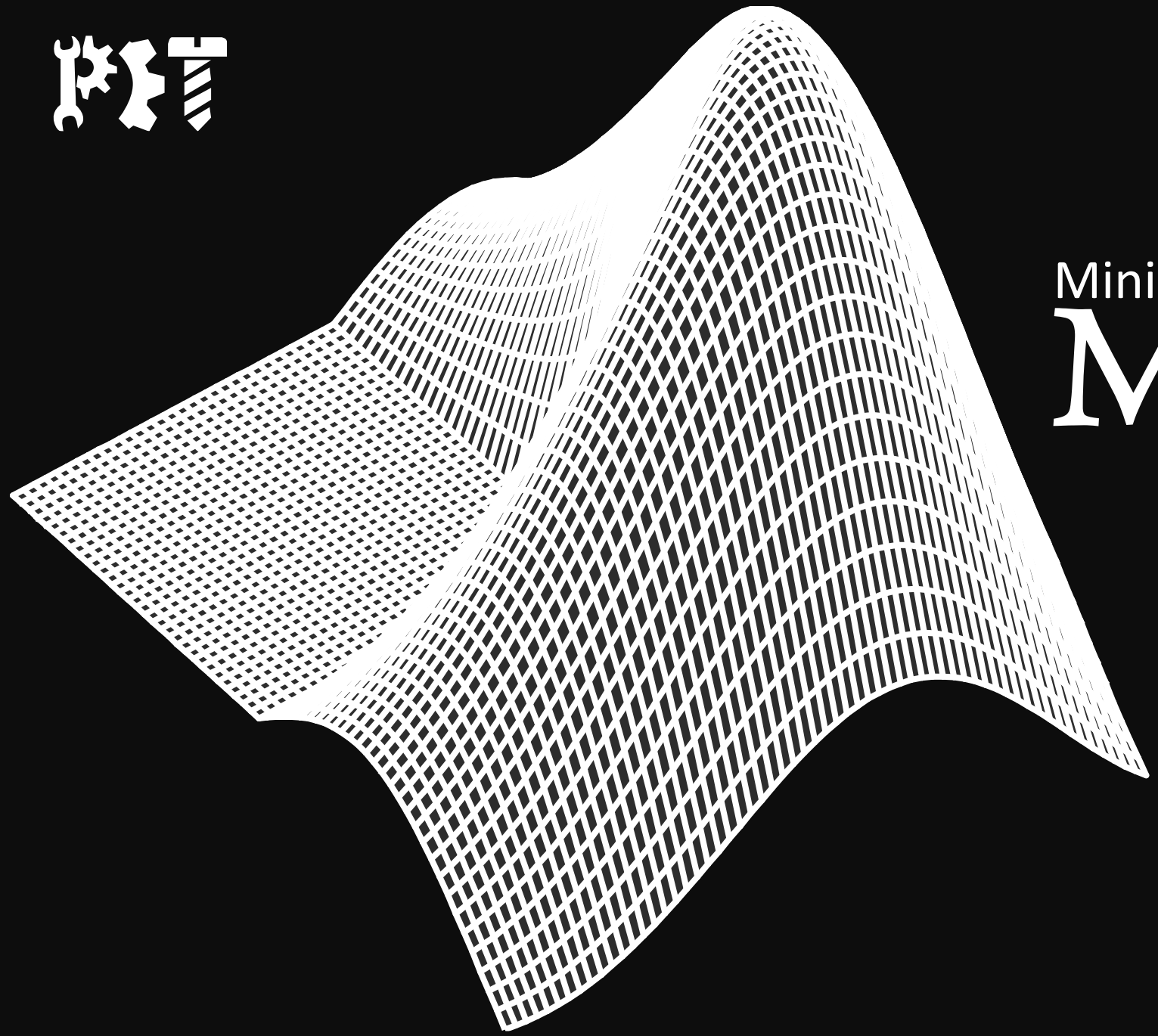


Minicurso

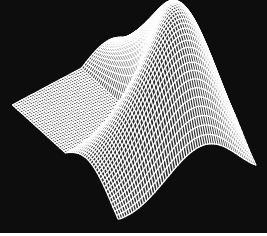
MATLAB

BÁSICO

MÓDULO 4

Roteiro do último módulo

- Máximos e mínimos
- Cálculo diferencial e integral numérico
- Cálculo diferencial e integral simbólico



