Maxima在线性代数的应用

作者：蔡炎龍

转换重排：dbzhang

目录

1 简介

这篇文章，是介绍Maxima 这套数学软件，在学习线性代数的应用。

Maxima 是一个所谓的「电脑代数系统」（Computer Algebra System, CAS），这种系统比较为人熟知的还有Mathematica 和Maple 等等。我们选定Maxima 做为我们使用的程序，主要有三个原因：

免费 Maxima 是免费，又是各平台都有的。所有的人可以在自己的电脑上练习。

功能完整 Maxima 虽然不要钱，并不代表不好。Maxima 不论计算或图形功能都十分完整。事实上，Maxima 是最早的全功能CAS 系统Macsyma 的后代。

具代表性 许多新的CAS 系统，如Maple, Mathematica 都多少受到Macsyma 的启发。所以学会Maxima，要学会Maple 或Mathematica 等软件都是很容易的事。

这篇文章主要是介绍线性代数相关功能。我们不假设同学已会基本的Maxima 使用方式，所以我们会用到的概念，也许不纯粹是线性代数的，也会一并介绍。专就线性代数而言，我们要会

的其实并不多。想要快速进入状况，可以跳过前面的部份，直接看线性代数相关指令，在操作上有问题时，再回头看有问题部份的相关说明即可。

如果同学们比较喜欢使用Mathematica，Maple，或是Matlab 等商业软件也是可以的。我们系上的电脑室有提供这些软件，可以上机试用看看。

2 基本概念

我们先介绍一下Maxima 操作的方式。

2.1 Maxima 当计算器

我们先来看，如果我们要把Maxima 当计算器用，会是什么情况？

```
(%i1) 1+1;
(%o1) 2
(%i2) 3*4*7;
(%o2) 84
(%i3) 9/3;
(%o3) 3
```

到目前为止，似乎还没什么特别。除了可以做复杂一点点的运算，和平常的计算机或数值计算软件也没什么不同。以下的例子就不一样了：

(%i4) 7/3;

(%o4) $\frac{7}{3}$

(%i5) 1/2+2/3;

(%o5) $\frac{7}{6}$

从(%o4)我们看到，7/3这种运算，Maxima 不是告诉我们2.3333...，而是分数的形式！难道Maxima 真的懂分数？不要怀疑，这就是所谓电脑代数系统(CAS) 的特长。我们可以像(%o5)的例子一样，输入个分数的四则运算试试即知。

如果坚持要用浮点数，那只要加个float 指令即可：

```
(%i6) float(7/3);
(%o6) 2.333333333333334
```

为了完整，我们顺便再介绍指数，根号，阶乘表示法：

```
(%i7) 2^10;
(%o7) 1024
(%i8) sqrt(9);
(%o8) 3
(%i9) 5!;
(%o9) 120
```

我们可以看出，这些运算不是自然的数学符号，就是和我们平常电脑程序语言的写法。

2.2 指令结尾

在上面的例子中，我们发现，在Maxima 下指令，结束时一定要打上分号「;」，让Maxima 知道我们下的指令已结束。为什么要多这一个动作，主要是为了有时打比较长的指令可以换行之故。另一个结束方式是打入「 $ 」的符号。不同於分号的地方是「运算结果不会显示出来」：

```
(%i10) 2+3$
(%i11) 2+3;
(%o11) 5
```

有一些CAS 程序，如Matehmatica 是用分号表示不显示运算结果。不过Maxima 中分号已用上，必需用其他字元。

2.3 离开Maxima

离开Maxima 打入“quit();” 即可。

当然，很多人可能会觉得奇怪，为什么不是打入“quit” 就好了呢？原来像这种程序导向的语言，什么动作其实都是执行一个函数。所以我们事实上是执行一个叫「离开」的函数。这函数没有引数，所以就成了quit() 的形式。

2.4 结果的引用

我们时常会需要引用前面的结果，这时就用百分比符号“%” 。比方说：

```
(%i12) 7/3;
```

(%o12) $\frac{7}{3}$

```
(%i13) float(%);
```

(%o13) 2.333333333333334

Maxima 也可以指定使用第几个输出的结果，不过自己定一个标签可能是最好的方式。比方说，我们可以这样用：

```
(%i14) myresult:34+(65*72)/119;
```

