Curso de GNU Octave / Matlab e Aplicações para Engenheiros

Tiago Oliveira Weber

13 de março de 2018

Apresentação do Professor

Prof. Dr. Tiago Oliveira Weber

Formação

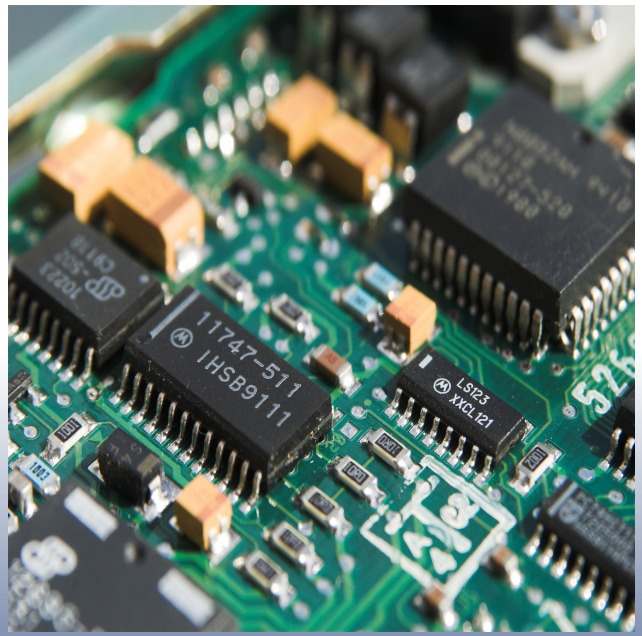
- Formado em Engenharia Elétrica na Universidade Federal de Santa Maria (UFSM)
- Doutor em Microeletrônica pela Universidade de São Paulo (USP)

Contato

- E-mail: tiago.weber@ufsc.br

Palavras Iniciais

- Fazer tarefas complexas sem necessitar muitas linhas de código;
- Fazer protótipos iniciais sem se preocupar com detalhes de implementação de baixo nível;



Conteúdo do Curso

Parte 1

- Conhecendo o GNU Octave e o Matlab
- Manipulando dados
- Condicionais e Laços
- Gerando Gráficos
- Utilizando Funções de Interação com Usuário
- Criando Funções
- Fazendo Depuração

Conteúdo do Curso

Parte 2

- Solucionando problemas matemáticos e de engenharia
- Interagindo com programas externos e códigos em C++

Programas que deve estar instalado:

- GNU Octave (preferencialmente) ou Matlab.

Conhecendo o GNU Octave e o Matlab

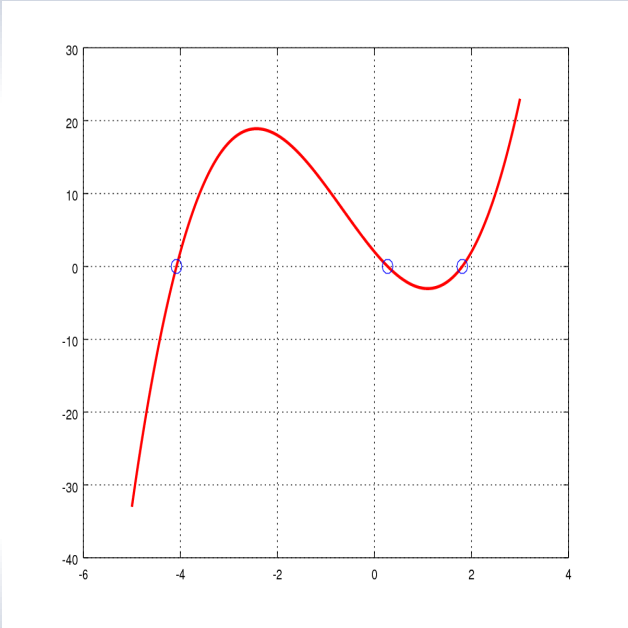
MATLAB (MAtrix LABoratory)

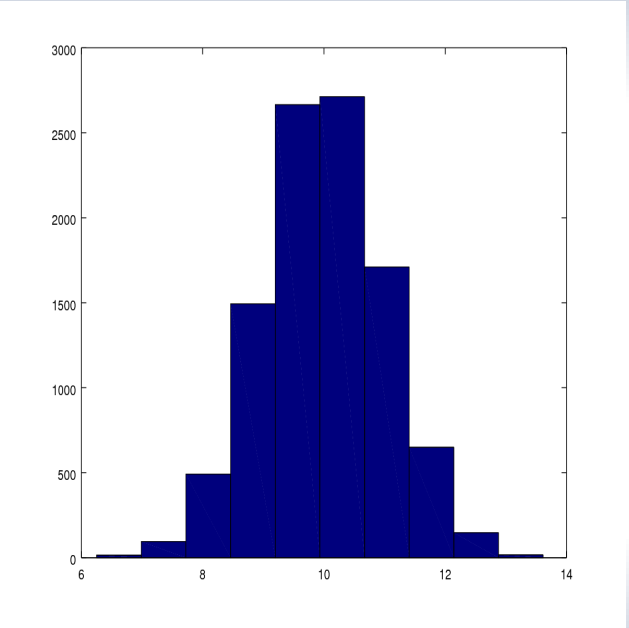
- É uma linguagem de programação de alto nivel com foco principal em cálculo com matrizes, processamento de sinais e elaboração de gráficos;
- Proprietário

GNU Octave

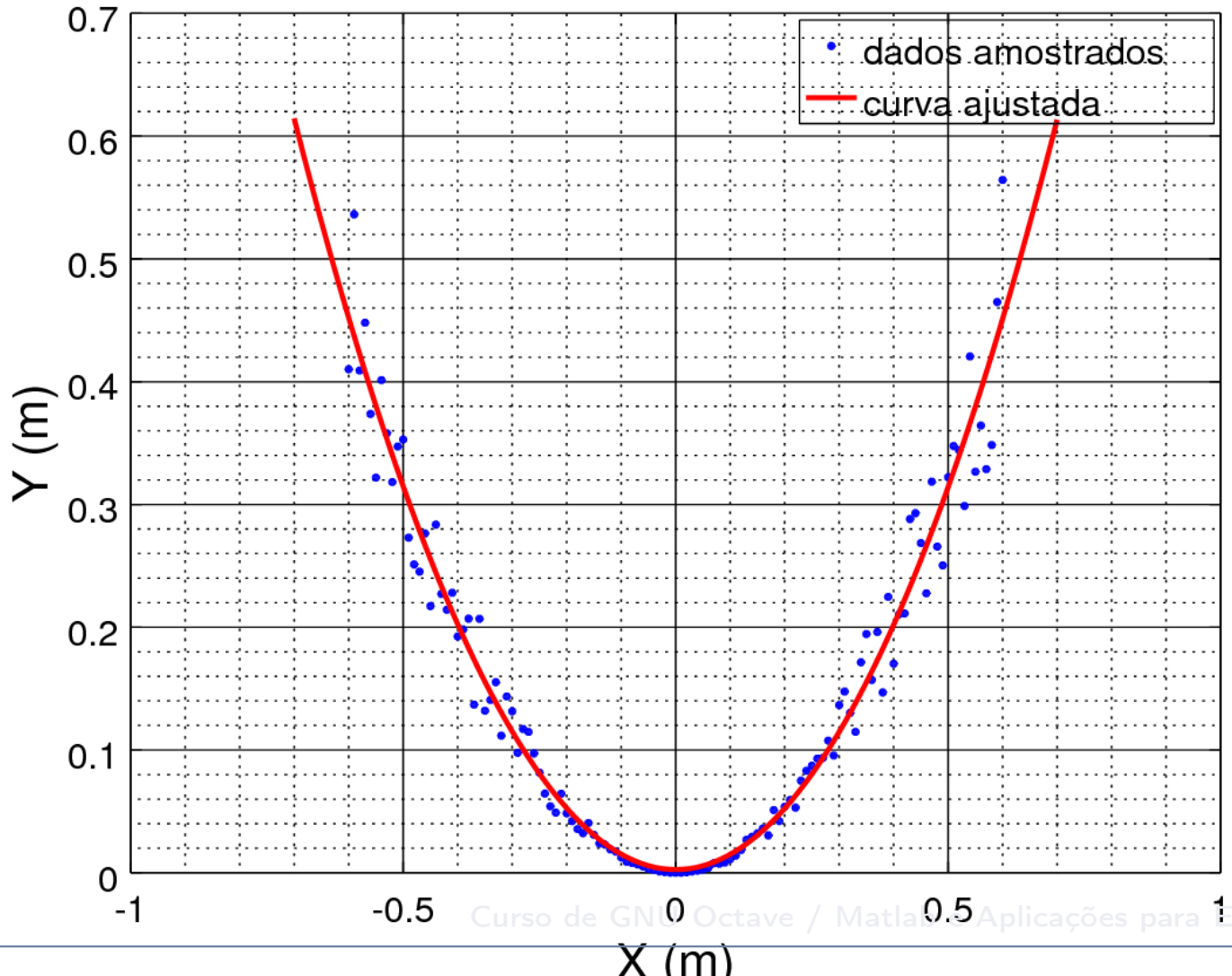
- É uma linguagem de programação altamente compatível com o MATLAB;
- É open-source disponível sob a licença GNU General Public License (GPL);

