一、Octave的功能demo

In [20]:
```
r = roots([1 -1 -1])
```

```
r =

  -0.61803
   1.61803
```

In [1]:
```
pkg load symbolic
```

In [2]:
```
syms x
```

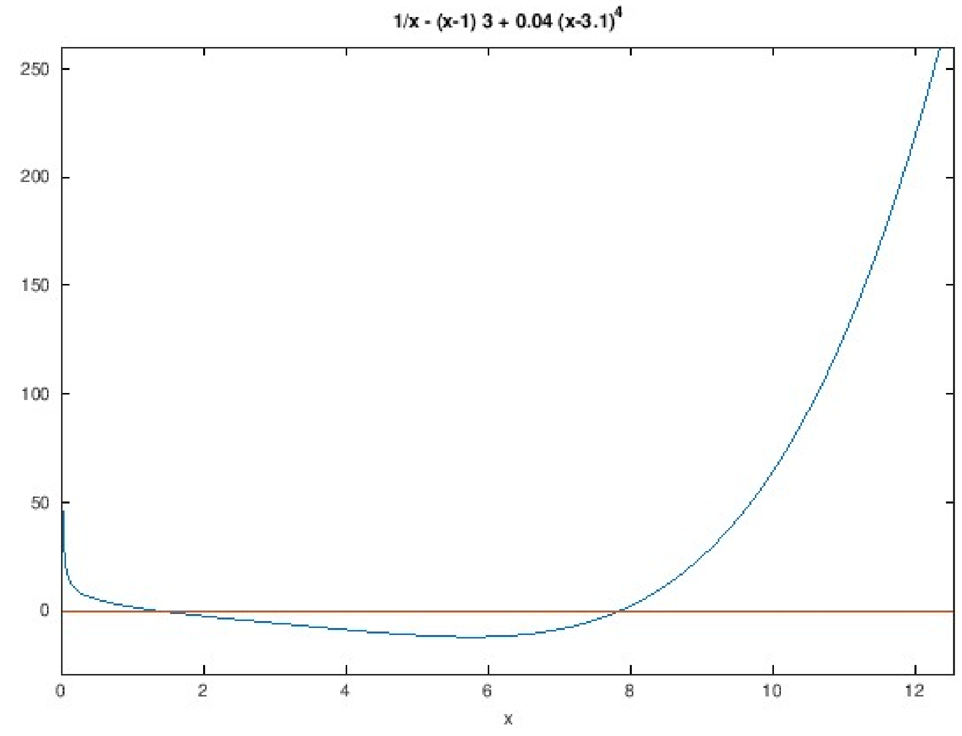
```
Symbolic pkg v2.9.0: Python communication link active, SymPy v1.5.1.
```

In [4]:
```
solve(1/x == (x-1))
```

```
ans = (sym 2x1 matrix)

  [       ___]
  [1    \/ 5 ]
  [- - -----]
  [2      2  ]
  [          ]
  [       ___]
  [1    \/ 5 ]
  [- + -----]
  [2      2  ]
```

In [12]:
```
f = inline('1/x - (x-1)*3 + 0.04*(x-3.1)^4');
```

In [17]:
```
ezplot(f, [ 0,4*pi]); hold on; plot(0:13,zeros(1,14)); hold off
```



In [15]:
```
help ode23
```

```
'perl' ۺ₩ Χǿeij
ɭ
warning: help: Texinfo formatting filter exited abnormally; raw Texinfo source o
f help text follows...
'ode23' is a function from the file D:\Octave\Octave-5.2.0\mingw64\share\octave\
5.2.0\m\ode\ode23.m



Additional help for built-in functions and operators is
available in the online version of the manual.  Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at https://www.octave.org and via the help@octave.org
mailing list.
```

In [21]:
```
whos
```

```
Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
        ans         1x1                          8  double
        f           1x1                          0  function_handle
        r           2x1                         16  double
        x           1x1                         30  sym

Total is 5 elements using 54 bytes
```

In [50]:
```
x = int64(65536)
```

```
x = 65536
```

In [56]:
```
whos
```

```
Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
   c    ans         1x1                         16  double
        i           1x1                          8  double
        x           1x1                          8  int64

Total is 3 elements using 32 bytes
```

