第四章 变量类型和函数

Variable Types and Functions

4.1 变量类型（Class）

- 字符串变量（char）
- 数值型变量，包括双精度（double）、单精度（single）等。

```
>> clear all
>> str='Every good boy does fine.';
>> x=1:5:100;
>> class(str)

ans =

char

>> class(x)

ans =

double
```

- Whos能够查看变量类型（class）以及变量所占的内存（Bytes）
  - 每个字符占2 bytes

```
>> whos
  Name        Size                Bytes  Class     Attributes

  ans         1x6                    12  char
  str         1x25                   50  char
  x           1x20                  160  double
```

- Help class可提供所有变量类型
- 字符、数值、逻辑（logical）、结构（struct）、单元(cell)等

```
>> help class
 CLASS  Return class name of object.
    S = CLASS(OBJ) returns the name of the class of object OBJ.

    Possibilities are:
      double          -- Double precision floating point number array
                         (this is the traditional MATLAB matrix or array)
      single          -- Single precision floating point number array
      logical         -- Logical array
      char            -- Character array
      cell            -- Cell array
      struct          -- Structure array
      function_handle -- Function Handle
      int8            -- 8-bit signed integer array
      uint8           -- 8-bit unsigned integer array
      int16           -- 16-bit signed integer array
      uint16          -- 16-bit unsigned integer array
      int32           -- 32-bit signed integer array
      uint32          -- 32-bit unsigned integer array
      int64           -- 64-bit signed integer array
      uint64          -- 64-bit unsigned integer array
      <class_name>    -- MATLAB class name for MATLAB objects
      <java_class>    -- Java class name for java objects
```

4.2 结构变量（struct）

- 如何把实验被试的个人信息保存到一个变量里？

```
>> subject.id = 213;
subject.age = 28;
subject.hand = 0;
subject.gender  = 1;
>> subject
```

变量名称

```
subject =
```

子变量（fields）/字段名称

```
    id: 213
   age: 28
  hand: 0
gender: 1
```

```
>> subject.age

ans =

    28
```

- 如何把所有被试的个人信息保存在一个变量里？
  - 结构变量本身也可以是数组/向量或矩阵

```
>> subject(2).id = 301;
subject(2).age = 25;
subject(2).hand = 0;
subject(2).gender = 0;

subject(3).age = 43;
subject(3).id = 200;
subject(3).hand = 1;
subject(3).gender = 0;
```

```
>> subject(1).data = rand(1,5);
subject(2).data = rand(1,7);
subject(3).data = rand(1,4);
```

子变量data的内容是数组/向量

- 对结构变量的内容进行操作。
  - 例如，计算每个左利手的被试的所有实验试次（trial）的平均数据。

```
>> count = 0;
for i=1:length(subject)
    if subject(i).hand == 0
        count = count+1;
        meanData(count) = mean(subject(i).data);
    end
end
meanData

meanData =

    0.4513    0.5012
```

是否为左利手被试

共有几个左利手被试

计算平均数据

试一试：

meanData(1,count)=subject(i).id;

meanData(2,count)=mean(subject(i),data);

mean是matlab自带，计算平均值

std是matlab自带，可计算标准差

- 结构变量的子变量可以是任何变量类型。

```
>> subject(1).hand = 'left';
>> subject(1)

ans =

        id: 213
       age: 28
      hand: 'left'
    gender: 1
      data: [0.1588 0.5997 0.4789 0.8852 0.1340]
```

子变量hand的类型是字符串

- 结构变量的子变量（fields）也可以是结构变量！

```
>> animalSubject(1).id = 23;
animalSubject(1).data = rand(1,5);
animalSubject(2).id = 46;
animalSubject(2).data = rand(1,4);
>> allSubjects.human = subject;
allSubjects.animal = animalSubject   ;
allSubjects

allSubjects =

     human: [1x3 struct]
    animal: [1x2 struct]
```

```
>> allSubjects.human(3).age

ans =

    43
```

子变量animal和自变量human的类型都是结构变量

4.3 单元变量（cell）

- 如何把不同长度的字符串放在一个变量里？

```
>> days = ['Monday','Tuesday','Wednesday','Thursday','Friday']

days =

MondayTuesdayWednesdayThursdayFriday

>> days = ['Monday';'Tuesday';'Wednesday';'Thursday';'Friday']
??? Error using ==> vertcat
CAT arguments dimensions are not consistent.
```

```
>> days = {'Monday','Tuesday','Wednesday','Thursday','Friday'}

days =

    'Monday'    'Tuesday'    'Wednesday'    'Thursday'    'Friday'
```

