

UNICEP

MINI - MANUAL PARA O USO DO MAPLE

Prof. Dr. Artur Darezzo Filho

Prof.Dr. Edson de Oliveira

2008

Mini-manual para o uso do Maple

O aplicativo Maple 7

Ao abrir o Maple nos deparamos com uma página de trabalho que apresenta um símbolo > em frente ao qual podemos declarar comandos a serem executados. Essa região pode ser modificada para uma região do tipo texto, ou vice-versa, com o uso da tecla F5 ou utilizando os ícones T e Σ do menu. As declarações são feitas por números, operadores e funções. O uso mais simples do Maple consiste em:

*digitar a expressão a ser executada
escrever um ponto e vírgula no final de cada expressão
acionar a tecla enter*

Desta forma, o Maple avalia a expressão na cor vermelha e a resposta aparece na linha seguinte, centralizada, na cor azul e, em notação matemática padrão.

Se no final de uma expressão digitarmos dois pontos " : " em vez do ponto e vírgula " ; " o Maple efetua e avalia, armazena na memória mas não mostra o resultado.

Os símbolos + , - , * , / e ^ denotam as operações de adição, subtração, multiplicação, divisão e potenciação, respectivamente. O símbolo ! denota fatorial.

Exemplos:

```
> 6 + 5 - 8;
3

> (12*3)/2;
18

> 3+5!;
123

> 3^4 - 16^3;
-4015
```

Para acionar o sistema de ajuda digitamos no prompt o sinal ? seguido da expressão da qual se deseja a informação. Por exemplo, digitando

```
> ?matrix
```

e acionando a tecla **enter** nos são retornadas informações sobre o comando **matrix**. O sistema de ajuda também pode ser acionado através do menu, no item **help**.

Ao encontrar uma falha em um comando, o Maple responde com uma mensagem de erro indicando o tipo de falha. Algumas delas são: sintaxe (equivoco na digitação, nome incorreto do comando, uso de palavras reservadas, etc.), erro de domínios de funções matemáticas, cálculos que excedem a capacidade de memória ou de computação do sistema ou do aplicativo. É sempre importante corrigi-los pois erros não corrigidos levam a mensagens de erros nos comandos subsequentes.

```
> 3/(6-3!);
Error, numeric exception: division by zero

> 1234567890^9876543210;
Error, numeric exception: overflow (o cálculo excede a capacidade de memória )
```

Digitando o comando **restart** e acionando a tecla **enter**, limpamos a memória interna do Maple, usada para estocar os cálculos. Daí para frente, tudo passa a ocorrer como se o arquivo tivesse sido aberto naquele momento.

Pacotes no Maple

O Maple 7 possui pacotes que fornecem uma série de recursos para resolver problemas específicos de determinados assuntos. Por exemplo, ele possui pacote para teoria dos grupos (o pacote *group*), pacotes para geometria (os pacotes *geometry* e *geom3d*), para teoria dos números (*numtheory*), para álgebra linear (*linalg*), para cálculo (*student*), para gráficos (*plots*), objetos gráficos básicos (*plottools*), etc.

Para acessar as funções de um pacote, como por exemplo o pacote para cálculo, digitamos o comando *with(student)*: . Digitando (;) em vez de (:) , podemos ver a listagem de todas as funções definidas no pacote.

> **with(student)**:

[*D, Diff, Doubleint, Int, Limit, Lineint, Product, Sum, Tripleint, changevar, completesquare, distance, equate, integrand, intercept, intparts, leftbox, leftsum, makeproc, middlebox, middlesum, midpoint, powsubs, rightbox, rightsum, showtangent, simpson, slope, summand, trapezoid*]

Números no Maple

O Maple reconhece vários tipos numéricos. Os mais utilizados são os seguintes : **integer** (inteiros) , **fraction** (fração), **float** e **complex** (complexos).

Float, são números reais representados no sistema de ponto flutuante e se constituem de uma seqüência de dígitos e um ponto decimal, multiplicados ou não, por uma potência de 10.

Exemplos: 1.75, 3.49, 0.3675*10¹².

Observemos que 1,75 (com vírgula) o Maple não aceita; devemos usar 1.75 (um ponto setenta e cinco).

Os complexos, são os números da forma $a + b*I$, onde a e b são números inteiros, fracionários ou floats e $I = \sqrt{-1}$.

Por padrão o valor numérico de uma expressão é dado pelo Maple no tipo mais abrangente que aparece na expressão.

