Nesta lista, pede-se que sejam estudados alguns comandos do Scilab, os códigos utilizados para tanto encontram-se ao final do documento.

Fazendo todos os comandos pedidos na introdução da lista, obtêm-se o seguinte navegador de variáveis e os gráficos abaixo dele :

Navegador de variáveis

Nome	Value	Tipo	Visibilidade
A	[0, 1; -2, -3]	Real	local
B	[-1.5, -0.5; 1, 0]	Real	local
C	3x3	Real	local
D	3x3	Real	local
a	[5, 2, 1]	Real	local
ans	3	Real	local
b	[0, 0, 12]	Real	local
d	[-1 - 0i, 0 - 0i; 0 ...	Real	local
f	N/A	r (Lista T)	local
p	[-1 - 0i; 0 - 0i]	Real	local
p1	1x1	Polinômio	local
p2	1x1	Polinômio	local
r	2	Real	local
t	15	Real	local
v	[0.707 - 0i, -0.4...	Real	local
x	1x17	Real	local
y	17x1	Real	local

Tabela 1 - Navegador de variáveis da introdução

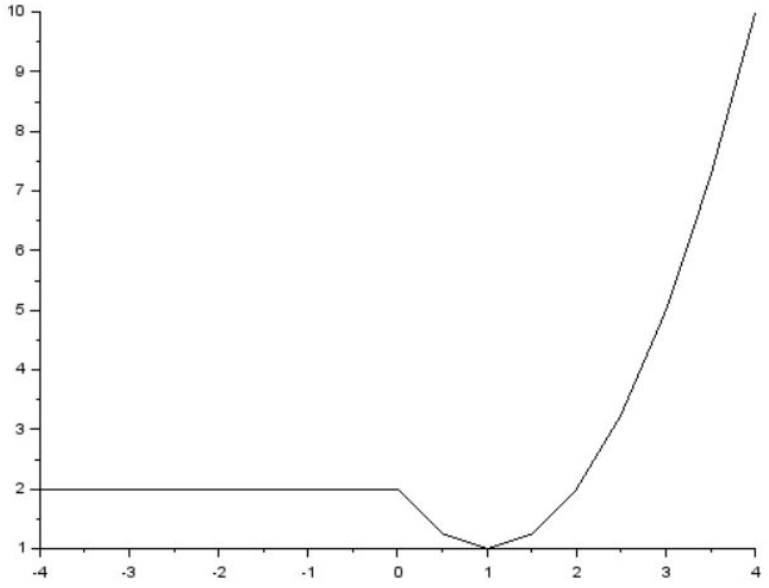