Algumas coisas que veremos

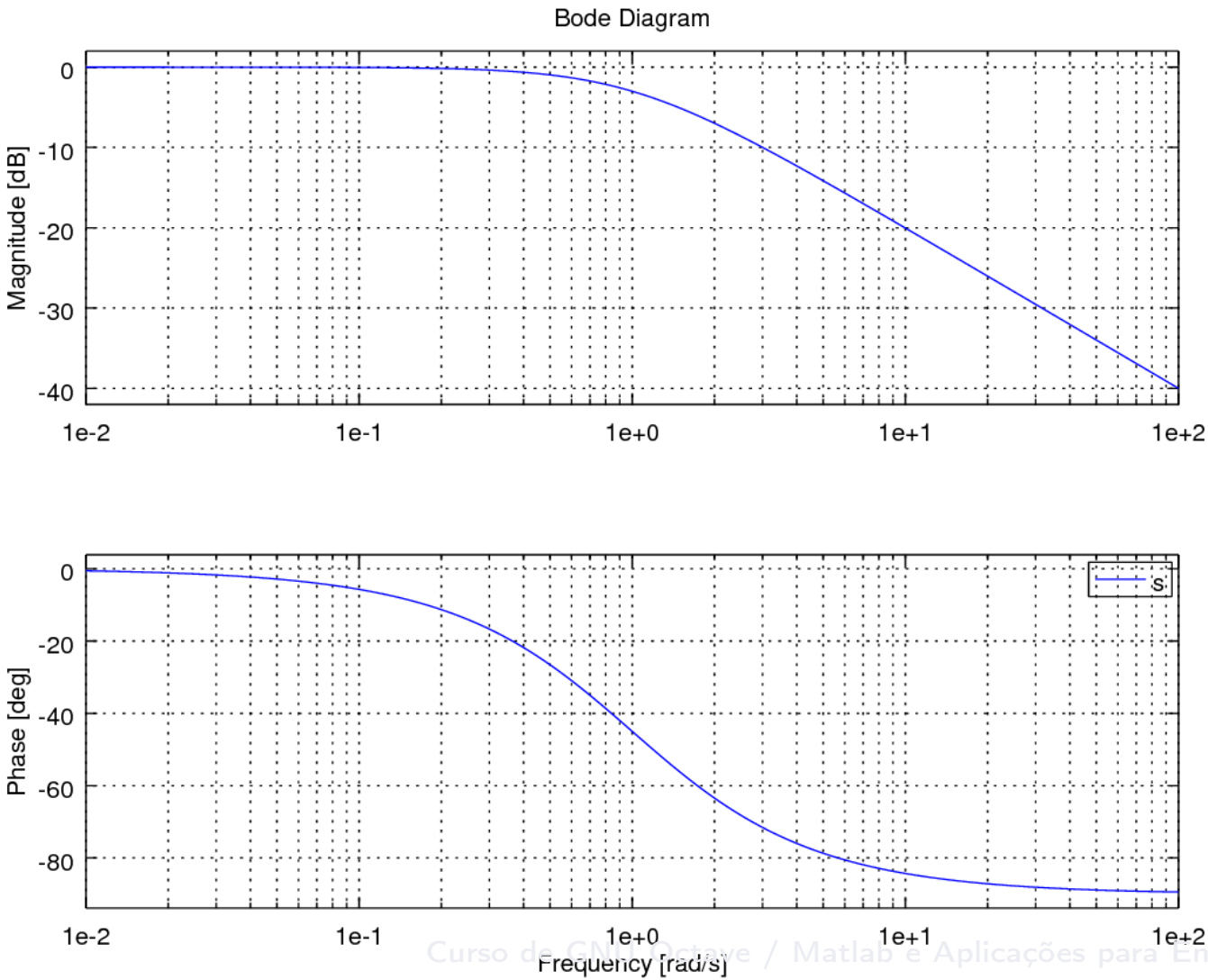




Algumas coisas que veremos



Algumas coisas que veremos



Diferenças entre GNU Octave e Matlab

- A maior parte das sintaxes que funcionam no Matlab, funcionam também no Octave; O contrário nem sempre é verdade. Ex.:
  - Octave suporta auto-incremento e operadores atributivos: i++; ++i; i+=1;...
  - Operador de negação no Octave pode ser '~' ou '!'. No MATLAB, apenas '~'.
- toolboxes do Matlab Vs pacotes do Octave;
- Simulink (Matlab)

A Interface

Abrindo o GNU Octave

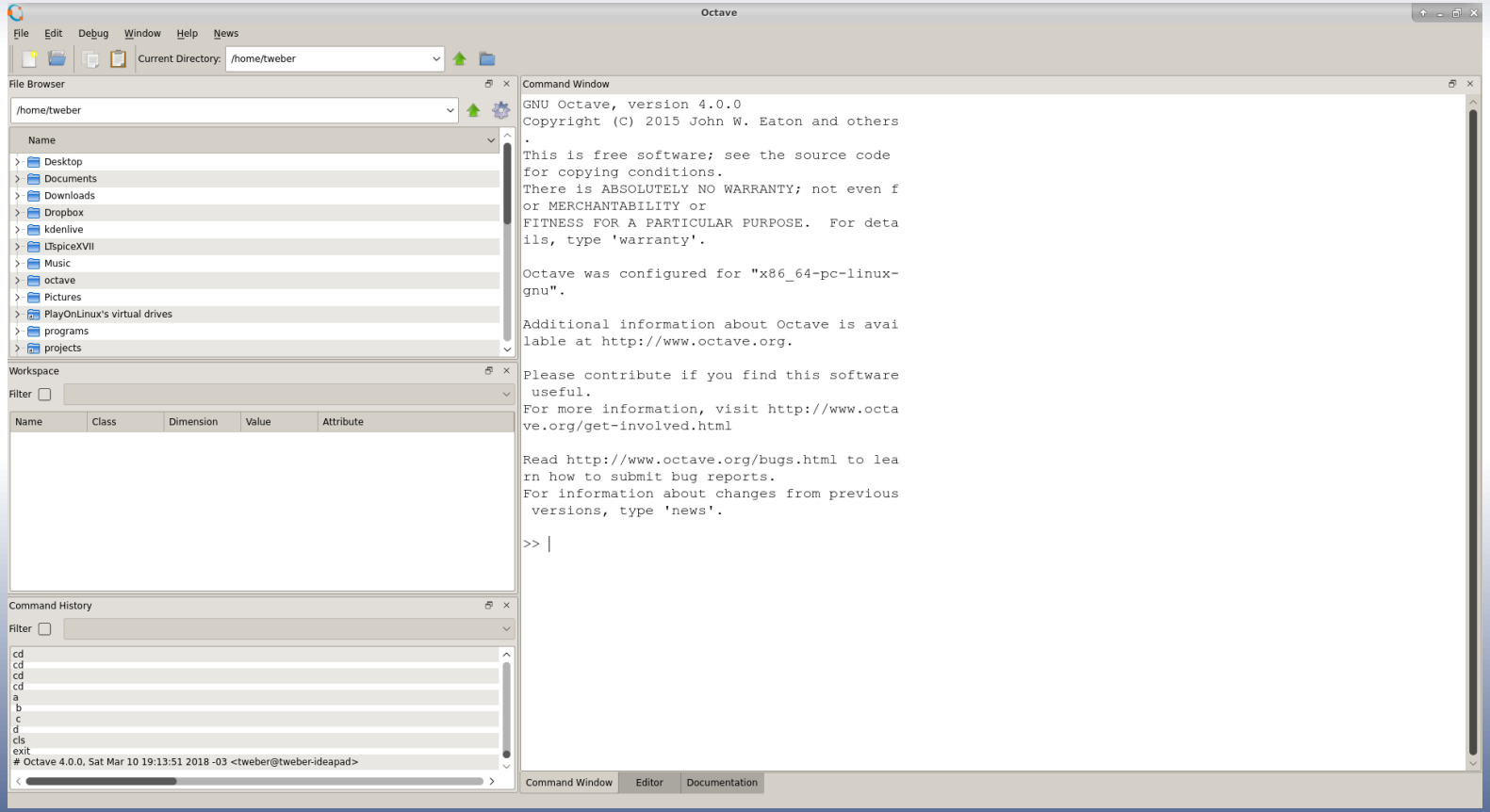
em modo gráfico	em linha de comando
`octave`	`octave --no-gui`