Exemplos:

> **15+12;**

27

> **10+3.9;**

13.9

> **18/12;** (o Maple por padrão simplifica frações de inteiros)

$\frac{3}{2}$

Observemos que no último exemplo o Maple só simplificou a expressão. Ele utilizou na resposta a forma fracionária, que é a forma mais abrangente da expressão dada. A forma decimal que seria a esperada não aconteceu. Para obtê-la podemos usar:

> **18./12;** (escrevendo o número 18 na forma decimal 18.0)

1.500000000

A função **evalf** avalia expressões no tipo float, com 10 dígitos, incluindo a parte inteira. Seguem alguns exemplos:

> **evalf(18/12);**

1.500000000

> **evalf(3437*Pi/23);** (Pi é a notação para o número irracional 3,1415....)

469.4632153

Podemos especificar o número de dígitos que desejarmos para a representação decimal dos números. O padrão é 10 e o limite prático é 500000, dependendo muitas vezes dos recursos do computador.

Por exemplo, se quisermos uma representação decimal para o número π com 80 dígitos, procedemos:

> **evalf(Pi,80);**

3.141592653589793238462643383279502884197169399375

Podemos usar a função **convert** para transformar a representação dos números .

> **convert(18/12,float);**

1.500000000

Um outro exemplo de aplicação da função **convert**.

convert(3.777777,fraction);

$\frac{539663}{142852}$

O comando para a raiz quadrada é **sqrt**.

> **sqrt(50);** (observemos que o tipo de número mais abrangente na expressão é inteiro)

$5\sqrt{2}$

> **convert(sqrt(50),float); evalf(sqrt(50));**

7.071067812

7.071067810

Para outras raízes:

> **root[3](10.);** (raiz cúbica de 10)

2.154434690

Pode-se também usar a forma de potências:

> **evalf(10^(1/3));**

2.154434690

Algumas funções básicas da Teoria dos Números

igcd	retorna o máximo divisor comum de dois ou mais números
ilcm	retorna o mínimo múltiplo comum de dois ou mais números
iquo(n,m)	retorna o quociente da divisão euclidiana de n por m
irem(n,m)	retorna o resto da divisão euclidiana de n por m
is(n=m)	verifica se n é igual a m

Funções logarítmicas, exponenciais e trigonométricas

Existe uma grande quantidade de funções reconhecidas pelo Maple. Para verificá-las podemos usar o **Help** do menu ou dar busca no tópico **inifcn**.

Por exemplo, para o logaritmo de x na base e o comando é **ln(x)**.

No caso de logaritmos de bases quaisquer, utilizamos **log[Basis](x)** onde Basis representa a base escolhida.

O comando para a função exponencial é **exp(x)**.

Os seguintes comandos são utilizados para calcular funções trigonométricas:

sin(x)	função seno	cot(x)	função cotangente
cos(x)	função cosseno	sec(x)	função secante
tan(x)	função tangente	csc(x)	função cossecante

Exemplos:

> csc(Pi/4);	$\sqrt{2}$
> exp(3);	e^3
> evalf(%);	20.08553692
> ln(100.);	4.605170186
> log[10](100000.); (cálculo do logaritmo de 100.000 na base 10)	5.000000000
> cos(5.);	.2836621855
> sec(7*Pi/4);	$\sqrt{2}$

> **evalf(sin(Pi/3));**

.8660254040

Nas funções trigonométricas, o valor de x é sempre dado em radianos (ou seja, o padrão do Maple é radiano). Se, por exemplo, quisermos o valor do $\sin(150^\circ)$, devemos converter 150° em radianos.

Procedemos assim:

> **sin(convert(150*degrees,radians));**

$\frac{1}{2}$

Computação Simbólica

A computação simbólica é uma das ferramentas mais importantes do aplicativo Maple. Ela nos permite obter cálculos algébricos exatos envolvendo objetos matemáticos.

Por padrão, o Maple simplifica expressões reduzindo os monômios semelhantes.

Vejam alguns exemplos.

> **$x^3 - 3x^2 - 6x + 11x^2 - 3x$;**

$x^3 + 8x^2 - 9x$

O comando para fatorar é **factor** .

> **factor(%);**

$x(x+9)(x-1)$

Podemos dar nomes a expressões, mediante o comando de atribuição **:=** .

> **$p := (x-1)^2(x-2)^4$;**

$p := (x-1)^2(x-2)^4$

Para liberar um nome de variável, já utilizado, utilizamos o comando **p:='p'**.

Feito isso, **p** reassume a sua característica de símbolo.

> **$z := 'p'$;**

$z := p$

Para expandir expressões, usamos o comando **expand** .

