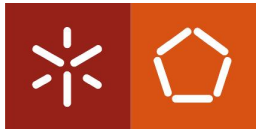
# Métodos Numéricos

A. Ismael F. Vaz

Departamento de Produção e Sistemas
Escola de Engenharia
Universidade do Minho
aivaz@dps.uminho.pt

Licenciatura em Engenharia Informática

Ano lectivo 2011/2012



# Conteúdo



# Conteúdo



# O que é o MATLAB?

## MATLAB

Começou como sendo um programa iterativo para cálculos com matrizes e transformou-se numa linguagem matemática de alto nível.
O seu desenvolvimento permite agora, por exemplo, a resolução de equações diferenciais e o desenho de gráficos a duas e três dimensões.
Possuindo o MATLAB uma linguagem de programação poder-se-ia dizer que qualquer algoritmo pode ser implementado em MATLAB.

## O que aprender?

Neste caso apenas iremos introduzir os comandos básicos e a criação de *scripts* necessários aos capítulos seguintes.
O MATLAB possui uma *toolbox* (entre outras) que fornece um conjunto de algoritmos para optimização.

# O que é o MATLAB?

## MATLAB

Começou como sendo um programa iterativo para cálculos com matrizes e transformou-se numa linguagem matemática de alto nível.
O seu desenvolvimento permite agora, por exemplo, a resolução de equações diferenciais e o desenho de gráficos a duas e três dimensões.
Possuindo o MATLAB uma linguagem de programação poder-se-ia dizer que qualquer algoritmo pode ser implementado em MATLAB.

## O que aprender?

Neste caso apenas iremos introduzir os comandos básicos e a criação de *scripts* necessários aos capítulos seguintes.
O MATLAB possui uma *toolbox* (entre outras) que fornece um conjunto de algoritmos para optimização.

# Só existe o MATLAB?

## Ferramentas similares

Outros programas similares são o *Mathematica* e o *Maple*

## *Mathematica*

Mais vocacionado para manipulação simbólica, embora o MATLAB também já incorpore algumas destas funcionalidades.

## *Maple* – Exemplo

Resolver a equação diferencial com condições iniciais

$$\frac{d^2y}{dx^2}(x) - 3y(x) = x, \quad y(0) = 1 \quad \frac{dy}{dx}(0) = 2$$

```
dsolve( diff(y(x),x,x) - 3*y(x) = x, y(0)=1, D(y)(0)=2, y(x) );
```

# Só existe o MATLAB?

## Ferramentas similares

Outros programas similares são o *Mathematica* e o *Maple*

## *Mathematica*

Mais vocacionado para manipulação simbólica, embora o MATLAB também já incorpore algumas destas funcionalidades.

## *Maple* – Exemplo

Resolver a equação diferencial com condições iniciais

$$\frac{d^2y}{dx^2}(x) - 3y(x) = x, \quad y(0) = 1 \quad \frac{dy}{dx}(0) = 2$$

```
dsolve( diff(y(x),x,x) - 3*y(x) = x, y(0)=1, D(y)(0)=2, y(x) );
```

# Só existe o MATLAB?

## Ferramentas similares

Outros programas similares são o *Mathematica* e o *Maple*

## *Mathematica*

Mais vocacionado para manipulação simbólica, embora o MATLAB também já incorpore algumas destas funcionalidades.