Abrindo o Matlab

em modo gráfico	em linha de comando
`matlab`	`matlab -nojvm`
	`matlab -nodesktop`

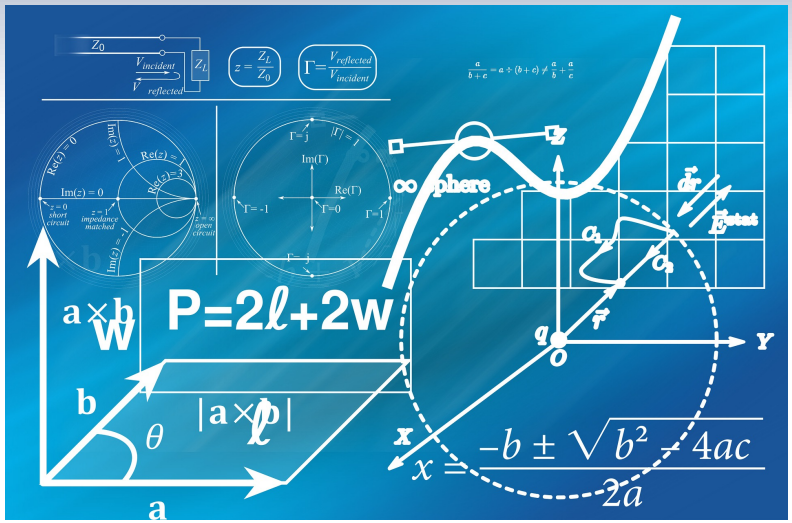
Utilizando a Interface

- A partir de agora, trataremos mais diretamente o GNU Octave. No entanto, no Matlab a interface é similar.



Utilizando a Interface

- utilizando o GNU Octave como calculadora



Utilizando a Interface

- utilizando o GNU Octave como linguagem de programação
- usando o editor
- criando e executando scripts / pastas onde o GNU Octave busca os scripts

O comando mais importante

help

Uso: help nome do comando

Manipulando dados

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Manipulando dados (operações com variáveis)

Código

```
a = 5;
b = 5;
ans = a+b;
```

Resposta

10

Manipulando dados (somando vetores ou matrizes)

Código

```
a = [2 3];
b = [5 3];
ans = a+b;
```

Resposta

```
7   6
```

Manipulando dados (multiplicando matrizes)

Número de linhas da matriz A deve ser igual ao número de colunas da matriz B.

Código

```
a = [2 3];
b = [3; 4];
ans = a*b;
```

Dica: a função size() retorna o tamanho do vetor ou matriz em linhas e colunas.

Resposta

18

Manipulando dados (multiplicação elemento a elemento)

As duas matrizes devem ter as mesmas dimensões.

Código

```
a = [2 3];
b = [3 4];
ans = a.*b;
```

Resposta

```
6   12
```

Manipulando dados (usando índices)

Acessando valor dentro de vetor ou matriz

Código

```
a = [2 3];
ans = a(1);
```

Resposta

2

Código

```
a = [2 3 ; 4 5];
ans = a(2,1);
```

Resposta

4

Manipulando dados (concatenando vetores)

Código

```
a = [2 3];
b = [4 5];
ans = [a, b]
```

Resposta

2 3 4 5

Manipulando dados (concatenando vetores)

Código

```
a = [2 3];
b = [4 5];
ans = [a; b]
```

Resposta

$$\begin{matrix} 2 & 3 \\ 4 & 5 \end{matrix}$$

Manipulando dados (concatenando vetores)

Exercício de Fixação

A partir das matrizes A = [1 2], B = [3 4], C = [5 6] e D = [7 8], crie a matriz [1 2 3 4; 5 6 7 8]

Manipulando dados (concatenando strings)

Concatenando strings

Código

```
nome1 = 'abra';
nome2 = 'cadabra';
ans = [nome1 nome2];
```

Resposta

abracadabra

Manipulando dados (criando vetores)

Vetor de zeros

Código

```
ans = zeros(1,5);
```

Resposta

0 0 0 0 0

Manipulando dados (criando vetores)

Vetor de um‘s

Código

```
ans = ones(1,5);
```

Resposta

```
1   1   1   1   1
```

Manipulando dados (criando vetores)

Criar vetor a partir de valor inicial, passo, valor final

Código

```
ans = 1:2:10
```

Resposta

```
1   3   5   7   9
```

Manipulando dados (filtrando vetor)

Código

```
a = 1:2:10
ans = a(a>4)
```

Resposta

5 7 9

Manipulando dados (filtrando vetor)

Exercício de fixação

Crie um vetor cujo primeiro valor é 5 e que vai com passo de 0.34 até 10; Filtre para mostrar apenas os valores maiores que 7 e menores que 8.

Dica:

Operador	Descrição
<	menor
>	maior
==	igualdade
&	E lógico
|	OU lógico

Manipulando dados (ruído)

Vetor de ruído com distribuição uniforme entre 0 e 1

Código

```
ans = rand(5,1);
```

Resposta

0.946623183011257 4
0.397195484558705 1
0.388166800377667 2
0.833164342896185 4
0.848749674321606 1

Manipulando dados (ruído gaussiano)

Vetor de ruído com distribuição gaussiana com média 10 e desvio padrão 1

Código

```
ans = 10+randn(5,1);
```

Resposta

8.68601143055238 7
9.90695137737933 4
10.2742171604448 3
9.83162439682920 6
12.0725122320448 9

Condicional (if)

Código

```
a = 5;

if (a>3)
   b = 'Iei!';
else
   b = 'Wow!';
end
ans = b;
```

Resposta

Iei!

Condicional (switch)

Código

```
yesno = "yes"

   switch yesno
     case {"yes" "y"}
       value = 1;
     case {"no" "n"}
       value = 0;
     otherwise
       value = -1;
   endswitch
   ans = value;
```

Resposta

1

Loop (for)

Código

```
for a=1:5
   x(a) = a.^2;
end
ans = x;
```

Resposta

1 4 9 16 25

Loop (for)

Código

```
a = [2 4 6 2 5]
for i=1:length(a)
   x(i) = a(i).^2;
end
ans = x;
```

Resposta

4 16 36 4 25

Loop (while)

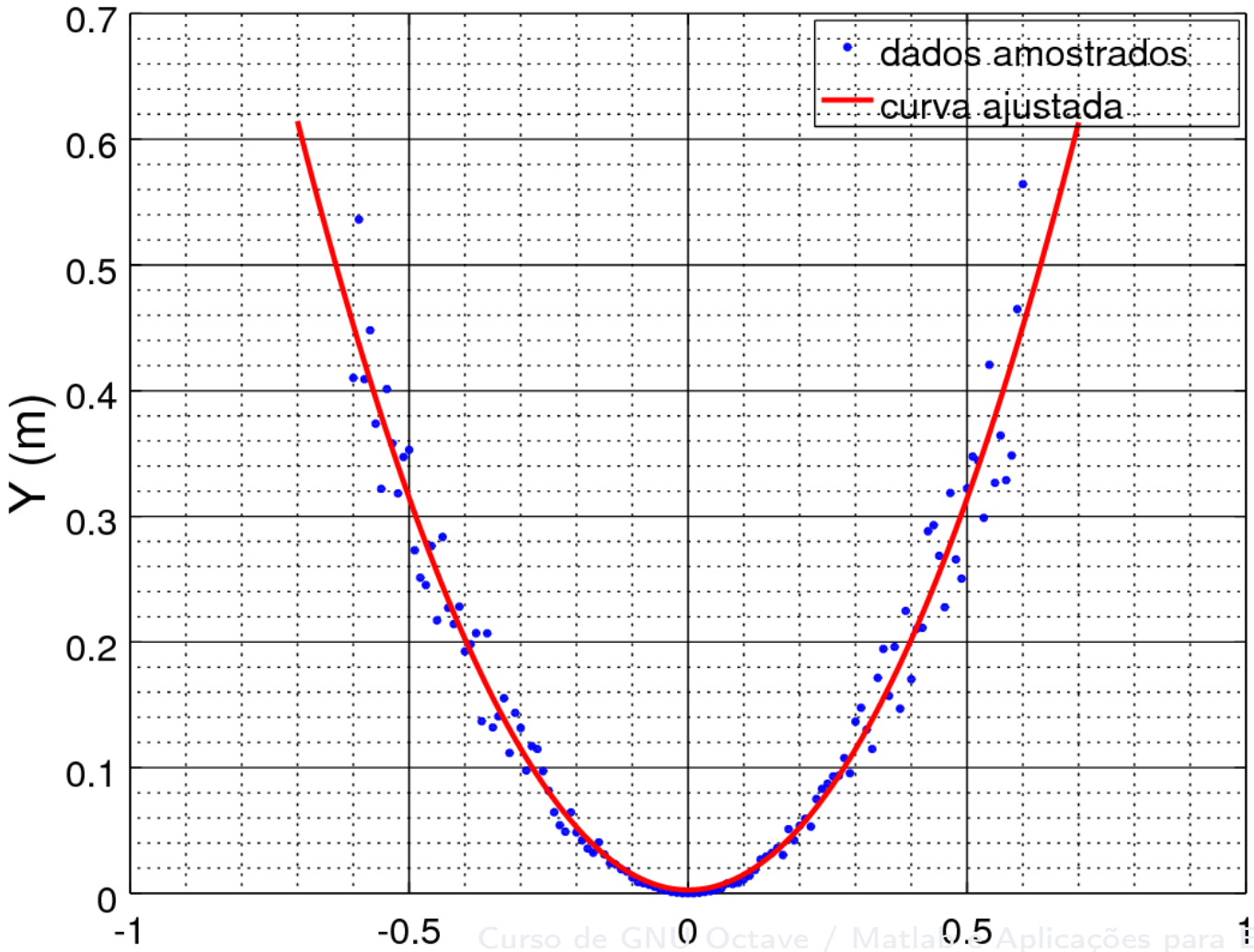
Código

```
i = 1;
while (i<6)
   x(i) = i.^2;
   i=i+1;
end
ans = x;
```