> **expand(p);**

$x^6 - 10x^5 + 41x^4 - 88x^3 + 104x^2 - 64x + 16$

> **$q := \text{expand}((x-1)^3)$;**

$q := x^3 - 3x^2 + 3x - 1$

> **$r := \text{quo}(p,q,x)$** : (retorna o quociente da divisão polinomial de **p** por **q**)

$r := x^3 - 7x^2 + 17x - 15$

> **$\text{rem}(p,q,x)$** : (retorna o resto da divisão polinomial de **p** por **q**)

$x^2 - 2x + 1$

> **$\text{gcd}(p,q)$** : (retorna o máximo divisor comum de **p** e **q**)

$(x-1)^2$

> **lcm(p,q):** (retorna o mínimo múltiplo comum de **p** e **q**)

$$(x-2)^4(x^3-3x^2+3x-1)$$

> **coeff(p,x^5):** (retorna o coeficiente de x^5)

$$-10$$

> **degree(p):** (retorna o grau do polinômio)

$$6$$

> **subs(x=-3,p):** (avalia a expressão **p** para $x = -3$)

$$10000$$

Algumas manipulações algébricas

Numa expressão racional muitas vezes é útil cancelar os fatores comuns do numerador e do denominador.

O comando **normal** realiza esta tarefa e apresenta numerador na forma expandida.

> **exp1:=(x^2-y^2)/(x-y)^3;**

$$\text{exp1} := \frac{x^2 - y^2}{(x - y)^3}$$

> **normal(exp1):**

$$\frac{y + x}{(-x + y)^2}$$

Os comandos **numer** e **denom** do Maple são usados, respectivamente para destacar o numerador e o denominador da fração.

> **fr:=((x-3)/(x+2)^2);**

$$\text{fr} := \frac{x - 3}{(x + 2)^2}$$

> **numer(fr):** (extrai o numerador de uma expressão racional)

$$x - 3$$

> **denom(fr):** (extrai o denominador de uma expressão racional)

$$(x + 2)^2$$

Listas e arrays

O comando **seq** pode ser usado para definir seqüências finitas. Por exemplo, a seqüência dos cubos de 1 a 5 é obtida por:

> **S:=seq(i^3,i=1..5);**

Quando digitamos uma seqüência entre colchetes criamos uma **lista**.

> $L := [a, b, c, d, e, f]$:

$L := [a, b, c, d, e, f]$

O Maple preserva a ordem e a repetição numa lista. Assim, $[r, s, t]$, $[s, t, r]$, $[r, r, s, t, r]$ são listas diferentes.

Como a ordem é preservada, é possível extrair um particular elemento de uma lista.

Assim, para extrair o **n-ésimo** elemento de uma lista **L** usamos o comando $L[n]$. Para extrair do **n-ésimo** ao **m-ésimo** termo, usamos $op(n..m, L)$.

Para transformar uma lista **K** numa seqüência, usamos $op(K)$.

> $L[3]$:

c

> $op(3..5, L)$:

c, d, e

> $K := [seq(5i, i=4..9)]$:

$K := [20, 25, 30, 35, 40, 45]$

> $K1 := op(K)$:

$K1 := 20, 25, 30, 35, 40, 45$

O número total de elementos de uma lista **L** é obtido por $nops(L)$.

> $nops(L)$:

6

A função $map(\text{comando}, \text{lista})$ aplica um determinado comando a cada elemento de uma dada lista.

> $T := [0, -\pi/2, -\pi, 3\pi/2]$:

$T := \left[0, -\frac{1}{2}\pi, -\pi, \frac{3}{2}\pi \right]$

> $map(\cos, T)$: (retorna a lista dos cossenos dos elementos da lista **T**)

$[1, 0, -1, 0]$

> $map(abs, T)$: (retorna a lista dos valores absolutos dos elementos da lista **T**)

$\left[0, \frac{1}{2}\pi, \pi, \frac{3}{2}\pi \right]$

Arrays são extensões dos conceitos de listas. São tipos de tabelas de dimensões 0, 1 ou mais e produzem, por exemplo, vetores, matrizes, etc.

O comando a seguir retorna uma lista de 3 elementos. Na lista, os elementos estão especificados entre colchetes.

> **array(1..3,[1,4,16]);**

[1, 4, 16]

No exemplo abaixo é definido um **array** de dimensão 2 (matriz) consistindo de três linhas (primeira entrada 1..3) e duas colunas (segunda entrada 1..2). Os elementos da primeira linha são dados pela primeira lista e os da segunda linha, pela segunda lista e os da terceira linha, pela terceira lista.

> **B:=array(1..3,1..2,[[3,4],[5,5],[8,0]]);**

$$B := \begin{bmatrix} 3 & 4 \\ 5 & 5 \\ 8 & 0 \end{bmatrix}$$

Para selecionar um elemento especificamos sua linha e a coluna .

> **B[3,2];** **B[1,2];**

0
4

Funções

Funções de uma variável são definidas por **nome da função:= variável -> expressão**, onde a seta é composta do símbolo de subtração - , seguido do símbolo de desigualdade > .