In [66]:
```
comp_num = 3+4j
```

```
comp_num =  3 + 4i
```

In [67]:
```
(4+3i)*(4+9i)
```

```
ans = -11 + 48i
```

In [74]:
```
imag(comp_num)
```

```
ans =  4
```

In [61]:
```
format short
```

In [75]:
```
name = 'I am Hu'
```

```
name = I am Hu
```

In [78]:
```
A = [1, 2, 3; ...
     3, 2, 1; ...
     4, 5, 3]
```

```
A =

   1   2   3
   3   2   1
   4   5   3
```

In [83]:
```
transpose(A)
```

```
ans =

   1   3   4
   2   2   5
   3   1   3
```

In [81]:
```
b = [1, 2,3]'
```

```
b =

   1
   2
   3
```

In [84]:
```
b = [1;2;3; 'a']
```

```
b =





a
```

In [ ]:
```

```

In [85]:
```
cell_array = {1,2,3,'a','2.43'}
```

```
cell_array =
{
  [1,1] =  1
  [1,2] =  2
  [1,3] =  3
  [1,4] = a
  [1,5] = 2.43
}
```

In [87]:
```
linspace(0,1, 11)
```

```
ans =

 Columns 1 through 8:

   0.00000   0.10000   0.20000   0.30000   0.40000   0.50000   0.60000   0.70000

 Columns 9 through 11:

   0.80000   0.90000   1.00000
```

In [89]:
```
t = (0:100)/100*pi;
```

In [91]:
```
a = [1 2 3]; b = [3 2 1];
```

In [92]:
```
a
```

```
a =

   1   2   3
```

In [93]:
```
b
```

```
b =

   3   2   1

```

In [96]:
```
a'*b
```

```
ans =

   3   2   1
   6   4   2
   9   6   3

```

In [98]:
```
dot(a,b')
```

```
ans =  10
```

In [101]:
```
cross(a',b)
```

```
warning: cross: taking cross product of column by row
warning: called from
    cross at line 59 column 7
ans =

  -4
   8
  -4

```

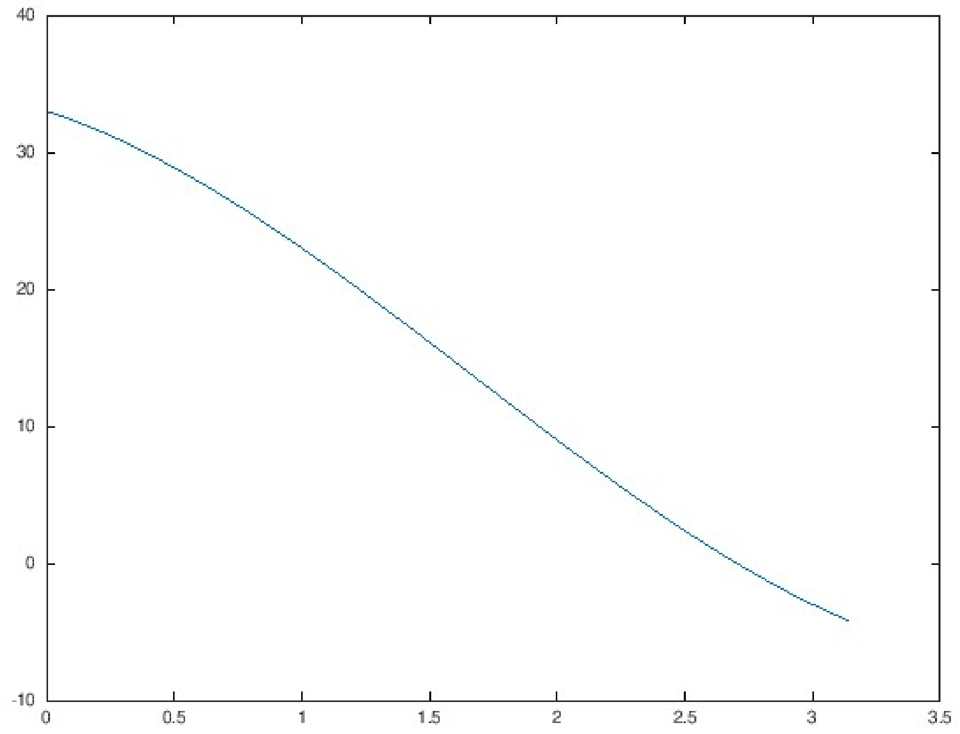
In [102]:
```
p = [1;-5; -6;33]
```

```
p =

    1
   -5
   -6
   33