Resposta

1 4 9 16 25

Gerando Gráficos

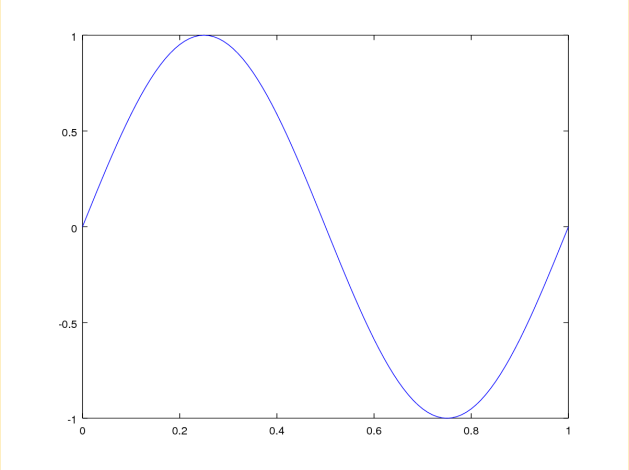


Gerando Gráficos

Código

```
t = 0:0.01:1;
y = sin(2*pi*t);
plot(t,y)
```

Resposta

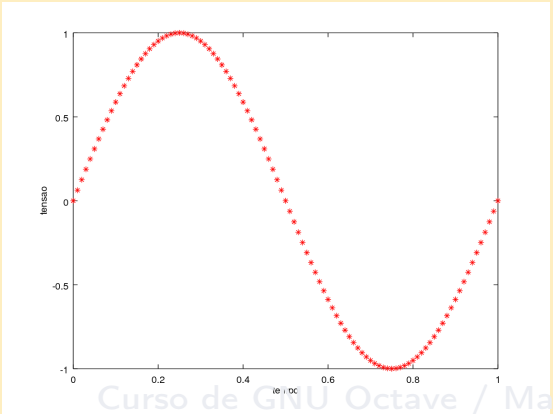


Gerando Gráficos

Código

```
t = 0:0.01:1;
y = sin(2*pi*t);
plot(t,y,'*r')
xlabel('tempo');
ylabel('tensao');
```

Resposta

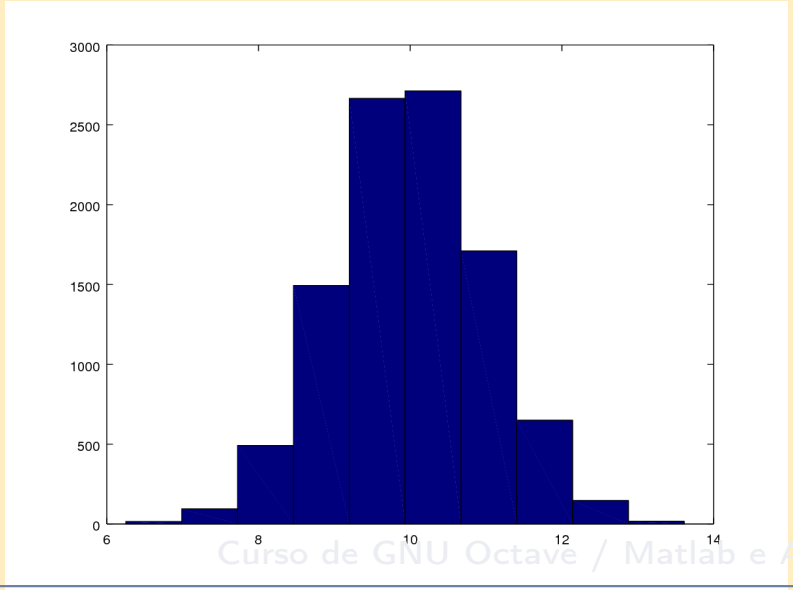


Gerando Gráficos (histograma)

Código

```
y = 10+randn(1,10000);
hist(y)
```

Resposta



Gerando Gráficos Aperfeiçoados (preâmbulo)

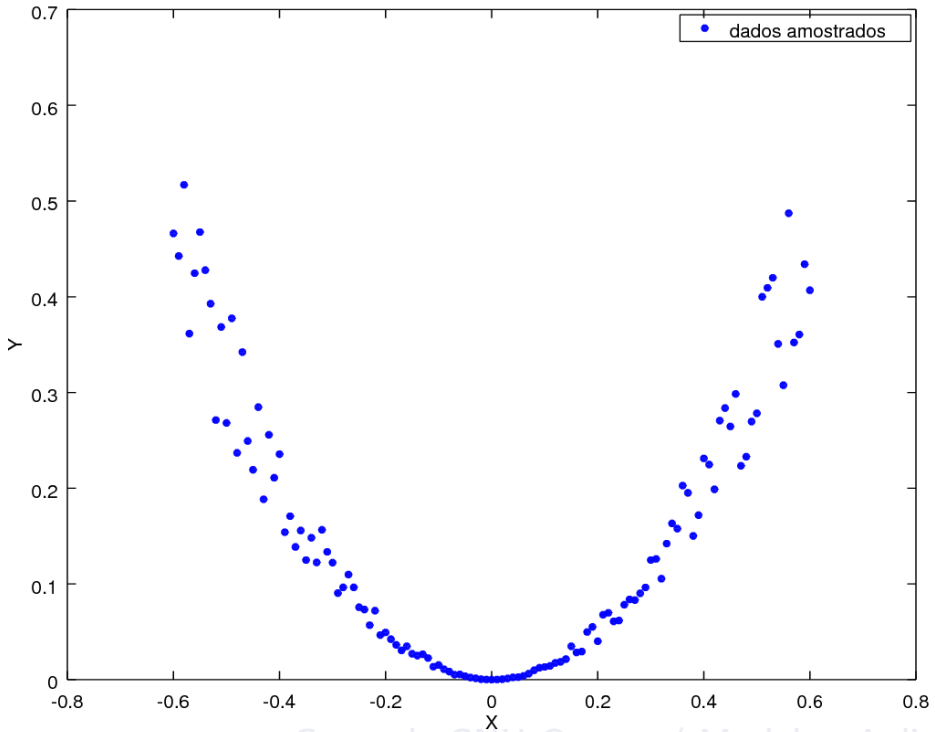
Código

```
x = -0.6:0.01:0.6
y = (x.^2).*(1+0.6*rand(1,length(x)));

plot(x,y,'.')
xlabel('X');
ylabel('Y');
legend('dados␣amostrados')
```

Gerando Gráficos Aperfeiçoados (preâmbulo)

Resposta



Gerando Gráficos Aperfeiçoados (preâmbulo 2)