> **f:=x->x^3 ;**

$$f := x \rightarrow x^3$$

> **f(3);** **f(y+1);**

$$27 \\ (y+1)^3$$

> **(f(x+h)-f(x))/h;**

$$\frac{(x+h)^3 - x^3}{h}$$

> **simplify(%);**

$$3x^2 + 3xh + h^2$$

Consideremos a expressão **a** definida por :

> **a:=x^3+3*x^2-4*x+2;**

$$a := x^3 + 3x^2 - 4x + 2$$

Observemos que o Maple não avalia a para $x = 2$, pois ele entende a como uma expressão e não como uma função.

O comando `unapply(E,x)` transforma em função de x , qualquer expressão algébrica E .

> `g:=unapply(a,x):`

$$g := x \rightarrow x^3 + 3x^2 - 4x + 2$$

> `g(2):`

14

Para definir funções reais mais complicadas, como por exemplo $\begin{cases} x^2, & x > 3 \\ x - 5, & x \leq 3 \end{cases}$ podemos empregar o comando `piecewise`.

> `h:=x->piecewise(3<x,x^2,x<=3,x-5):`

$$h := x \rightarrow \text{piecewise}(3 < x, x^2, x \leq 3, 5)$$

> `h(x):`

$$\begin{cases} x^2 & 3 < x \\ x - 5 & x \leq 3 \end{cases}$$

> `h(-1):` `h(3):` `h(6):`

-6

-2

36

O comando `limit(g(x), x=a)` determina o **limite** da função g no ponto $x=a$.

> `limit(g(x),x=-1):`

8

O comando `diff(g(x), x)` determina a **primeira derivada** da função g .

> `derivadag:=diff(g(x),x):`

$$\text{derivadag} := 3x^2 + 6x - 4$$

Com este comando, o Maple retorna a derivada de $g(x)$ como uma expressão e não como função.

Para se obter a função derivada, utilizamos o comando `D(g)`.

> `D(g):`

$$x \rightarrow 3x^2 + 6x - 4$$

> `D(g)(1):` (calcula a derivada no ponto $x=1$)

5

Também podemos usar o comando `subs(x=1, derivadag)`, para efetuar a substituição de $x=1$ na expressão da derivada de g acima.

> **subs(x=1,derivadag):**

5

> **diff(g(x),x,x):** (retorna a derivada segunda)

$6x + 6$

> **diff(g(x),x\$3):** (retorna a derivada terceira)

6

> **D(D(g)): (D@@2)(g):**

$x \rightarrow 6x + 6$

$x \rightarrow 6x + 6$

> **(D@@4)(g):** (função derivada de ordem 4)

0

O comando **int(g(x) , x)** determina a **integral indefinida** da função **g** a menos de constante.

> **int(g(x),x):**

$\frac{1}{4}x^4 + x^3 - 2x^2 + 2x$

> **int(g(x),x=-1..3):** (calcula a integral definida para x de -1 até 3.)

40

Se quisermos apenas indicar a integral, sem resolvê-la, utilizamos:

> **Int(g(x),x=-1..3):**

$\int_{-1}^3 x^3 + 3x^2 - 4x + 2 dx$

Para calcular o valor da integral anterior procedemos:

> **value(%):**

40

Funções de várias variáveis são definidas de maneira similar:

nome da função := (variável 1,variável 2,...,variável n) -> expressão

> **H:=(x,y)->x^2+2*y:**

$H := (x, y) \rightarrow x^2 + 2y$

> **H(-1,2): H(b+1,b-1):**

$$(b+1)^2 + 2b - 2$$

Solução de equações: o comando solve

O comando `solve(equação, { variável })`, resolve equações analiticamente.

Caso a variável não seja indicada (sua indicação é opcional), o Maple faz a escolha.

Se no lugar da equação escrevermos uma expressão, o Maple considera expressão = 0.

Se no conjunto de soluções aparecer uma expressão do tipo $x = x$, isto significa que x pode assumir qualquer valor.

> `solve(x^4-2*x^2-1/2*x^3+x,{x});`

$$\{x=0\}, \{x=\frac{1}{2}\}, \{x=\sqrt{2}\}, \{x=-\sqrt{2}\}$$

> `solve(3*x+9*y=0);`

$$\{y=y, x=-3y\}$$

Podemos associar a seqüência de soluções de uma equação a uma variável.

> `solve(3*x^2-5*x+7);`

$$\frac{5}{6} + \frac{1}{6}I\sqrt{59}, \frac{5}{6} - \frac{1}{6}I\sqrt{59}$$

> `u:=%;`

$$u := \frac{5}{6} + \frac{1}{6}I\sqrt{59}, \frac{5}{6} - \frac{1}{6}I\sqrt{59}$$

A variável u é uma seqüência de dois elementos. Podemos acessar cada solução separadamente.

