Cell[BoxData[

RowBox[{

RowBox[{"-", "2"}], "+",

RowBox[{"6", " ", "x"}], "-",

RowBox[{"2", " ",

SuperscriptBox["x", "2"]}], "+", "y", "-",

RowBox[{"3", " ", "x", " ", "y"}], "+",

RowBox[{

SuperscriptBox["x", "2"], " ", "y"}], "+",

SuperscriptBox["y", "2"], "-",

RowBox[{"3", " ", "x", " ",

SuperscriptBox["y", "2"]}], "+",

RowBox[{

SuperscriptBox["x", "2"], " ",

SuperscriptBox["y", "2"]}]}]], "Output",

CellChangeTimes->{3.6300190123484373`\*^9}]

Cell[BoxData[

RowBox[{

RowBox[{

SuperscriptBox["x", "2"], " ",

RowBox[{"Cos", "[", "x", "]"}]}], "+",

RowBox[{"2", " ", "x", " ",

RowBox[{"Sin", "[", "x", "]"}]}]}]], "Output",

CellChangeTimes->{3.6300196719601655`\*^9}]

Simplify

Expand

Factor

Cancel

Sqrt

Exp

您可以在下一个计算中按下 Ctrl+L (或 Shift+Ctrl+L) 调用您最近的输入（或输出）.

用 EscintEsc 输入 \[Integral] 并且用 Esc\[ThinSpace]dd\[ThinSpace]Esc 输入 \[DifferentialD]：

用 Ctrl+\_ 输入下限，然后用 Ctrl+% 输入上限：

TraditionalForm[%]

Esc\[ThinSpace]sum\[ThinSpace]Esc 求和符号 \[Sum]

Esc\[ThinSpace]prod\[ThinSpace]Esc 连乘符号 \[Product]

Esc\[ThinSpace]int\[ThinSpace]Esc 积分符号 \[Integral]

Esc\[ThinSpace]dd\[ThinSpace]Esc 积分中使用的微分符号 \[DifferentialD]

Esc\[ThinSpace]pd\[ThinSpace]Esc 偏导数 \[PartialD]

Esc\[ThinSpace]sqrt\[ThinSpace]Esc 根号 \[Sqrt]

Ctrl+\_ or Ctrl+- 将光标移到积分的下限位置或下标位置

Ctrl+^ or Ctrl+6 将光标移到积分的上限位置或上标位置

Ctrl+& or Ctrl+7 将光标移到和或连乘的上限位置或上面

Ctrl+% or Ctrl+5 上，下之间切换

Ctrl+Space 从上，下位置返回

使用 "算术" 节中的运算符，用户可以输入任何代数表达式. 还可以使用空格表示乘号. 注意不要忘了在 x y 中的空格. 如果没有空格而输成 xy， Mathematica 将把它理解成名为 xy 的单一符号，而不是两个符号 x 和 y 的乘积.

当输入比较复杂的表达式时，正确的使用圆括号是重要的. 例如，表达式 x^(4y) 的输入形式为 x^(4y). 若丢了圆括号，得到的将是 x^4 y. 多使用括号不会有坏处，但是想知道究竟什么时候你需要使用圆括号，请看 "运算符输入形式".

用户可以把整个 Mathematica 看作一个使用变换规则的集合来处理不同表达式的简单系统.

D[f,x] (偏) 导数 \[PartialD]f/\[PartialD]x

Integrate[f,x] 不定积分 \[Integral]f dx

Sum[f,{i,Subscript[i, min],Subscript[i, max]}] 和式 \!\(\\*

UnderoverscriptBox["\[Sum]",

RowBox[{"i", "=",

SubscriptBox["i",

StyleBox["min", "TI"]]}],

SubscriptBox["i",

StyleBox["max", "TI"]],

LimitsPositioning->True]\ f\)

Solve[lhs==rhs,x] 求解 x 的方程

Series[f,{x,Subscript[x, 0],order}] f 在点 x=Subscript[x, 0] 的幂级数展开式

Limit[f,x->Subscript[x, 0]] 极限 Underscript[lim, x->Subscript[x, 0]]f

Minimize[f,x] 关于 x 的 f 的最小值

Simplify

. 这是点乘  
\* 这是[叉乘](http://www.baidu.com/s?wd=%E5%8F%89%E4%B9%98&hl_tag=textlink&tn=SE_hldp01350_v6v6zkg6" \t "_blank)

替换算符 /. 使用户可以对一个特定表达式进行变换. 然而，有时用户想要定义一个总是 被使用的变换规则. 例如，无论 x 何时出现， x 总是被 3 替换.