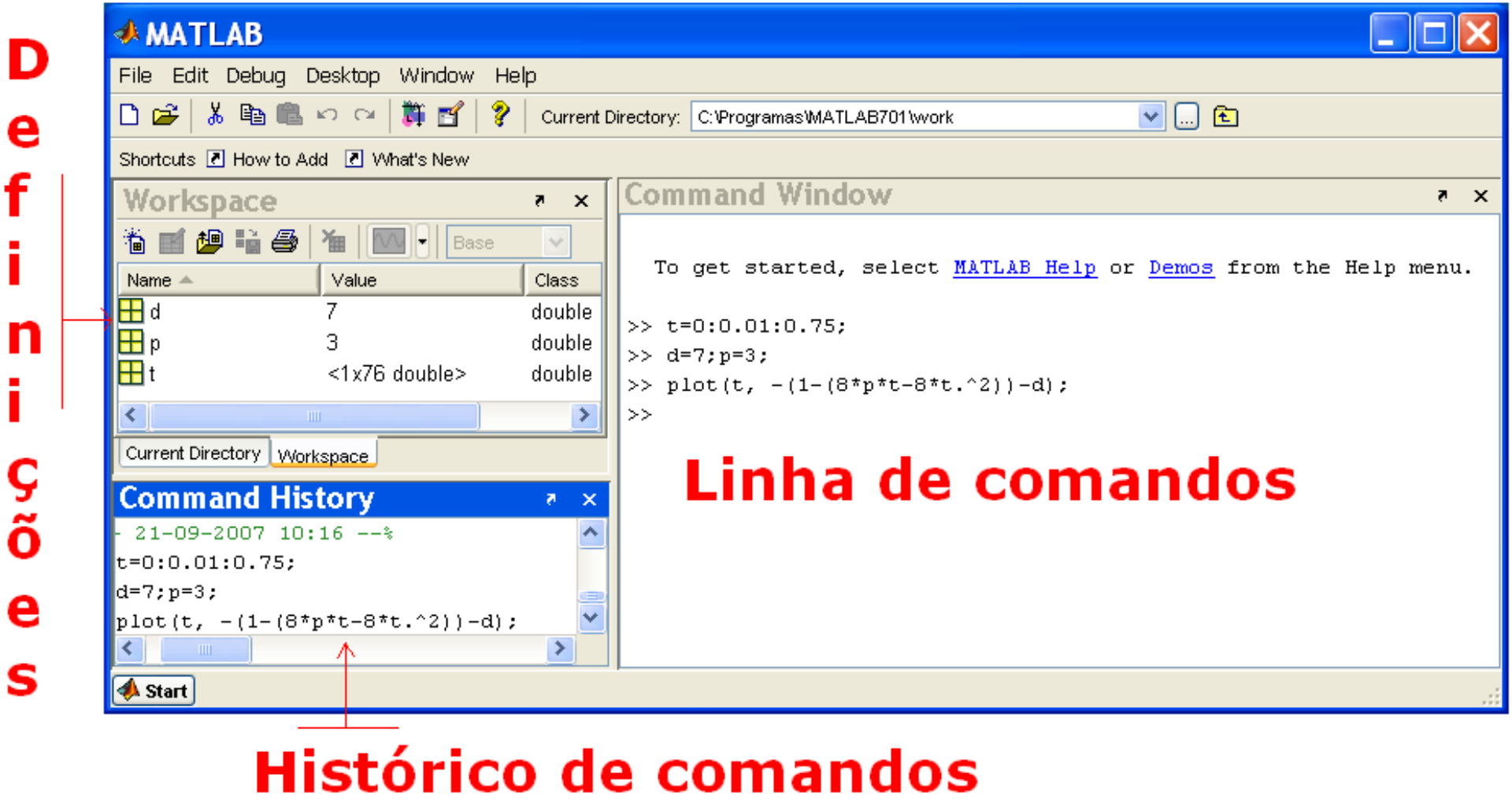
## *Maple* – Exemplo

Resolver a equação diferencial com condições iniciais

$$\frac{d^2y}{dx^2}(x) - 3y(x) = x, \quad y(0) = 1 \quad \frac{dy}{dx}(0) = 2$$

*dsolve( diff(y(x),x,x) - 3*y(x) = x, y(0)=1, D(y)(0)=2, y(x) );*

# Ambiente MATLAB



# Operações básicas

## Aritméticas

```
>> 2+3*2-1.5*2^2
ans =
     2
```

`ans` é uma variável *built-in* que é criada sempre que um resultado não é atribuído.

## Variáveis *built-in* – constantes

```
>> pi
ans =
    3.1416
```

# Operações básicas

## Funções

```
>> a=2*sin(pi)^2+3*exp(1)+sqrt(2)+cosh(2)
a =
   13.3313
```

## Ajuda

```
>> help acos
 ACOS   Inverse cosine.
    ACOS(X) is the arccosine of the elements of X. Complex
    results are obtained if ABS(x) > 1.0 for some element.

    See also cos, acosd.
...
```

# Operações básicas – Formatos

## Precisão

O MATLAB usa sempre aritmética com precisão de 15 algarismos significativos, no entanto, por defeito apenas mostra 4.

```
>> format long
>> a
a =
   13.33125473883387
>> format short
>> a
a =
    13.3313
>> format short e
>> a
a =
   1.3331e+001
```

# Operações básicas com escalares

## O ;

Os comandos que terminam com o ; não são impressos no ecrã.

```
>> a=log(2)+log10(2)+log2(2);
>> a
a =
    1.9942
>> a=log(2)+log10(2)+log2(2)
a =
    1.9942
```

Útil quando se pretende efectuar vários comandos seguidos (*scripts*) e não se pretende ver determinados resultados.

# Operações básicas com vectores e matrizes

## Vectores linha

```
>> a=[1 2 3]
a =
     1     2     3
>> a=[1,2,3]
a =
     1     2     3
```

## Vectores coluna

```
>> a=[1;2;3]
a =
     1
     2
     3
```

## O(A) transposto(a)

```
>> a=[1;2;3]'
a =
     1     2     3
```

## O(A) transposto(a)

```
>> a=[1,2,3]'
a =
     1
     2
     3
```

# Operações básicas com vectores e matrizes

## Operações com vectores – Funções

```
>> a
a =
     1     2     3
>> b=sin(a)
b =
    0.8415    0.9093    0.1411
```

## Operações com vectores

```
>> a+b
ans =
    1.8415    2.9093    3.1411
>> a^2
??? Error using ==> mpower
Matrix must be square.
```

# Operações básicas com vectores e matrizes

## Operações elemento a elemento

```
>> a.^2
ans =
     1     4     9
>> (a+b).^2
ans =
    3.3910    8.4640    9.8666
>> a.*b
ans =
    0.8415    1.8186    0.4234
```

## Operações com vectores

```
>> 2*a+b.^2
ans =
    2.7081    4.8268    6.0199
```

# Operações básicas com vectores e matrizes

## Operações elemento a elemento

```
>> a*b
??? Error using ==> mtimes
Inner matrix dimensions must agree.
>> a*b'
ans =
    3.0834
```

## Operações com vectores

```
>> a'*b
ans =
    0.8415    0.9093    0.1411
    1.6829    1.8186    0.2822
    2.5244    2.7279    0.4234
```

# Operações básicas com vectores e matrizes

## Matriz

```
>> A=[ 1 2 3; 4 5 6]

A =

     1     2     3
     4     5     6
```

Matriz $2 \times 3$. O MATLAB é *case sensitive*, *i.e.*, `A` é diferente de a.

## Matriz transposta

```
>> A=[ 1 2 3; 4 5 6]'

A =

     1     4
     2     5
     3     6
```

Matriz $3 \times 2$.

# Operações básicas com vectores e matrizes

## Definição

```
>> A+B
ans =
     3     5
     5     7
     7     9
>> A'+B'
ans =
     3     5     7
     5     7     9
>> (A+B)'
ans =
     3     5     7
     5     7     9
```

## Soma

```
>> A=[2 3; 3 4; 4 5]
A =
     2     3
     3     4
     4     5
>> B=[1 2; 2 3; 3 4]
B =
     1     2
     2     3
     3     4
```

# Operações básicas com vectores e matrizes

## Produto

```
>> A*B
??? Error using ==> mtimes
Inner matrix dimensions ...
>> A*B'
ans =
     8     13     18
    11     18     25
    14     23     32
```

O número de colunas da primeira matriz tem de ser igual ao número de linhas da segunda matriz.

## Produto elemento a elemento

```
>> A.*B
ans =
     2      6
     6     12
    12     20
>> A^2
??? Error using ==> mpower
Matrix must be square.
>> A.^2
ans =
     4      9
     9     16
    16     25
```

# Sistemas lineares

Um sistema linear pode ser representado na forma de matricial como $Ax = b$, em que $A$ é uma matriz (dos coeficientes), $x$ é a solução do sistemas e $b$ é um vector (dos termos independentes).