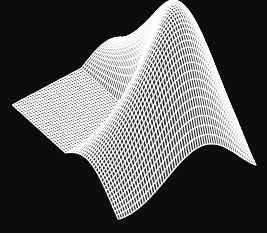
Mínimos e Máximos de uma função

→x = fminbnd(‘função’, x1,x2)

Usado para encontrar o mínimo e o máximo de uma função. X1 e x2 devem ser os limites do intervalo que se deseja avaliar.

→[x valor] = fminbnd(‘função’,x1,x2)

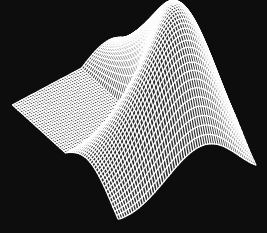
Usado para atribuir o resultado do comando em um valor



Mínimos e Máximos de uma função

→Exemplo

```
>> [x valor]=fminbnd('x^3 -12*x^2 +40.25*x -
36.5',0,8)

x = %valor onde a função apresenta o mínimo valor%

5.6073

valor = %o mínimo valor da função no intervalo
especificado%

-11.8043
```

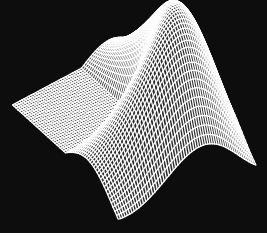


Mínimos e Máximos de uma função

→Exemplo

Quando se desejar encontrar o máximo valor da função em um dado intervalo, basta multiplicar toda a função desejada por “-1” e manter a mesma estrutura de comando

```
>> [x valor]=fminbnd+('-x^3 12*x^2 -40.25*x +36.5',0,8)

x =

2.3927

valor =

-4.8043
```



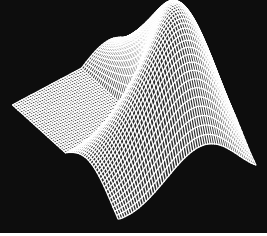
Diferenciação

Dados dois pontos muito próximos podemos aproximar a derivada da forma a seguir

$$\frac{dy}{dx} \approx \frac{y_2 - y_1}{x_2 - x_1}$$

→ diff

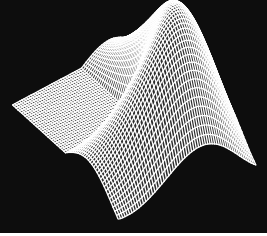
Calcula a diferença entre dois pontos adjacentes de um vetor e armazena o resultado em um novo vetor



Diferenciação

→ Exemplo

```
>> x = [0 1 2 3 4]

x =

     0     1     2     3     4

>> diff(x)

ans =

     1     1     1     1

>> y = [2 3 5 1 9 0]

y =

     2     3     5     1     9     0

>> diff(y)

ans =

     1     2    -4     8    -9
```



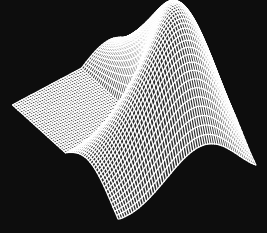
Diferenciação

→ OBS.:

Se *diff* for utilizado com uma matriz, a função tratará cada coluna da matriz como um vetor e calculará as diferenças para as colunas.

