Introdução ao MATLAB

Prof: José Manoel Baltazar

Introdução

Esta apresentação visa passar os conceitos básicos do uso do Matlab.

MATLAB: Comandos Gerais

- **Variáveis:**
  - **<variável = expressão>**
  - **Ex: A=125; A=2500/20;**
- **Vetores:**
  - **<valor_inicial : incremento : valor_final>**
  - **Ex: Z = 6: -1: 1**

    Z = 6 5 4 3 2 1
- **Matrizes:**
  - **Ex: A = [1 2 3 ; 4 5 6 ; 7 8 9]**

    A = 1 2 3

    4 5 6

    7 8 9

Operação com Matrizes

- + Adição
- - Subtração
- * Multiplicação
- / Divisão à direita
- \ Divisão à esquerda
- ^ Exponenciação
- ' Transposta

Exemplos:

- **Adição e Subtração:**
  - **A = [1 2 3 ; 4 5 6 ; 7 8 9];**

    B = A' ;

    C = A + B

    C = 2 6 10

    6 10 14

    10 14 18
  - **C = B - A**

    C = 0 2 4

    -2 0 2

    -4 -2 0

- **Multiplicação:**
  - X = [-1 0 2];

    Y = [-2; -1; 1];

    ans = X*Y

    ans = 4
- **Divisão:**
  - A = [1 2 3 ; 4 5 6 ; 7 8 9];

    B = A' ;

    X = A\B

    X = -0.2334 -0.3333 -1.1344

    0.4667 -2.3333 -3.7311

    0.1000   3.0000   5.1989

- **Transposta:**
  - **A = [1 2 3 ; 4 5 6 ; 7 8 9];**

    B = A'

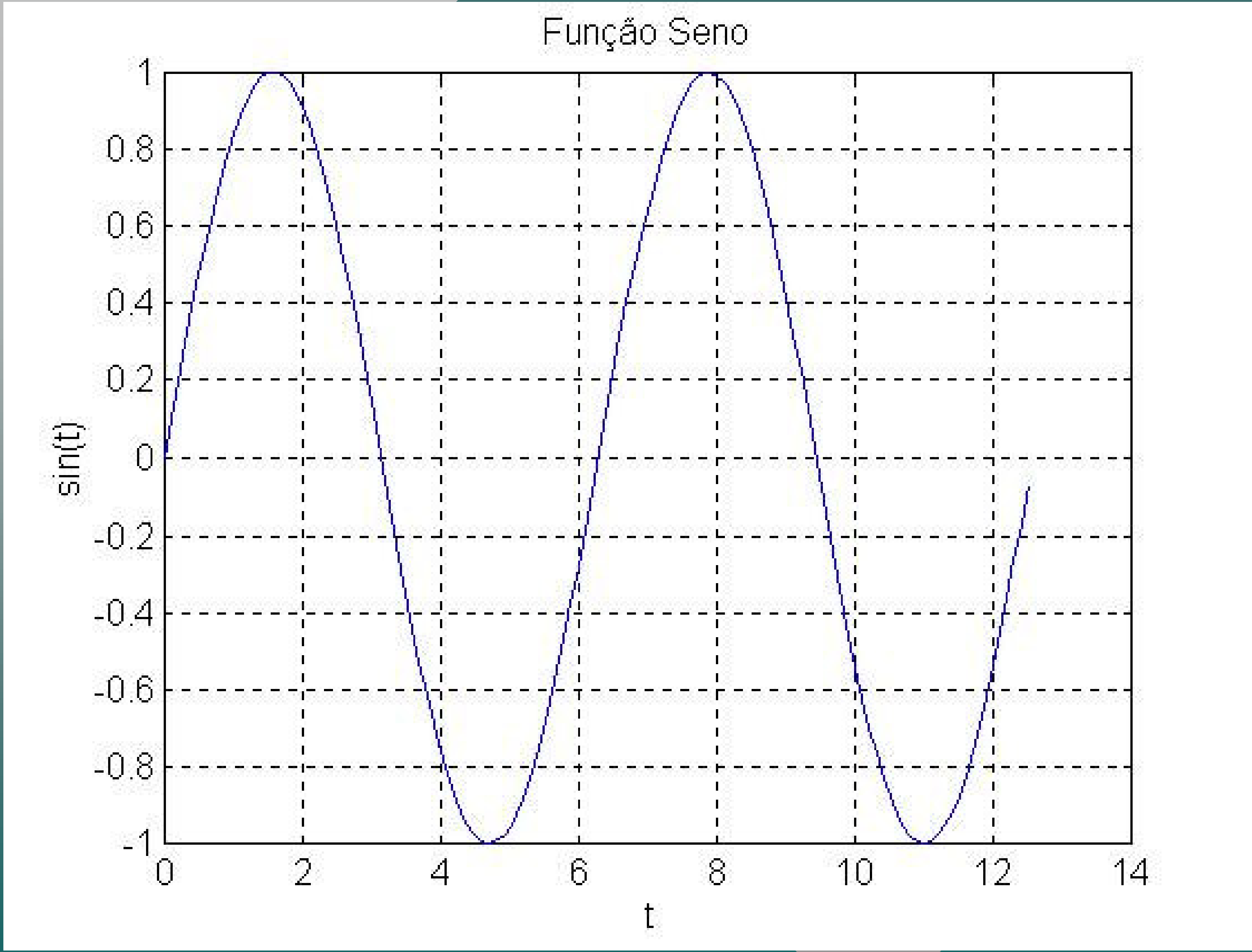
    C = 1 4 7

    2 5 8

    3 6 9

Gráficos

- **Forma Básica:**
  - **Ex: Código para plotar um gráfico do seno no matlab:**
    - **» t=0:.1:4*pi;**
    - **» Y=sin(t);**
    - **» whitebg;**
    - **» plot(t,Y,'b');**
    - **» grid;**
    - **» xlabel('t');**
    - **» Ylabel('sin(t)');**
    - **» Title('Função Seno');**