Imagem 1 - Janela gráfica 0 da introdução

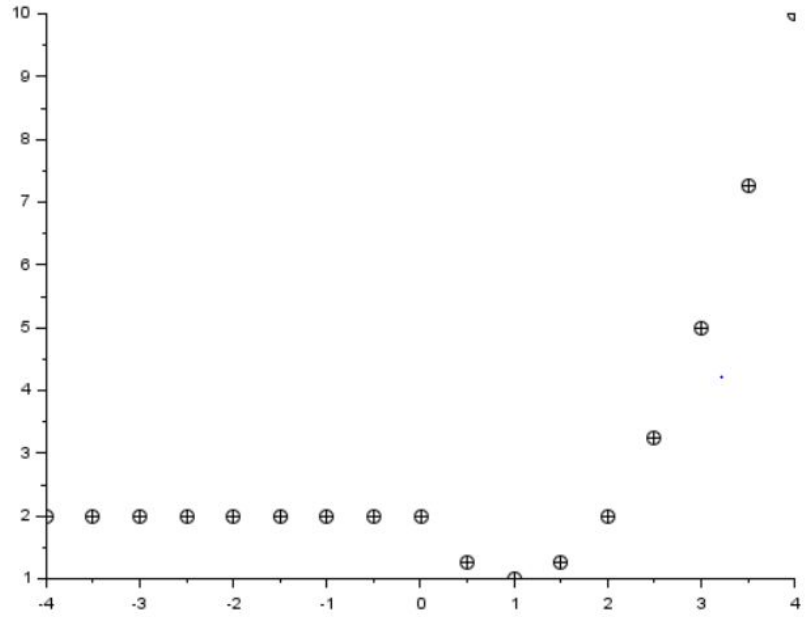

Imagem 2 - Janela gráfica 1 da introdução

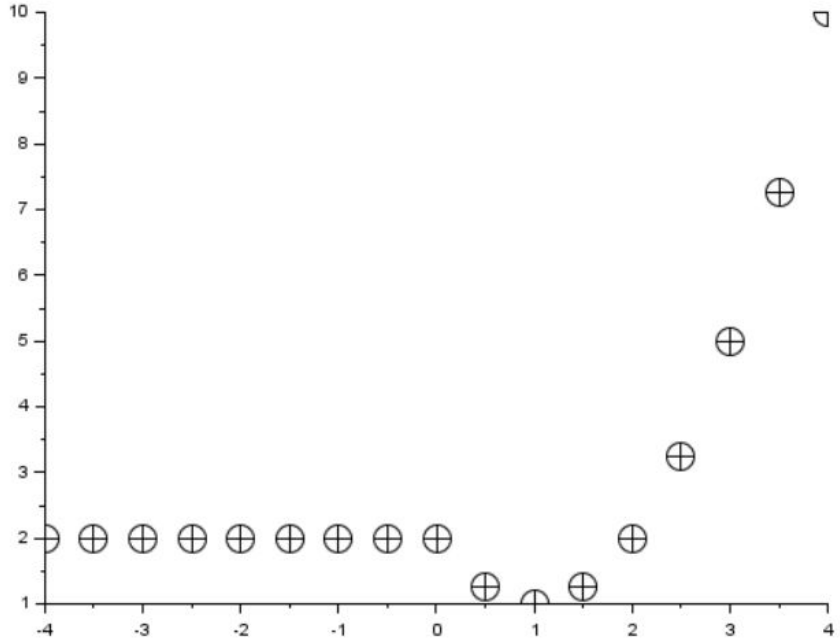

Imagem 3 - Janela gráfica 2 da introdução

Já na parte de macros, ao se executar o arquivo teste, o navegador de variáveis obtido é o seguinte :

Navegador de variáveis

	Nome	Value	Tipo	Visibilidade
	ans	3.61	Real	local

Tabela 2 - Navegador de variáveis do teste 1

Fazendo as modificações pedidas e executando o arquivo teste.sce novamente, obtêm-se o seguinte navegador de variáveis e os gráficos que se encontram abaixo dele.

Navegador de variáveis

	Nome	Value	Tipo	Visibilidade
	a	1	Real	local
	b	0	Real	local
	t1	1x1	Booleano	local
	t2	1x1	Booleano	local
	x	1x11	Real	local
	y	1x11	Real	local