Tendo em mãos os vetores $diff(x)$ e $diff(y)$, para calcular a derivada basta realizar o seguinte cálculo

$$diff(y)\ ./diff(x)$$

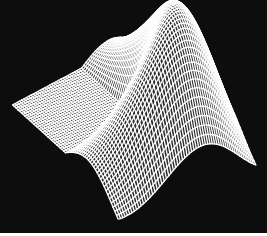


Integração Num. – Quadratura de Simpson

→ i=quad(@(x)(fun),a,b) **ou** (‘fun’,a,b)

Atente para o fato que a função deve considerar a sintaxe de operações elemento a elemento

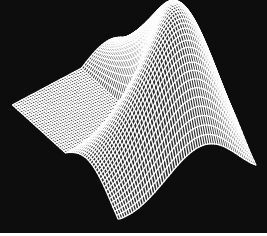
```
>> i=quad('x.^2',0,2)

resi =

    2.6667

>> i=quad(@(x)(x.^2),0,2)

resi =

    2.6667
```



Integração Num. – Quadratura de Simpson

→ i=dblquad(@(x,y)(fun),a,b,ay,by) **ou** (‘fun’,a,b,ay,by)

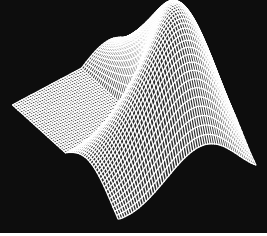
```
>> i=dblquad(@(x,y)(x.^2+y),0,2,0,3)

i =

    17

>> i=dblquad('x.^2+y',0,2,0,3)

i =

    17
```



Integração Num. – Quadratura de Simpson

→ i=triplequad(@(x,y,z)(fun),a,b,ay,by,az,bz) **ou** (‘fun’,a,b,ay,by,az,bz)

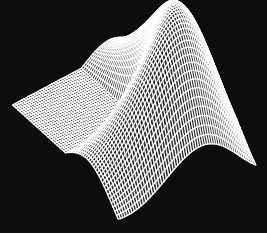
```
>> i=triplequad(@(x,y,z)(x.^2+y+z),0,2,0,3,0,1)

i =

    20

>> i=triplequad('x.^2+y +z',0,2,0,3,0,1)

i =

    20
```



Integração Num. – Quadratura de Simpson

→ **Atente** para a ordem de declaração das variáveis, isso também será sua ordem de integração

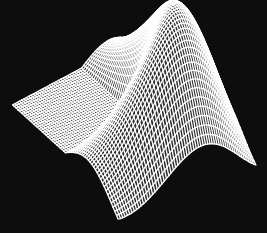
```
>> i=triplequad(@(y,z,x)(x.^2+y+z),0,2,0,3,0,1)

i =

    17
```



Integração Num. – Integral

→ i=integral(fun,a,b) | fun=@(x) x.^2

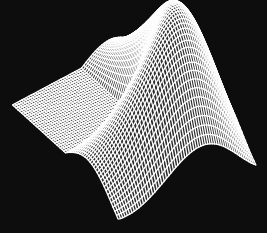
```
>> fun = @(x) x.^2;
>> i=integral(fun,0,2)

i =

    2.6667
```



Integração Num. – Integral

→ i=integral2(fun,a,b,ay,by) | fun=@(x,y) x.^2 +y

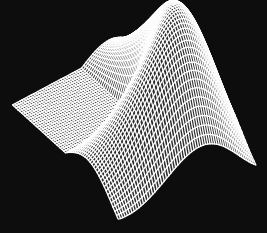
```
>> fun = @(x,y) x.^2+y;
>> i=integral2(fun,0,2,0,3)

i =

   17.0000
```



Integração Num. – Integral

→ i=integral3(fun,a,b,ay,by,az,bz) | fun=@(x,y,z) x.^2 +y +z

```
>> fun = @(x,y,z) x.^2+y+z;
>> i=integral3(fun,0,2,0,3,0,1)

i =

   20.0000
```



Matemática Simbólica

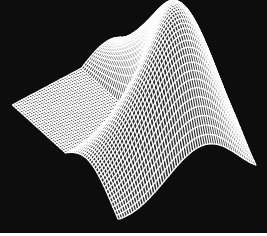
→ syms

Usado para declarar variáveis simbólicas.