> `u[1];` (retorna a primeira solução da equação)

$$\frac{5}{6} + \frac{1}{6}I\sqrt{59}$$

> `u[2];` (retorna a segunda solução da equação)

$$\frac{5}{6} - \frac{1}{6}I\sqrt{59}$$

O comando `solve`, pode ser usado também para resolver um **sistema de equações**. A sintaxe é :

`solve({ equação 1, equação 2, ..., equação n, { variável 1, variável 2, ..., variável m })`

Exemplo: Resolver o sistema linear

$$\begin{cases} x + y + z = 1 \\ 2x + y + z = 3 \\ x - y + 2z = 0 \end{cases}$$

> **solve**($\{x+y+z=1, 2*x+y+z=3, x-y+2*z=0\}, \{x,y,z\}$);

$$\{x=2, y=0, z=-1\}$$

Resolução numérica: o comando fsolve

O comando **fsolve** é o equivalente numérico de **solve**.

Ele resolve equações produzindo, em geral, uma única solução real aproximada e do tipo ponto flutuante. Entretanto, no caso de polinômios, o comando pesquisa todas as raízes reais.

> **fsolve**($x^4-3*x^3-12*x^2-36*x+72$);

$$1.316049770, 5.780542834$$

No caso de polinômios, a opção **complex** força o Maple pesquisar também as raízes complexas, além das reais.

> **fsolve**($x^4-3*x^3-12*x^2-36*x+72, \{x\}, \text{complex}$);

$$\{x = -2.048296302 - 2.295397344 I\}, \{x = -2.048296302 + 2.295397344 I\}, \\ \{x = 1.316049770\}, \{x = 5.780542834\}$$

Gráficos

Para se obter gráficos de **dimensão 2** o comando básico é

plot (**f**, **x=a..b**, **y=c..d**, **opções**)

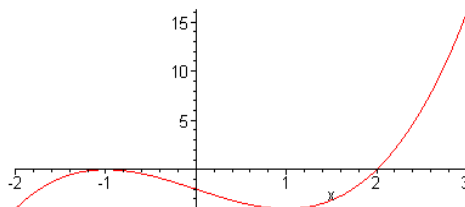
onde **f** é uma função de uma variável real, **x = a..b** é o intervalo de variação de **x**, **y = c..d** é o intervalo de variação de **y** e opções são dados disponíveis no sistema, que podem melhorar a visualização do gráfico conforme a necessidade.

Os elementos dos intervalos devem ser constantes reais.

Os dois últimos argumentos não são essenciais para se obter o gráfico. A variação de **x**, também não. Neste caso, o intervalo $-10..10$ é usado como padrão e o comando é **plot** (**f,x**). Se não fornecermos a variação de **y**, ela será deduzida a partir dos valores calculados.

Vamos obter gráfico da função $f(x) = x^3 - 3x - 2$ no domínio $[-2, 3]$.

> **plot**($x^3-3*x-2, x=-2..3$);



Se clicarmos na região do gráfico, a terceira barra do menu se altera fornecendo novas opções para a exploração do gráfico. Por padrão, o Maple fornece gráficos em escalas que nos proporciona uma visualização apropriada e, em geral, utiliza escalas diferentes nos eixos.

Clicando no ícone **1:1** (no ambiente do Maple) obtemos o gráfico na escala 1 para 1. Isto pode ser feito também, usando a opção **scaling=constrained**.

O gráfico do segmento de reta ligando os pontos (x_0, y_0) e (x_1, y_1) é obtido empregando o comando $[[x_0, y_0], [x_1, y_1]]$.

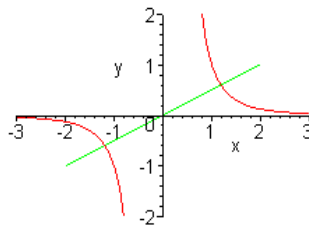
Funções com singularidades podem ser melhor visualizadas se delimitarmos os valores da imagem.

Podemos obter os gráficos de várias funções ao mesmo tempo, utilizando o comando:

plot({gráfico 1, gráfico 2,..gráfico n}, x=a..b, opções).

O próprio sistema nos oferece cores diferentes para os gráficos, o que melhora a sua visualização.

> **plot({1/x^3, [[2, 1], [-2, -1]]}, x=-3..3, y=-2..2, scaling= constrained);**



No pacote **plots** do Maple existe a função **display({g1, g2, ..., gn})** que permite desenhar vários gráficos g_1, g_2, \dots, g_n , na mesma figura, bem como escolher domínios e cores para cada um.

A função **display** não entende gráficos com diferentes eixos ou vários títulos.

Podem também ocorrer erros se misturarmos intervalos finitos e intervalos infinitos.