## Sistema

$$\left\{ \begin{array}{llll} x_1 & +2x_2 & +3x_3 & = 1 \\ 4x_1 & +5x_2 & +6x_3 & = 2 \\ 7x_1 & +8x_2 & +9x_3 & = 3 \end{array} \right. \equiv \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

$$\textit{i.e.} \quad A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad \text{e} \quad b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

# Resolução de Sistemas lineares

## O exemplo anterior

```
>> A=[1 2 3; 4 5 6; 7 8 9]; b=[1 2 3]';
>> x=A\b
Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 2.203039e-018.
x =
   -0.3333
    0.6667
         0
>> A*x-b
ans =
  1.0e-015 *
         0
   -0.2220
         0
```

# Resolução de Sistemas lineares

## Curiosidades

```
>> det(A)
ans =
     0
>> [L,U]=lu(A)
L =
    0.1429    1.0000         0
    0.5714    0.5000    1.0000
    1.0000         0         0
U =
    7.0000    8.0000    9.0000
         0    0.8571    1.7143
         0         0   -0.0000
```

# Resolução de Sistemas lineares

## Não fazer – Porquê?

```
>> x=inv(A)*b
Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 2.203039e-018.
x =
     0
     0
     0
```

# Acesso a elementos de vectores e matrizes

## Acesso a vectores

```
>> b=sin(b);
>> b(1)
ans =
    0.8415
>> b(2:3)
ans =
    0.9093
    0.1411
>> b(:)
ans =
    0.8415
    0.9093
    0.1411
```

## Acesso a matrizes

```
>> A(2,2)
ans =
     5
>> A(2,:)
ans =
     4     5     6
>> A(:,1)
ans =
     1
     4
     7
>> A(1:2,2:3)
ans =
     2     3
     5     6
```

# Matrizes e vectores especiais

## Zeros e uns

```
>> zeros(2,3)
ans =
     0     0     0
     0     0     0
>> ones(2,1)
ans =
     1
     1
>> t=1:1:3
t =
     1     2     3
>> t=1.1:0.1:1.2
t =
    1.1000    1.2000
```

## Identidade

```
>> eye(3)
ans =
     1     0     0
     0     1     0
     0     0     1
>> rand(3,2)
ans =
    0.6068    0.7621
    0.4860    0.4565
    0.8913    0.0185
>> randn(2,2)
ans =
    1.1892    0.3273
   -0.0376    0.1746
```

# Modificação de elementos em vectores e matrizes

## Atribuições

```
>> A=ones(3,3);
>> A(1:2,1)=3
A =
     3     1     1
     3     1     1
     1     1     1
>> A(3,1:2:3)=4
A =
     3     1     1
     3     1     1
     4     1     4
```

## Troca de valores

```
>> A([1 2],1)=2*A([1 2],1)
A =
     6     1     1
     6     1     1
     4     1     4
>> A([1 3],1)=A([3 1],1)
A =
     4     1     1
     6     1     1
     6     1     4
```

# Operadores lógicos e o *find*

## find

```
>> v=[0.1 0.4 0.12 0.13]
v =
    0.1000    0.4000    0.1200    0.1300
>> i=find(v>0.12)
i =
     2     4
>> a=v(i)
a =
    0.4000    0.1300
>> i=find(v==0.1)
i =
     1
>> i=find(v~=0.1)
i =
     2     3     4
```

# Operadores lógicos e o *find*

## Operadores lógicos

| Símbolo | Representa | Símbolo | Representa |
|---|---|---|---|
| > | Maior que | >= | Maior ou igual que |
| < | Menor que | <= | Menor ou igual que |
| ~= | Diferente de | == | Igual a |
| ~ | Negação | & | E |
| \| | Ou | | |

# Funções básicas

## Funções

| Função | Descrição |
|---|---|
| max | Elemento máximo de um vector |
| min | Elemento mínimo de um vector |
| sum | Soma de todos os elementos |
| mean | Média aritmética |
| stdev | Desvio padrão |

# Mensagens e *display* de variáveis

## Ficheiros

```
>> x = 0:.1:1; y = [x; exp(x)];
>> fid = fopen('exp.txt','w')
fid =
     3
>> fprintf(fid,'%6.2f  %12.8f\n',y);
>> fclose(fid);
```

## Conteúdo de *exp.txt*

```
  0.00     1.00000000
  0.10     1.10517092
  0.20     1.22140276
  0.30     1.34985881

...

  0.90     2.45960311
  1.00     2.71828183
```

# Mensagens e *display* de variáveis

**Terminal**

```
>> x=1;y=2;
>> fprintf('Duas variáveis: %5.1f %6.2f\n',x,y);
Duas variáveis:   1.0   2.00
>> x=[1 2];y=[3 4];
>> fprintf('Dois vectores: %5.1f %6.2f\n',x,y);
Dois vectores:   1.0   2.00
Dois vectores:   3.0   4.00
```

**Terminal**

```
>> x
x =
     1     2
>> disp(x)
     1     2
```

**Terminal**

```
>> x=1.23;
>> s=sprintf('Uma string %4.2f',x)
s =
Uma string 1.23
```

# Scripts

## Definição

Um *script* trata-se da execução de uma série de comandos. Os *scripts* são guardados em ficheiros de extensão `.m` e por isso designámos por *M-Files* (ficheiros M).