(%o14) $\frac{8726}{119}$

```
(%i15) float(myresult);
```

(%o14) 73.32773109243698

2.5 重要常数

Maxima 当然有内建e 或是π 常常用到的数，只是表示法奇怪一点。e 是%e 而π 是%pi 。

2.6 定义变量

Maxima定义变量的想法有点特别，在定义一个变数时，其时是给某个数字、矩阵，或想要定义的任何式子等等一个标签。让我们来看几个例子：

```
(%i16) a: 37;
```

(%o16) 37

```
(%i17) a;
```

(%o17) 37

```
(%i18) b: 22+100*(375-128);
```

(%o18) 24722

```
(%i19) a+b;
```

(%o19) 24759

2.7 函数

Maxima 函数的定义和使用非常直觉，我们看几个例子就知道：

(%i20) `f(x) := 3*x^2 + 5;`

(%o20) $f(x) := 3x^2 + 5$

(%i21) `f(2);`

(%o21) 17

(%i22) `g(x,y) := sin(x)*cos(y);`

(%o22) $g(x, y) := \sin(x) * \cos(y);$

(%i23) `g(2*%pi,4);`

(%o23) 0

重点就是，在定义函数时要用":=" 去定义。比较一下和变数定义的不同，想想为什么要有两种不一样的定义方式。

3 进阶使用

3.1 列式而不运算

我们先计算一个瑕积分，用到无穷大的部份Maxima 是以inf 表示：

(%i1) `integrate(%e^(-x^2),x,0,inf);`

(%o1) $\frac{\sqrt{\pi}}{2}$

还记得这在微积分是怎么积出来的吗？Maxima 居然会积！不过，今天这不是我们的重点。今天重点是，有时你不是要秀答案，只是要列出式子。我们要怎么样让Maxima 不要太自动就算出来呢？答案是加个"'"号在前面，例如：

(%i2) `'integrate(%e^(-x^2),x,0,inf);`

(%o2) $\int_0^{\infty} \frac{1}{e^{x^2}}\, dx$

3.2 kill 指令

有时我们设定了一堆变数，函数，后来又不想再用下去，可以用kill 指令。而kill(all) 更是把我们定义过的变数，函数全部删除。看些例子就更加清楚：

(%i3) `f(x) := 3*x^2 + 5;`

(%o3) $\mathrm{f(x)} := 3\mathrm{x}^2 + 5$

(%i4) `f(x);`

(%o4) $3\mathrm{x}^2 + 5$

(%i5) `kill(all);`

(%o5) done

(%i6) `f(x);`

(%o6) f(x)

3.3 ev的使用

我们可以把Maxima 的ev 指令想成一个独立的环境。有点像在写程序时的函式一样, 并不会影响到其他的运作。第一种ev 的应用是把我们设成不要执行的指令执行:

(%i7) `f:'integrate(x^2,x)`

(%o7) $\int \mathrm{x}^2\,\mathrm{dx}$

(%i8) `ev(f,integrate)`

(%o8) $\frac{\mathrm{x}^3}{3}$

另一个很有用的使用方式是, 我们有个式子, 比方说:

(%i9) `f: a*x^2 + b*x + c;`

(%o9) $\mathrm{ax}^2 + \mathrm{bx} + \mathrm{c}$

假设我们想令一个式子是$a = 1, b = -2, c = -8$ 的情况, 我们当然可以先令各个变数是这样, 们问题是这么一来, f 也永远是$x^2 - 2x - 8$, a, b, c 这三个变数也不再是「符号」, 而是有值的。为了避免这个问题, 我们可以用ev 指令, 在下了这个指令后, 我们可以发现, 并没有变动到原来a, b, c 或是f :

(%i10) g: ev(f, a=1, b=-2, c=-8);

(%o10) $x^2 - 2x - 8$

(%i11) a;