```

In [103]:
```
polyval(p, 1.2)
```

```
ans =  20.328
```

In [105]:
```
plot(t, polyval(p, t))
```



In [62]:
```
pi
```

```
ans =  3.1416
```

功能1： 数值计算

In [107]:
```
mymat
```

In [109]:
```
M
```

```
M =

   1   2   3
   4   5   6
   7   8   9
```

对基本数值的处理：整数与浮点数

In [1]:
```
int16(12394723465432.23)  % try int32, int64 with different numbers
```

```
ans = 32767
```

In [9]:
```
int8(1.53)
```

```
ans = 2
```

对字符串的处理功能，如同c语言的string.h。当然，python和bash等具有正则表达式处理能力的编程环境在string的处理能力上远胜于此

例题（初等算术）：设三角形三边长分别为$a=4, b=3, c=2$,求三角形面积(记$s=\frac{a+b+c}{2}$,海伦公式$S=\sqrt{s(s-a)(s-b)(s-c)}$)

```
In [ ]:
```

例题（三角函数求值）：设$a=-24^o, b=75^o$,求$\frac{\sin(|a|+|b|)}{\sqrt{\tan(|a+b|)}}$

```
In [ ]: sin(30/180*pi)    % tips
```

MATLAB中默认的实数均为双精度（当然也可严格声明为整数或单精度），但是所有的函数运算均为浮点数运算!请区别如下命令的效果

```
In [5]: format long  % "输出"16位有效数字

In [2]: sin(30/180*pi)

        ans =  0.500000000000000

In [7]: format short   % 只"输出"5位有效数字

In [8]: sin(30/180*pi)

        ans =  0.50000

In [ ]:
```

例题（插值与拟合）-- 绘图见下一节， interp, spline, polyfit,etc.

```
In [ ]:

In [ ]:

In [ ]:
```

功能2： 数据可视化

```
In [ ]: Demo： 绘制函数图形

In [ ]: x = (0:20)/20*2*pi;

In [ ]: plot(x,sin(x),'b-*', x, cos(x), 'r-o');grid on;axis tight

In [ ]:

In [ ]:
```

例题：绘制心形曲线

```
In [ ]:
```

功能3、第三方功能包/工具箱

以符号计算为例，数学用户经常需要求解代数方程和微分方程，symbolic的 solve和dsolve提供了求解能力。octave的符号计算能力是由octave-forge提供的实现，需要独立安装。

```
In [1]: pkg load symbolic     # 第三方package需要显式导入
```

```
In [2]: syms x y
```

```
Symbolic pkg v2.9.0: Python communication link active, SymPy v1.5.1.
```

```
In [5]: [x,y] = solve(3*x+2*y==9,-2*x+5*y==11)
```

```
x = (sym)

  23
  --
  19

y = (sym)

  51
  --
  19
```

```
In [4]:
```

```
'perl' گⵣ␈Xǿeĳ
ļ
warning: help: Texinfo formatting filter exited abnormally; raw Texinfo source o
f help text follows...
'@sym/solve' is a function from the file D:\Octave\OCTAVE~1.0\mingw64\share\octa
ve\packages\symbolic-2.9.0\@sym\solve.m



Additional help for built-in functions and operators is
available in the online version of the manual.  Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at https://www.octave.org and via the help@octave.org
mailing list.
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

二、Octave变量

整形与浮点型

In [ ]:

In [ ]:

复数

In [ ]:

In [ ]:

重要的变量类型：字符串

In [ ]:
```
% test for strings
file = 'id.png';
path='/homw/work/app/';
strcat(path, file)    %  path + file  do not work!
```

How To : runover all files in a data folder? （Tips： files = dir())

In [ ]:

In [ ]:

In [ ]:

In [ ]:

重要的变量(数据结构)：矩阵与向量

1. 声明向量的方法: 给出全部元素 $a = 5, \sum_{n=1}^{+\infty} \frac{x^n}{n!}$

In [ ]:
```
vec_col = [0; 1; 2; 3; 4; 5; 6]    % add comma ";" if you do not want output
```

In [ ]:
```
vec_row = [0 .1 0.32 3.2 4.93847362 5e-3 6.]
```