Código

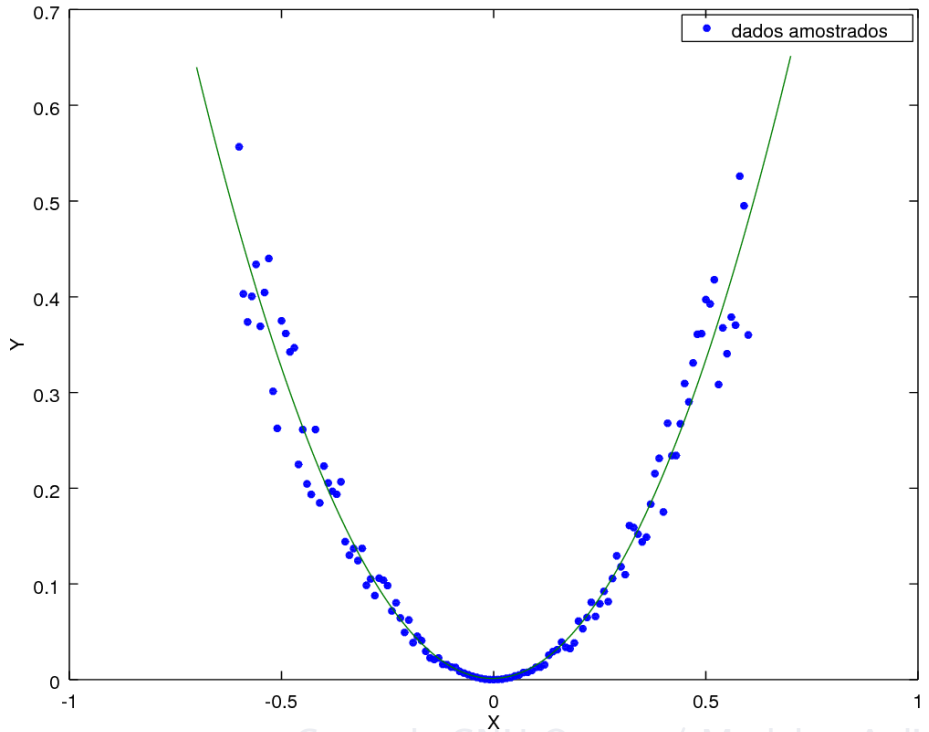
```
x = -0.6:0.01:0.6
y = (x.^2).*(1+0.6*rand(1,length(x)));

p = polyfit(x,y,2);    % faz o fitting da curva para
    polinômio de segunda ordem
xp = -0.7:0.001:0.7;

plot(x,y,'.',
     xp,polyval(p,xp));
xlabel('X');
ylabel('Y');
legend('dados␣amostrados')
```

Gerando Gráficos Aperfeiçoados (preâmbulo 2)

Resposta



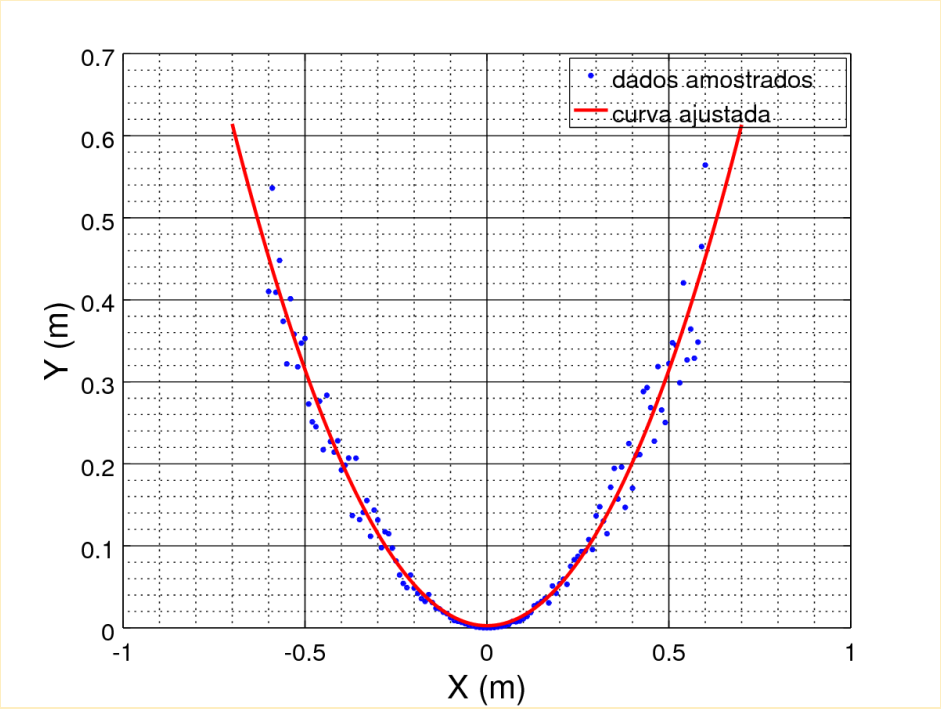
Gerando Gráficos Aperfeiçoados

Código

```
x = -0.6:0.01:0.6
y = (x.^2).*(1+0.6*rand(1,length(x)));
p = polyfit(x,y,2);
xp = -0.7:0.001:0.7;
plot(x,y,'.','markersize',5, 'color',[0, 0, 1],
     xp,polyval(p,xp),'linewidth',2,'color','red')
grid minor;
xlabel('X (m)','fontsize',20);
ylabel('Y (m)','fontsize',20);
lgd = legend('dados amostrados','curva ajustada','
    location','north','color','white');
set(lgd,"fontsize", 15);
set (gca, "fontsize", 15)
```

Gerando Gráficos Aperfeiçoados

Resposta



Criando Funções

Exemplo de Código de Função

```
function [retval1, retval2] = product_and_sum (input1,
    input2)

retval1 = input1.*input2;

retval2 = input1+input2;

end
```

Utilizando Funções de Interação com Usuário e com Arquivos (Imprimindo resultados na tela)

Código

```
a = 10;
fprintf(1, 'O projeto está funcionando há %d anos!',a);
```

Resultado

O projeto está funcionando há 10 anos

Imprimindo resultados em um arquivo

Código

```
fid=fopen('arquivo.txt');
a = 10;
fprintf(fid, 'O projeto está funcionando há %d anos!',a);
fclose(fid);
```

Solicitando informações do usuário

Código

```
distancia = input("Qual a distância em metros?");
continuar = yes_or_no("Deseja continuar?");
opcao = menu("título","opcao1","opcao2");
```

Fazendo Depuração

Comando	Descrição
`keyboard`	quando colocado no código, permite acesso ao terminal e pausa execução
`dbstep`	executa o comando da linha atual e vai para a próxima
`dbcont`	executa o código até encontrar um breakpoint
`dbstop`	sai do modo debug
`dbup`	sobe um nível na árvore enquanto no modo debug
`dbdown`	desce um nível na árvore enquanto no modo debug
`debug_on_error(1)`	entra no modo debug automaticamente quando encontra erro
`debug_on_warning(1)`	entra no modo debug automaticamente quando encontra warning

Exercício 1

- Utilize a função "randperm"como base para escolher aleatoriamente um índice de um vetor. Use o help para descobrir como usar o comando.

Exemplo:

Digamos que o vetor a = [1 2 4 5 10];
A cada iteração do seu comando/função, queremos que ele retorne aleatoriamente 1 dos itens contidos neste vetor a.

Exercício 2

- Escreva em um arquivo de texto: "Hello World. Today is "e preencha com a data de hoje através do comando "date".

Exemplo:

"Hello World. Today is 14-Mar-2018!"

Exercício 3

Crie os seguintes vetores:

A = [100,99,98, ..., 2, 1];

B = $[0, \frac{1}{99}, \frac{2}{99}, \ldots 1]$

- OBS.: será interessante utilizar os operador ":"na construção deles

Exercício 4

- Plote um gráfico de uma onda senoidal de 0 até 4 milisegundos, com frequência de 1 kHz e amplitude 5. Se a freqûencia estiver certa, você deverá ver 4 periódos completos da onda. Faça o eixo horizontal mostrar o tempo em milisegundos.

Exercício 5

Faça um script que:

- pergunte ao usuário:
  - Frequência;
  - Amplitude;
  - Offset;
- plote o gráfico da onda resultante de forma que o eixo horizontal mostre o tempo (e não o número de iterações) para 4 períodos da onda.

