符号运算软件Maxima简介

作者：Richard H. Rand
翻译：dbzhang

目录

1 简介

在Linux 中运行Maxima，只需键入

maxima<回车>

计算机将显示如下的欢迎词:

```
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function bug_report()
provides bug reporting information.
(%i1)
```

此处(%i1)是一个“标签”。每一个输入或输出行都贴有一个标签，每一行都可以在随后的会话中通过标签被调用。标签i代表该行是你输入的命令，标签o代表该行为机器的响应。永远不要尝试使用形如 %i1 或 %o5 的变量名，那将会和采用该标签的行相混淆。

Maxima 对字符的大小写是敏感的。所有内建函数的函数名都是小写的（sin, cos, save, load,等）。内建的常数采用小写形式（%e, %pi, inf, 等）。如果你键入SIN(x)或者Sin(x)，
Maxima 认为你指代其他的函数而不是内建的sin 函数。用户自定义函数和变量可以采用大写或小写的形式。注意foo(XY), Foo(Xy), FOO(xy) 是不同的函数。

2 特殊键和符号

1. 要结束一个Maxima会话，键入quit();
2. 要终止一次计算而不退出Maxima，键入^C。（这儿^ 代表Ctrl键，因此^C 意味着先按住Ctrl键，然后再按下C。）了解这一点在有些情况下对你是很重要的，比如，在一次计算需要耗费太长时间的时候。举例如下：

```
(%i1) sum (1/x^2, x, 1, 10000);

Maxima encountered a Lisp error:

 Console interrupt.

Automatically continuing.
To reenable the Lisp debugger set *debugger-hook* to nil.
(%i2)
```

3. 要告诉Maxima你已经完成了命令的输入，键入分号(;)并回车。注意到单独一个回车并不代表输入的结束。
4. 另一个可以代替分号(;) 的终止符是一个美元符号($)，而且，它可以使Maxima不回显计算结果，当你在进行一次有着很长结果的计算，并且你不想浪费时间显示结果的时候，这会很有用。
5. 如果你想重复一条你已经给出的命令，比如说在(%i5) 行，你可以在上述的行号前加两个单引号(") 的方法来避免再次输入，比如，"%i5。（注意这只是简单地输入(%i5) 行，而不是把它再做一次- 试试吧。）

6. 如果你想引用Maxima上一步计算的结果，你可以用它的o 标签，也可以使用专门的百分号(%)。

7. 标准量e (自然对数的底数)，i (-1 的平方根) 和p (3.14159?) 分别表示成%e，%i，和%pi。注意这里% 只是作为一个前缀使用, 与用% 来查询先前计算结果的用法完全无关。

8. 为了把一个值赋给一个变量，Maxima使用冒号(:)，而不是等号。等号被用来表示方程或等式。

3 算术

常见的算术操作符有:

+ 加法

- 减法

***** 标量乘法

/ 除法

^或** 幂运算

. 矩阵乘法

sqrt(x) x的平方根

Maxima输出的特点是严格的算术（有理）运算。例如：

```
(%i1) 1/100 + 1/101;
                                201
(%o1)                           -----
                                10100
```

如果计算中涉及无理数，它们将保持符号形式：

```
(%i2) (1 + sqrt(2))^5;
                                        5
(%o2)                       (sqrt(2) + 1)
(%i3) expand (%);
(%o3)                      29 sqrt(2) + 41
```

尽管如此,将计算结果用小数显示出来往往是有用的。这可以通过在你想要展开的表达式后面加上“ ,numer”来实现。

```
(%i4) %, numer;
(%o4)                    82.01219330881976
```

注意在这里使用了% 来引用上一步的结果. 在这个版本的Maxima 里, numer 给出16 位的有效数字, 但最后一位往往是不可靠的. 尽管如此, Maxima 可以通过使用bfloat函数来提供任意高的的精度：

```
(%i5) bfloat (%o3);
(%o5)                  8.201219330881976B1
```

显示的有效数字的位数由Maxima的变量fpprec控制，它的缺省值是16：

```
(%i6) fpprec;
(%o6)                          16
```