```
In [ ]: vec_H = 0:.1:1            % 声明向量的常用方法2: given h
```

```
In [ ]: vec_N_lin = linspace(-1,1,7)   % 声明向量的常用方法3 :  given N
```

```
In [ ]: vec_N_log = logspace(0,2,10)    % spaced in log
```

```
In [ ]: plot(vec_N_log, zeros(length(vec_N_log),1),'ro')
```

```
In [ ]: clear    %% 注意回去执行前面的plot命令 -- 变量找不到了! !
```

```
In [ ]: clc; clf; %% 注意: 这两个命令只有在m自带开发环境下有用, 作用分别是清除命令窗口、清除图形窗口;
        % 若要在Jupyter中清除输出, 尝试右键菜单里的"Clear XXX"
```

```
In [ ]:
```

```
In [ ]:
```

2. 向量基本运算：加法、数乘、点积、外积、混合积 等

```
In [ ]: vec_row - 1
```

```
In [ ]: prod_in1 = dot(vec_col, vec_col)
```

```
In [ ]: prod_in2 = dot(vec_col, vec_row)
```

```
In [ ]: prod_in3 = vec_row * vec_col    % important !!, try to exchange the place?
```

```
In [ ]: prod_out1 = cross(vec_col, vec_N_lin')    % this is wrong, too long vector for defin
        ition of cross
```

```
In [ ]: cross([1,2,3]', [3;5;9])
```

How TO：实现混合积?

```
In [ ]: # dot(a, cross(b,c));
```

试理解区别:

```
In [ ]: vec_row * vec_col
```

```
In [ ]: vec_col * vec_row
```

矩阵(Matrix)

矩阵声明方法1：直接输入所有元素（想一下，在什么时候有用？）

```
In [ ]: A_Mat =  [2 3 0; 3 -1 2; 3 0 -2]
```

In [ ]:
```
A_Mat_alt = [2, 3, 0; ...
             3,-1, 2; ...
             3, 0,-2]
```

矩阵声明方法2： 列向量 × 行向量 = 矩阵

In [ ]:
```
B_Mat = vec_col * vec_row
```

这个特性有时能让表达式变得简洁，高效地表达数学用户的意图。

In [ ]:
```

```

In [ ]:
```

```

用户可以象对c语言的数组一样访问矩阵或向量中的元素，如a(3)表示向量a的第三个元素，而A(3,2)表示矩阵A的第三行、第二列的元素。与c数组不同的是MATLAB矩阵和向量的下标是从1开始计数的！除了上述最基本的矩阵元素操作外，MATLAB提供了其他矩阵操作指令以简化用户操作，如

In [ ]:
```
A = rand(5,5);
B = A(2:3, 1:2);    % 取出一个2x2的子矩阵
```

In [ ]:
```
A(2,:) = [];           % 删除第二行
```

In [ ]:
```
A = [A, zeros(size(A,1),1)]   %矩阵右边拼接一列
```

In [ ]:
```
B = A'
```

In [ ]:
```
C = transpose(A)   % 矩阵转置
```

In [ ]:
```

```

In [11]:
```
ones(5,5)
```

```
ans =

   1   1   1   1   1
   1   1   1   1   1
   1   1   1   1   1
   1   1   1   1   1
   1   1   1   1   1
```

In [12]:
```
zeros(5,5)
```

```
ans =

   0   0   0   0   0
   0   0   0   0   0
   0   0   0   0   0
   0   0   0   0   0
   0   0   0   0   0
```

In [13]:
```
mat_magic = magic(6)
```

```
mat_magic =

   35    1    6   26   19   24
    3   32    7   21   23   25
   31    9    2   22   27   20
    8   28   33   17   10   15
   30    5   34   12   14   16
    4   36   29   13   18   11
```

In [14]:
```
diag(mat_magic)
```