如同在 "定义变量" 讨论的那样，这可以通过 x=3 给 x 赋 以值 3 来实现. 一旦用户做了赋值 x=3，无论 x 何时出现， x 总是被 3 替换.

用户可以把符号的值定义成任何表达式，而不仅是能定义成数. 应该注意，一旦给出了定义，这个定义将一直被使用，直到用户明确改变或消除该定义. 在使用 Mathematica 时，忘记清除已经赋给符号的值是常见的出错原因. 为了避免错误，当使用完设置的变量后，应立即清除该变量的值

.x=value 定义 x 的值，值将一直被使用

x=.或clear[x]清除定义给 x 的值

f[x\_]:=x^2 定义函数 f

?f 显示 f 的定义

Clear[f] 清除所有 f 的定义

= 和 := 不仅用来定义函数，还用来给变量赋值. x=value 立即求 value 的值，并将其赋于 x. 另一方面，x:=value 不立即求值 value. 而是在每次使用 x 时才计算其值.

尽管 := 比 = 用得多一些，但还有必须用 = 定义函数的一个重要情形. 当进行一个运算得到具有符号参数 x 的结果时，还需要进一步得到对应于不同 x 的结果. 一种方式是用 /. 将适当的规则用于 x. 通常用 = 去定义变量 x 的函数就较方便一些.

lhs->rhs 给出规则后就计算 rhs

lhs:>rhs 使用规则时计算 rhs

与赋值的情形类似，当用确定的值代替表达式的值时用 ->，而当给出一个求值的命令时用 :>.

Matlab

cd(‘E:\’)

> mkdir longlanbin

>> addpath longlanbin

4、MathType还可以导出和导入为windows图元文件emf、矢量图片格式gif，这意味着如果你有一个公式是图片格式的，可以转换为这两种格式，然后用MathType导入并编辑。

最后来一个Matlab运算结果转换为MathType公式的例子：

syms x y;

f=sin(x)\*exp(y)+log(y)\*cos(x);

f=diff(diff(f,x,2),y,2)

fs=latex(f)

f =

-sin(x)\*exp(y)+1/y^2\*cos(x)

fs =

-\sin \left( x \right) {e^{y}}+{\frac {\cos \left( x \right) }{{y}^{2}}}

在MathType中选择转换选项后，把上面的代码直接复制进去就可以看到公式了。



12 findsum\_mex.c

13 link /dll /LIBPATH:"D:\PROGRA~2\MATLAB\R2013a\extern\lib\win64\microsoft" libmx.lib libmex.lib libmat.lib /MACHINE:X64 kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib /nologo /manifest /incremental:NO /implib:"findsum\_mex.x" /MAP:".\findsum\_mex.mexw64.map" /NODEFAULTLIB:LIBCMT /OUT:findsum\_mex.mexw64 findsum\_data.obj findsum\_initialize.obj findsum\_terminate.obj findsum.obj findsum\_api.obj findsum\_mex.obj @findsum\_mex\_mex.arf

14 正在创建库 findsum\_mex.x 和对象 findsum\_mex.exp

15 'cmd' 不是内部或外部命令，也不是可运行的程序

16 或批处理文件。

17 gmake: \*\*\* [findsum\_mex.mexw64] Error 1

matlab中如何查看函数的源代码

open +函数名

edit +函数名

首先，built-in 函数是不公开的，其次，built-in 函数并非是用我们熟知的matlab语言写的，所以，即使给你了，你也未必能看懂。matlab 最初是在两个高效的底层矩阵计算包 LINPACK 和 EISPACK 上发展出来的，这两个包的升级版本叫做 LAPACK （关于这些的介绍，你可以参考：http://www.mathworks.com/company ... porates-lapack.html）。能写出这些包并且对其维护的人都是全球顶尖编程高手，他们不仅要求编程牛逼，还要精通线性代数和矩阵运算。所以，一般人根本不可能接触到matlab这些底层的东西。曾经有很多matlab爱好者向mathworks 询问并索要过built-in函数源代码，都被拒绝了，他们收到的回复是：

You can read and edit the source code for M-file functions that ship with

MATLAB; the only way to read and edit the built-in functions like FFT is to

get a job with The MathWorks. We don't distribute the built-in source code

for MATLAB.