(%o11) a

4 线性代数相关指令

这节我们正式介绍线性代数相关，也就是矩阵相关的指令。

4.1 矩阵及向量

我们先来看矩阵和向量的定义方式。前面说过，在Maxima 里，所谓设定一个变数的值，只不过是给某个数字或矩阵等等一个名称。我们这里就举应用在矩阵和向量时的情况：

(%i1) A:matrix([1,2,3],[-2,8,3],[1,4,9]);

(%o1) $\begin{pmatrix} 1 & 2 & 3 \\ -2 & 8 & 3 \\ 1 & 4 & 9 \end{pmatrix}$

(%i2) v: [2,3,5];

(%o2) $[2, 3, 5]$

我们可以看出，要定义一个矩阵，就是把矩阵一列列的输入；定义一个向量，其实和我们用手写向量出来也差不多。不过，问题是我们在线性代数常常要把向量写成「行向量」，而非如上的「列向量」表示方式。我们可以用下面两种不同的方式达成：

(%i3) v: transpose([2,3,5]);

(%o3) $\begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix}$

(%i4) v: matrix([2],[3],[5]);

(%o4) $\begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix}$

其实向量应该是一个一列或一行的矩阵, 但是Maxima 提供了简单定义列向量的方法。这里要强调一点, 一般来说因为矩阵乘法的关系, 我们写成列向量和行向量差别很大。不过Maxima 其实不太在意这点: 它可以聪明地发现你要做的事, 并且正确得计算出来! 简单的说, 一般而言, 我们不需要麻烦得定义行向量, 用列向量即可。

4.2 矩阵的表示和截取

这节我们讨论矩阵的抽象表示和取出一个矩阵行，列，甚至entry 的方法。这在很多理论和计算的尝试会用到。Maxima 是一个CAS 系统，所以我们可以完全用符号去定义一个矩阵，比方说：

```
(%i5) A: matrix([a[1,1],a[1,2]],[a[2,1],a[2,2]]);
```

(%o5) $\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}$

你也可以做完全抽象的代数计算：

```
(%i6) c*A;
```

(%o6) $\begin{pmatrix} a_{1,1}\, c & a_{1,2}\, c \\ a_{2,1}\, c & a_{2,2}\, c \end{pmatrix}$

如此一来，我们要试著导出一些定理就非常方便!

现在，我们重新把A 定义成一个实数矩阵，再看看怎么样找出A 的某一列，某一行，或某个entry。

```
(%i7) A: matrix([1,2,3],[-2,8,3],[1,4,9]);
```

(%o7) $\begin{pmatrix} 1 & 2 & 3 \\ -2 & 8 & 3 \\ 1 & 4 & 9 \end{pmatrix}$

```
(%i8) row(A,1);
```

(%o8) $\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$

```
(%i9) col(A,2);
```

(%o9) $\begin{pmatrix} 2 \\ 8 \\ 4 \end{pmatrix}$

```
(%i10) A[2,3];
```

(%o10) 3

4.3 矩阵向量之四则运算

我们要做矩阵加法、减法、乘法非常直觉而容易。乘法用的运算元是“.”。我们假设有了前面矩阵A 和向量v 的定义，来看以下的例子：

```
(%i11) A.v;
```

(%o11) $$\begin{pmatrix} 23 \\ 35 \\ 59 \end{pmatrix}$$

你也可以定义非向量的矩阵试试矩阵的乘法。比方说，两个矩阵A , B 的乘积是A.B，要注意A*B 并不会得到矩阵相乘的结果！到底A*B 是什么意思，大家不妨自己试试，看可不可以找出其中的意义。

向量内积的做法和你想的一样：

```
(%i12) w: [2,3,5];
```

(%o12) $[2, 3, 5]$

```
(%i13) w.w;
```

(%o13) 38

你可能发现了一个问题，那就是我们上面内积的例子是用列向量。那行向量可以吗？可以的！Maxima会聪明的知道你想做什么，不信可以试试看。

矩阵和向量的纯量乘法是用平常的“*” 号：

```
(%i14) 2*A;
```

(%o14) $$\begin{pmatrix} 2 & 4 & 6 \\ -4 & 16 & 6 \\ 2 & 8 & 18 \end{pmatrix}$$

现在我们来看一下有可能会产生误会的地方。假设我们现在要算$A \cdot A$ ，你可能会想是A^2，结果并不正确！其实A^2 是把A 的每一个entry 都平方。正确计算$A \cdot A$ 要用A^^2