```
ans =

   35
   32
    2
   17
   14
   11
```

In [15]:
```
diag(diag(mat_magic))
```

```
ans =

Diagonal Matrix

   35    0    0    0    0    0
    0   32    0    0    0    0
    0    0    2    0    0    0
    0    0    0   17    0    0
    0    0    0    0   14    0
    0    0    0    0    0   11
```

In [111]:
```
tril(magic(3))
```

```
ans =

   8   0   0
   3   5   0
   4   9   2
```

In [119]:
```
K = randn(4,5)
```

```
K =

  -0.612098  -0.615771  -0.395308   0.392939  -0.492821
   0.835864  -0.990653  -1.759737   1.590874  -0.716475
  -0.211443   0.781518  -0.332782   0.447544  -0.247363
   1.615263  -0.059295  -1.361493   1.847058  -0.780874
```

```
In [125]: K(2:end-1,3:end)
```

```
ans =

  -1.75974   1.59087  -0.71648
  -0.33278   0.44754  -0.24736
```

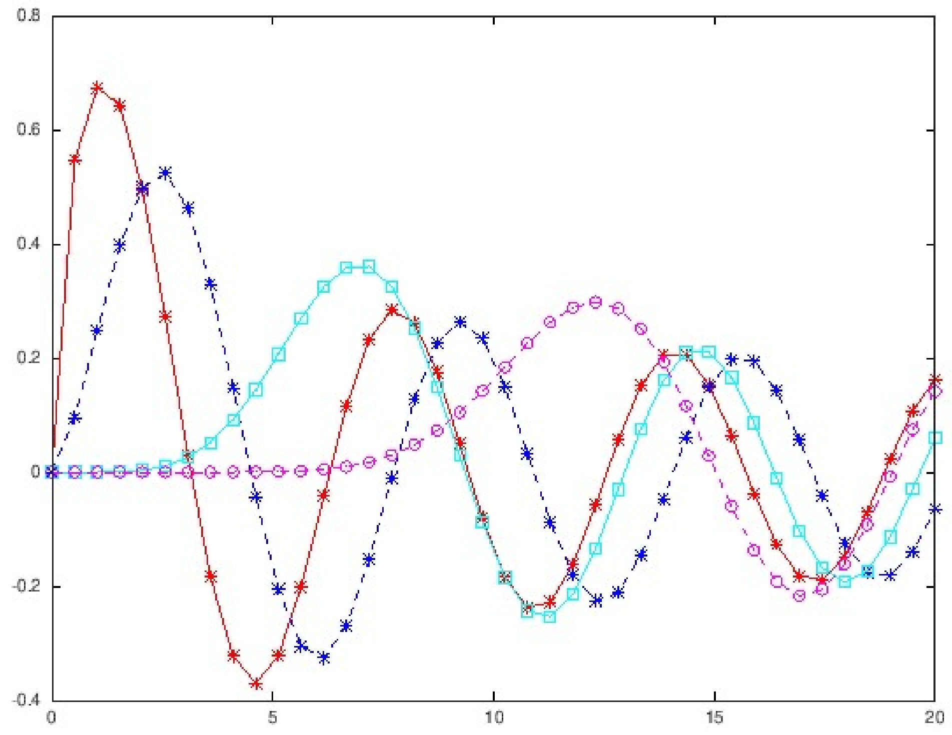
```
In [130]: [ones(3,3),rand(3,1); zeros(3,4)]
```

```
ans =

   1.00000   1.00000   1.00000   0.48638
   1.00000   1.00000   1.00000   0.52184
   1.00000   1.00000   1.00000   0.26431
   0.00000   0.00000   0.00000   0.00000
   0.00000   0.00000   0.00000   0.00000
   0.00000   0.00000   0.00000   0.00000
```

三、 函数(Function)

除了如前面我们已经用过的打印函数/命令、绘图函数/命令以及常用的数学库函数之外，特殊函数在工程计算中有广泛应用的