在这儿我们重置fpprec 以产生100个有效数字：

```
(%i7) fpprec: 100;
(%o7)                          100
(%i8) ''%i5;
(%o8) 8.20121933088197564152489730020812442785204843859314941221\
23712401731241875401104126661238495501605 61B1
```

注意在(%i8)中用两个单引号(")来重复命令(%i5)。Maxima可以处理非常大的数而不用近似值：

```
(%i9) 100!;
(%o9) 93326215443944152681699238856266700490715968264381621468 59\
29638952175999932299156089414639761565182862536979208272237582 51\
18521091686400000000000000000000000
```

4 代数

当我们看到Maxima 在代数上的便利后，作为一个使符号运算的计算机工具，它的重要性就变得更明显了。这里有一个分解多项式的例子：

```
(%i1) (x + 3*y + x^2*y)^3;
                              2             3
(%o1)                      (x  y + 3 y + x)
(%i2) expand (%);
       6  3      4  3       2  3       3      5  2       3  2
(%o2) x  y  + 9 x  y  + 27 x  y  + 27 y  + 3 x  y  + 18 x  y
                                          2      4        2      3
                                  + 27 x y  + 3 x  y + 9 x  y + x
```

现在假定我们想把上面表达式中的x用5/z 替代

```
(%i3) %o2, x=5/z;
           2        3                  2                3
      135 y    675 y    225 y   2250 y    125   5625 y    1875 y
(%o3) ------ + ------ + ----- + ------- + --- + ------- + ------
        z          2        2         3       3       4          4
                  z        z         z       z       z          z
                                                2          3
                                          9375 y    15625 y         3
                                       + ------- + -------- + 27 y
                                            5          6
                                           z          z
```

Maxima 内建函数ratsimp 用来进行通分操作：

```
(%i4) ratsimp (%);
           3  6        2  5          3           4
(%o4) (27 y  z  + 135 y  z  + (675 y  + 225 y) z
          2          3          3             2          2
 + (2250 y  + 125) z  + (5625 y  + 1875 y) z  + 9375 y  z
          3   6
 + 15625 y )/z
```

表达式也可以使用factor进行因式分解：

```
(%i5) factor (%);
                              2              3
                      (3 y z  + 5 z + 25 y)
```

```
(%o5)                    ----------------------
                                      6
                                     z
```

Maxima可以求解非线性代数方程系统的精确解。在这个例子中，我们使用函数solve解一个三元方程组，三个未知数分别为a, b, c:

```
(%i6) a + b*c = 1;
(%o6)                         b c + a = 1
(%i7) b - a*c = 0;
(%o7)                         b - a c = 0
(%i8) a + b = 5;
(%o8)                          b + a = 5
(%i9) solve ([%o6, %o7, %o8], [a, b, c]);
             25 sqrt(79) %i + 25       5 sqrt(79) %i + 5
(%o9) [[a = -------------------, b = -----------------,
             6 sqrt(79) %i - 34        sqrt(79) %i + 11
     sqrt(79) %i + 1         25 sqrt(79) %i - 25
c = ---------------], [a = -------------------,
           10                6 sqrt(79) %i + 34
     5 sqrt(79) %i - 5         sqrt(79) %i - 1
b = -----------------, c = - ---------------]]
     sqrt(79) %i - 11                10
```

注意上面的显示由一个“列表”组成，也就是说，由两个方括号[...]括住的一些表达式，它本身还包含两个列表，每个列表包含方程组的一组解。Maxima可以轻松处理三角问题，函数trigexpand利用sum-angles公式使得每个三角函数的参数尽可能简单。

```
(%i10) sin(u + v) * cos(u)^3;
                              3
(%o10)                  cos (u) sin(v + u)
(%i11) trigexpand (%);
                 3
(%o11)        cos (u) (cos(u) sin(v) + sin(u) cos(v))
```

与此相反，函数trigreduce把一个表达式转换成几项和的形式，每一项只含有1个sin或cos:

```
(%i12) trigreduce (%o10);
       sin(v + 4 u) + sin(v - 2 u)   3 sin(v + 2 u) + 3 sin(v)
(%o12) --------------------------- + -----------------------
                    8                             8
```