Tabela 3 - Navegador de variáveis do teste 2

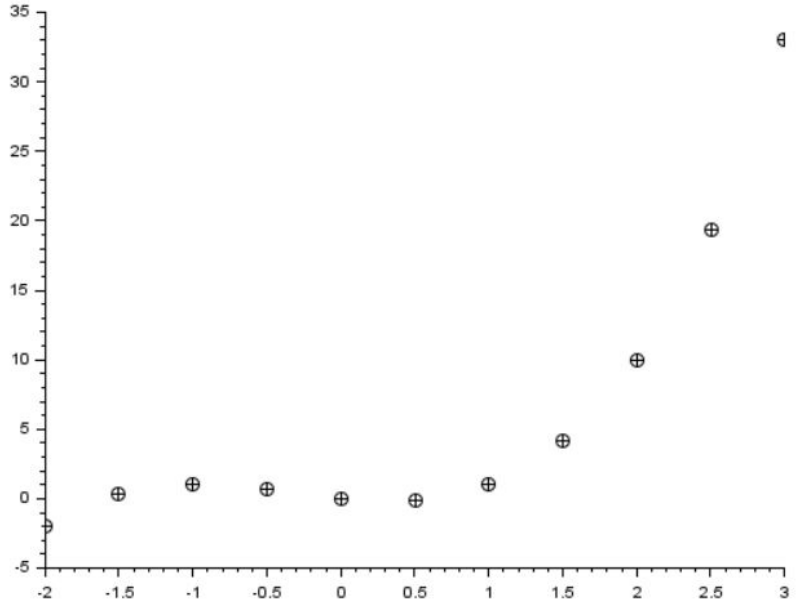

Imagem 4 - Janela gráfica 0 do teste 2

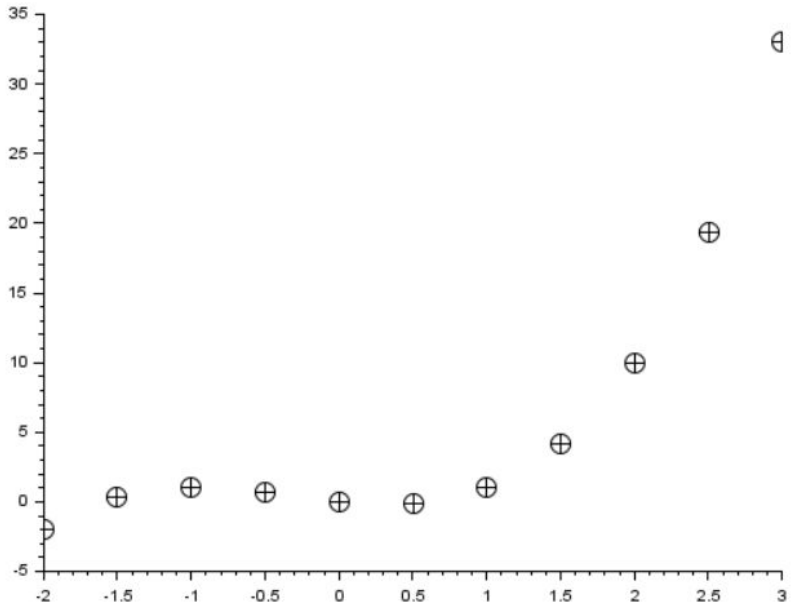

Imagem 5 - Janela gráfica 1 do teste 2

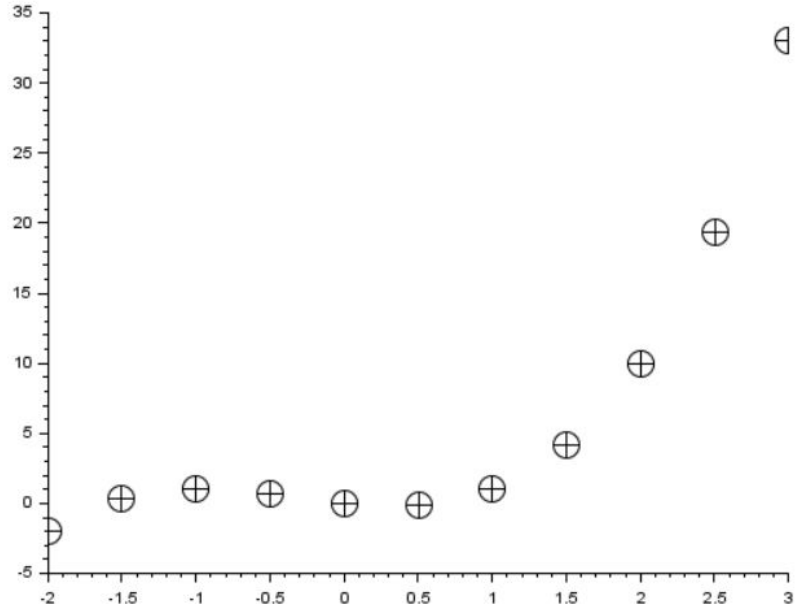

Imagem 6 - Janela gráfica 2 do teste 2

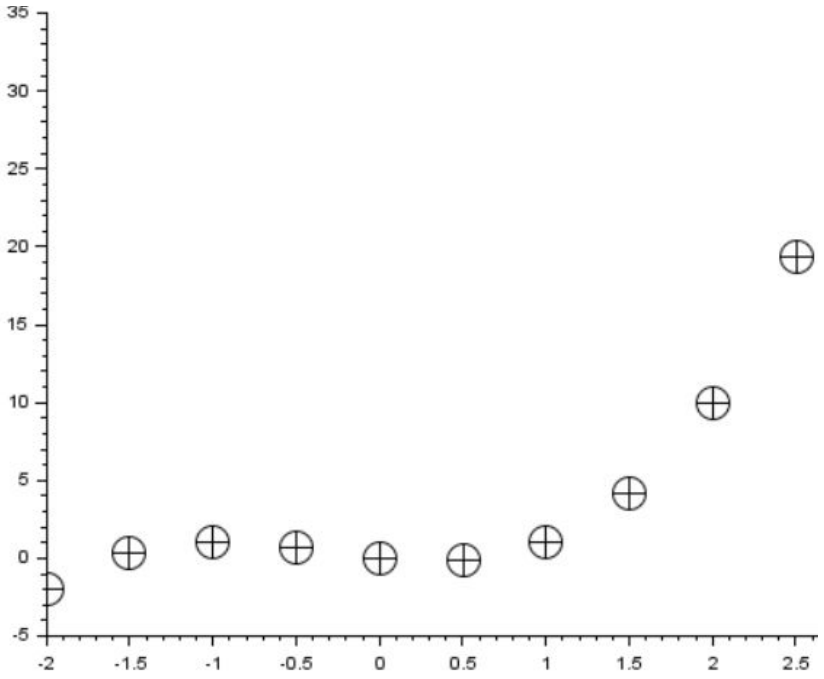

Imagem 7 - Janela gráfica 3 do teste 2

Códigos

Introdução :

```
clear;
clc;
xdel(winsid());

//Objetos:
//Definição de uma constante:
a=1
//Escrevendo números complexos:
a=2+%i
b=-5-3*%i
//Expressão Booleana:
//Verificação se a é igual a 1:
//O resultado da operação abaixo é o elemento booleano "F" (false):
a==1

//Matrizes e vetores:

//Vetor constante:
v=[1 2 3 4 5]
```

```
//ou
v=1:5

//Matriz constante:
A=[2 2 3
    0 0 7
    5 9 -1]
//ou
A=[2 2 3;0 0 7;5 9 -1]

//Podemos formar matrizes usando operações com objetos definidos anteriormente:
a=1;b=2;
//Observe que se colocarmos ponto e vírgula ao final da expressão, o resultado não é
mostrado na tela, o que pode ser conveniente em algumas situações.

A=[a+b %pi 3
     b^2 0 atan(a)
     5 sin(b) -1]


//Podemos formar matrizes e vetores de zeros:

//Elemento zero:
B=zeros()

//Matriz de zeros com 2 linhas e 3 colunas:
B=zeros(2,3)

//Matriz de zeros com as mesmas dimensões da matriz A:
A=[2 2 3;0 0 7;5 9 -1];
B=zeros(A)

//De modo semelhante, podemos formar matrizes e vetores de uns:

//Matriz de uns com 2 linhas e 3 colunas:
C=ones(2,3)