4.4 矩阵相关函数

我们要计算矩阵的行列式值，求转置矩阵, 矩阵的秩等等的基本运算，Maxima 当然也都有（A还是我们之前定义的矩阵）：

(%i15) `transpose(A);`

(%o15) $\begin{pmatrix} 1 & -2 & 1 \\ 2 & 8 & 4 \\ 3 & 3 & 9 \end{pmatrix}$

(%i16) `determinant(A);`

(%o16) 54

(%i17) `rank(A);`

(%o17) 3

我们当然也可以手动计算行列式值。但这时需要知道矩阵第i,j 这个位置的子式(minor), 也就是A 矩阵去掉第i 列, 第j 行所成的矩阵, 这指令叫minor:

(%i18) `minor(A,1,1);`

(%o18) $\begin{pmatrix} 8 & 3 \\ 4 & 9 \end{pmatrix}$

矩阵的余因子(cofactor) 在Maxima 中并没有定义, 好在我们自己可以很容易定一个cofactor 函数:

(%i19) `cofactor(M,i,j):=(-1)^(i+j)*determinant(minor(M,i,j));`

(%o19) $\mathrm{cofactor}(\mathrm{M},\mathrm{i},\mathrm{j}) := (-1)^{\mathrm{i}+\mathrm{j}}\mathrm{determinant}(\mathrm{minor}(\mathrm{M},\mathrm{i},\mathrm{j}))$

(%i20) `cofactor(A,1,1);`

(%o20) 60

我们在计算反矩阵等会用到的古典伴随矩阵(classical adjoint matrix) 也很容易算出来:

(%i21) `adjoint(A);`

(%o21) $\begin{pmatrix} 60 & -6 & -18 \\ 21 & 6 & -9 \\ -16 & -2 & 12 \end{pmatrix}$

说到反矩阵，要用Maxima 求出来也是易如反掌:

(%i22) `invert(A);`

(%o22) $\begin{pmatrix} \frac{10}{9} & -\frac{1}{9} & -\frac{1}{3} \\ \frac{7}{18} & \frac{1}{9} & -\frac{1}{6} \\ -\frac{8}{27} & -\frac{1}{27} & \frac{2}{9} \end{pmatrix}$

或是你也可以用前面的方式求反矩阵:

```
(%i23) A^^(-1);
```

(%o23) $\begin{pmatrix} \frac{10}{9} & -\frac{1}{9} & -\frac{1}{3} \\ \frac{7}{18} & \frac{1}{9} & -\frac{1}{6} \\ -\frac{8}{27} & -\frac{1}{27} & \frac{2}{9} \end{pmatrix}$

在解线性方程组常用到的梯形矩阵也是容易得很：

```
(%i24) echelon(A);
```

(%o24) $\begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & \frac{3}{4} \\ 0 & 0 & 1 \end{pmatrix}$

4.5 使用模组

用了Maxima 一阵子，你可能会预期它该会的都会。比方说求一个矩阵的trace，这应该够容易了吧?

事情并不是那么简单。Maxima 本身是「不会」算trace 的！当然我们可以自己写个小程序，不过先别急。我们可以使用使用适当的模组来做这件事。

所谓模组就是一段小程序，通常是增加一些指令，供你使用。你也许会觉得奇怪，那为什么Maxima 不一开始就把这些模组都加进来? 那是因为如此一来太占用内存，也许很多对某些人重要的指令你永远也不用去用!