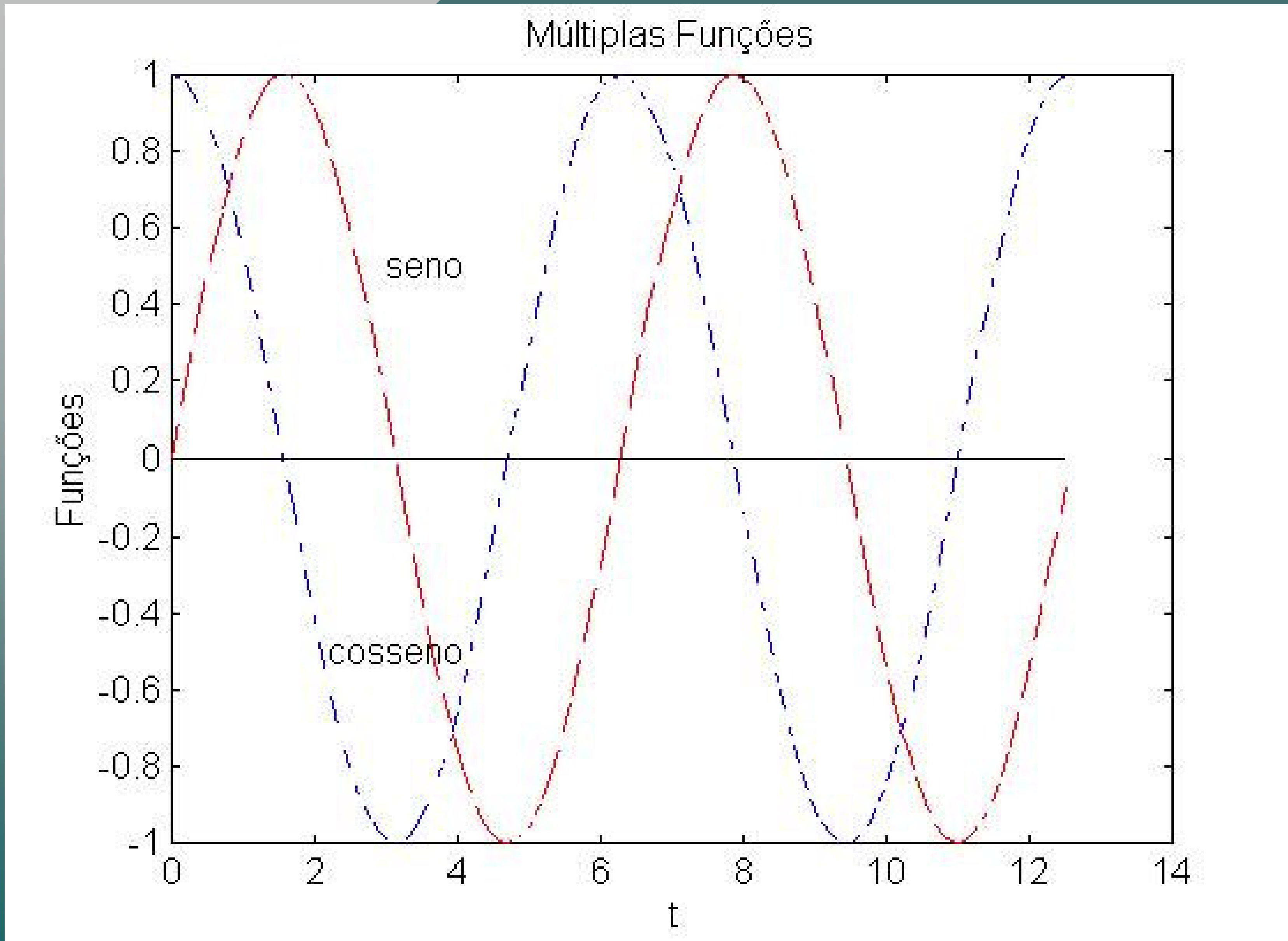


Função Seno
sin(t)
t
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14

Múltiplas Linhas

- Ex: Código para plotar um gráfico de múltiplas funções no matlab:

```
» t=0:.1:4*pi;
» Y=sin(t);
» Z=cos(t);
» W=0*t;
» plot(t,Y,'--',t,Z,'-.b',t,W,'black')
» text(3,0.5,'seno')
» text(2.2,-0.5,'cosseno')
» Ylabel('Funções')
» title('Múltiplas Funções')
» Xlabel('t')
```



Múltiplas Funções
seno
cosseno
Funções
t
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14

Comandos para Execução de Expressões

- **[v1,v2,...,vm]=feval(‘<função>’,a1,a2,...an)**
  - **Exemplo:**

>> x = rem(5,3)

x = 2

>> y = feval(‘rem’,5,3)

y = 2

- **Exemplo 2:**

```
Tabela=['sqrt(x)   '
        'exp(x)    '
        '1/x+5*x']
N=input('Escolha a Expressão: ');
X=input('Defina o Argumento: ');
A=eval(Tabela(N, ))
```

Operações com Polinômios

- Avaliação
  - **$P(x) = 3x^5 - 2x^4 + 5x^3 + 7x^2 - 3x + 1$ nos pontos**

    x = 1,2,3,4 e 5

    ```
    >>c = [3 -2 5 7 -3 1];
    >>x = 1:5;
    >>Y = polyval(c,x)
    ```

    y = 11 127 757 2981 8911

- Adição e Subtração

  - $c = 7x^3 + x^2 + x - 3$

    >>a = [5 –4 0 1];