--

Steve Lord

[slord@mathworks.com](mailto:slord@mathworks.com)

).请问可以查看Matlab中函数的源代码吗?

Matlab除了buildin函数和mex/dll文件看不到原码，其他如工具箱等都可

以直接看到代码，首先确认该文件安装在matlab中，即which

filename.m存在，然后可以edit filename.m

正常情况下用ctrl+C是可以的，如果matlab陷入死循环，老是提示busy，这时的ctrl+C就不起作用，只能任务管理器强行终止。

ctrl+break 也行

替换函数subs()

符号转化成数值输出：eval() numeric() 数值转化成符号或表达式 sym()

[Fa,F2]=solve('Fa\*cos(fii)+F2\*cos(fi)-F1\*cos(fi)', 'Fa\*sin(fii)+F2\*sin(fi)-F1\*sin(fi)-8000','Fa','F2');

Fa=eval(Fa)

F2=eval(F2)

这样才能求出数值解来。Solve()是符号求解函数。

在m文件中写了一个简单的程序，突然发现cos(pi/2)的值不为零，因此两个矩阵在相乘后，本来是0的结果却用两个很大的的数的除法来表示。原因是在matlab中，pi是3.1416，而不是一个无理数。我想请教各位的是，在我得到结果后，是不是可以怎样处理一下，使结果看起来更加的清晰?

syms pi

cos(pi/2)

ans =0

可能是你设置了除法显示的格式，用输入"format long g"就可以了

matlab中更改变量后如何更新

solve(TT(1,1)==a3,x)

T=sym('[l1 l2;l3 l4]')

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

注意：eqn和varn可以是符号表达式，也可以是字符串表达式，但是使用符号表达式时不能有“=”号，假如说varn没有给出，使用findsym函数找出默认的求解变量。返回的g是一个结构体，以varn为字段。由于符号求解的局限性，好多情况下可能得到空矩阵，此时只能用数值解法

解方程A=solve('a\*x^2 + b\*x + c')

解方程组B=solve('a\*u^2 + v^2', 'u - v = 1', 'a^2 - 5\*a + 6')

发现运行结果很多都是用分数表示的，没有计算出具体值？

用format rat ，不过用vpa倒是可以解决这个问题了，

你添加的function还是script。 如果添加function的话，中间定义的变量都在workspace里面看不到。script就可以看到。 文件--新建--script。

clear;

clc;

syms pi;

syms c1 c2 c3 c4 c5 c6  d2 a2 d4 a3 d6

%输入下面矩阵时，应注意避免多余空格造成的输入错误

A10=[cos(c1) 0 -sin(c1) 0;sin(c1) 0 cos(c1) 0;0 -1 0 0;0 0 0 1];

A21=[cos(c2) -sin(c2) 0 cos(c2)\*a2;sin(c2) cos(c2) 0 sin(c2)\*a2;0 0 1 d2;0 0 0 1];

A32=[cos(c3) 0 sin(c3) -cos(c3)\*a3;sin(c3) 0 -cos(c3) -sin(c3)\*a3;0 1 0 0;0 0 0 1];

A43=[cos(c4) 0 -sin(c4) 0;sin(c4) 0 cos(c4) 0;0 -1 0 d4;0 0 0 1];

A54=[cos(c5) 0 sin(c5) 0;sin(c5) 0 -cos(c5) 0;0  1 0 0;0 0 0 1];

A65=[cos(c6) -sin(c6) 0 0;sin(c6) cos(c6) 0 0;0  0 1 d6;0 0 0 1];

A01=simplify(inv(A10));

A12=simplify(inv(A21));

A23=simplify(inv(A32));

A34=simplify(inv(A43));

A45=simplify(inv(A54));

A56=simplify(inv(A65));

A61=simplify(A21\*A32\*A43\*A54\*A65);

A60=simplify(A10\*A21\*A32\*A43\*A54\*A65);