Solucionando problemas matemáticos e de engenharia

Parte 2

Encontrando zeros de polinômios

$$p(x) = x^3 + 2x^2 + x - 10$$

Código

```
c = [1, 2, -8, 2];
ans = roots (c)
```

Resultado

-4.080604400570926
1.809785920127422
0.270818480443506 9

Encontrando zeros de polinômios

$$p(x) = x^3 + 2x^2 + x - 10$$

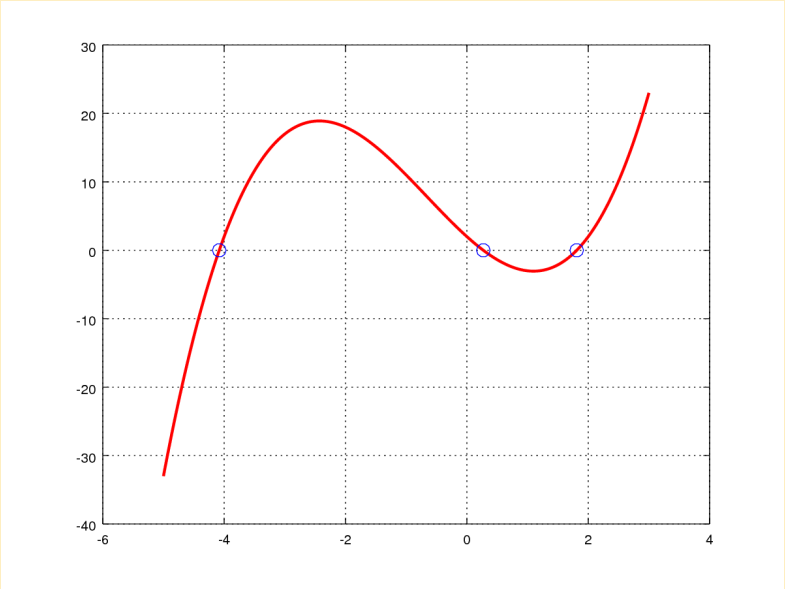
Código

```
c = [1, 2, -8, 2];
roots_c = roots(c);

x=-5:0.01:3;
plot(x,polyval(c,x),'r','linewidth',2,
     roots_c,zeros(1,3),'ob','markersize',10)
grid on;
```

Encontrando zeros de polinômios

Resultado



Resolvendo Sistemas de Equações Lineares

$$\begin{cases} 2x + y + z = 180 \\ x + 3y + 2z = 310 \\ 2x + y + 4z = 240 \end{cases}$$

- Colocar sistema de equações na forma $A \cdot x = b$
- Fazer a operação: $x = A \setminus b$

Resolvendo Sistemas de Equações Lineares

$$\begin{cases} 2x + y + z = 180 \\ x + 3y + 2z = 310 \\ 2x + y + 4z = 240 \end{cases}$$

Código

```
# Colocando na forma A*x = b,
A = [ 2 1 1;
      1 3 2;
      2 1 4]

b = [180; 310; 240];
ans = A \ b;        # equivalente a x = inv(A)*b
```

Resultado

```
42    % x
76    % y
20    % z
```

Resolvendo Sistemas de Equações Lineares (exemplo 2)

- a técnica A produz 9 peças por dia;

$$y = 9 \cdot x;$$

- a técnica B produz 12 peças por dia, mas demora 5 dias para iniciar operações;

$$y = 12 \cdot (x - 5)$$

Quanto tempo até a técnica B produzir mais peças que a 9?

Resolvendo Sistemas de Equações Lineares (exemplo 2)

Técnica	Peças/dia	Atraso (dias)	Equação	Equação rearranjado
A	9	0	$y = 9 \cdot x$	$-9 \cdot x + y = 0$
B	12	5	$y = 12 \cdot (x - 5)$	$-12 + y = -60$

Código

```
# Colocando na forma A*x = b,
A = [ -9, 1;
      -12, 1];

b = [0; -60];
ans = A \ b;          # equivalente a x = inv(A)*b
```

Resultado

```
  20    -> dias
 180    -> peças
```

Resolvendo Equações Não-lineares

- resolve equações na forma F(x) = 0

Uso: `fsolve (fcn, x0)`
Onde: `fcn` é uma função que deve aceitar um vetor com as variáveis de entrada e retornar um vetor com os resultados de cada equação `x0` é o ponto inicial a ser testado (semente)
Exemplo:

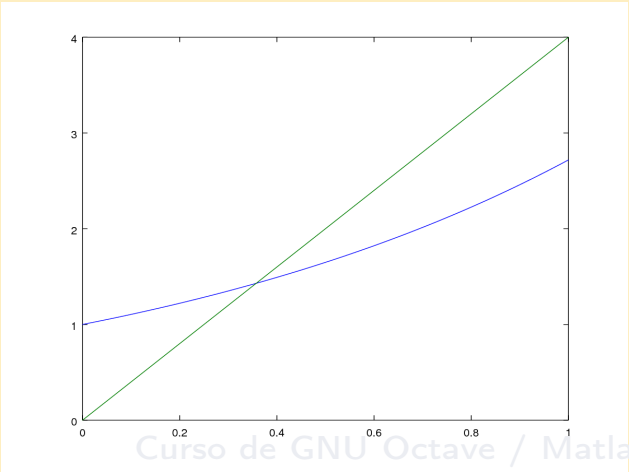
$$\begin{cases} y = e^x \\ y = 4x \end{cases}$$

Resolvendo Equações Não-lineares

Visualizando o problema

```
x=0:0.01:1
plot(x,e.^x,
     x,4*x)
grid minor;
```

Resultado



Resolvendo Equações Não-lineares

Forma 1: considere uma função única de resposta e minimize ela.

Código (função não-linear)

```
function y = fn(x)
  a1 = e.^x;
  a2 = 4*x;
  y = abs(a1-a2);
end
```

Código (Resolvendo)

```
ans = fsolve(@fn,[0]);
```

Resultado

0.3574029558465378

Resolvendo Equações Não-lineares

Outra forma: considere as funções separadas, igualando elas a zero.

Código (função não-linear)

```
function f_to_minimize = fn2(x)
  f_to_minimize(1) = e.^x(1) - x(2); % x(2) is our y
  f_to_minimize(2) = 4*x(1) - x(2);
end
```

Código (Resolvendo)

```
ans = fsolve(@fn2,[0,0]);
```

Resultado

0.357402929259072 1.42961171703629

Resolvendo Equações Não-lineares (resistor e diodo)

Função não-linear baseada no circuito

```
function [y] = res_diode(x)
Vd=x(1); %x(1) é a tensão no diodo
Vsupply=5;
Id = 1.3e-15*(e^(Vd/0.026)-1); % corrente no diodo
Ir = (Vsupply-Vd)/1e3; %corrente no resistor (lei de ohm)

y = Id-Ir;
end
```

Resolvendo Equações Não-lineares (resistor e diodo)