函数realpart 和imagpart 返回一个复数表达式的实部和虚部

```
(%i13) w: 3 + k*%i;
(%o13)                        %i k + 3
(%i14) w^2 * %e^w;
                                 2   %i k + 3
(%o14)                (%i k + 3)  %e
(%i15) realpart (%);
                 3       2                3
(%o15)         %e  (9 - k ) cos(k) - 6 %e  k sin(k)
```

5 微积分

Maxima可以计算导数和积分，按Taylor 级数展开，求极限和常微分方程的精确解。我们从把符号f 定义为如下的x 的函数开始：

```
(%i1) f: x^3 * %e^(k*x) * sin(w*x);
                          3   k x
(%o1)                    x  %e    sin(w x)
```

我们计算f 对自变量x 的导数：

```
(%i2) diff (f, x);
         3   k x               2   k x
(%o2) k x  %e    sin(w x) + 3 x  %e    sin(w x)
                                                    3   k x
                                               + w x  %e    cos(w x)
```

现在我们求f 对x 的不定积分：

```
(%i3) integrate (f, x);
            6      3  4      5  2    7   3
(%o3) (((k w  + 3 k  w  + 3 k  w  + k ) x
```

```
       6      2  4      4  2      6   2
 + (3 w  + 3 k  w  - 3 k  w  - 3 k ) x
            4       3  2      5         4       2  2      4
 + (- 18 k w  - 12 k  w  + 6 k ) x - 6 w  + 36 k  w  - 6 k )
   k x                  7      2  5      4  3    6     3
 %e    sin(w x) + ((- w  - 3 k  w  - 3 k  w  - k  w) x
         5       3  3      5     2
 + (6 k w  + 12 k  w  + 6 k  w) x
       5       2  3       4              3       3      k x
 + (6 w  - 12 k  w  - 18 k  w) x - 24 k w  + 24 k  w) %e
             8      2  6      4  4      6  2    8
 cos(w x))/(w  + 4 k  w  + 6 k  w  + 4 k  w  + k )
```

稍微改变一下语法可以得到定积分：

```
(%i4) integrate (1/x^2, x, 1, inf);
(%o4)                           1
(%i5) integrate (1/x, x, 0, inf);

Integral is divergent
 -- an error.  Quitting.  To debug this try debugmode(true);
```

下面我们使用符号f (在%i1中已定义)和双曲正弦函数来定义符号g，并在x = 0 点展开为泰勒级数（高达3阶）：

```
(%i6) g: f / sinh(k*x)^4;
                         3   k x
                        x  %e    sin(w x)
(%o6)                   -----------------
                               4
                          sinh (k x)
(%i7) taylor (g, x, 0, 3);
                        2    3   2          2    3   3
         w    w x   (w k  + w ) x    (3 w k  + w ) x
(%o7)/T/ -- + --- - -------------- - ---------------- + . . .
          4    3            4                  3
         k    k          6 k                6 k
```

当x 趋向于0时g 的极限计算如下：

```
(%i8) limit (g, x, 0);
                                  w
(%o8)                             --
                                   4
                                  k
```

Maxima也允许导数已非计算形式表示（注意引号）：

```
(%i9) 'diff (y, x);
                                  dy
(%o9)                             --
                                  dx
```

(%i9) 中的引号操作符表示（不求值）。没有它的话，Maxima将得到0：

```
(%i10) diff (y, x);
(%o10)                             0
```

使用引号操作符我们可以写微分方程：

```
(%i11) 'diff (y, x, 2) + 'diff (y, x) + y;
                            2
                           d y   dy
(%o11)                     --- + -- + y
                             2   dx
                           dx
```

Maxima的ode2 函数可以求解一阶和二阶的常微分方程：

```
(%i12) ode2 (%o11, y, x);
              - x/2          sqrt(3) x             sqrt(3) x
(%o12) y = %e       (%k1 sin(---------) + %k2 cos(---------))
                              2                     2
```