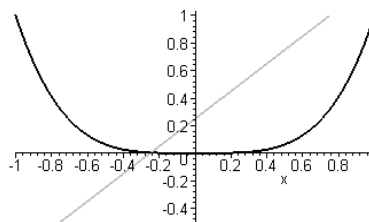
> **with(plots):** (para chamar o pacote de gráficos)

Utilizando a função **display** :

> **g1:=plot(x+1/4, x=-3/4..3/4, color=grey):**

> **g2:=plot(x^4, x=-1..1, color=black):**

> **display({g1, g2}):**



Existem diversos recursos opcionais que podemos usar se quisermos alterar o padrão do sistema. Estão listados abaixo, alguns deles.

color=c (cores): onde **c** representa a cor escolhida . As cores pré-definidas, são: aquamarine, black, blue, navy, coral, cyan, brown, gold, green, gray, grey, khaki, magenta, maroon, orange, pink, plum, red, sienna, tan, turquoise, violet, wheat, white, yellow.

style=s (estilos): onde **s** é point, line ou patch . O padrão é line.

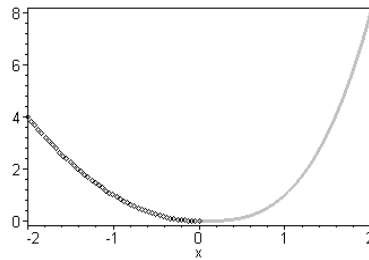
symbol=s: define símbolos a serem usados no lugar dos pontos no desenho; **s** pode ser box, cross, circle, point e diamond.

thickness=n (espessura): define a espessura **n** das linhas no desenho. O padrão é zero.

axes=f: estabelece como os eixos serão desenhados; **f** pode ser normal, frame e boxed. O default é normal.

Exemplo:

- > **with(plots):**
- > **h1:=plot(x^2, x=-2..0, style=point, symbol=diamond, color=magenta):**
- > **h2:=plot(x^3, x=0..2, style=line, thickness=3, color=brown):**
- > **display(h1,h2, axes=boxed):**

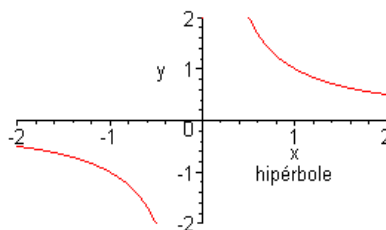


Se quisermos colocar um texto no ponto de coordenadas cartesianas (a,b) do desenho, utilizamos a sintaxe:

textplot([a, b, `texto`])

Exemplo:

- > **with(plots):**
- > **u1:=plot(1/x,x=-2..2, y=-2..2):**
- > **u2:=textplot([1, -1, `hipérbole`]):**
- > **display(u1,u2):**

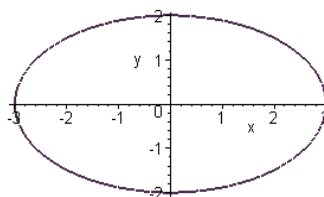


O pacote **plots** contém o comando **implicitplot**, que permite desenhar gráficos de expressões implícitas.

A sintaxe é:

implicitplot (expressão, x=a..b, y=c..d, opções)

- > **with(plots):**
- > **implicitplot(4*x^2+9*y^2=36, x=-3..3, y=-2..2, thickness=2,color=violet):**

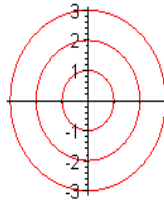


O comando **plot** também pode ser usado para desenhar curvas parametrizadas. Por exemplo, se uma curva é dada por $x = f(t)$, $y = g(t)$, $a \leq t \leq b$, o seu gráfico é obtido por :

plot([f(t), g(t), t=a..b], opções)

Vamos, como exemplo, obter os círculos concêntricos de raios 1, 2 e 3.

- > **plot({ [cos(t), sin(t), t=0..2*Pi] , [2*cos(t), 2*sin(t), t=0..2*Pi] , [3*cos(t), 3*sin(t), t=0..2*Pi]}, color=red, scaling=constrained);**



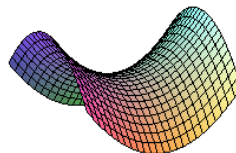
Utilizando o comando **plot3d**, podemos desenhar superfícies em três dimensões.

A sintaxe é:

plot3d(expressão, variação de x, variação de y, opções)

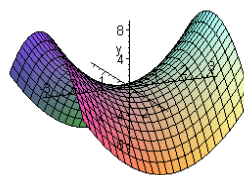
Este comando é similar ao comando **plot**. Clicando sobre a região do gráfico, a terceira barra do menu se modifica e várias opções para a manipulação do gráfico se apresentam.