→ pretty(S)

Usado para expressar uma expressão matemática como é escrita por nós humanos

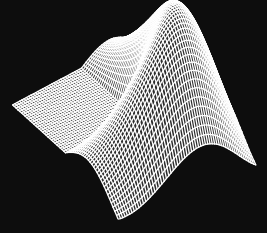
→ S nesse caso é a expressão ou função que desejamos representar



Matemática Simbólica

→ Exemplo

```
>> syms x y z
>> S=x^2 +2*y +1/z

S =

2*y + x^2 + 1/z

>> pretty(S)
        2   1
2 y + x  + -
           z
```



Matemática Simbólica – Eqs. Algébricas

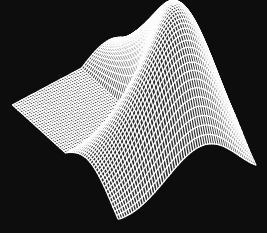
→ `h=solve(eq)` **ou** `h=solve(eq,var)`

Note que quando não declaramos um valor de igualdade o MATLAB atribui zero a esse valor.

solve('eq') => solve('eq' == 0)

```
>> syms x
>>  h = solve(x^2 + 2*x + 1)

h =

 -1
 -1
```

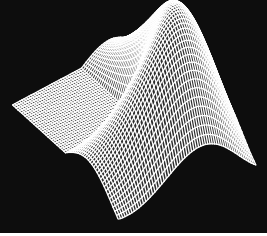
```
>> syms x
>>  h = solve(x^2 + 2*x + 1 == 0)

h =

 -1
 -1
```



Matemática Simbólica – Eqs. Algébricas

Note que devemos atribuir a igualdade com (==)

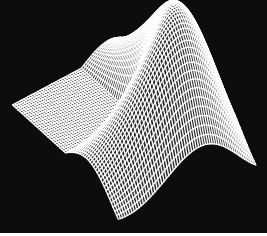
```
>> syms x
>> h = solve(x^2 + 2*x + 1==2)

h =

 - 2^(1/2) - 1
   2^(1/2) - 1
```



Matemática Simbólica – Eqs. Algébricas

Note que a 'eq.' também pode ser expressa como uma variável simbólica

```
>> syms x
>> eq=x^2+2*x+1==2;

>> raizes=solve(eq)

raizes =

 - 2^(1/2) - 1
   2^(1/2) - 1
```

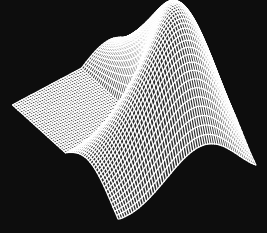


Matemática Simbólica – Sistema de Eqs.

→ (saída)=solve(eq1,eq2,...,eqn) **ou**

→ (saída)=solve(eq1,eq2,...,eqn,var1,var2,...,varn)

Se o número **n** de equações é igual ao número de variáveis nas equações, o MATLAB apresenta uma solução numérica para todas as variáveis

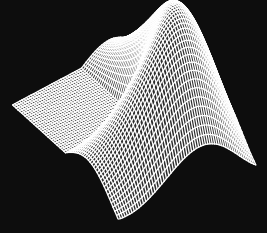


Matemática Simbólica – Sistema de Eqs.

→ (saída)=solve(eq1,eq2,...,eqn) **ou**

→ (saída)=solve(eq1,eq2,...,eqn,var1,var2,...,varn)

Se o número de variáveis for **maior** que a de equações, o MATLAB apresenta uma solução para **n** variáveis em termos do restante das variáveis

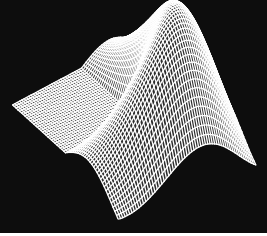


Matemática Simbólica – Sistema de Eqs.

→ (saída)=solve(eq1,eq2,...,eqn) **ou**

→ (saída)=solve(eq1,eq2,...,eqn,var1,var2,...,varn)

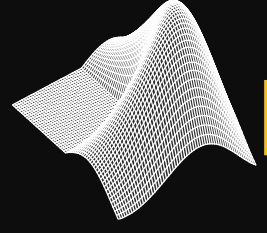
Quando o número de variáveis supera o número de equações você pode escolher para que variáveis o sistema será resolvido (usa-se a segunda sintaxe)



Matemática Simbólica – Sistema de Eqs.