## Ficheiro `bioinf.m`

```
x = 0:.1:1;
y = exp(x);
fprintf('%4.2f  %8.4f\n',x,y);
```

## Execução

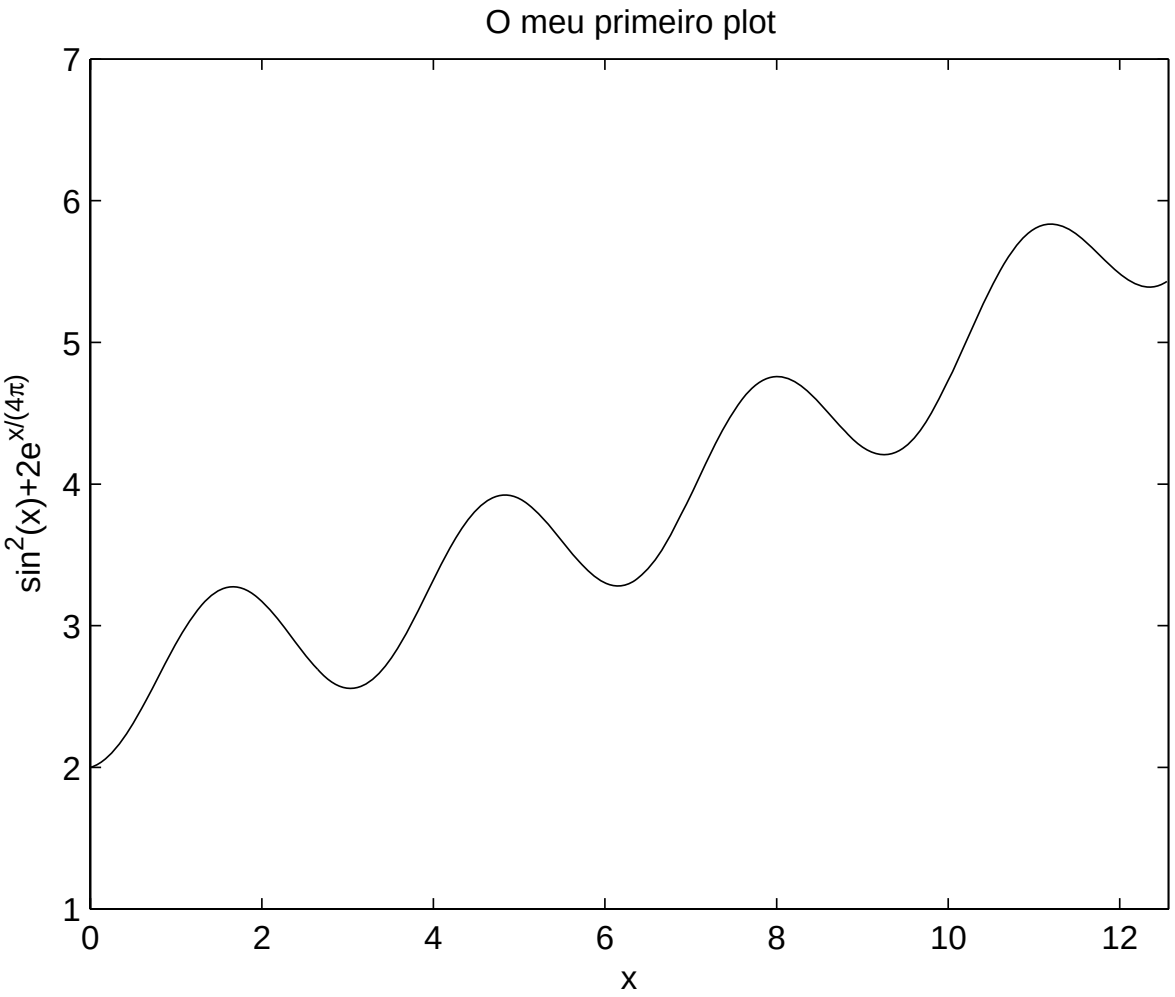
```
>> bioinf
0.00    0.1000
0.20    0.3000
...
2.01    2.2255
2.46    2.7183
```

# Desenho de gráficos 2D

## Plot

```
>> x=0:0.05:4*pi;
>> plot(x,sin(x).^2+2*exp(x/(4*pi)));
>> xlabel('x');
>> ylabel('sin^2(x)+2e^{x/(4\pi)}');
>> title('O meu primeiro plot');
>> axis([0 4*pi 1 7]);
```