6 矩阵运算

Maxima 可以计算行列式，以及带符号元素(也就是说，带有代数变量的元素)的矩阵的逆, 特征值和特征向量。我们从一个元素一个元素地输入一个矩阵m开始：

```
(%i1) m: entermatrix (3, 3);

Is the matrix  1. Diagonal  2. Symmetric  3. Antisymmetric  4. General
Answer 1, 2, 3 or 4 :
4;
Row 1 Column 1:
0;
Row 1 Column 2:
1;
Row 1 Column 3:
a;
Row 2 Column 1:
1;
Row 2 Column 2:
0;
Row 2 Column 3:
1;
Row 3 Column 1:
1;
Row 3 Column 2:
1;
Row 3 Column 3:
0;

Matrix entered.
                           [ 0  1  a ]
                           [         ]
(%o1)                      [ 1  0  1 ]
                           [         ]
                           [ 1  1  0 ]
```

下面我们求它的转置，行列式和逆矩阵：

```
(%i2) transpose (m);
                           [ 0  1  1 ]
                           [         ]
```

```
(%o2)                          [ 1  0  1 ]
                               [         ]
                               [ a  1  0 ]
(%i3) determinant (m);
(%o3)                             a + 1
(%i4) invert (m), detout;
                             [ - 1   a    1  ]
                             [               ]
                             [  1   - a   a  ]
                             [               ]
                             [  1    1   - 1 ]
(%o4)                        -----------------
                                   a + 1
```

在(%i4)中，修饰符detout 将行列式的值保持在逆矩阵元素的外边。作为检验，我们用矩阵m 乘以它的逆（注意这儿用小数点表示矩阵的乘法）：

```
(%i5) m . %o4;
                                  [ - 1   a    1  ]
                                  [               ]
                                  [  1   - a   a  ]
                  [ 0  1  a ]     [               ]
                  [         ]     [  1    1   - 1 ]
(%o5)             [ 1  0  1 ] . -----------------
                  [         ]          a + 1
                  [ 1  1  0 ]
(%i6) expand (%);
          [   a       1                                  ]
          [ ----- + -----         0              0       ]
          [ a + 1   a + 1                                ]
          [                                              ]
          [                   a       1                  ]
(%o6)     [       0         ----- + -----         0       ]
          [                 a + 1   a + 1                ]
          [                                              ]
          [                                 a       1    ]
```

```
         [       0               0        ----- + -----  ]
         [                                a + 1   a + 1  ]
(%i7) factor (%);
                             [ 1  0  0 ]
                             [         ]
(%o7)                        [ 0  1  0 ]
                             [         ]
                             [ 0  0  1 ]
```

要求得矩阵m 的特征值和特征向量，我们使用函数eigenvectors：

```
(%i8) eigenvectors (m);
           sqrt(4 a + 5) - 1  sqrt(4 a + 5) + 1
(%o8) [[[- -----------------, -----------------, - 1],
                  2                  2
                 sqrt(4 a + 5) - 1    sqrt(4 a + 5) - 1
[1, 1, 1]], [1, - -----------------, - -----------------],
                       2 a + 2              2 a + 2
    sqrt(4 a + 5) + 1  sqrt(4 a + 5) + 1
[1, -----------------, -----------------], [1, - 1, 0]]
         2 a + 2            2 a + 2
```

在%o8中，第一个元组（triple）给出了m的特征值，第二个给出了它们各自的重数（此处都是不重复的）。下面的三个元组给出了m 相应的特征向量。为了从这些表达式中提取一个特征向量，我们使用part 函数：

```
(%i9) part (%, 2);
                sqrt(4 a + 5) - 1    sqrt(4 a + 5) - 1
(%o9)     [1, - -----------------, - -----------------]
                      2 a + 2              2 a + 2
```

7 Maxima编程

到现在为止，我们已经在交互模式下用过了Maxima，就象个计算器一样，然而，对于那些扯进了反复控制次序的计算，还是运行一个程序来得方便。这里我们展示了一个短且简单的程序，用来计算一个有着两个变量x 和y 的函数f 的临界点。这个程序提示用户输入函数f，然后它