- > **restart;**
- > **plot3d(y^2-x^2, x=-3..3, y=-3..3);**



Como pode ser visto, o Maple por padrão, apresenta o desenho na forma de rede opaca, sem os eixos. Usando a opção **axes=normal** o sistema de eixos é introduzido.

- > **plot3d(y^2-x^2, x=-3..3, y=-3..3, axes=normal);**



Matrizes

Uma outra maneira de representar matriz, além do comando array, é com a sintaxe:

matrix(m, n, [a₁₁ , ..., a_{1n} , a₂₁ , ..., a_{2n} , ..., a_{m1} , ..., a_{mn}]);

onde *m* é o número de linhas, *n* o número de colunas e a_{i,j} é o elemento linha i coluna j.

Exemplos:

> **A:=matrix(3,4,[1,2,-3,2,2,7,-1,8,3,0,1,-1]);**

$$A := \begin{bmatrix} 1 & 2 & -3 & 2 \\ 2 & 7 & -1 & 8 \\ 3 & 0 & 1 & -1 \end{bmatrix}$$

> **B:=matrix(3,4,[-1,2,-1,3,-2,5,-1,4,0,0,1,-3]);**

$$B := \begin{bmatrix} -1 & 2 & -1 & 3 \\ -2 & 5 & -1 & 4 \\ 0 & 0 & 1 & -3 \end{bmatrix}$$

> **C:=matrix(3,3,[1,2,3,4,5,6,0,1,-1]);**

$$C := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 0 & 1 & -1 \end{bmatrix}$$

> **P:=matrix(2,3,2);** (matriz de ordem 2x3 cujos elementos são todos iguais a 2)

$$P := \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}$$

Para os exemplos que seguem, vamos considerar as matrizes A, B e C definidas acima.

> **evalm(A+B);** (efetua a soma de matrizes)

$$\begin{bmatrix} 0 & 4 & -4 & 5 \\ 0 & 12 & -2 & 12 \\ 3 & 0 & 2 & -4 \end{bmatrix}$$

> **evalm(C&*A);** (efetua o produto)

$$\begin{bmatrix} 14 & 16 & -2 & 15 \\ 32 & 43 & -11 & 42 \\ -1 & 7 & -2 & 9 \end{bmatrix}$$

> **transpose(C);** (transpõe matrizes)

$$\begin{bmatrix} 1 & 4 & 0 \\ 2 & 5 & 1 \\ 3 & 6 & -1 \end{bmatrix}$$

> **det(C);** (retorna o valor do determinante da matriz)

> **inverse(C):** (retorna a inversa)

$$\begin{bmatrix} -\frac{11}{9} & \frac{5}{9} & -\frac{1}{3} \\ \frac{4}{9} & -\frac{1}{9} & \frac{2}{3} \\ \frac{4}{9} & -\frac{1}{9} & -\frac{1}{3} \end{bmatrix}$$

> **A[1,3]:** (extrai o elemento da primeira linha e terceira coluna)

$$-3$$

> **row(A,2):** (extrai a segunda linha de A)

$$[2, 7, -1, 8]$$

> **col(A,2):** (extrai a segunda coluna de A)

$$[2, 7, 0]$$

> **submatrix(A,2..3,1..2):** (extrai a segunda e terceira linhas e primeira e segunda colunas de A)

$$\begin{bmatrix} 2 & 7 \\ 3 & 0 \end{bmatrix}$$

> **stackmatrix(A,B):** (justapõe matrizes como linhas)

$$\begin{bmatrix} 1 & 2 & -3 & 2 \\ 2 & 7 & -1 & 8 \\ 3 & 0 & 1 & -1 \\ -1 & 2 & -1 & 3 \\ -2 & 5 & -1 & 4 \\ 0 & 0 & 1 & -3 \end{bmatrix}$$

> **concat(A,B):** (justapõe matrizes como colunas)

$$\begin{bmatrix} 1 & 2 & -3 & 2 & -1 & 2 & -1 & 3 \\ 2 & 7 & -1 & 8 & -2 & 5 & -1 & 4 \\ 3 & 0 & 1 & -1 & 0 & 0 & 1 & -3 \end{bmatrix}$$

A função **linsolve(A,Y):** do pacote **linalg** determina o vetor X que satisfaz a equação matricial $A \cdot X = Y$. Se a matriz A tem m linhas e n colunas e o vetor Y tem n coordenadas então o vetor X terá m coordenadas, se a solução existir.

Exemplo:

> **A:=matrix(3,3,[1,1,1,2,1,1,1,-1,2]); Y:=matrix(3,1,[1,0,0]);**

$$A := \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ 1 & -1 & 2 \end{bmatrix}$$

$$Y := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

> **X:=linsolve(A,Y):**

$$X := \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

Programação no Maple

O Maple possui uma linguagem de programação de alto nível e compatível com os recursos de sua própria estrutura. Uma forma prática de se construir pequenos programas "executáveis" no Maple são os chamados **procedure** (procedimento).