# Desenho de gráficos 2D



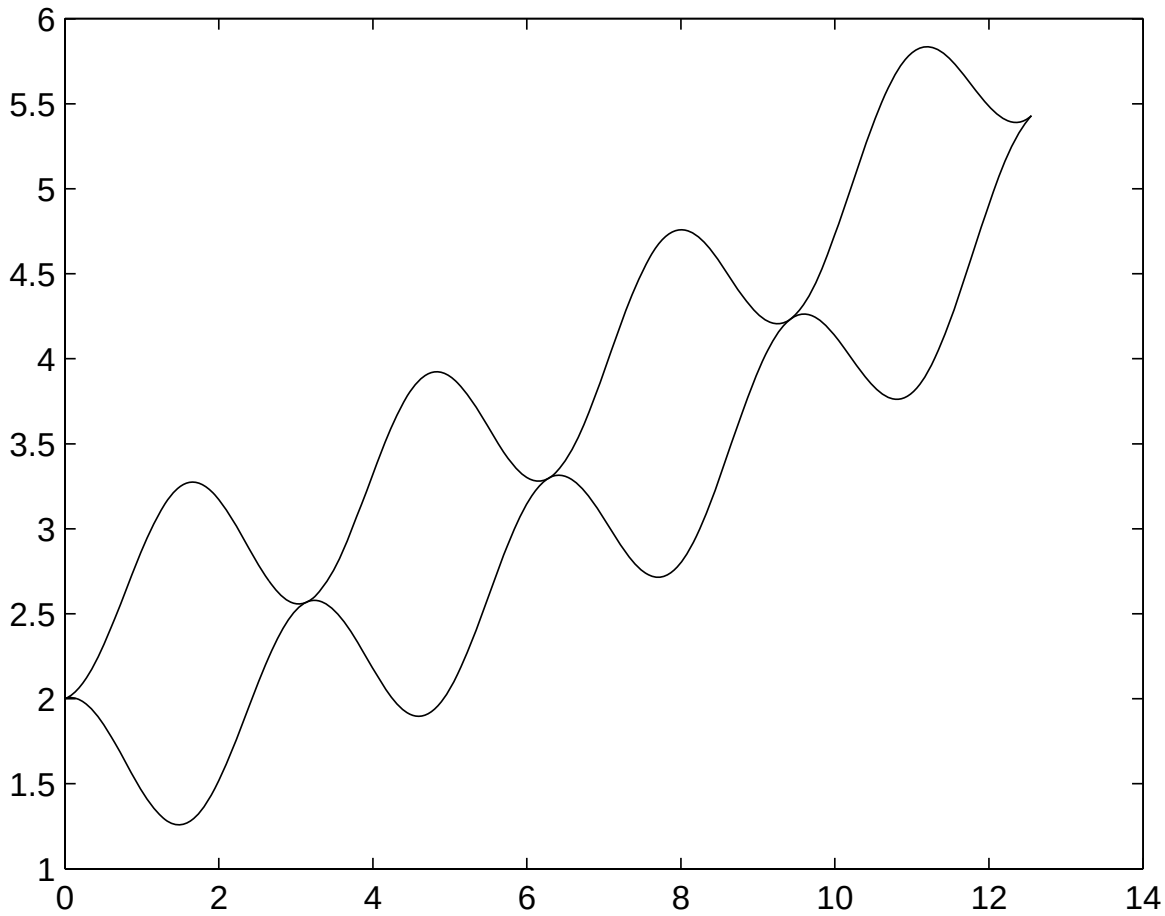
# Desenho de gráficos 2D

## Sobreposição

```
>> x=0:0.05:4*pi;
>> plot(x,sin(x).^2+2*exp(x/(4*pi)));
>> hold on;
>> plot(x,-sin(x).^2+2*exp(x/(4*pi)));
>> hold off;
>> x=0:0.05:4*pi;
>> y1=sin(x).^2+2*exp(x/(4*pi));
>> y2=-sin(x).^2+2*exp(x/(4*pi));
>> plot(x,y1,x,y2);
```

Dois gráficos quase idênticos (atenção às cores).

# Desenho de gráficos 2D

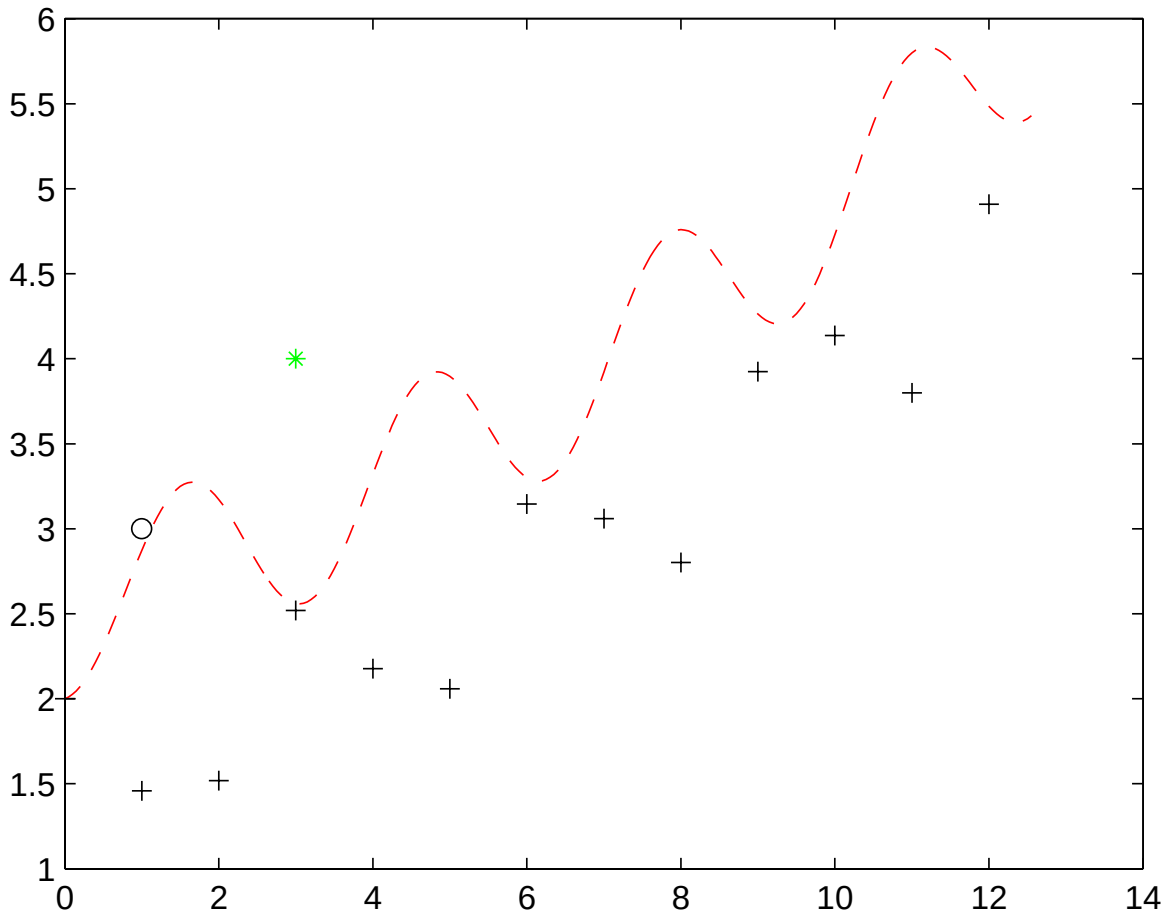


# Desenho de gráficos 2D

## Usando marcas e tipos de linhas

```
>> x=0:0.05:4*pi;
>> y=0:1:4*pi;
>> plot(x,sin(x).^2+2*exp(x/(4*pi)),'--r',1,3,'ok',...
3,4,'*g',y,-sin(y).^2+2*exp(y/(4*pi)),'+k');
```