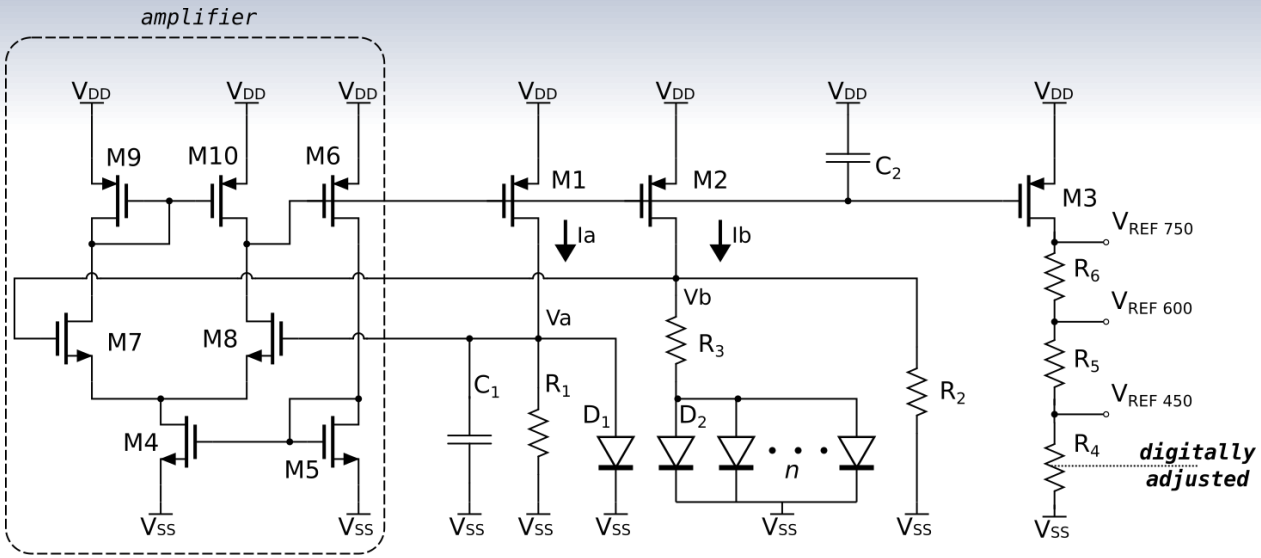
Código

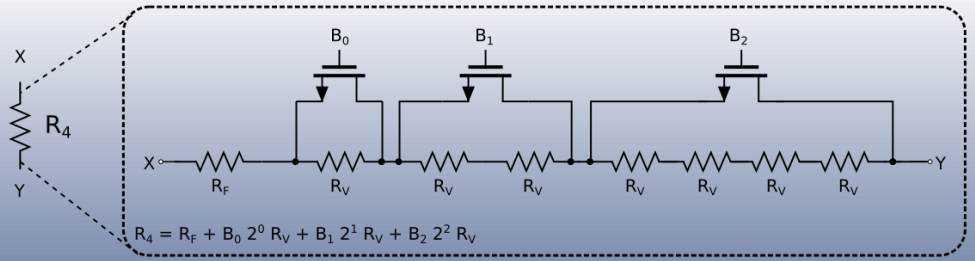
```
ans = fsolve(@res_diode,[0]);
```

Resultado

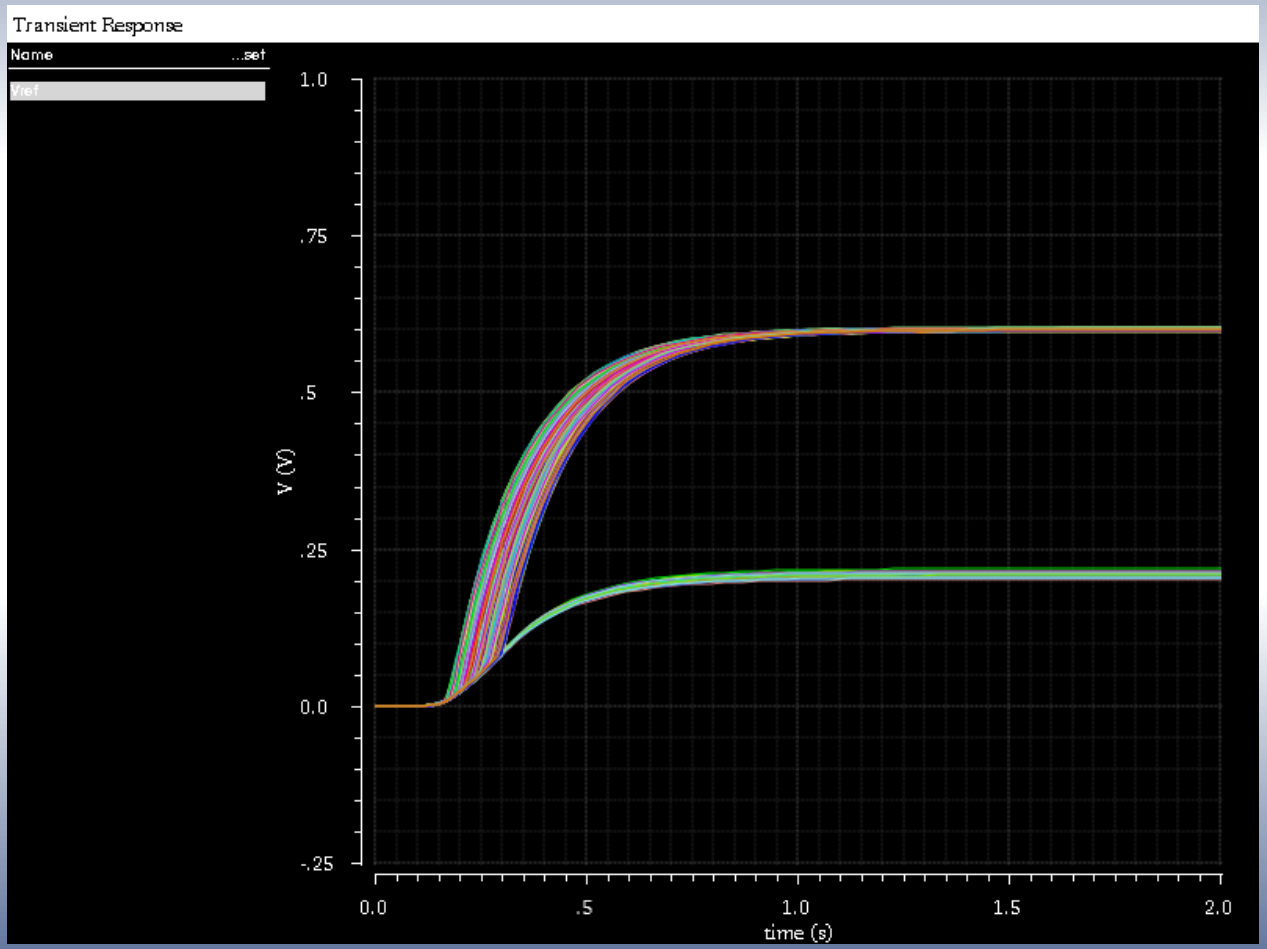
0.7492099696750156

Resolvendo Equações Não-lineares (Topologia Bandgap)





Resolvendo Equações Não-lineares (Análise Bandgap)



Algoritmos Genéticos

- Pacote GA (Genetic Algorithm)
  `https://octave.sourceforge.io/ga/function/ga.html`
- Uso: `[x,fval] = ga (fitnessfcn, nvars)`
- Uso: `[x,fval] = ga (fitnessfcn, nvars, A, b, Aeq, beq, LB, UB, nonlcon, options)`
- Onde:
  - fitnessfcn: função objetivo a minimizar
    - deve receber vetor com dimensão 1 por nvars;
    - deve retornar um valor escalar;
  - nvars: número de variáveis do problema
  - options: estrutura com os parâmetros da otimização (ver gaoptimset)

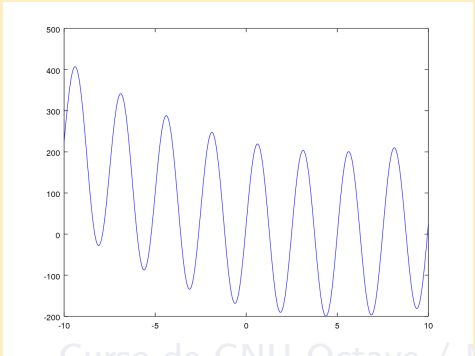
Algoritmos Genéticos

Função Custo Simples

```
function [y] = gafcn(x)
y = (x-5).^2+200*sin(0.8*pi*x);
end
```

Plotando

```
x = -10:0.01:10;
plot(x,gafcn(x))
```



Algoritmos Genéticos

Resolvendo com GA (sem mexer nas opções)

```
pkg load ga
ans = ga(@gafcn,1)
```

Resultados

1.879939685790021

Algoritmos Genéticos

Resolvendo com GA

```
pkg load ga
options =   gaoptimset('EliteCount',5,'Generations',10,'
    PopulationSize',200,'CrossoverFraction',0.1); %
    usualmente não utilizar CrossoverFraction tão baixo
LB = -10;
UB = 10;
ans = ga(@gafcn,1,[] ,[] ,[] ,[] , LB, UB, [], options)
```

Resultados

4.375973901291995

Simulated Annealing

Resolvendo com SA

```
pkg load optim
LB = -10;
UB = 10;
nt = 100; % número de reduções de temperatura
ns = 20; % iterações entre reduções
rt = 0.9; % fator de redução de temperatura
maxevals = 1000;
neps = 5;
functol = 1e-10;
paramtol = 1e-3;
verbosity = 1;
minarg = 1;
control = {LB,UB,nt,ns,rt,maxevals,neps,functol,paramtol,
    verbosity,minarg}

ans = samin("gafcn",{1},control)
```

Simulated Annealing

Resultado

4.354602297872638

Controle

- Pacote "control"
- Criar função de transferência:

```
s = tf('função de transferência')
```

Código

```
pkg load control

s = tf ('s');
G = 1/(s+1);  %modelo contínuo no tempo
```

