Maxima

Daniel Miranda

2 de Junho de 2017

Breve histórico

O Maxima é derivado do sistema Macsyma, desenvolvido no MIT nos anos de 1968 a 1982 como parte do Projecto MAC. O MIT transferiu uma cópia do código fonte do Macsyma para o Departamento de Energia em 1982; essa versão é agora conhecida como Macsyma DOE. Uma cópia do Macsyma DOE foi mantida pelo Professor William F. Schelter da Universidade do Texas, desde 1982 até a sua morte em 2001. Em 1998, Schelter obteve autorização do Departamento de Energia para liberar o código fonte do Macsyma DOE sob a Licença Pública GNU, e em 2000 iniciou o projeto Maxima no SourceForge para manter e desenvolver o Macsyma DOE, agora chamado Maxima.

O que é o Maxima

Segundo os seus criadores:

> *O Maxima é um software para manipulação de expressões simbólicas e numéricas, incluindo diferenciação, integração, expansão em series de Taylor, transformadas de Laplace, resolução de equações diferenciais ordinárias, sistemas de equações lineares, manipulação de vetores, matrizes e tensores. O Máxima também faz gráficos de funções e dados em duas e três dimensões.*

De modo mais geral, o Máxima é um exemplo de **Sistema de Álgebra Computacional**, que abreviaremos por CAS (do inglês *computer algebra system*). Os CAS são softwares que facilitam a realização de manipulações matemáticas simbólicas. Para compreendermos as vantagens de um CAS é importante compreendermos a distinção de **computação simbólica** e **computação númerica**, em computação simbólica as manipulações matemáticas de números, símbolos, expressões e fórmulas, são realizadas de forma **exata**, ao contrário da computação numérica que lida apenas com números de ponto flutuante (e, portanto, aproximações).

Em geral um CAS é possui as seguintes funcionalidades:

- Um sistema de manipulação algébrica capaz simplificar e manipular expressões usando propriedades e fórmulas matemáticas: assim por exemplo um CAS é capaz de realizar operações simbólicas com números e matrizes, expandir e simplificar expressões, resolver equações, diferenciar e integrar funções;
- Um sistema de manipulação numérica capaz de trabalhar com precisão aritmética arbitrária;
- Um sistema gráfico capaz de produzir gráficos de funções e dados em duas ou três dimensões;
- Uma linguagem de programação de alto nível, permitindo aos utilizadores implementar os seus próprios algoritmos;

Operações Aritméticas

Com o Máxima é possível fazer operações aritméticas de modo análogo à uma calculadora:

```
( %i1)

7.53+5.22;

(%o1)      12.75
```

Observe que os comandos do Maxima terminam com ;
No máxima e $*$ e / representam o produto e a divisão:

(%i2)

```
72/22.1;
```

(%o2) 3.257918552036199

No máxima o ^ representa a potência, assim por exemplo 2^{10} pode ser calculado:

(%i3)

```
2^10
```

(%o3) 1024

Para representarmos expressões, formulas e ou funções no Maxima usamos apenas parenteses. Ou seja, nesses casos não se usa chaves e nem colchetes:
Por exemplo, para calcularmos: $(33.1 + 22.4)^{\frac{1}{2}}$

```
( %i4)

(33.1+22.4)^(1/2);

(%o4)      7.44983221287567
```

Por exemplo, para entrarmos a expressão: $(1+(b+1)^3)(1+b)^2$

(%i5)

```
(1+(b+1)^3)*(1+b)^2;
```

(%o5) $(b+1)^2\left((b+1)^3+1\right)$

Vários comandos podem ser inseridos na mesma linha, desde que separados por ponto e virgula:

(%i6)

```
2^1;2^2;2^3;2^4;
```

(%o6) 2

(%o6) 4

(%o6) 8

(%o6) 16

Comentando o Código

Comentários podem ser inseridos nas linhas de entrada do máxima colocando o texto entre /* e */.

```
( %i7)

2+2          /*Esse texto exemplifica a utilização de comentários

(%o7)     4
```