# Desenho de gráficos 2D

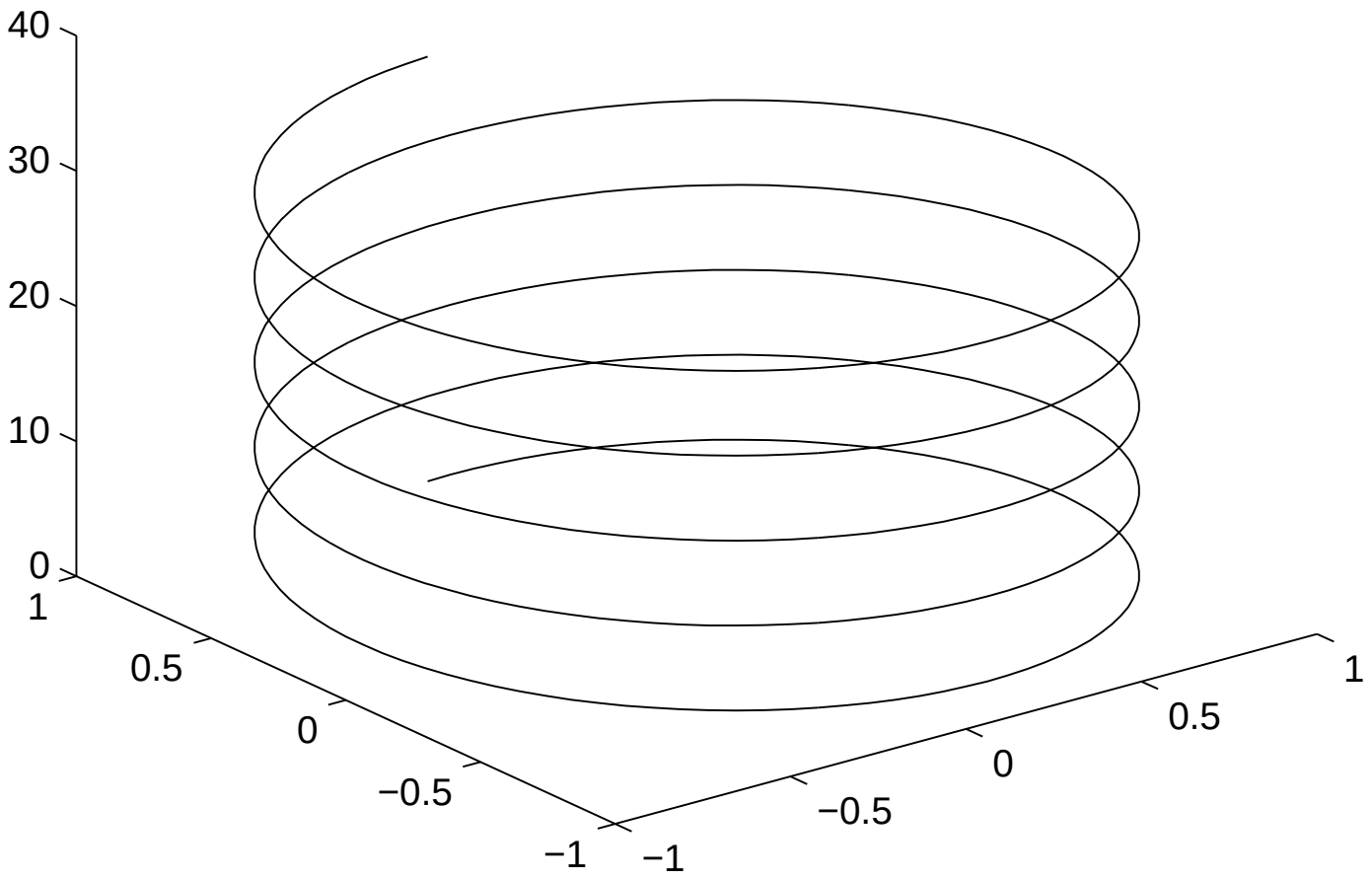


# Desenho de gráficos 3D

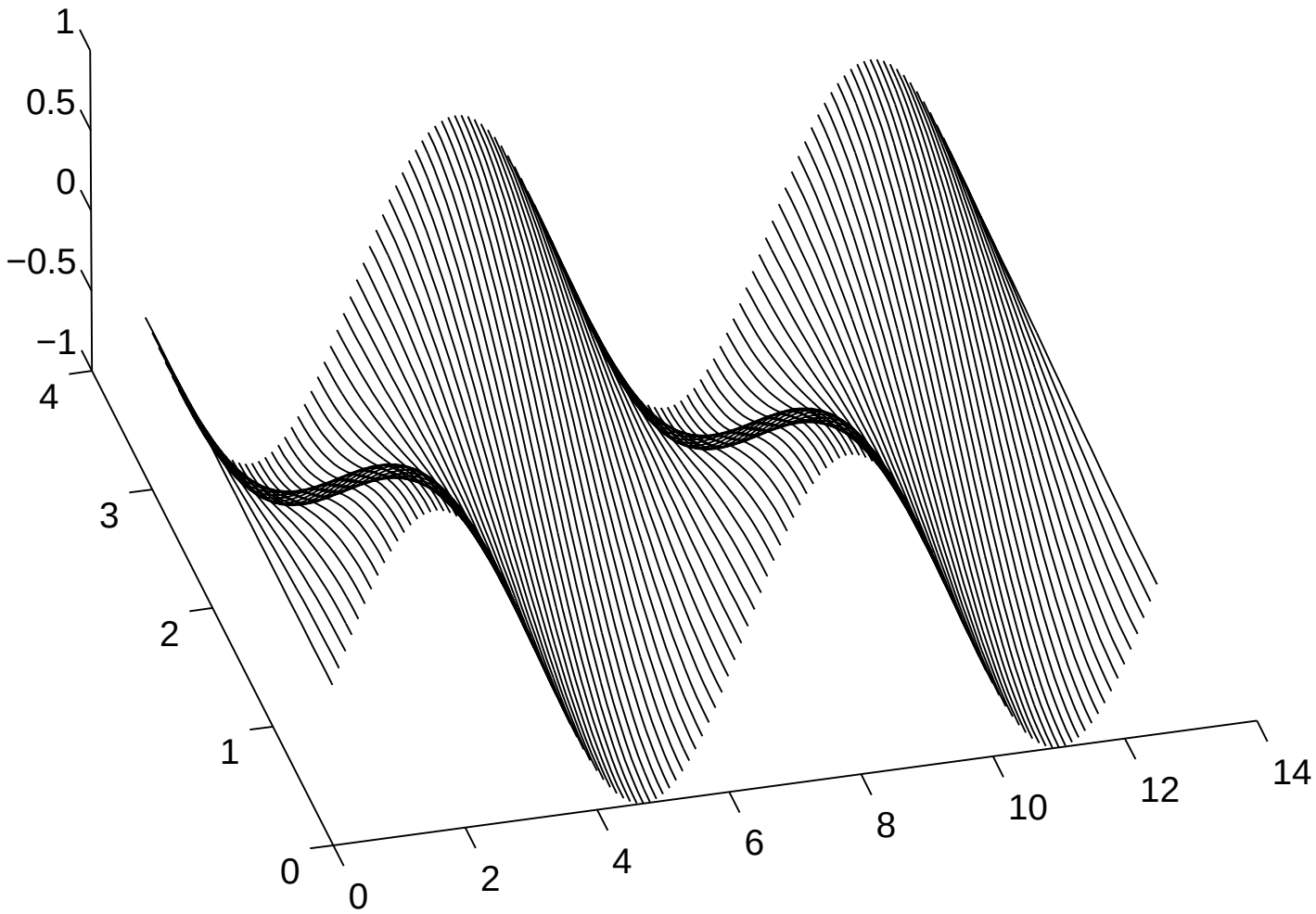
## Plot3

```
>> t = 0:pi/50:10*pi;
>> plot3(sin(t),cos(t),t);
>> [x,y]=meshgrid(0:0.1:4*pi,0:0.1:pi);
>> plot3(x,y,sin(x).*cos(y));
>> surf(x,y,sin(x).*cos(y));
```