    >>b = [2 5 –1 4];

    >>c = [a+b]

    7 1 -1 -3

- **Idem Subtração**

- Multiplicação
  - $a(x) = 3x^2 - 5x + 4$
  - $b(x) = 2x - 1$

```
>>a = [3 -5 4];
>>b = [2 -1];
>>c = conv(a,b)
          c = 6  -13  13  -4
```

$g(x) = 6x^3 - 13x^2 + 13x - 4$

- Divisão
  - **$P(x) = 2x^4 - 3x^3 + 4x^2 - 5x + 6$ por**
  - **$e(x) = x^2 - 3x + 1$**

```
>>d = [2 –3 4 –5 6];
>>e = [1 –3 1];
>>[q,r] = deconv(d,e)
                q = 2   3   11
                 r = 0   0  25  -5
```

$q(x) = 2x^2 - 3x + 11x$ **$r(x) = 25x - 5$**

- Derivação
  - **$p(x) = x^4 + 2x^3 - 12x^2 + 14x - 5$**

```
>>p = [1  2  12  14  –5];
>>p1 = polyder(p)
            p1 = 4  6  -24  14
>>p2 = polyder(p1)
            p2 = 12  12  -24
```

$P'(x) = 4x^3 + 6x^2 - 24x + 14$

•Cálculo de Raízes

- MATLAB possui duas poderosas funções para o cálculo de raízes de equações.
- A função **roots** obtém todas as raízes de uma equação algébrica e a função **fzero** determina uma raiz de uma equação algébrica ou transcendente.

- Função roots

$p(x) = x^4 + 2x^3 - 13x^2 - 14x + 24$

```
>>c = [1  2  -13  -14  +24];
>>r = roots(c)

    r =  -4.000   3.000   -2.000   1.000
```

- Função fzero

$f(x) = 3x^2\sin(x)\exp(-x)$, $x_0 = 2$

```
    function y=f(x)
  y  = 3*x^2*sin(x)*exp(-x)  com x0=2

  >>raiz = fzero('f',2)
  >>raiz = 3.1416
```

Aplicações

- **O Matlab pode ser usado na resolução de problemas matemáticos e computacionais:**
  - **Resolução de Equações Diferenciais**
    - **Os comandos do MATLAB para resolver equações diferenciais ordinárias são:**

-ode23 : Resolver equação diferencial. método baixa ordem.

-ode23p : Resolver e plotar soluções.

-ode45 : Resolver equação diferencial. Método para alta ordem .

Equação de Lorenz :

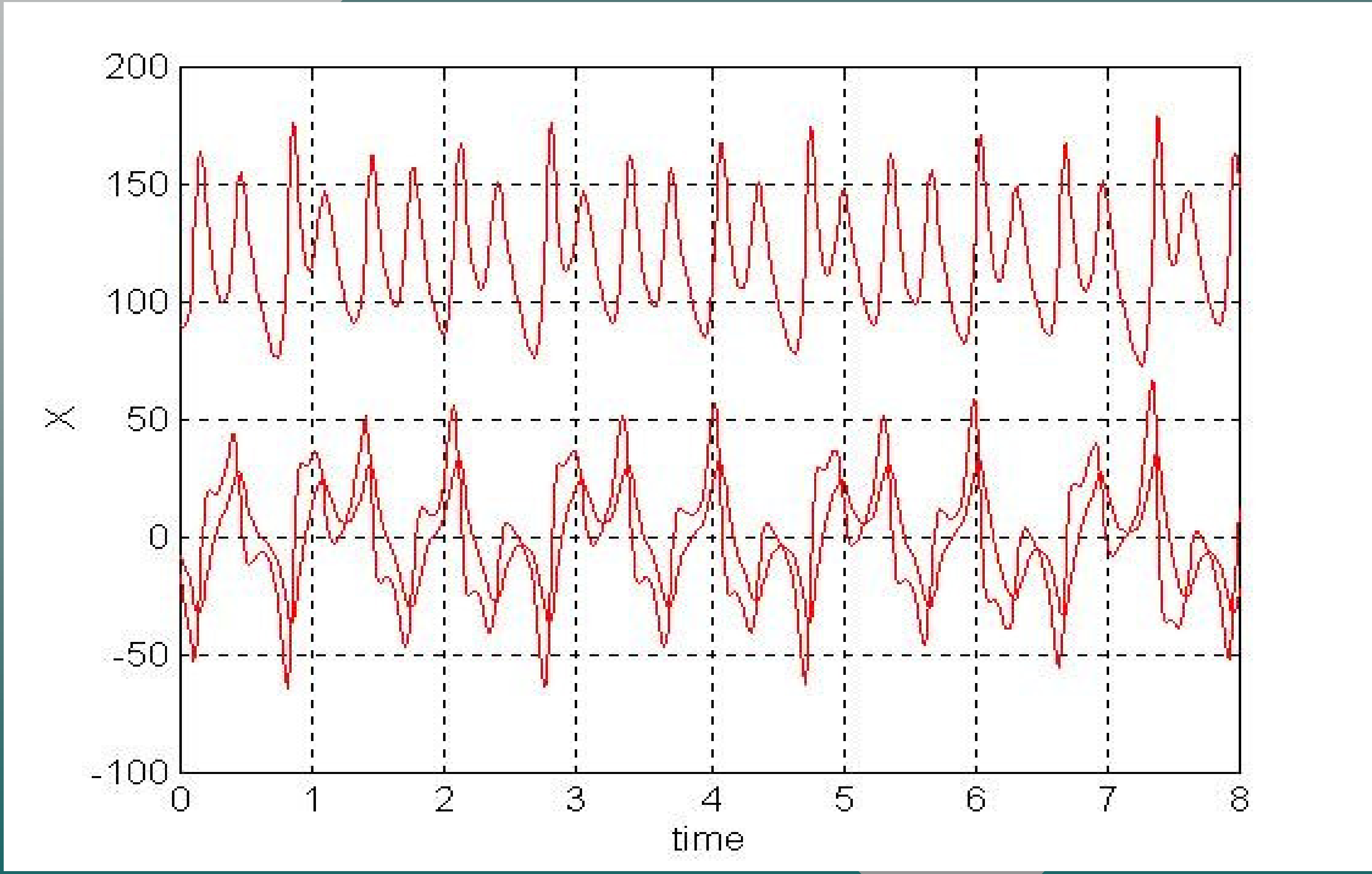
Para isso, cria-se subprogramas e funções através de um editor de texto (notepad), que serão chamadas através do Matlab:

```
% Solução da equação de Lorenz
global r
r= input('Entre um Valor para a constante r: ');
simtime= input('Entre com o tempo de execução: ');
acc= input('Entre um valor para a precisão: ');
initx = [-7.69 -15.61 90.39]';
% Chamada da função ode45 para resolver equações
[t x]=ode45('f505',0,simtime,initx,acc);
% Plota o gráfico do Resultado contra o Tempo
whitebg('w');
figure(1); plot(t,x,'red');
Xlabel('time'); Ylabel('X');
grid;
figure(2); plot(x(:,1),x(:,3),'red');
Xlabel('X'); ylabel('Z');
grid;
```