→ [x1, x2, x3]=solve(eq1,eq2,eq3)

Representação das saídas está esquematizada acima.
Note que x1, x2 e x3 podem ter mais de um valor, sendo assim vetores colunas.

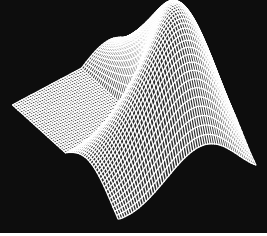


Matemática Simbólica – Sistema de Eqs.

→ Exemplo

A seguir demonstraremos como resolver o seguinte sistema de equações

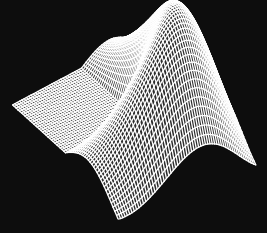
$$(\text{I})\ 10x + 12y + 6t = 0$$
$$(\text{II})\ 5x - y = 13t$$



Matemática Simbólica – Sistema de Eqs.

→ Exemplo

```
>> syms x y t
>> S=10*x+12*y+16*t;
>> [x1 y1]=solve(S,'5*x-y=13*t')

x1 =

2*t


y1 =

-3*t
```



Matemática Simbólica – Sistema de Eqs.

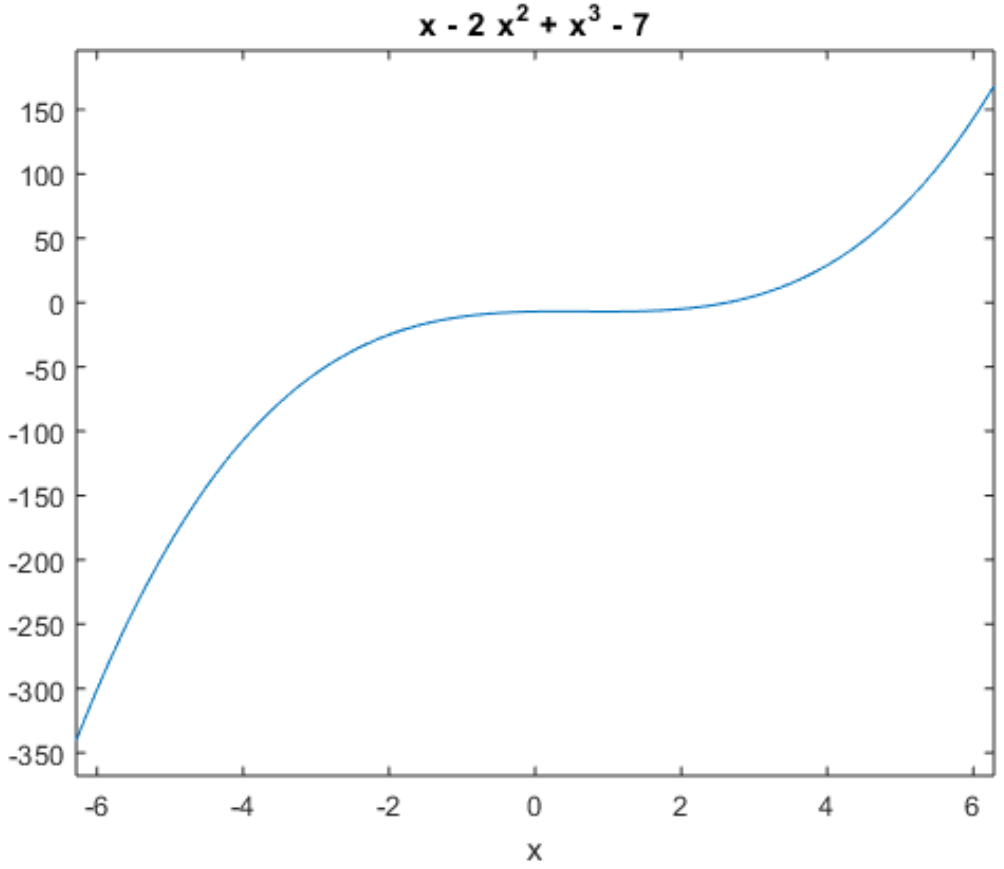
→ Exemplo

```
>> syms x y t
>> S=10*x+12*y+16*t;
>> T=5*x-y-13*t;
>> [x1 y1]=solve(S,T)

x1 =

2*t


y1 =

-3*t
```