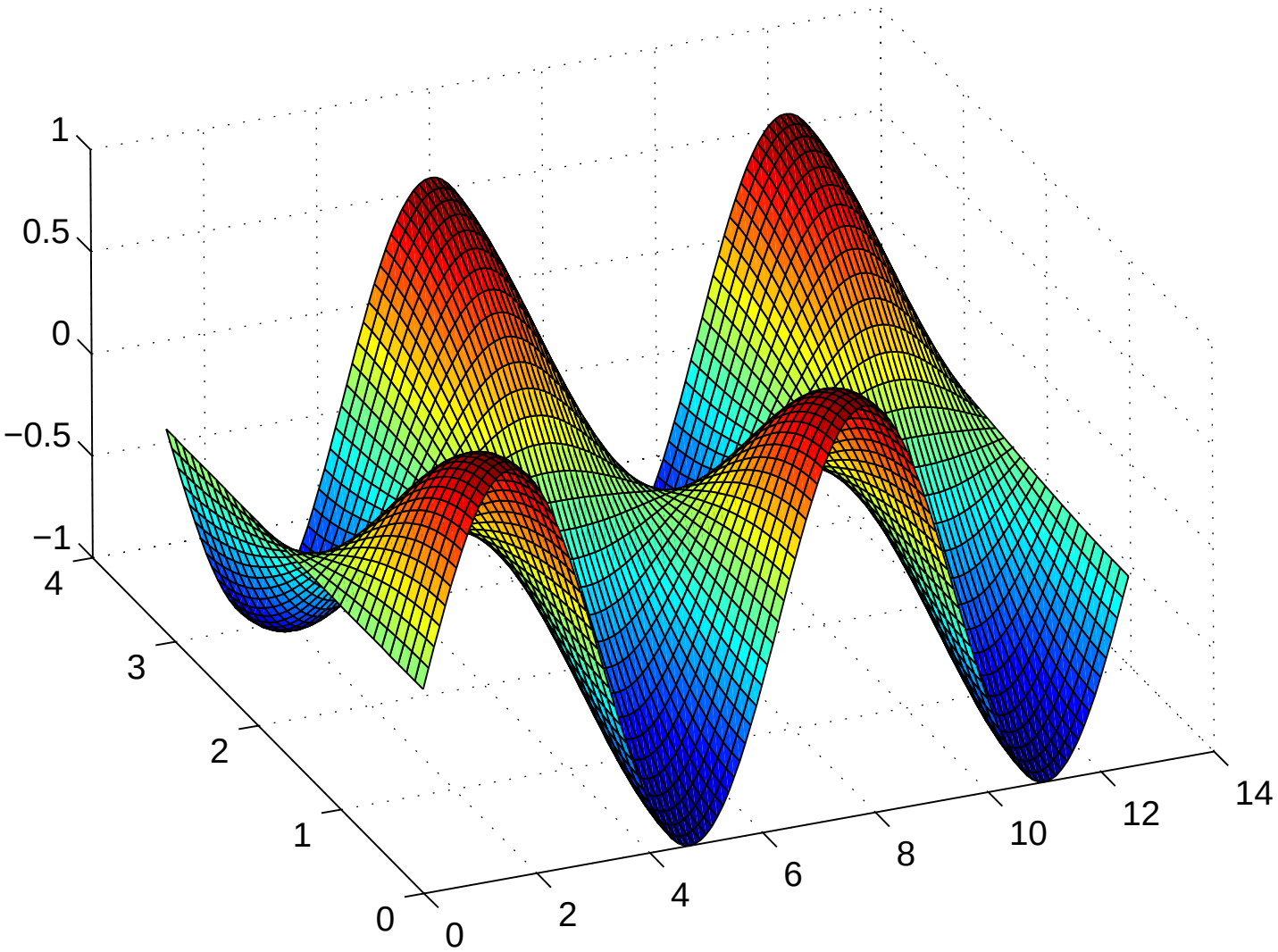
# Desenho de gráficos 3D



# Desenho de gráficos 3D



# Desenho de gráficos 3D

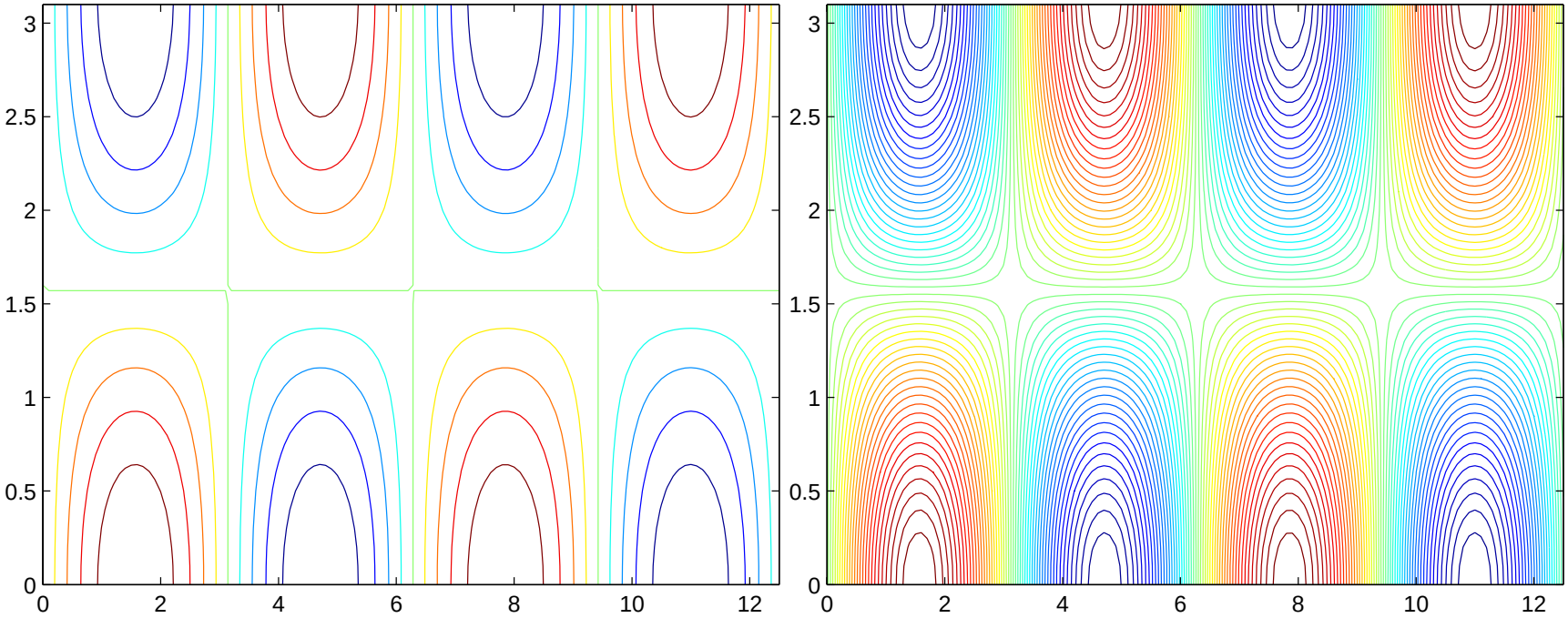


# Desenho de curvas de nível

## Contour

```
>> [x,y]=meshgrid(0:0.1:4*pi,0:0.1:pi);
>> contour(x,y,sin(x).*cos(y));
>> contour(x,y,sin(x).*cos(y),50);
```

# Desenho de curvas de nível




# Funções MATLAB

As funções em MATLAB são escritas em ficheiros M. O nome do ficheiro deve corresponder ao nome da função

## Execução

```
>> simples(2)
ans =
     4
>> a=simples([1 2])
a =
     1     4
>> b=simples([1 2; 2 3])
b =
     1     4
     4     9
```

## simples.m

```
function f = simples(x)
% O quadrado de x

% .^ para ...
f=x.^2;
```

## Help

```
>> help simples
  O quadrado de x
```

# Funções MATLAB – O `if`

myfun.m

```
function [a, b] = myfun(x,y)
%
% Argumentos de entrada:
%   x - O meu primeiro argumento
%   y - O meu segundo argumento
%
% Argumentos de saída:
%   a - O meu primeiro argumento de saída
%   b - O meu segundo argumento de saída

% Isto já não aparece no Help
```

# Funções MATLAB – O `if`

## `myfun.m` – Cont.

```
% Verificar os argumentos de entrada
[d1,d2]=size(x);
if d2 ~= 1
    error('Só aceito vectores coluna');
else
    if nargin < 2
        y=ones(d1,1); % Valor por defeito para o y