```
In [6]: t = linspace(0,20,40);
        plot(t,besselj(0.5, t), 'r*-'); hold on;
        plot(t,besselj(1.5, t), 'b*--');
        plot(t,besselj(5.5, t), 'cs-');
        plot(t,besselj(10.5, t), 'mo--');
```



这里我们看一个工程计算的例子

In [ ]:
```
% TO DO ...
```

In [131]:
```
load triangle
```

In [132]:
```
whos
```

```
Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
        K           4x5                        160  double
        M           3x3                         72  double
        ans         6x4                        192  double
        tri       240x3x2                    11520  double

Total is 1493 elements using 11944 bytes
```

In [136]:
```
tri2 = flipdim(tri,);
```

```
error: flip: DIM must be a positive integer
error: called from
    flip at line 70 column 5
    flipdim at line 36 column 5
```

In [139]:
```
v1 = tri(1,3,:)
```

```
v1 =

ans(:,:,1) = 0
ans(:,:,2) =  0.10000
```

In [140]:
```
whos
```

```
Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
        K           4x5                        160  double
        M           3x3                         72  double
        ans         1x3x2                       48  double
        tri       240x3x2                    11520  double
        tri2        1x3                         24  double
        v1          1x1x2                       16  double

Total is 1480 elements using 11840 bytes
```

In [141]:
```
M\[3.4 2.4 2.4]'
```

```
warning: matrix singular to machine precision, rcond = 1.54198e-18
ans =

  -1.838889
  -0.055556
   1.727778
```

In [142]:
```
(M+0.5*eye(3)) \ [3.4 2.4 2.4]'
```

```
ans =

  -3.05366
  -0.89756
   3.25854
```

In [144]:
```
cond(M+0.5*eye(3))
```

```
ans =    3.8131e+16
```

自定义函数

In [7]:
```
function y = my_func(x)
y = 3*x.^2 + 2*x + 18;
end
```

In [8]:
```
my_func([1.2, 1.3, 1.5])
```

```
ans =

   24.720   25.670   27.750
```

In [146]:
```
A = 0.5*ones(3,3)
```

```
A =

   0.50000   0.50000   0.50000
   0.50000   0.50000   0.50000
   0.50000   0.50000   0.50000
```

In [145]:
```
M
```

```
M =

   1   2   3
   4   5   6
   7   8   9
```

In [147]:
```
A*M
```

```
ans =

   6.0000   7.5000   9.0000
   6.0000   7.5000   9.0000
   6.0000   7.5000   9.0000
```

In [150]:
```
M./A
```
```
ans =

    2    4    6
    8   10   12
   14   16   18
```

In [153]:
```
M.^2
```
```
ans =

    1    4    9
   16   25   36
   49   64   81
```

In [152]:
```
M*M
```
```
ans =

    30    36    42
    66    81    96
   102   126   150
```

In [155]:
```
M.^A
```
```
ans =

   1.0000   1.4142   1.7321
   2.0000   2.2361   2.4495
   2.6458   2.8284   3.0000
```

In [159]:
```
[5,3,7] >= [5,7,1]
```
```
ans =