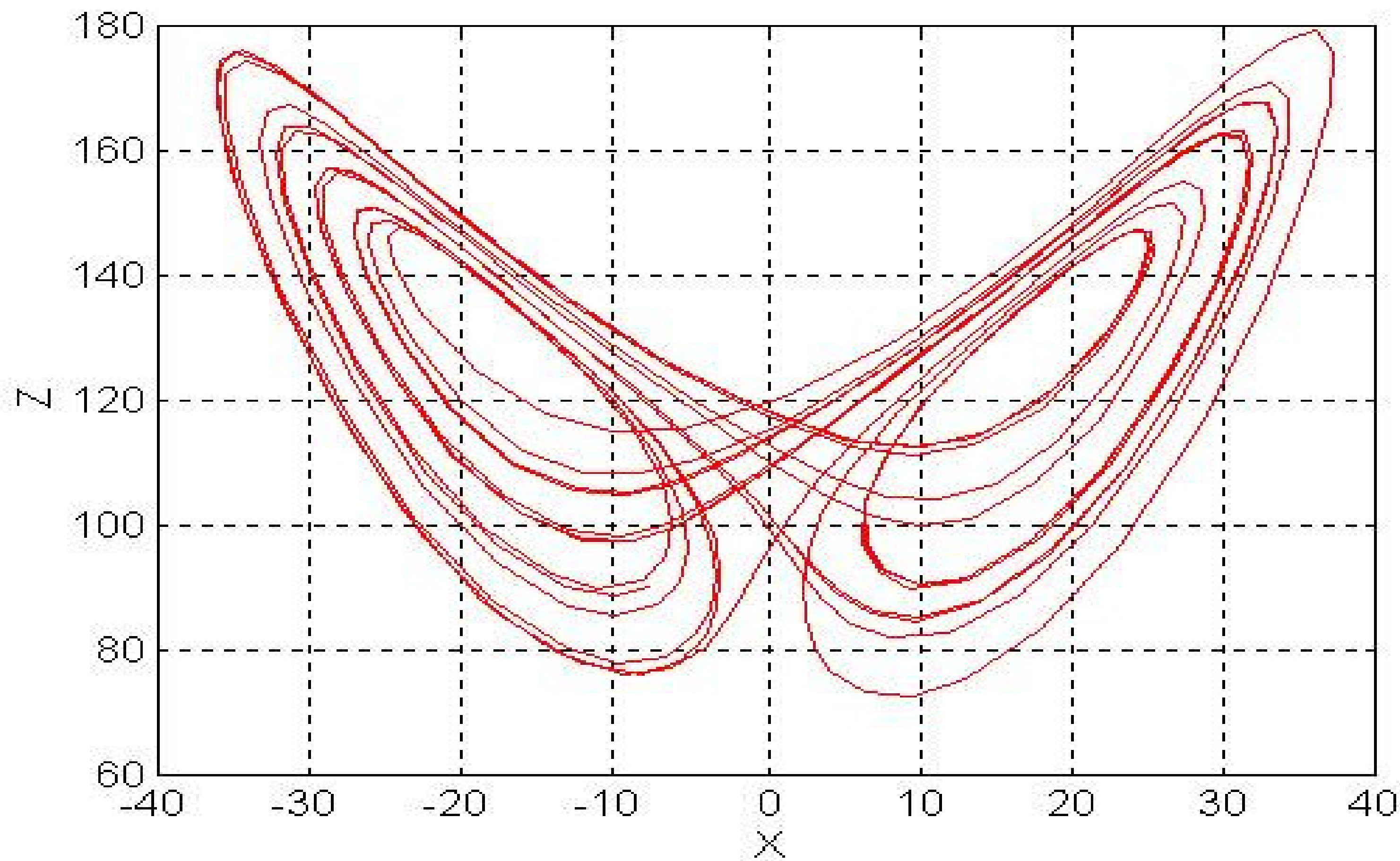
Código da função f505, utilizado pelo subprograma anterior:

```
Function fv=f505(t,x)
%x, y e z são representados por x(1), x(2) e x(3)
global r
fv=zeros(3,1); fv(1)=10*(x(2)-x(1));
fv(2)=r*x(1)-x(2)-x(1)*x(3);
fv(3)=x(1)*x(2)-8*x(3)/3;
```

•Gráfico da solução da equação de lorenz, onde cada variável é colocada contra o tempo.