计算f_x 和f_y 的偏导数，然后，它使用Maxima 命令`solve` 去获得$f_x = f_y = 0$的解。这个程序是在Maxima 之外用一个文本编辑器写得，然后用`batch` 命令装载进Maxima。下面是程序列表：

```
/* ---------------------------------------------------------------------
   this is file critpts.max:
   as you can see, comments in maxima are like comments in C

   Nelson Luis Dias, nldias@simepar.br
   created 20000707
   updated 20000707
   --------------------------------------------------------------------- */
critpts():=(
   print("program to find critical points"),
/* ---------------------------------------------------------------------
   asks for a function
   --------------------------------------------------------------------- */
   f:read("enter f(x,y)"),
/* ---------------------------------------------------------------------
   echoes it, to make sure
   --------------------------------------------------------------------- */
   print("f = ",f),
/* ---------------------------------------------------------------------
   produces a list with the two partial derivatives of f
   --------------------------------------------------------------------- */
   eqs:[diff(f,x),diff(f,y)],
/* ---------------------------------------------------------------------
   produces a list of unknowns
   --------------------------------------------------------------------- */
   unk:[x,y],
/* ---------------------------------------------------------------------
   solves the system
   --------------------------------------------------------------------- */
   solve(eqs,unk)
)$
```

这个程序(实际上是个没有参数的函数) 叫做critpts。每一行都是一个有效的。可以由键盘上执行的Maxima 命令。它们由逗号隔开。偏导数被贮存在一个叫做eqs 的变量中。未知数贮存在unk 中。这里是它运行的例子：

```
(%i1) batch ("critpts.max");

batching #p/home/robert/tmp/maxima-clean/maxima/critpts.max
(%i2) critpts() := (print("program to find critical points"),
f : read("enter f(x,y)"), print("f = ", f),
eqs : [diff(f, x), diff(f, y)], unk : [x, y], solve(eqs, unk))
(%i3) critpts ();
program to find critical points
enter f(x,y)
%e^(x^3 + y^2)*(x + y);
                 2    3
                y  + x
f =  (y + x) %e
(%o3) [[x = 0.4588955685487 %i + 0.35897908710869,
y = 0.49420173682751 %i - 0.12257873677837],
[x = 0.35897908710869 - 0.4588955685487 %i,
y = - 0.49420173682751 %i - 0.12257873677837],
[x = 0.41875423272348 %i - 0.69231242044203,
y = 0.4559120701117 - 0.86972626928141 %i],
[x = - 0.41875423272348 %i - 0.69231242044203,
y = 0.86972626928141 %i + 0.4559120701117]]
```

8 Maxima函数的不完全列表

更详细的说明可查看Maxima的参考手册(Maxima安装目录)/doc/html/maxima_toc.html。在程序Maxima运行时，你也可以使用describe(function name) 来查看某一函数的说明。

allroots(a) 求解多项式方程a所有的（复数）根，并把它们以数值格式（i.e. 采用16位有效数字）列出来。

append(a,b) 将列表b追加到列表a，产生一个单一列表。

batch(a) 加载并运行一个文件名为a的程序。

coeff(a,b,c) 给出表达式a中b的c次方项的系数。

concat(a,b) 生成符号ab，比如concat(y,44)的结果为y44。

cons(a,b) 将a加入列表b的头部。

demoivre(a) 将表达式a中的复指数项变换为等价的三角形式。

denom(a) 给出表达式a的分母。

depends(a,b) 声明a是自变量b的函数。这在书写微分方程的时候很有用。

desolve(a,b) 使用拉普拉斯变换求解线性常微分方程a的未知量b。

determinant(a) 给出方阵a的行列式。

diff(a,b1,c1,b2,c2,...,bn,cn) 给出a对变量bi的ci阶偏导数。diff(a,b,1)可简写为diff(a,b)。'diff(...) 代表不经过计算（unevaluated）的求导，这在书写微分方程的时候很有用。

eigenvalues(a) 返回两个列表，第一个列表是矩阵a的本征值，第二个是本征值对应的重复次数。

eigenvectors(a) 包含eigenvalues所有功能，并且计算矩阵a的本征向量。

entermatrix(a,b) 引导用户一个一个元素地输入一个a × b 的矩阵。

ev(a,b1,b2,...,bn) 在bi的条件下计算表达式a的值。bi可以是方程、方程构成的列表（比如solve返回的结果）或者赋值语句，在这种情况下，ev将bi“插入”到表达式a中。bi还可以是关键词numer（它让结果以数值格式显示），detout（它使任一矩阵的逆矩阵把行列式的值作为系数放在矩阵外），或者diff（它要求所有的微分都必须计算，即'diff被diff替代）。对manual command(即，不在用户自定义函数内)，ev可以省略，于是可简写为a,b1,b2,...,bn.