Syms nx ox ax x ny oy ay y nz oz az z

T40=[nx ox ax x;ny oy ay y;nz oz az z;0 0 0 1]

A01\*T40

A21\*A32\*A43

c1=pi/2;c3=pi/2;c2=0;c4=0;c5=0;c6=0;

A10 A21 A32 A43 A54 A65

A01 =

[ cos(c1), sin(c1), 0, 0]

[ 0, 0, -1, 0]

[ -sin(c1), cos(c1), 0, 0]

[ 0, 0, 0, 1]

A12 =

[ cos(c2), sin(c2), 0, -a2]

[ -sin(c2), cos(c2), 0, 0]

[ 0, 0, 1, -d2]

[ 0, 0, 0, 1]

A61 =

[ - cos(c6)\*(sin(c2 + c3)\*sin(c5) - cos(c2 + c3)\*cos(c4)\*cos(c5)) - cos(c2 + c3)\*sin(c4)\*sin(c6), sin(c6)\*(sin(c2 + c3)\*sin(c5) - cos(c2 + c3)\*cos(c4)\*cos(c5)) - cos(c2 + c3)\*cos(c6)\*sin(c4), sin(c2 + c3)\*cos(c5) + cos(c2 + c3)\*cos(c4)\*sin(c5), d4\*sin(c2 + c3) - a3\*cos(c2 + c3) + a2\*cos(c2) + d6\*sin(c2 + c3)\*cos(c5) - (d6\*cos(c2 + c3)\*sin(c4 - c5))/2 + (d6\*cos(c2 + c3)\*sin(c4 + c5))/2]

[ cos(c6)\*(cos(c2 + c3)\*sin(c5) + sin(c2 + c3)\*cos(c4)\*cos(c5)) - sin(c2 + c3)\*sin(c4)\*sin(c6), - sin(c6)\*(cos(c2 + c3)\*sin(c5) + sin(c2 + c3)\*cos(c4)\*cos(c5)) - sin(c2 + c3)\*cos(c6)\*sin(c4), sin(c2 + c3)\*cos(c4)\*sin(c5) - cos(c2 + c3)\*cos(c5), a2\*sin(c2) - a3\*sin(c2 + c3) - d4\*cos(c2 + c3) - d6\*cos(c2 + c3)\*cos(c5) - (d6\*sin(c2 + c3)\*sin(c4 - c5))/2 + (d6\*sin(c2 + c3)\*sin(c4 + c5))/2]

[ cos(c4)\*sin(c6) + cos(c5)\*cos(c6)\*sin(c4), cos(c4)\*cos(c6) - cos(c5)\*sin(c4)\*sin(c6), sin(c4)\*sin(c5), d2 + d6\*sin(c4)\*sin(c5)]

[ 0, 0, 0, 1]

>>

机器人末端执行器工作空间的求解问题转化

为机器人手腕的中心工作空间的求解问题。即转化

为用图解法求解3自由度机器人的工作空间的问

题。

[X Y Z]=meshgrid(-900:10:900,-900:10:900,-900:10:900);

>> size(X)

ans =

181 181 181

>> numel(X) %求数组元素个数

ans = 5929741

如何debug一段for i=1:1:n循环程序，查找错误点，难点在于，假设i=1:1:n,n是一个很大的数，当运行至一半时突然出错，怎样高效查错呢？

h=plot3(x,y,z,’o-’) ;set(h,withen,10)

view(AZ ,EL)

 Here are some examples:  
  
    AZ = -37.5, EL = 30 is the default 3-D view.  
    AZ = 0, EL = 90 is directly overhead and the default 2-D view.  
    AZ = EL = 0 looks directly up the first column of the matrix.  
    AZ = 180 is behind the matrix.  
  
    view(2) sets the default 2-D view, AZ = 0, EL = 90.  
    view(3) sets the default 3-D view, AZ = -37.5, EL = 30.

有木有擅长C语言，实验室一师姐需要一外援，采集程序做好了，控制程序的matlab的都做好了，需要转换成c程序，然后把采集的量加进来，有偿劳动！！

c++可以调用matlab

用matlab编译器和命令，转换一下程序即可

可以用引擎方式，也可使用dll

用getfullmatrix命令