//Matrizes diagonais:

//Matriz diagonal com os elementos da diagonal principal indo de 1 a 5:
D=diag(1:5)
//Extraindo os elementos da diagonal principal:
A=[1 2 3
    4 5 6
    7 8 9]
B=diag(A)
```

```
//Formando uma matriz diagonal com os elementos da diagonal principal de uma matriz:
C=diag(diag(A))

//Operações:

//Matriz identidade:
A=diag(ones(1,3))

//Soma de matrizes:
B=A+A
//Somar 1 a todos os elementos de uma matriz:
C=B+1
//Multiplicação de matrizes:
A=[1 2 3;4 5 6;7 8 9]
C=[1 2 0;0 0 1;0 2 3]
D=A*C

//Multiplicação elemento a elemento:
A=[1 0 0;0 2 3;5 0 4]
B=[2 0 0;0 2 2;0 0 3]
C=A.*B

//Extração da linha 2:
a=C(2,:)
//Extração da coluna 3:
b=C(:,3)
//Extração da última linha:
b=C($,:)

//Traço de uma matriz:
A=[1 2 3; 4 5 6; 7 8 9]
t=trace(A)

//Rank (característica) de uma matriz:
r=rank(A)

//Matriz transposta:
B=A'

//Inversa de matriz:
A=[0 1;-2 -3]
B=inv(A)
A*B
```

```
//Determinante de uma matriz:
d=det(A)

//Polinômios:

//Polinômio em x com raízes em 0 e -1:
v=[0 -1]
p1=poly(v,'x')

//Polinômio em z com coeficientes 1 e 2 e 1:
p2=poly([1 2 1], 'z', 'coeff')

//Funções racionais:

//p1: numerador:
p1=poly(v,'s')

//p2: denominador:
p2=poly([5 2 1], 's', 'coeff')

//Função racional:
f=p1/p2

//Extração dos coeficientes:
a=coeff(p2)

//Cálculo de raízes:
p=roots(p1)

//Autovalores e autovetores:

//d - matriz diagonal cujos elementos são os autovalores.
//v - matriz cujas colunas são os autovetores.
[v,d]=spec(A)