Valores Exatos e Aproximados

O exemplo abaixo mostra que para divisão de inteiros o Máxima apenas simplifica a fração

```
( %i8)
```

```
72/22;
```

(%o8) $\frac{36}{11}$

Se no exemplo anterior, desejarmos uma aproximação numérica para $\frac{36}{11}$ podemos obtê-la usamos o comando para ponto flutuante *float*() como abaixo:

```
( %i9)
```

```
float(72/22);
```

(%o9) 3.272727272727273

O Maxima interpreta a entrada 22.1 como um número real e realiza as contas envolvendo números reais em ponto flutuante:

(%i10)

```
72/22.1;
```

(%o10) 3.257918552036199

Outro modo de obter a representação decimal de um número real é adicionando uma virgula e o comando numer depois do número real:

(%i11)

```
14/17;
```

(%o11) $\frac{14}{17}$

(%i12)

```
14/17, numer;
```

(%o12) 0.82352941176471

Se desejarmos que os resultados e os cálculos efetuados sejam expressos com um determinado número de casas decimais e/ou algarismos significativos podemos estabelecer essa precisão mediante a fixação de um valor à variável interna global fpprec (float point precision, precisão de ponto flutuante), que por padrão no Máxima é 16)
Para calcularmos as primeiras 30 casas decimais de *pi*, primeiro definimos a precisão de ponto flutuante para 30:

(%i13)

```
fpprec:30;
fpprintprec:30;
```

(%o13) 30

E agora usamos o comando %pi para acessar a constante π

(%i14)

```
%pi,numer;
```

(%o14) 3.14159265358979323846264338328*b*0

Constantes Matemáticas

O número π deve ser escrito desta forma %pi.
(%i15)

```
%pi;
```

(%o15) π

De modo análogo temos que o número de Euler e deve ser escrito como **%e** e a constante imaginária $i = \sqrt{(-1)}$ deve ser escrita como **%i**
(%i16)

```
%e,numer;
```

(%o16) 2.718281828459045

Como exemplo calcularemos $e^{\pi * i}$:

```
( %i17)
```

```
%e^(%pi*%i);
```

```
(%o17)      -1
```

Números Complexos podem ser escritos utilizando a constante **%i**. Assim por exemplo podemos calcular o produto entre $2+3i$ e $4+2i$:

(%i18)

```
(3+3* i)*(4+5*i);
```

(%o18) $3 * \%i + 3) * (5 * \%i + 4)$

Para que o Maxima expanda o produto anterior podemos usar o comando **expand**:

(%i19)

```
expand(%);
```

(%o19) $27 * \%i - 3$

A constante ∞ é representada no máxima por **inf** enquanto que $-\infty$ é representado por $-\infty$

Funções Matemáticas

sqrt(x)	$\sqrt{a}$	raiz quadrada de x;
abs(x)	$\|a\|$	módulo de x;
n!	$n!$	n fatorial;
exp(x)	e^a	exponencial de x;
log(x)	$\ln(x)$	logaritmo natural de x;
sin(x)	$\sin(x)$	seno de x (em radianos);
cos(x)	$\cos(x)$	cosseno de x (em radianos);
tan(x)	$\tan(x)$	tangente de x (em radianos);
asin(x)	$\arcsin(x)$	arco-seno de x (em radianos);
acos(x)	$\arccos(x)$	arco-cosseno de x (em radianos);

Para calcularmos 12!

(%i20)

```
12!
```

(%o20) 479001600

Para calcularmos π^2:

(%i21)

```
float(%pi^2);
```

(%o21) 9.869604401089358

P ara calcularmos $\cos\left(\frac{\pi}{4}\right)$:

```
( %i22)
```

```
cos(%pi/4);
```

(%o22) $\frac{1}{\sqrt{2}}$

O Maxima não possui uma função pré-definida para logaritmo de base 10 ou de outras bases. Podemos definir o logaritmo na base 10 através do comando