我们要算一个矩阵的trace，要使用nchar1 这个模组，这个模组提供了mattrace 指令去计算trace。

使用的方法如下，先以

```
(%i25) load("nchar1");
```

读入nchar1 模组，接著就可以使用这个模组提供的指令：

```
(%i26) A: matrix([1,2,3],[2,2,1],[3,3,1]);
```

(%o26) $\begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \\ 3 & 3 & 1 \end{pmatrix}$

```
(%i27) mattrace(A);
```

(%o27) 4

5 线性代数应用实例

5.1 特征值和特征向量

我们这里讨论线性代数很重要的特征值相关的计算。我们定义一个矩阵A，计算特征值和特征向量时我们都以这个矩阵为主要讨论对象:

```
(%i1) A: matrix([4,0,1],[2,3,2],[1,0,4]);
```

(%o1) $\begin{pmatrix} 4 & 0 & 1 \\ 2 & 3 & 2 \\ 1 & 0 & 4 \end{pmatrix}$

我们计算一下特征值：

```
(%i2) eigenvalues(A);
```

(%o2) $[[5,3],[1,2]]$

怎么样，很方便吧...等等，特征值怎么会出来两个向量呢！？原来，真正的特征值是放在结果的第一个list 当中，也就是5 和3。那第二个list 代表什么呢？代表的就是每个特征值的几何重数, 也就是每个特征值对应的特征向量空间之维度。换言之，这是比较完整的特征值资讯！

我们也可以用eigenvectors 计算特征向量。事实上，eigenvectors 也会把特征值列出来，所以是包含前面eigenvalues 功能的指令。不过如果我们一开始就介绍eigenvectors，看到那有点复杂的结果大家可能会昏倒。现在已经会了eigenvalues，大概就没问题了：

```
(%i3) eigenvectors(A);
```

(%o3) $[[[5,3],[1,2]],[1,2,1],[1,0,-1],[0,1,0]]$

第一部份和eigenvectors 输出一样，就是说我们有5 有和3 两个特征值，其mutiplicities 分别是1 和2。因此，对於5 应该要有一个对应的特征向量，即$[1,2,1]$，对於3会有两个，分别是接下来的$[1,0,-1]$ 和$[0,1,0]$。这些向量会生成相对应特征值的向量空间。

5.2 手动特征值的计算

上一节介绍Maxima 内建特征值计算，并不一定每个人都喜欢。比方说显示的方式比较特别，另外就是不是一步一步算的，心里有时也有不踏实的感觉。因此，我们这里介绍一下如何用Maxima 一步一步的把特征值求出来。

我们再用一次上一节的例子：

```
(%i4) A: matrix([4,0,1],[2,3,2],[1,0,4]);
```

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

我们先求特征多项式，也就是$A - tI$ 的行列式值：

```
(%i5) f: charpoly(A,t);
```

(%o5) $\mathrm{f} : \mathrm{t} + (3 - \mathrm{t})(4 - \mathrm{t})^2 - 3$

如果想要看到比较漂亮的式子, 可以将f 展开:

```
(%i6) expand(f);
```

(%o6) $-\mathrm{t}^3 + 11\mathrm{t}^2 - 39\mathrm{t} + 45$

我们还可以将f 做因式分解, 这样就可以清楚看到A 有几个特征值, 和各特征值的代数重数:

```
(%i7) factor(f);
```

(%o7) $-(\mathrm{t} - 5)(\mathrm{t} - 3)^2$

这样我们就求得A 的特征值是5 和3 。
另一个解法是，我们可以求$f = 0$ 的零根。做法是使用solve 指令：

```
(%i8) solve(f=0, t);
```

(%o8) $[\mathrm{t} = 5, \mathrm{t} = 3]$

当然，我们算法正确，应该是得到和前面一样的结果。

5.3 解线性方程组

线性代数的核心问题，就是解线性方程组。解线性方程组一样可以用上一节介绍的solve 指令来解。我们来看一个简单的例子，并且用Maxima 来解。

我们考虑下面的线性方程组：

$$x + 2y + 3z = 6$$

$$2x - 3y + 2z = 14$$

$$3x + y - z = -2$$

我们一样可以用前面用过的solve 指令来解：

```
(%i9) eq1: x + 2*y + 3*z = 6;
```

(%o9) $x + 2y + 3z = 6$

```
(%i10) eq2: 2*x - 3*y + 2*z = 14;
```

(%o10) $2x - 3y + 2z = 14$

```
(%i11) eq3: 3*x + y - z = -2;
```

(%o11) $3x + y - z = -2$

```
(%i12) solve([eq1, eq2, eq3],[x,y,z]);
```

(%o12) $[[x = 1, y = -2, z = 3]]$

5.4 手动求特征向量空间的基底

我们在前面介绍过, 使用eigenvectors 指令就可以求出特征向量空间的一组基底。我们再用一次前面的矩阵:

```
(%i13) A: matrix([4,0,1],[2,3,2],[1,0,4]);
```