•Gráfico da solução da equação de Lorenz, para r=126.52, usando precisão de 0.000005 e tempo de execução igual a 8



Aplicações

Considere a equação diferencial de segunda ordem chamada de *Equação de Van der Pol*

$$x + (x^2 - 1) . x + x = 0$$

Pode-se reescrever esta equação como um sistema acoplado de equações diferenciais de primeira ordem

$$x_1 = x_1 . (1-x_2^2) - x_2$$
$$x_2 = x_1$$

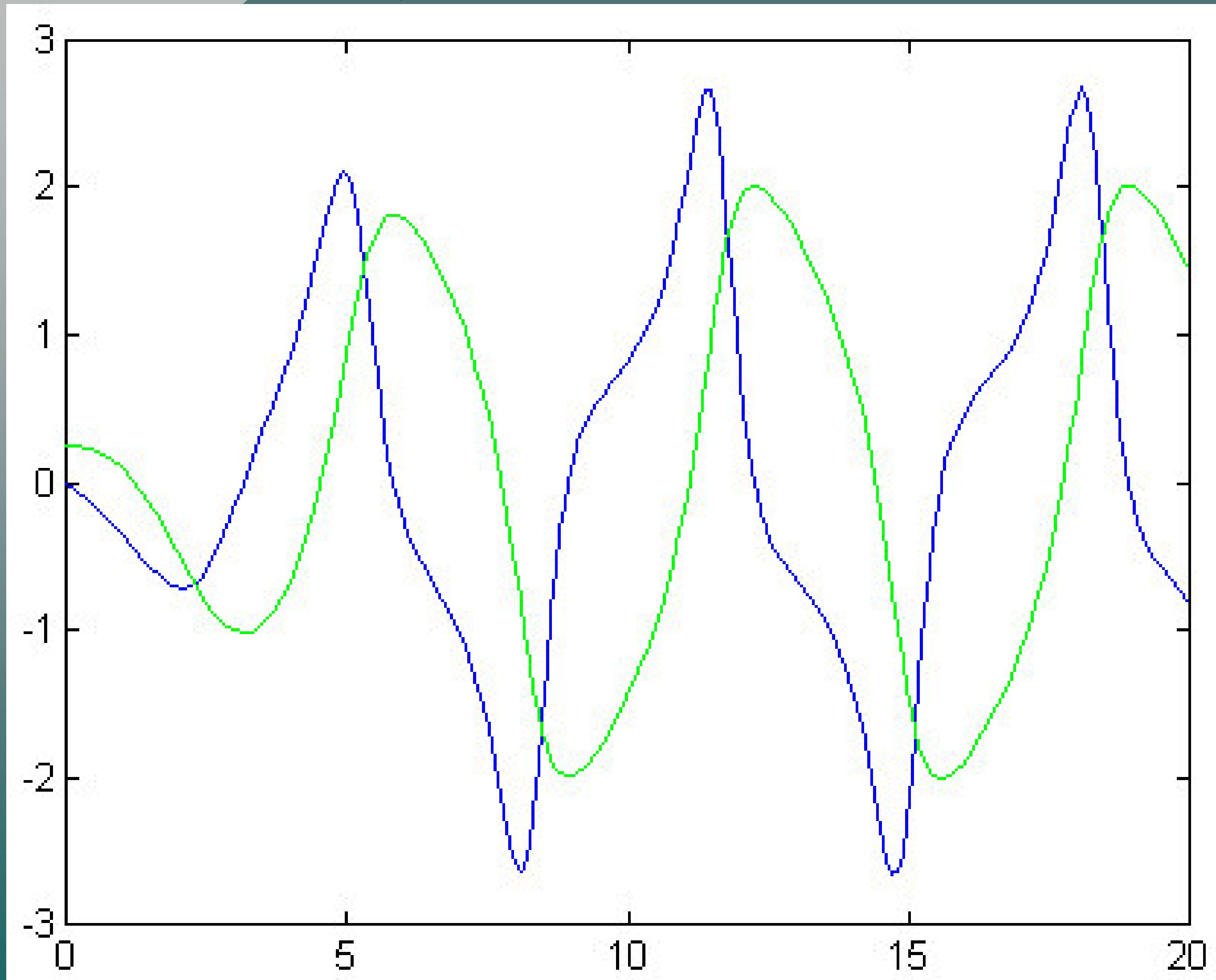
O primeiro passo para simular esse sistema é criar um arquivo ".m" contendo essas equações diferenciais. Por exemplo, o arquivo volpol.m:

```
function xdot=volpol(t,x)
 xdot=[0 0]
 xdot(l)=x(l).*(1- x(2).^2) - x(2);
 xdot(2)=x(l);
```

Para simular a equação diferencial no intervalo $0 \le t \le 20$, utiliza-se o comando ode23

```
>>t0 = 0; tf  = 20;
>>x0 = [0 0.25];
>>[t,x] = ode23('volpol', t0, tf, x0);
>>plot(t,x)
```

- Gráfico da solução de Van der Pol no intervalo $0 \leq t \leq 20$



Aplicações

- Sistema de Duffing :

O sistema de Duffing pode ser escrito:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -p_2 x_1 - x_1^3 - p_1 x_2 + q\cos(\omega\, t) \end{cases}$$