  1  0  1
```

In [163]:
```
function y = my_func(x)
    y = 3*x.*x + 2*x + 18;
%    return y;
endfunction
```

In [161]:
```
my_func(1.2)
```
```
ans =  24.720
```

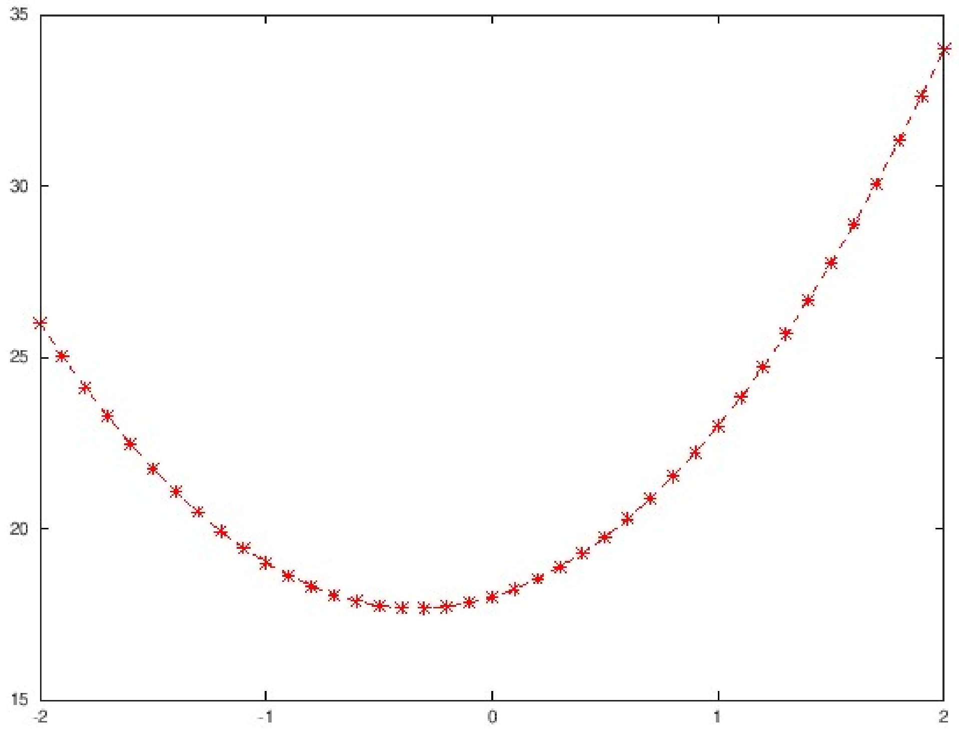
内联函数

In [9]:
```
my_func_inline = @(x) 3*x.^2 + 2*x + 18;
```