(%o13) $\begin{pmatrix} 4 & 0 & 1 \\ 2 & 3 & 2 \\ 1 & 0 & 4 \end{pmatrix}$

我们已经求出A 的特征值是5 和3 , 我们这里用特征值3 做范例, 看看怎么样能求出对应的特征向量。我们现在要求的就是什么样的向量v , 会满足$(A - 3I)v = 0$ 。这里我们可以用ident 指令可以很容易造出$n \times n$ 的单位矩阵。以下我们就把大略的设定做好:

```
(%i14) I: ident(3);
```

(%o14) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i15) v:[x,y,z];
```

(%o15) $[x, y, z]$

```
(%i16) u: (A-3*I).v;
```

(%o16) $\begin{pmatrix} z + x \\ 2z + 2x \\ z + x \end{pmatrix}$

我们现在就是要看什么样的x, y, z 会让u 是零向量。这个例子其实用手解也很容易, 但是我们给Maxima 一个机会。我们要做的就是解一个线性方程组:

(%i17) `eq1: u[1,1]=0;`

(%o17) $z + x = 0$

(%i18) `eq2: u[2,1]=0;`

(%o18) $2z + 2x = 0$

(%i19) `eq3: u[3,1]=0;`

(%o19) $z + x = 0$

(%i20) `solve([eq1,eq2,eq3],[x,y,z]);`

(%o20) $[[x = -\%r1, y = \%r2, z = \%r1]]$

这看来有点可怕的%r1 和%r2 是什么呢? 原来这只是表示两个参数, 换成我们一般的写法, 我们可能会写成$x = -t, y = s, z = t$ 。至此, 我们已找到特征向量的一般表示式, 如果要找到一组基底也很容易, 我们先令ans 代表前面解出的式子, 再把$(\%r1, \%r2)$ 代入$(1, 0)$, $(0, 1)$ 即可:

(%i21) `ans: %;`

(%o21) $[[x = -\%r1, y = \%r2, z = \%r1]]$

(%i22) `ev(ans, %r1=1, %r2=0);`

(%o22) $[[x = -1, y = 0, z = 1]]$

(%i23) `ev(ans, %r1=0, %r2=1);`

(%o23) $[[x = 0, y = 1, z = 0]]$

我们可能会希望把结果设成两个向量v_1, v_2 , 方便以后使用。我们可以再用ev 来做到这样的事:

(%i24) `v1: ev([x,y,z], %o22)`

(%o24) $[-1, 0, 1]$

(%i25) `v2: ev([x,y,z], %o23)`

(%o25) $[0, 1, 0]$

6 Maxima 的绘图功能

6.1 二维绘图

Maxima 二维绘图的指令是用plot2d。比方说，我们要画$4x^3-2x-2$ 这个函数，设定x 轴的范围是从-5 到5，就下这个指令：

```
(%i1) plot2d([4 * x^3-2 * x-2],[x,-5,5]);
```

6.2 三维绘图

三维绘图也一样容易，只要改用plot3d 的指令即可：

```
(%i2) plot3d(cos(-x^2+y^3/4),[x,-4,4], [y,-4,4]);
```

Geomview 是一个UNIX 的软件，Maxima 可以运用Geomview 做出非常漂亮的3D 图形。我们来看上个例子以Geomview 输出的结果。

```
(%i3) plot3d(cos(-x^2+y^3/4),[x,-4,4], [y,-4,4], [plot_format,geomview]);
```

Geomview 不但可以画出漂亮3D 图形，更重要的是它可以弥补Maxima 的一些缺点。比方说，Maxima 本身的3D 绘图不可以同时显示两个或两个以上函数图形（2D可以），但利用Geomview，这样的绘图变成可能。

6.3 点绘图

有很多绘图的应用，就只需要画出点，或是用一些点来描述一些函数。这事实上比画函数还简单，但是Maxima 直到5.9.2 版才有这样的功能。详情请参考(5.9.2 之后的) 使用手册。

6.4 多个函数的绘图

如果要比较几个函数，要如何下指令呢？我们来看个例子就明白了：