    else
        [d3,d4]=size(y);
        if d4 ~= 1
            error('Só aceito vectores coluna');
        end
        if d3 ~= d1
            error('Dimensões de x e y não são iguais');
        end
    end
end
```

# Funções MATLAB – O `if`

## myfun.m – Cont.

```
if nargout < 1
    error('Pelo menos um argumento de saída');
end

a=2*x+y;

if nargout > 1
    b=3*x+2*y;
end
```

# Funções MATLAB – O `if`

## Execução

```
>> myfun([1 2])
??? Error using ==> myfun
Só aceito vectores coluna
>> myfun([1 2]')
??? Error using ==> myfun
Pelo menos um argumento de saída
>> f=myfun([1 2]')
f =
     3
     5
>> f=myfun([1 2]',[1 2 3])
??? Error using ==> myfun
Só aceito vectores coluna
```

# Funções MATLAB – O `if`

## Execução

```
>> f=myfun([1 2]',[1 2 3]')
??? Error using ==> myfun
Dimensões de x e y não são iguais
>> [f1,f2]=myfun([1 2]',[1 2]')
f1 =
     3
     6
f2 =
     5
    10
```

# Funções MATLAB – O `for`

## Execução

```
>> for i=1:2
fprintf('%d --> %d\n',i,2*i);
end
1 --> 2
2 --> 4
```

# Funções MATLAB – O `inline`

## Execução

```
>> g=inline('sin(x)');
>> g(1)
ans =
    0.8415
>> g=inline('sin(x)*a','x','a');
>> g(1,2)
ans =
    1.6829
```

# Zeros de funções

## A função fzero

```
>> g=inline('cos(x)');
>> fzero(g,1.1)
ans =
    1.5708
```