Plotando Funções Simbólicas

→ `ezplot(S)`

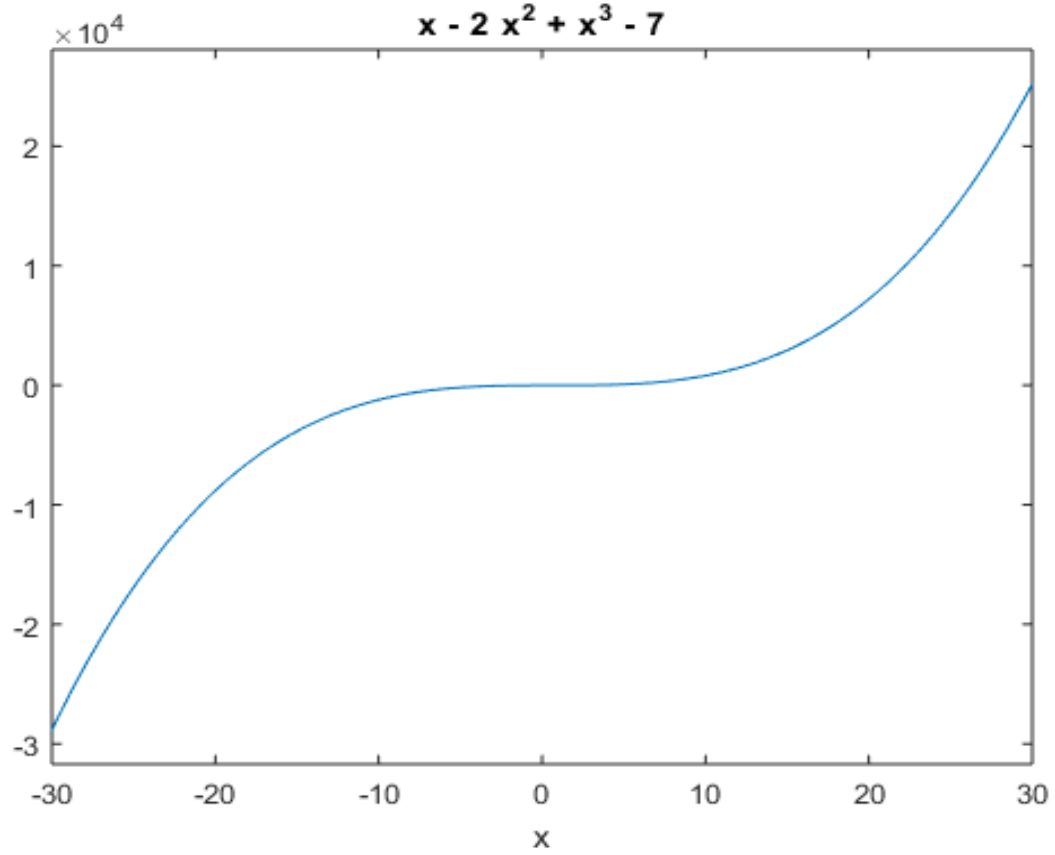
```
>> syms x
>> S=x^3 -2*x^2 +x-7;
>> ezplot(S)
```