A sintaxe para se construir procedimentos é a seguinte:

- > **Nome:=proc**(argumentos)
instruções contendo os argumentos
- > **end:**

As variáveis utilizadas na construção de um procedimento podem ser locais ou globais. Os valores das variáveis globais são reconhecidos pelo Maple fora do procedimento.

Se o procedimento não especificar o caráter da variável que está sendo utilizada, o Maple decide sobre isso e avisa o usuário.

Para se escrever expressões na tela, podemos usar o comando **print** (imprimir).

A sintaxe do comando é:

print (expressão 1, expressão 2, etc).

As expressões podem ser valores numéricos ou comentários. Em caso de comentários, esses devem estar entre aspas simples (`).

> **print(`o valor de pi é aproximadamente`, evalf(Pi));**

o valor de pi é aproximadamente 3.141592654

> **u:=3*x;**

u := 3 x

> **print(`o cubo de u é`, u^3);**

o cubo de u é 27 x³

Exemplo:

Estabelecer um procedimento para calcular a soma dos **n** primeiros números inteiros positivos.

- > **soma:=proc(n)**
- > **local s;**
- > **s:= factor(sum(i,i=1..n));**
- > **print(`a soma dos`, n, ` primeiros números inteiros positivos é`, s) ;**
- > **end:**

Definido o procedimento, podemos operar com ele como uma função qualquer.

> **soma(300);**

a soma dos, 300, primeiros números inteiros positivos é 45150

> **soma(n);**

a soma dos, n, primeiros números inteiros positivos é $\frac{1}{2} n (n + 1)$

- **declaração condicional**

Para decisões, o Maple oferece a declaração condicional, cujos operadores lógicos

if (se), **elif** (ou se) **else** (ou então), **then** (então)

apresentam as seguinte sintaxes:

- > **if** condição **then** seqüência de comandos
- > **elif** condição **then** seqüência de comandos
- > **else** seqüência de comandos **fi;**

Vamos usá-las no exemplo seguinte.

Exemplo:

Estabelecer um procedimento para definir uma função que valha -2 para $x \leq 0$, $\frac{4x}{\pi} - 2$ para x entre 0 e $\frac{\pi}{2}$ e $\cos(x)$ para $\frac{\pi}{2} \leq x$.

- > **f:=proc(x)**
- > **if x<=0 then -2**
- > **elif x>0 and x<evalf(Pi/2) then evalf(4*x/Pi - 2)**
- > **else**
- > **cos(x)**
- > **fi;**
- > **end;**

> **f(-1); f(1); evalf(3*sqrt(2));**

-2
-.726760456
4.242640686

- **A declaração repetição**

Durante o desenvolvimento de um algoritmo deparamo-nos muitas vezes com situações onde certa instrução é repetida várias vezes.

Para isso temos o comando **for**.

A sua utilização segue um esquema **for-do-od**, da seguinte forma:

- > **for** variável **from** expressão **to** expressão **do**
expressões a serem repetidas
- > **od;**

O esquema é bastante legível se adotarmos as seguintes traduções: **for** (para), **from** (a partir de), **to** (preposição a) e **do** (faça).

Como exemplo, vamos calcular os cubos de 1, 2, 3, 4, 5, 6.

```
> for j from 1 to 6 do
> j^3
> od;
```

```
1
8
27
64
125
216
```

- **Somas iterativas**

Em processos **iterativos** (recursivos) devemos utilizar o conceito da reatribuição dinâmica de variáveis. Suponhamos, por exemplo, que estamos interessados em somar os números de 1 a 100. Para isso começamos com a soma $s = 0$.

Na primeira etapa fazemos $s = s + 1$ (agora, s vale 1). Na segunda etapa fazemos $s = s + 2$ (agora, s vale 3). Na terceira etapa fazemos $s = s + 3$ (agora, s vale 6). Na quarta etapa fazemos $s = s + 4$ (agora, s vale 10), e assim, sucessivamente.

Ao chegarmos na centésima etapa, obteremos $s = 1 + 2 + 3 + \dots + 100$.

No Maple, esta soma é obtida do seguinte modo:

```
> s:=0:
> for j from 1 to 100 do
> s:=s+j:
> od:
> soma100:=s;
```

```
soma100 := 5050
```

Exemplo: Estabelecer um procedimento para se obter a soma dos n primeiros números ímpares.

```
> si:=proc(n)
> local s, i:
> s:=0:
> for i from 1 by 2 to n do
> s:=s+i:
> od:
> end:
```

A soma dos 50 primeiros números ímpares é:

```
> si(50):
```

```
625
```