expand(a) 展开表达式a。

exponentialize(a) 将a中的所有三角函数转换为它们对应的复指数形式。

factor(a) 对表达式a进行因式分解。

freeof(a,b) 如果a不是表达式b的一部分，返回true。

grind(a) Displays a variable or function a in a compact format.When used with writefile and an editor outside of Maxima, it offers a scheme for producing batch files which include Maxima-generated expressions.

ident(a) 返回一个a × a的单位矩阵。

imagpart(a) 返回a的复数部分。

integrate(a,b) 计算a对变量b的不定积分。

integrate(a,b,c,d) 计算a在区间$b \in [c, d]$上的定积分。积分限c,d可以分别取minf（负无穷大），inf（正无穷大）。

invert(a) 计算方阵a的逆矩阵。

kill(a) 从当前的Maxima环境中移除变量a以及它的属性。

limit(a,b,c) 计算当b趋近于c时a的极限。与积分函数integrate一样，c可以取inf或minf。

lhs(a) 给出方程a的等号左边部分。

loadfile(a) 从磁盘的当前目录中加载文件名为a的文件。该文件必须具有正确的格式（i.e. 由save命令创建）。

makelist(a,b,c,d) 创建一个a（假定a以b为自变量）的列表，从b=c到b=d依次将a追加到列表。

map(a,b) Maps the function a onto the subexpressions of b.

matrix(a1,a2,...,an) 创建一个以ai为行向量的矩阵a，每一个行向量是一个包含m个元素的列表[b1, b2, ..., bm]。

num(a) 给出表达式a的分子。

ode2(a,b,c) 求解一阶或二阶常微分方程a，其中b是c的函数。

part(a,b1,b2,...,bn) 首先取表达式的第b1部分，然后再在该部分中再取b2部分，依次...

playback(a) Displays the last a (an integer) labels and their associated expressions. If a is omitted, all lines are played back. See the Manual for other options.

ratsimp(a) 化简并以两个多项式的商的形式显示。

realpart(a) 返回a的实部。

rhs(a) 给出方程a的等号右边部分。

save(a,b1,b2,..., bn) 在磁盘的当前目录下创建包含变量、函数或矩阵bi的文件a。该文件可以在以后的会话中用loadfile命令重新载入。如果b1取all的话，每一个符号（包括标签）都可以得以保存。

solve(a,b) 求解关于未知数b的代数方程a，将返回一个根的列表。简单起见，如果方程a是c=0的形式（即方程右侧为0）,a可以用表达式c替代。

string(a) 将表达式a转换为Maxima的线性表示（类似a的Fortran表示）就像是它被输入并放入一个缓冲区以用来进行可能的编辑。这样string以后的表达式不能用于后续的计算。

stringout(a,b1,b2,...,bn) 在当前缺省磁盘目录下创建关于变量bi（比如labels）的文件a。该文件采用文本格式并且不能被Maxima再次读入。尽管如此，这种字符串化的输出只需经过稍许修改就可用于Fortran，Basic或C程序。

subst(a,b,c) 将表达式c中的b用a来替换。

taylor(a,b,c,d) 将表达式a在b=c处展开为泰勒级数，展开的级次不超过$(b-c)^d$。Maxima也支持超过一个自变量的泰勒展开，详细资料查看手册。

transpose(a) 给出矩阵a的转置。

trigexpand(a) 这是一个三角化简的函数，它采用sum-of-angle公式使得每一个sin和cos函数的变量尽可能简单。例如：trigexpand(sin(x+y))的结果为cos(x) sin(y) + sin(x) cos(y).

trigreduce(a) 这是一个三角化简的函数，它采用三角恒等式将乘积或幂函数变换为sin或cos的和的形式，每一项只含有一个sin或cos。例如：trigreduce(sin(x)^2)的结果为(1 - cos(2x))/2.

trigsimp(a) 这也是一个三角化简的函数，它将表达式中的tan，sec等函数变换为cos和sin函数。变换过程中它也使用恒等式$sin()^2 + cos()^2 = 1$。