```
(%i4) plot2d([cos(x), sin(x), tan(x)], [x, -2*%pi, 2*%pi], [y,-2,2])$
```

这个例子会同时画出$\cos(x), \sin(x)$ 和$\tan(x)$ 的图形。

6.5 参数式绘图

我们仅简单举一参数式绘图之例子, 详情请参考Maxima 使用手册。

```
(%i5) plot2d([parametric, cos(t), sin(t), [t,-2*%pi,2*%pi], [nicks,80]]);
```

7 Maxima 的安装

在Maxima 的官方网站有不同版本的Maxima 供各平台使用：

http://maxima.sourceforge.net/

不过，不同平台可能有一些不同的选择。我概略说明一下我建议的安装方式。不管用Windows, Mac, 或是Linux，我都推荐使用TeXmacs 这个文书处理软件当界面，因为这样可以显示最漂亮的数学符号。

7.1 Windows

Windows 至少有三种可以执行Maxima 的方式，不管哪一种，都要先装xMaxima。首先就是在官方网站下载Maxima Windows 版。安装也很容易，下载后点两下就可以自动安装。

xMaxima 的缺点是纯文字显示，不能显示漂亮的数学符号。使用PC 的同学，当然可以试著安装Linux，采用下面介绍的方式使用Maxima。如果还没确定，或不想花那么多时间安装Linux，可以先试用有Maxima 的LiveCD。Linux 的LiveCD 是可以开机的CD，你只要放进你的电脑，用光碟开机，就可使用，不用灌Linux。

长庚大学黄朝锦教授提供了有TeXmacs （见后Linux 的说明）及Maxima 的LiveCD，你可在下面download。

ftp://math.cgu.edu.tw/pub/KNOPPIX/

请选择TeXmacs 的.iso 档，再烧成光碟即可。注意有一般光碟和DVD 版，看自己的需要下载。Windows 还可以装wxMaxima，这个界面比xMaxima 漂亮，不过还不及接下来要介绍，采用WinTeXmacs 的界面，所以我不详细介绍。

Windows （或其他平台），让Maxima 看来最漂亮的大概就是用WinTeXmacs。

7.2 Linux

不同的Linux 都有不同软件管理程序，像Maxima 大概所有管理程序都有提供，所以我不详细说明如何安装，只列出建议安装的套件：

Maxima Maxima 主程序。

TeXmacs 一个可打漂亮数学式子的编辑器，提供漂亮的Maxima 介面。

Geomview 配合Maxima 可画出高级3D 图形。

使用时，先执行TeXmacs，在里面执行Maxima 的session即可。这是我推荐的漂亮版Maxima。

7.3 Mac OS X

Mac OS X 是一个UNIX 系统，所以需要的程序和Linux 一样。首先你先要安装Apple 的X11 软件。这是因为UNIX 上用的X-Windows 系统当然和Mac OS X 的aqua 视窗系统不同，UNIX 软件大多只能用X-Windows 显示。

在安装TeXmacs 之前，你必需要有完整的LaTeX 系统。我强烈推荐用i-installer 安装：

http://www.rna.nl/tex.html

如果不知怎么做，可参考我的文章：

http://homepage.mac.com/yenlung/WebWiki/LaTeXonMac.html

接著，使用Fink 去安装Maxima, TeXmacs, Geomview:

http://www.rna.nl/tex.html

要注意的是Geomview 只有unstable 版。

8 关於这份文件......

8.1 原繁体中文版

Maxima 在线性代数的应用

本文件使用LaTeX2HTML , Version 2002-2-1 (1.71) 转换。

中文转换方法详见李果正《我的CJK》中「CJK 和LaTeX2HTML 的配合」之说明。本文使用李果正taiwan.perl 套件。

8.2 本文件

Maxima 在线性代数的应用

由dbzhang转换，并重新排版。