//Funções:

//Definição de uma função:
deff('[y]=teste(x)','if x<0 then y=-(x^2),else y=sin(x),end')
//Uma vez definida a função, podemos calcular seu valor no ponto x=pi/2:
y=teste(0.5*%pi)

//No caso de x ser um vetor, a sintaxe seria:
deff('[y]=h(x)','n=length(x);for i=1:n, if x(i)<0 then y(i)=2, else y(i)=1+(x(i)-1)^2, end,end');

//Plotar a função entre –4 e 4:
```

```
//Criando um vetor com os valores variando de -4 a 4, com passo de 0.5:
x=-4:0.5:4;

//Calculando a função:
y=h(x);

//Plotando o resultado:
plot2d(x,y)

//Criando uma nova janela grafica:
 set("current_figure",1)

//Plotando o resultado com astericos:
plot2d(x,y,-3)

//Criando uma nova janela grafica:
 set("current_figure",2)

//Aumentando o tamanho dos asteriscos:
xset("mark size",4)
3
plot2d(x,y,-3)
```

Teste 1:

```
clear;
clc;
xdel(winsid());

function [y]=teste(x)
y=x+x^2+sin(x*2*%pi);
endfunction

teste(0.5*%pi)
```

Teste 2:

```
clear;
clc;
xdel(winsid());


deff('[y]=test0(x)','y=x+x^2+sin(x*2*%pi)')
deff('[y]=test1(x)','y=-x+x^2+x^3')
deff('[y]=test2(x)','y=sqrt(x)')

x=-2:0.5:3;
a=1;
b=0;
t1=(a==1);
t2=(b>0.5);

if and([t1 t2]) then
        y=test0(x);
        elseif or([t1 t2]) then
        y=test1(x);
        else
        y=test2(x);
end,

plot2d(x,y,-3)
set("current_figure",1)
xset('mark size', 2)
plot2d(x,y,-3)
set("current_figure",2)
xset('mark size', 4)
plot2d(x,y,-3)
set("current_figure",3)
xset('mark size', 5)
plot2d(x,y,-3)
```