Plotando Funções Simbólicas

→ `ezplot(S,[xmin,xmáx])`

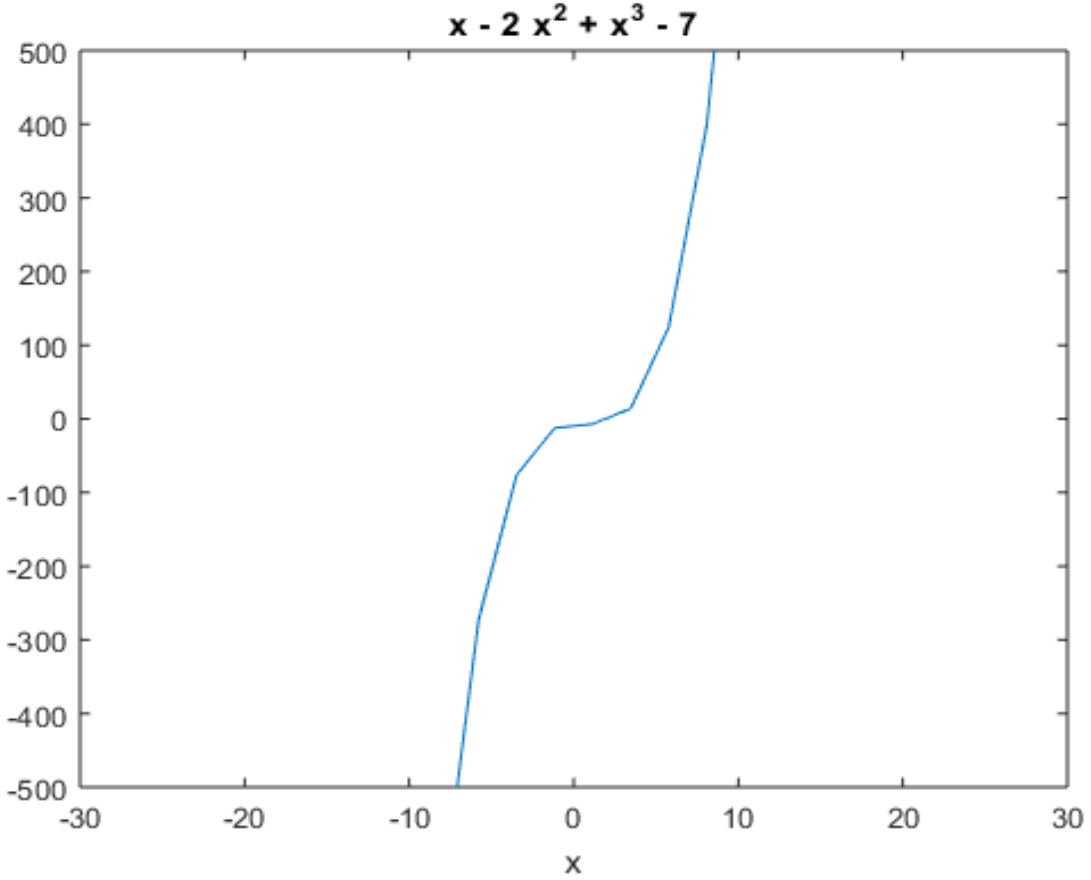
```
>> syms x
>> S=x^3 -2*x^2 +x-7;
>> ezplot(S,[-30,30])
```

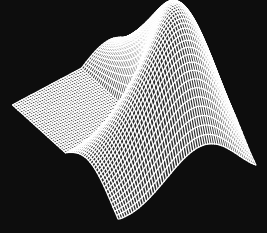


Plotando Funções Simbólicas

→ `ezplot(S,[xmin,xmáx,ymin,ymáx])`

```
>> syms x
>> S=x^3 -2*x^2 +x-7;
>> ezplot(S,[-30,30,-500,500])
```

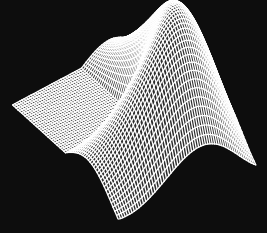




Avaliando Funções Simbólicas

→ res=subs(S,var,número)

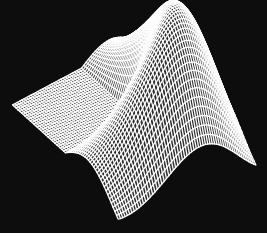
```
>> syms x
>> S=x^2;
>> res=subs(S,x,4)

res =

16
```



Avaliando Funções Simbólicas

→ res=subs(S,{var1,var2,...,varn},{n1,n2,...,nn})

```
>> syms x y z t

>> S=x^2+3*t+log(y)+exp(z);
>> res=subs(S,{x,t,y,z},{1,2,5,3})

res =

exp(3) + log(5) + 7
```