Controle

Código

```
pkg load control
num = [1];
den = [1 1];
s = tf(num,den);
```

Controle - Bode

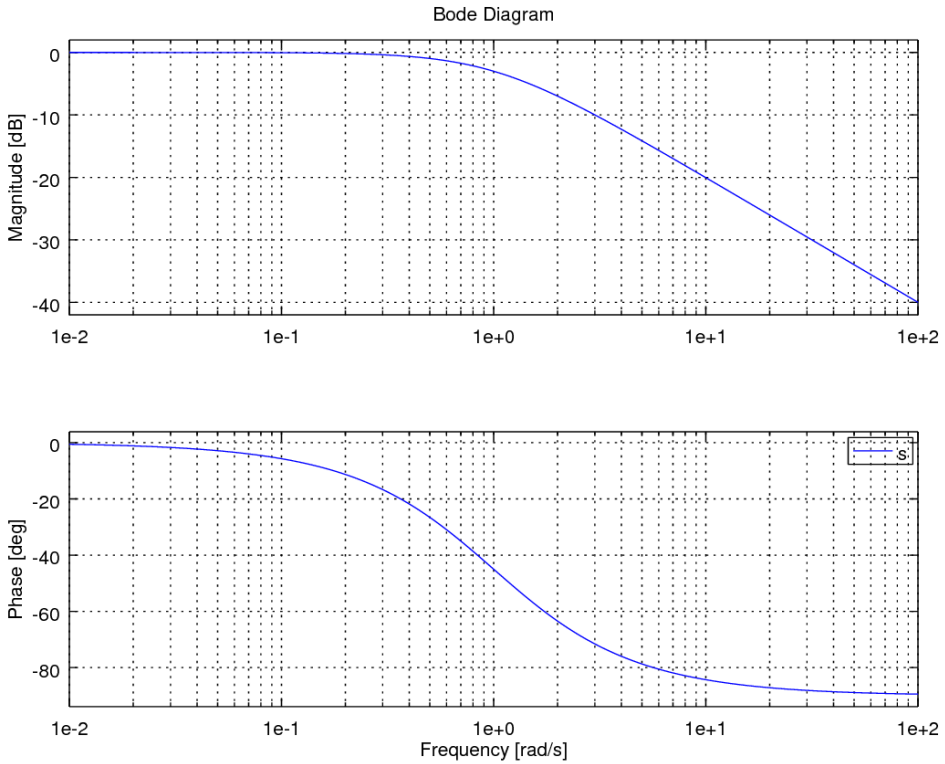
- Uso:
  - mostrar figura: `bode (SYS)`
  - retornar vetores: - `[mag, pha, W] = bode(SYS)`

Exemplo

```
pkg load control
num = [1];
den = [1 1];
s = tf(num,den);
bode(s)
```

Controle - Bode

Resultado

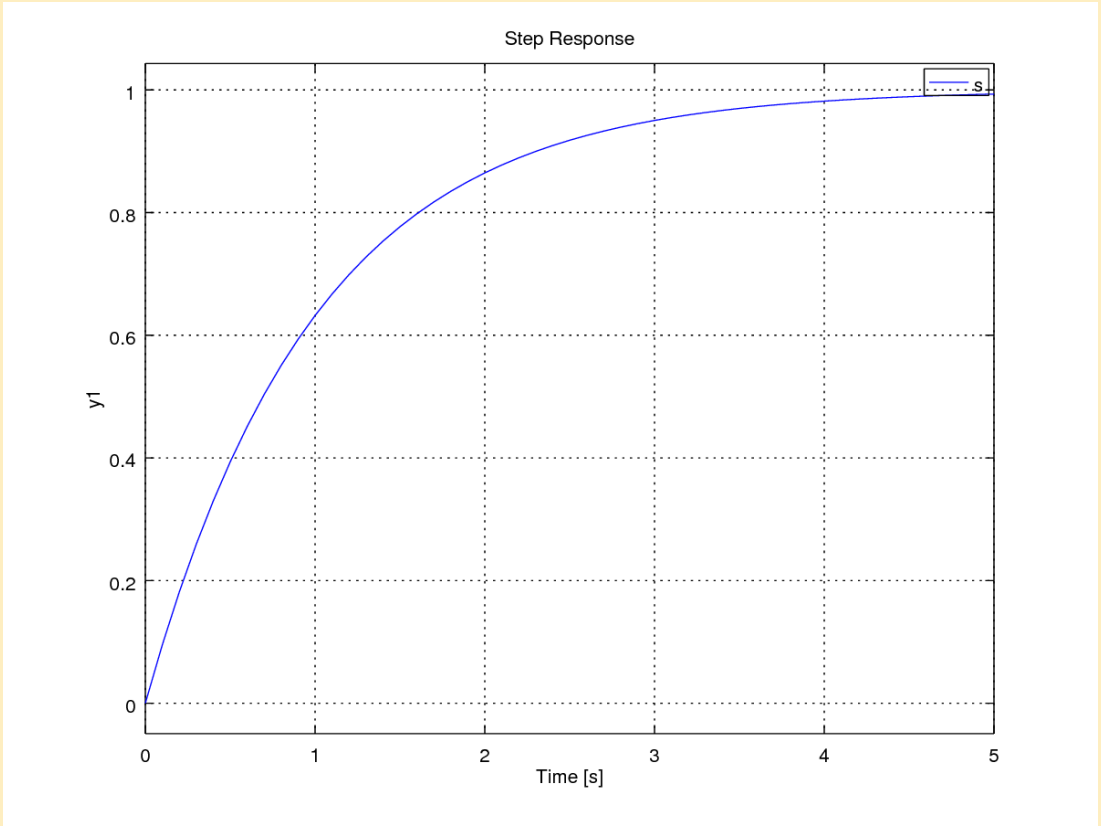


Controle - Resposta ao Degrau

```
pkg load control
num = [1];
den = [1 1];
s = tf(num,den);
step(s)
```

Controle - Resposta ao Degrau

Resultado



Controle - Resposta ao Impulso

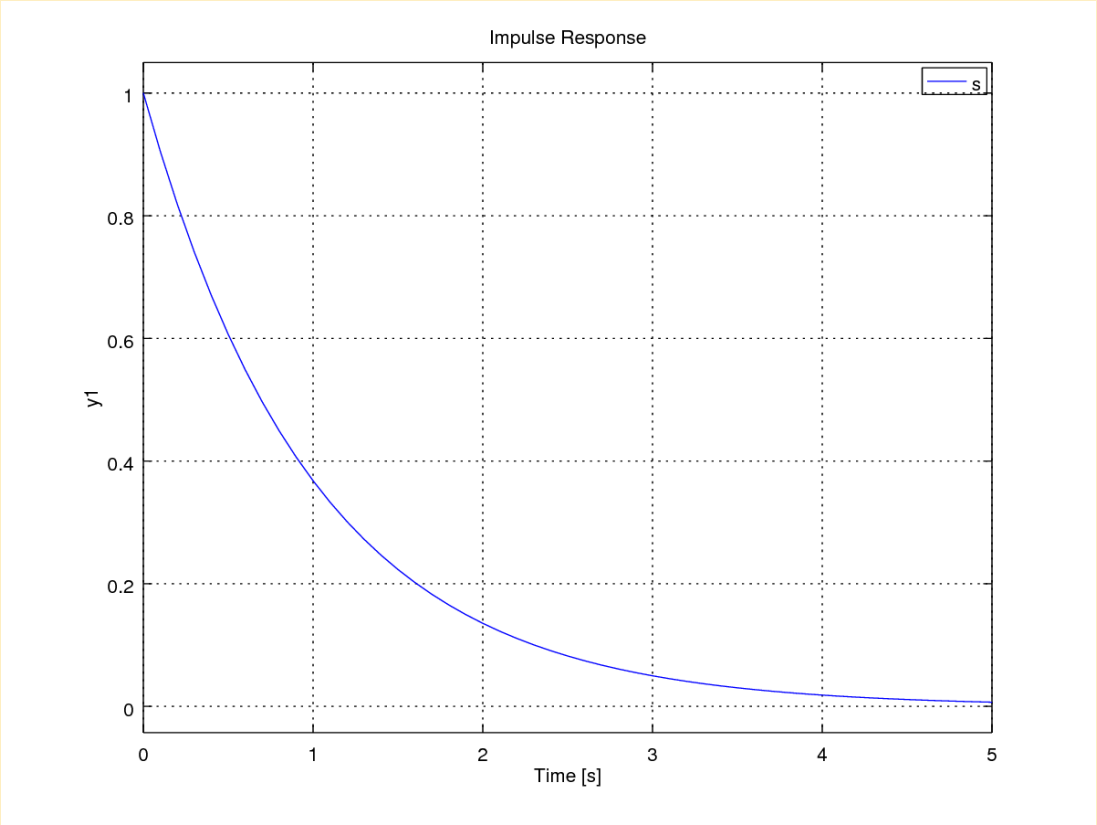
Código

```
pkg load control
num = [1];
den = [1 1];
s = tf(num,den);
impulse(s)
```

org$_{\text{babeleoe}}$

Controle - Resposta ao Impulso

Resultado



system()

- Utilizando o comando:

  `system('nome do comando')`

é possível executar comandos de sistema externos ao Octave.

Interagindo com programas em C++

- baseado em `https://octave.org/doc/v4.0.1/Getting-Started-with-Oct_002dFiles.html`
- inclua a biblioteca «octave/oct.h>"no programa em C++

Código em C++

```
#include <octave/oct.h>

DEFUN_DLD (helloworld, args, nargout, "Hello World Help
    String")
{
  int nargin = args.length ();
  octave_stdout << "Hello World has " << nargin << "
      input arguments and " << nargout << " output
      arguments.\n";

  return octave_value_list ();
}
```

Interagindo com programas em C++

- Compile o código utilizando "mkoctfile";

Compilação

```
mkoctfile helloworld.cc
```

Interagindo com programas em C++

Como executar de dentro do Octave

```
a = helloworld(1,2,3);
```