```
In [10]: my_func_inline([1.2, 1.3, 1.5])
```

```
ans =

   24.720   25.670   27.750
```

```
In [164]: t = -2:.1:2; plot(t,my_func(t),'r--*')
```



```
In [ ]:
```

测试其他第三方包

In [12]:
```
pkg list
```

```
Package Name          | Version | Installation directory
----------------------+---------+-----------------------
                audio  |   2.0.0 | ...\mingw64\share\octave\packages\audio-2.0.0
       communications  |   1.2.2 | ...\share\octave\packages\communications-1.2.2
              control  |   3.2.0 | ...\mingw64\share\octave\packages\control-3.2.0
       data-smoothing  |   1.3.0 | ...\share\octave\packages\data-smoothing-1.3.0
             database  |   2.4.4 | ...\share\octave\packages\database-2.4.4
            dataframe  |   1.2.0 | ...\share\octave\packages\dataframe-1.2.0
                dicom  |   0.2.2 | ...\mingw64\share\octave\packages\dicom-0.2.2
            financial  |   0.5.3 | ...\share\octave\packages\financial-0.5.3
                 fits  |   1.0.7 | ...\mingw64\share\octave\packages\fits-1.0.7
  fuzzy-logic-toolkit  |   0.4.5 | ...\octave\packages\fuzzy-logic-toolkit-0.4.5
                   ga  |  0.10.1 | ...\mingw64\share\octave\packages\ga-0.10.1
              general  |   2.1.0 | ...\mingw64\share\octave\packages\general-2.1.0
        generate_html  |   0.3.1 | ...\share\octave\packages\generate_html-0.3.1
             geometry  |   3.0.0 | ...\share\octave\packages\geometry-3.0.0
                  gsl  |   2.1.1 | ...\mingw64\share\octave\packages\gsl-2.1.1
                image  |  2.10.0 | ...\mingw64\share\octave\packages\image-2.10.0
   instrument-control  |   0.4.0 | ...\octave\packages\instrument-control-0.4.0
             interval  |   3.2.0 | ...\share\octave\packages\interval-3.2.0
                   io  |  2.4.13 | ...\mingw64\share\octave\packages\io-2.4.13
       linear-algebra  |   2.2.3 | ...\share\octave\packages\linear-algebra-2.2.3
                 lssa  |   0.1.3 | ...\mingw64\share\octave\packages\lssa-0.1.3
                ltfat  |   2.3.1 | ...\mingw64\share\octave\packages\ltfat-2.3.1
              mapping  |   1.2.1 | ...\mingw64\share\octave\packages\mapping-1.2.1
        miscellaneous  |   1.3.0 | ...\share\octave\packages\miscellaneous-1.3.0
                  nan  |   3.4.5 | ...\mingw64\share\octave\packages\nan-3.4.5
               netcdf  |  1.0.12 | ...\mingw64\share\octave\packages\netcdf-1.0.12
                nurbs  |  1.3.13 | ...\mingw64\share\octave\packages\nurbs-1.3.13
                  ocs  |   0.1.5 | ...\mingw64\share\octave\packages\ocs-0.1.5
               odepkg  |   0.8.5 | ...\mingw64\share\octave\packages\odepkg-0.8.5
                optim  |   1.6.0 | ...\mingw64\share\octave\packages\optim-1.6.0
          optiminterp  |   0.3.5 | ...\share\octave\packages\optiminterp-0.3.5
           quaternion  |   2.4.0 | ...\share\octave\packages\quaternion-2.4.0
             queueing  |   1.2.6 | ...\share\octave\packages\queueing-1.2.6
               signal  |   1.4.1 | ...\mingw64\share\octave\packages\signal-1.4.1
              sockets  |   1.2.0 | ...\mingw64\share\octave\packages\sockets-1.2.0
            sparsersb  |   1.0.6 | ...\share\octave\packages\sparsersb-1.0.6
              specfun  |   1.1.0 | ...\mingw64\share\octave\packages\specfun-1.1.0
              splines  |   1.3.3 | ...\mingw64\share\octave\packages\splines-1.3.3
           statistics  |   1.4.1 | ...\share\octave\packages\statistics-1.4.1
                  stk  |   2.6.1 | ...\mingw64\share\octave\packages\stk-2.6.1
              strings  |   1.2.0 | ...\mingw64\share\octave\packages\strings-1.2.0
               struct  |  1.0.16 | ...\mingw64\share\octave\packages\struct-1.0.16
             symbolic *|   2.9.0 | ...\share\octave\packages\symbolic-2.9.0
               tisean  |   0.2.3 | ...\mingw64\share\octave\packages\tisean-0.2.3
                  tsa  |   4.6.2 | ...\mingw64\share\octave\packages\tsa-4.6.2
                video  |   1.2.4 | ...\mingw64\share\octave\packages\video-1.2.4
              windows  |   1.4.0 | ...\mingw64\share\octave\packages\windows-1.4.0
               zeromq  |   1.5.0 | ...\mingw64\share\octave\packages\zeromq-1.5.0
```

In [13]:
```
pkg load image
pkg load statistics
```

In [14]:
```
pkg load optim
```

上述三个包在后续讲座中也会被用到.可以用pkg install XXX 或手工下载安装包到本地进行安装。我这个系统中，symbolic包就是本地安装的。

In [179]:
```
list = [1,2,3,1,2,3];
for name = list % 1:10
    printf('%f, \n',name);
end
```

```
1.000000,
2.000000,
3.000000,
1.000000,
2.000000,
3.000000,
```

In [169]:
```
list
```

```
list = ahuzhang
```

四、其他

In [ ]:

In [180]:
```
function y = df(x)
    y = 6*x - exp(x);
end
```

In [181]:
```
function y = f(x)
 y = 3*x^2 - exp(x);
end
```

In [186]:
```
iter = 0;
err = 1;
x0 = 2.0
x = x0;
format long;
while(err > 1e-8 && iter < 20)
    x0 = x;
    x = x0 - df(x0)\f(x0);
    err = norm(x - x0);
    iter = iter + 1;
    fprintf('iter %d: x = %18.15f, f(x) = %18.15f\n', iter, x, f(x));
end
```

```
x0 =  2
iter 1: x =  1.000000000000000, f(x) =  0.281718171540955
iter 2: x =  0.914155281832543, f(x) =  0.012372566882759
iter 3: x =  0.910017665783406, f(x) =  0.000030034837379
iter 4: x =  0.910007572548888, f(x) =  0.000000000179075
iter 5: x =  0.910007572488709, f(x) =  0.000000000000000
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

1. 安装octave-gui环境，并用pkg install 安装 symbolic 包（默认安装不包含）
2. 安装python3.6, 并用pip安装jupyter notebook, multiple_kernel以及octave_kernel等
3. 前面和课件中提到的命令和例子自己演练一遍，记录你的结果

```
In [ ]:
```