(%i23)

```
log10(x) := log(x) / log(10);
```

(%o23) $\log10(x) := \frac{\log(x)}{\log(10)}$

Agora podemos calcular $\log_1 0(100)$:

```
( %i24)

log10(100),numer;

(%o24)      2.0
```

Usando os Resultados Anteriores

Para convertemos a saída anterior para um valor numérico podemos utilizar o comando float(%). Nesse caso o símbolo % nos permite o último resultado apresentado: **(%o24)** $\frac{1}{\sqrt{2}}$

(%i25)

```
float(%);
```

(%o25) 0.70710678118655

Observação

Para sermos precisos o símbolo % é a variável de sistema que armazena a expressão de saída (por exemplo, %o1, %o2, %o3, ...) mais recentemente calculada pelo Maxima. Podemos acessar a enésima saída usando a variável %o(n)

```
( %i26)

sqrt(2+4^3);
```

(%o26) $\sqrt{66}$

```
( %i27)

5*cos(3)
```

(%o27) $5\cos(3)$

Agora podemos calcular $\sqrt{66} + 5\cos(3)$ através do comando:

```
( %i28)
```

```
%o11+%o12
```

(%o28) $\sqrt{66} + 5\cos(3)$

E podemos converter para ponto flutuante como:

```
( %i29)
```

```
float(%)
```

(%o29) 3.174075921633734

Definindo Variáveis

Uma ferramenta importante no Maxima é a capacidade de atribuir e manipular variáveis. Uma variável, em programação, é um identificador ao qual se pode atribuir valores. No Maxima a instrução de atribuição concretiza-se empregando o símbolo :

```
( %i30)

x:3;

(%o30)      3
```

Para visualizar o valor da variável x, utilizamos o comando print.

```
( %i31)

print(x);

(%o31)     3
```

Vamos definir $y = 5$ e calcular $x + y$ e $x^y + y^x$

```
( %i32)
```

```
y:5;
```

```
(%o32)      5
```

```
( %i33)
```

```
x+y;
```

```
(%o33)      8
```

```
( %i34)
```

```
x^y+y^x;
```

```
(%o34)      368
```

Se desejarmos remover um valor atribuído a uma variável podemos fazer isso através do comando **kill**: Esse comando sempre retorna *done* (i.e. feito);
Para limpar a variável x definida na entrada **??**:

```
( %i35)

kill{x};

(%o35)      done
```

Agora, se retornarmos o valor da variável x, utilizamos o comando print, teremos

```
( %i36)

print(x);

(%o36)      x
```

Uma variável pode armazenar expressões. Assim por exemplo podemos armazenar a expressão $\pi a^2 h$ na variável V através do comando

```
( %i37)

V:%pi*a^2*h;

(%o37)      π a^2 h
```

Usando o caractere $ podemos evitar que o máxima exiba uma saída: Assim por exemplo se queremos armazenar a expressão $\pi a^2 h$ na variável V, evitando que o Maxima retorne uma saída podemos fazer isso através do comando:

```
( %i38)
```

```
V:%pi*a^2*h$;
```

E nesse caso não a saída ?? não será exibida.

Uma variável pode também armazenar uma igualdade ou uma desigualdade. Se queremos armazenar a desigualdade $3a + 5b < 3$ para usos futuros:

```
( %i39)
```

```
desigualdade:3*a+5*b<3
```

(%o39) $3 * a + 5 * b < 3$

Somátorios e Produtórios

Para calcular $\sum_{n=1}^{j} a_n$ usamos o

$$\sum_{n=1}^{10} \frac{1}{2^n}$$

```
( %i1)
 sum(1/2^n,n, 1, 10);
(%o1)      1023/1024
( %i2)
 float(%);
(%o2)      0.9990234375
```

O máxima também aceita somas infinitas:

(%i3)

```
sum(1/x^2, x, 1, inf);
```

(%o3) $\sum_{x=1}^{\infty} \frac{1}{x^2}$

Para que o máxima simplifique o somatório, temos que adicionar a opção **simpsum=true**:

(%i4)

```
sum(1/x^2, x, 1, inf),simpsum=true;
```

(%o4) $\frac{\pi^2}{6}$

O máxima é manipula somatórios, mas em geral não os contraí:
(%i5)

```
soma1: sum(1/x^2, x, 4, inf)$
soma2:sum(1/x^3, x, 2, inf) $
soma1+soma2;
```

(%o5) $\left(\sum_{x=4}^{\infty} \frac{1}{x^2}\right) + \sum_{x=2}^{\infty} \frac{1}{x^3}$

Para que o maxima agrupe os somatórios usamos o comando **sumcontract**

(%i6)

```
sumcontract(%);
```

(%o6) $\left(\sum_{x=4}^{\infty} \frac{1}{x^2} + \frac{1}{x^3}\right) + \frac{35}{216}$

E para calcular o valor do somatório anterior:

(%i7)

```
%,simpsum=true;
```

(%o7) $\zeta(3) + \frac{\pi^2}{6} - \frac{85}{36}$

O próxima exemplo nos retorna as fórmulas fechadas para a soma dos n primeiros números , dos quadrados dos n primeiros números, dos cubos, e da quarta potência. O comando **simpsum=true**

(%i8)

```
[sum(i,i,1,n), sum(i^2,i,1,n), sum(i^3,i,1,n),sum(i^4,i,1,n)];
```

(%o8) $$\left[\frac{n^2+n}{2}, \frac{2\,n^3+3\,n^2+n}{6}, \frac{n^4+2\,n^3+n^2}{4}, \frac{6\,n^5+15\,n^4+10\,n^3-n}{30}\right]$$

Podemos também calcular produtórios

(%i9)

```
product(a+i, i, 1,5);
```

(%o9) $(a+1)\,(a+2)\,(a+3)\,(a+4)\,(a+5)$

Definindo uma lista com os elementos $1, 3, 5, 7, 9$ nessa ordem:

```
( %i1)

[1,3,5,7,9];

(%o1)      [1, 3, 5, 7, 9]
```

Elevando todos os elementos da lista anterior ao quadrado:

(%i2)

```
[1,3,5,7,9]^2;
```

(%o2) [1,9,25,49,81]

Somando os elementos de duas listas de mesmo tamanho:

(%i3)

```
[2,4,6,8]+[3,5,14,22];
```

(%o3) [5, 9, 20, 30]

Atribuindo uma lista a uma variável:
(%i4)

```
lista:[2,4,6,8]\$;
```

(Note que ocultamos a saída, utilizando o carácter $
Somando 3 a cada elemento da lista:
(%i5)

```
lista+3;
```

(%o5) $[5, 7, 9, 11]$

Calculando $\frac{a+1}{a^2}$ para cada elemento da lista

(%i6)

```
(lista+1)/lista^2;
```

(%o6) $[\frac{3}{4}, \frac{5}{16}, \frac{7}{36}, \frac{9}{64}]$

Calcular $sen(x)$ para os elementos da lista

(%i7)

```
sin(lista);
```

(%o7) $[sin(2), sin(4), sin(6), sin(8)]$

Convertendo os valores da saída anterior para ponto flutuante:

```
( %i8)

%,numer;

(%o8)
[0.90929742682568, −0.75680249530793, −0.27941549819893, 0.9893582466
```

Para criar uma lista onde os elementos são definidos por uma expressão, podemos usar o comando makelist.
Para criar a lista onde os elementos são da forma $\frac{1}{n}$ com n de 1 até 8:
(%i9)

```
makelist(1/n,n,1,8);
```

(%o9) $[1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8]$

Para criar uma lista com os 10 primeiros impares:
(%i10)

```
makelist(2*n+1,n,0,10);
```

(%o10) [1,3,5,7,9,11,13,15,17,19,21]

Para criar a lista onde os elementos são da forma $\frac{n}{n+1}$ com n de 1 até 5:

(%i11)

```
makelist(n/(n+1),n,1,5);
```

(%o11) [1/2,2/3,3/4,4/5,5/6]