Fixando os parâmetros :

p_1= 0.4

p_2= -1.1

ω = 1.8

e variando q

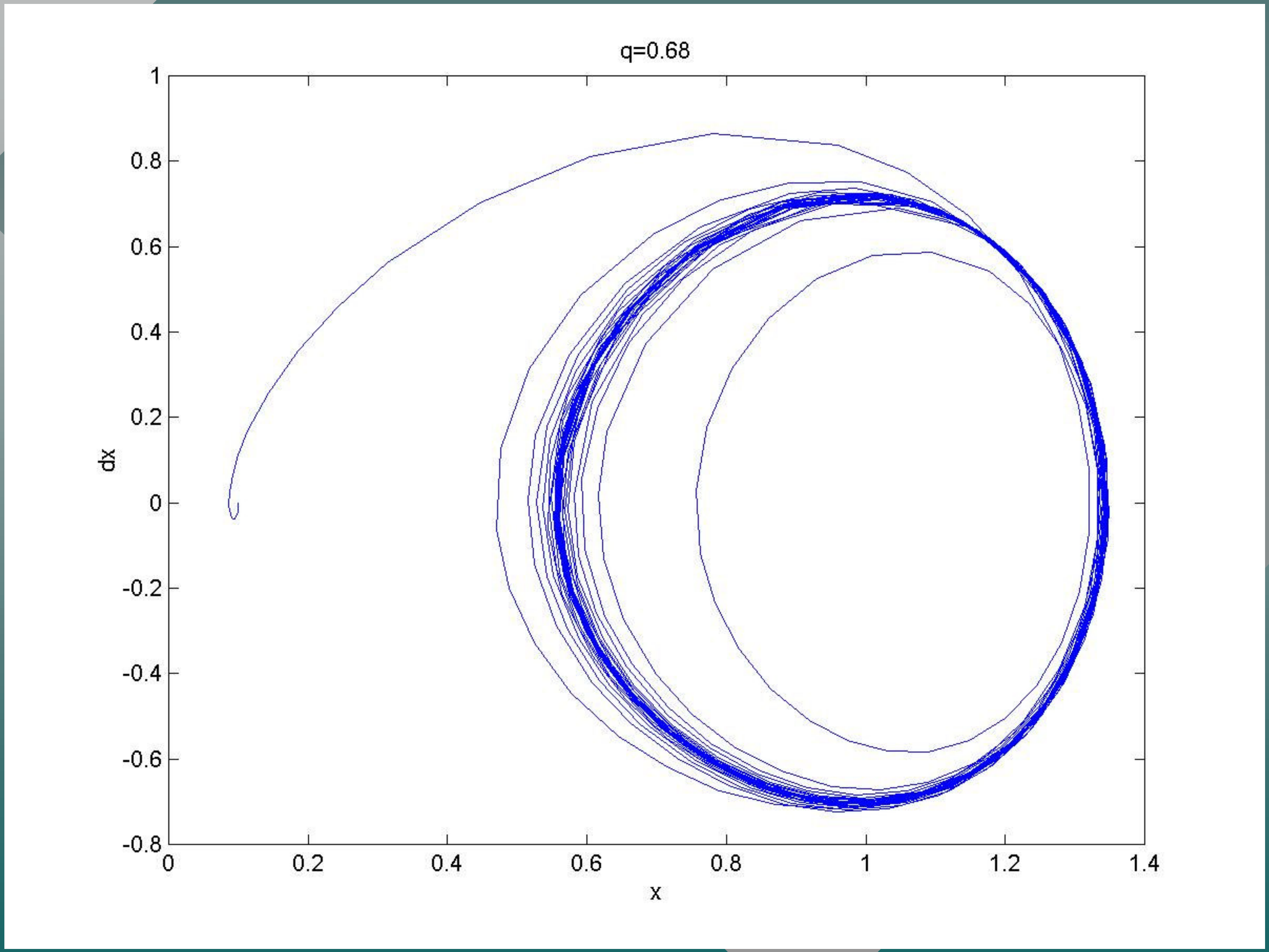
Equação de Duffing

```
function yprime=teste(t,y)
p=0.4;
p1=-1.1;
w=1.8;
q=0.68;
yprime=[y(2);(-p1*y(1))-(y(1)^3)-(p*y(2))+(q*cos(w*t))];
```