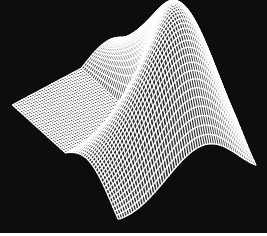


Diferenciação – Diferenciação simbólica

→ d=diff(S)

Usado para diferenciar simbolicamente.

```
>> syms x

>> S = 3*x + 4*x^3;

>> d=diff(S)

d =

12*x^2 + 3
```

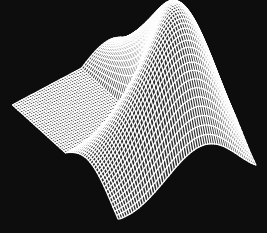


Diferenciação – Diferenciação simbólica

→ Exemplo

Também é possível fazer a derivação sem declarar vetores

```
>> syms x
>> d=diff(2*x +x^2)

d =

2*x + 2
```

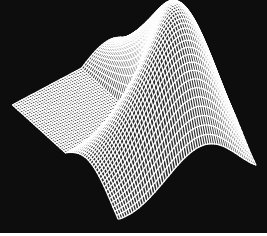


Diferenciação – Diferenciação simbólica

→ d=diff(S,var)

Usado para diferenciar parcialmente em que *var* trata-se da variável que se deseja diferenciar.

```
>> syms x y
>> d=diff(x^2 +2*x +y,y)

d =

1

>> d=diff(x^2 +2*x +y,x)

d =

2*x + 2
```

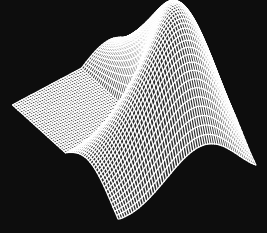


Diferenciação – Diferenciação simbólica

→ d=diff(S,var)

Usado para diferenciar parcialmente em que *var* trata-se da variável que se deseja diferenciar.

```
>> syms x y
>> d=diff(diff(x^2 +2*x +y,y),x)

d =

0

>> d=diff(diff(x^2 +2*x +y,x),y)

d =

0
```

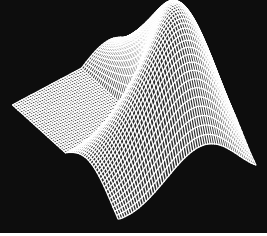


Integração – Integração simbólica

→ i=int(S)

Usado para integrar simbolicamente

```
>> syms x

>> S=x^2 +1;

>> i=int(S)

i =

(x*(x^2 + 3))/3
```

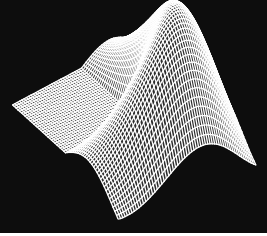


Integração – Integração simbólica

→ i=int(S, var)

Usado para integrar simbolicamente. Var trata-se da variável em que está integrando

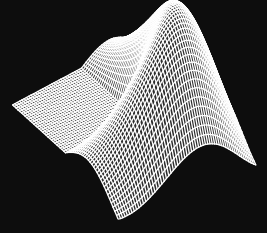
```
>> syms x z
>> resi=int(S,z)

i =

z*(x^2 + 1)
```



Integração – Integração simbólica

→ i=int(S, a, b) **ou** resi=int(S, var, a, b)

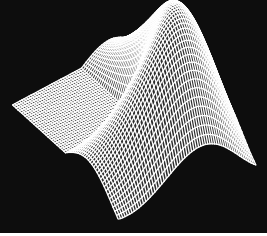
```
>> syms x
>> S=x^2;
>> i=int(S,0,2)

i =

8/3
```



Integração – Integração simbólica dupla

→ i=int(int(S, var1, a, b),var2,a1,b1)

```
>> syms x y
>> S=x^2+y;
>> i=int(int(S,x,0,2),y,0,3)

i =

17
```



Integração – Integração simbólica tripla

→ i=int(int(int(S, var1, a, b),var2,a1,b2),var3,a3,b3)

```
>> syms x y z
>> S=x^2+y+z;
>> i=int(int(int(S,x,0,2),y,0,3),z,0,1)

i =

20
```