Podemos criar uma lista com os valores:

$$\left[\sum_{n=1}^{1}\frac{1}{2^n},\sum_{n=1}^{2}\frac{1}{2^n},\sum_{n=1}^{3}\frac{1}{2^n},\ldots,\sum_{n=1}^{10}\frac{1}{2^n}\right]$$

(%i12)

```
makelist(sum(1/2^n, n, 1, k),k,1,10);
```

(%o12) $[\frac{1}{2},\frac{3}{4},\frac{7}{8},\frac{15}{16},\frac{31}{32},\frac{63}{64},\frac{127}{128},\frac{255}{256},\frac{511}{512},\frac{1023}{1024}]$

Manipulando Elementos de uma Lista

Vamos começar criando uma lista:

(%i2)

```
lista:makelist(n^2,n,1,7);
```

(%o2) [1,4,9,16,25,36,49]

Os elementos de uma lista são indexados a partir de um. Para retornar o enésimo elemento de uma lista usamos o comando lista[n].

ssim por exemplo, para retornar o primeiro elemento da lista anterior:

```
( %i3)

 lista[1];

(%o3)     1
```

Para retornar o quarto elemento da lista anterior:

```
( %i4)

 lista[4];

(%o4)     16
```

O comando **rest(n)** cria uma nova lista com os *n* primeiros elementos de uma lista removidos:

```
( %i5)

 rest (lista, 3);

(%o5)      [16,25,36,49]
```

O comando **length (lista);** retorna o comprimento de uma lista:

```
( %i6)

 length (lista);

(%o6)     7
```

O comando **append(lista1, lista2)** cria uma nova lista incluindo todos os elementos de lista1 e lista2 seguidos, incluindo os elementos da lista mais longa.

```
(%i7)
 lista1:[a,b,c,d]; lista2:[1,2,3,4,5,6];
```

(%o7) $[a, b, c, d]$

(%o7) $[1, 2, 3, 4, 5, 6]$

```
(%i8)
 append(lista1,lista2);
```

(%o8) $[a, b, c, d, 1, 2, 3, 4, 5, 6]$

os comandos **cons** e **endcons** permitem adicionar novos elementos a uma lista, no início e final de mesma, respectivamente;

```
( %i9)
 cons(X,lista1);
```

(%o9) $[X, a, b, c, d]$

```
( %i10)
 endcons(X,lista1);
```

(%o10) $[a, b, c, d, X]$

O comando **delete** apaga todas as ocorrências da entrada em uma lista

```
( %i11)

delete(a, [a,b,c,a,a,b]);

(%o11)      [b, c, b]
```

É possível realizar operações elementares, tais como a soma, subtração, multiplicação, com os elementos de uma lista:
Vamos calcular a soma dos 20 primeiros pares
(%i12)

```
pares:makelist(2*n,n,1,20);
```

(%o12)
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40]

Para somar os elementos da lista usamos o comando **apply**
(%i13)

```
apply("+", pares)
```

(%o13) 420

Para calcular o produto dos vinte primeiros pares:

```
( %i14)

apply("*", pares)

(%o14)     255108265612582846464000000
```

Para aplicar uma função a cada elemento de uma lista utilizamos o comando **map**:
Para calcular a raiz quadrada dos elementos da lista $[1, 6, 9, 13]$:
(%i15)

```
map(sqrt,[1,6,9,13]);
```

(%o15) $[1, \sqrt{6}, 3, \sqrt{13}]$

Assim por exemplo se $f(x) = x + sin(x)$ e queremos calcular os valores de $f(x)$ para os elementos da lista $[1, 2, 5, 7]$:
Começamos definindo a função $f(x)$ (falaremos mais sobre como definir funções no capítulo **??**
(%i16)

```
f(x):=x+sin(x);
```

(%o16) $f(x) := x + sin(x)$

Agora aplicamos $f(x)$ a lista
(%i17)

```
map(f,[1,2,5,7]);
```

(%o17) $[\sin(1) + 1, \sin(2) + 2, \sin(5) + 5, \sin(7) + 7]$

Matrizes

No Maxima define uma matriz do seguinte modo **A: matrix (lista1,lista2,..,listan);**
Vamos definir a matriz