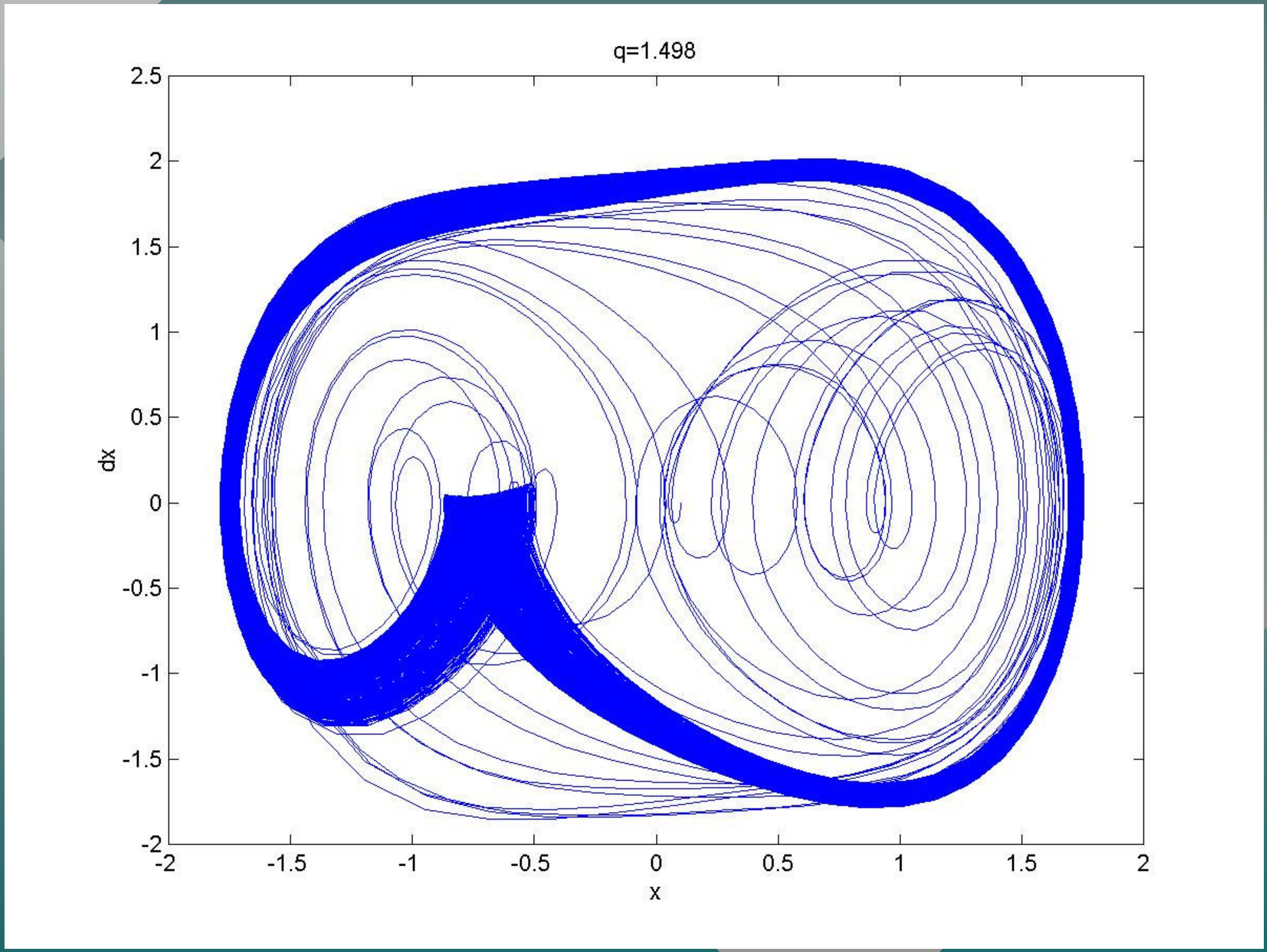
INTEGRADOR

```
[t,y]=ode45('teste',[100 5000],[0.1;0]);
y1=y(:,1);
y2=y(:,2);
plot(y1,y2,'b-')
xlabel('x');
ylabel('dx');
title('q=0.68');
```

Sendo q=0,68 :



Sendo q=1,498 :



Equação de Duffing

$$\dot{x} = y$$

$$\dot{y} = x - x^3 - \varepsilon y + \gamma \cos(\omega t)$$

com $\gamma = 0{,}3$; $\omega = 1$ e

ε é o parâmetro de controle escolhido.

Diagrama de Bifurcação - Equação de Duffing, $0.1 \le \varepsilon \le 0.5$

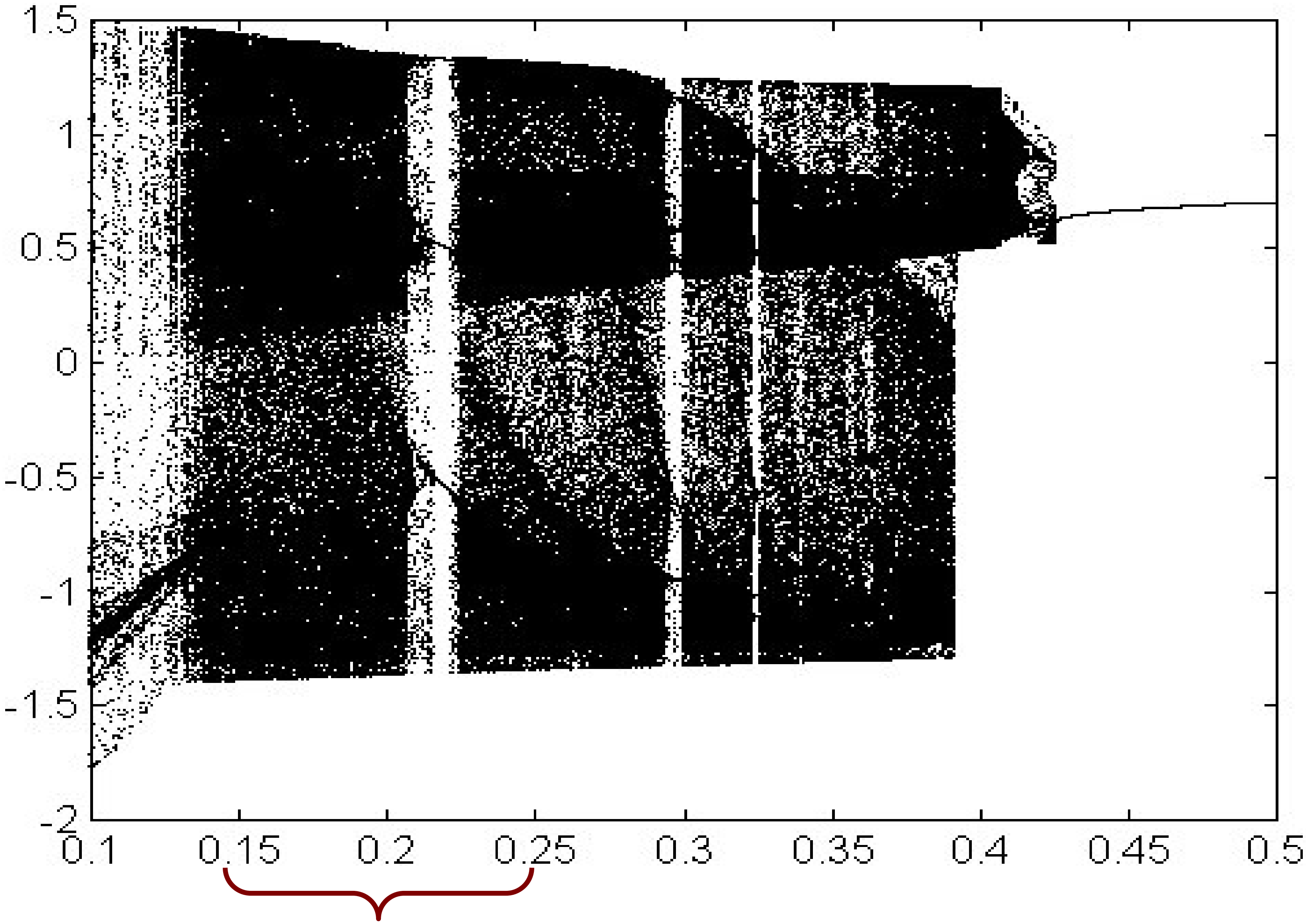


Diagrama de Bifurcação - Equação de Duffing, $0.1 \leq \varepsilon \leq 0.25$

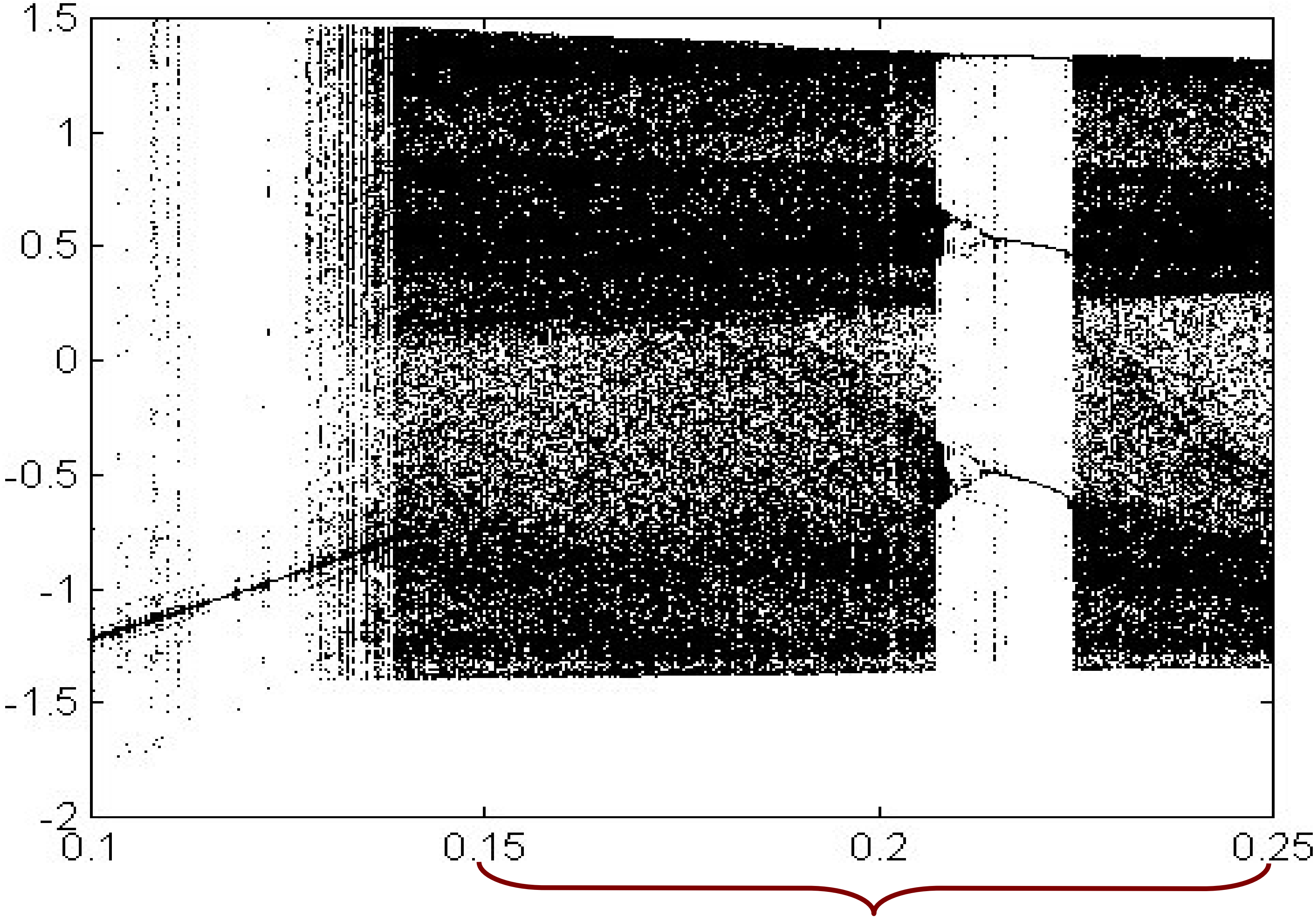


Diagrama de Bifurcação Tri-dimensional, $0.15 \leq \varepsilon \leq 0.25$