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & -1 & 0 \\ 4 & 2 & 1 \end{pmatrix}$$

(%i2)

```
A:matrix([1,2,3],[3,-1,0],[4,2,1]);
```

(%o2) $\begin{pmatrix} 1 & 2 & 3 \\ 3 & -1 & 0 \\ 4 & 2 & 1 \end{pmatrix}$

Vamos definir outra matriz

(%i3)

```
B: matrix ([1, -1, 2], [2, 1, 5], [0, 1, 3]);
```

(%o3) $$\begin{pmatrix} 1 & -1 & 2 \\ 2 & 1 & 5 \\ 0 & 1 & 3 \end{pmatrix}$$

Podemos calcular a soma $A + B$ e o produto AB

(%i4)

```
A+B;A.B
```

(%o4) $$\begin{pmatrix} 2 & 1 & 5 \\ 5 & 0 & 5 \\ 4 & 3 & 4 \end{pmatrix}$$

(%o4) $$\begin{pmatrix} 1 & -2 & 6 \\ 6 & -1 & 0 \\ 0 & 2 & 3 \end{pmatrix}$$

Cuidado o comando $A * B$ calcula o produto coordenada a coordenada e não o produto de matrizes:

```
( %i5)

C:matrix([2,1],[2,0]);

( %i6)

C.C; C*C;
```

$$(\%o6) \quad \begin{pmatrix} 6 & 2 \\ 4 & 2 \end{pmatrix}$$

$$(\%o6) \quad \begin{pmatrix} 4 & 1 \\ 4 & 0 \end{pmatrix}$$

A Matriz identidade $n \times n$ pode ser facilmente definida usando o comando **ident(n)**

(%i7)

```
id:ident(3);
```

(%o7)
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Se uma matriz for invertível, sua inversa pode ser calculada através do comando **invert**

(%i8)

```
invert (A);
```

(%o8)
$$\begin{pmatrix} -\frac{1}{23} & \frac{4}{23} & \frac{3}{23} \\ -\frac{3}{23} & -\frac{11}{23} & \frac{9}{23} \\ \frac{10}{23} & \frac{6}{23} & -\frac{7}{23} \end{pmatrix}$$

A transposta de uma matriz pode ser calculada através do comando **transpose**

(%i9)

```
transpose (A);
```

(%o9)
$$\begin{pmatrix} 1 & 3 & 4 \\ 2 & -1 & 2 \\ 3 & 0 & 1 \end{pmatrix}$$

O determinante de uma matriz pode ser calculado atraves do comando **determinant**

(%i10)

```
determinant(A);
```

(%o10) 23

A entrada $(2,3)$ da matriz A pode ser obtido através do comando:

```
( %i11)
```

```
A[2,3];
```

```
(%o11)      0
```

A terceira linha da matriz A pode ser obtida através do comando:

```
( %i12)
```

```
row(A, 3);
```

```
(%o12)      (4   2   1)
```

A primeira coluna de uma matriz pode ser obtida através do comando:

```
( %i13)
```

```
col(A,3);
```

$$(\%o13) \qquad \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix}$$

Uma matriz pode ser escalonada através do comando **triangularize(M)**

(%i14)

```
M:matrix([1,2,3,4],[5,6,7,z],[b,1,2,3]);
```

(%o14)
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & z \\ b & 1 & 2 & 3 \end{pmatrix}$$

(%i15)

```
triangularize(M);
```

(%o15)
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & -4 & -8 & z-20 \\ 0 & 0 & -4\,b & (2\,b-1)\,z-24\,b+8 \end{pmatrix}$$

Uma matriz pode ser escalonada com 1 nos pivôs através do comando **echelon(M)**

(%i16)

```
echelon(M);
```

(%o16)
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & -\frac{z-20}{4} \\ 0 & 0 & 1 & -\frac{(2\,b-1)\,z-24\,b+8}{4\,b} \end{pmatrix}$$