- 查看/索引单元变量的内容

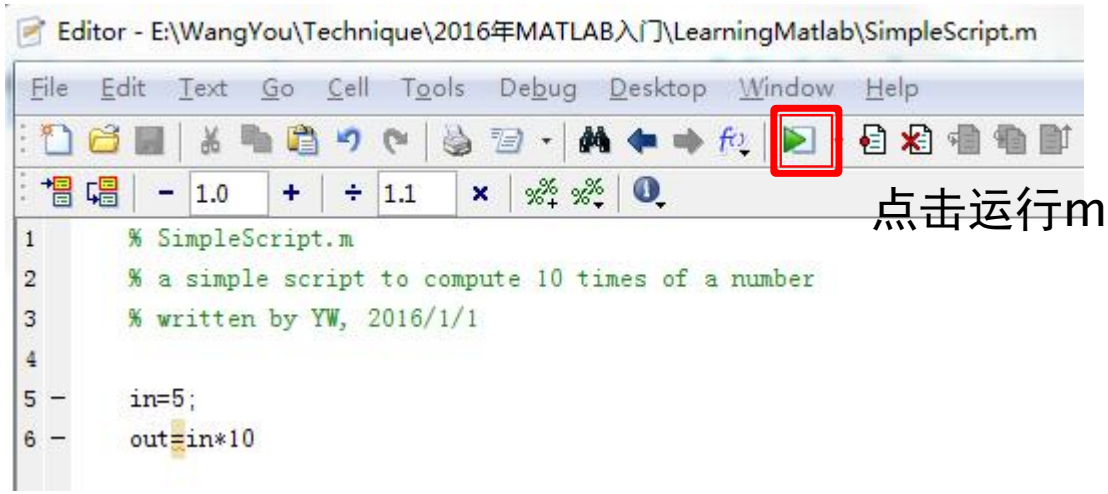
```
>> days{1}

ans =

Monday

>> days{1}(2)

ans =

o

>> x=days{1};x(2)

ans =

o
```

- 单元变量的内容可以是任何变量类型。

```
>> myCell{1} = 'this is a string of characters.';
myCell{2} = 2.^(1:5);
myCell{3} = allSubjects;
myCell{3}
newVector = myCell{2};
newVector(3)

ans =

     human: [1x3 struct]
    animal: [1x2 struct]


ans =

     8
```

单元变量myCell的内容包括字符串、向量、结构变量

- 新建一个m文件SimpleScript（脚本），计算某个数的10倍。

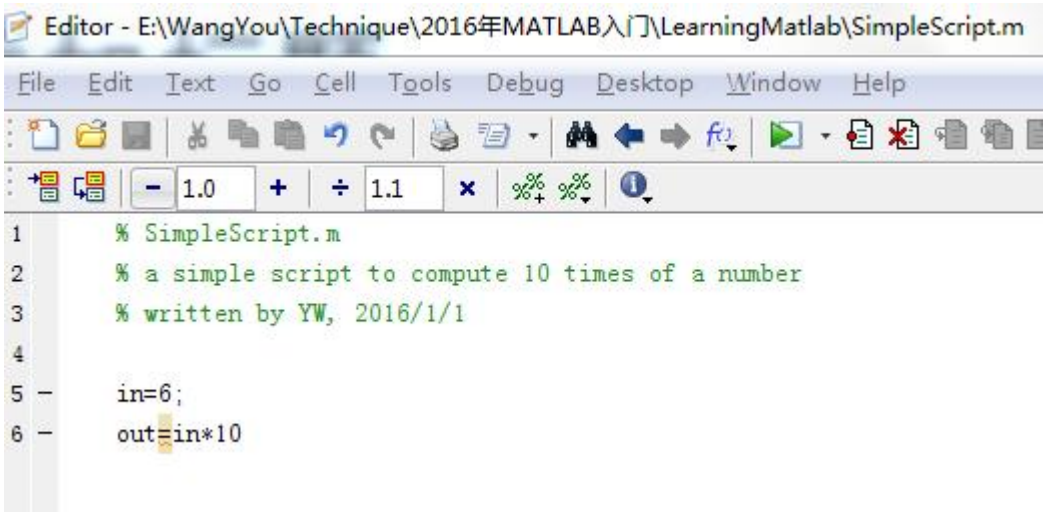


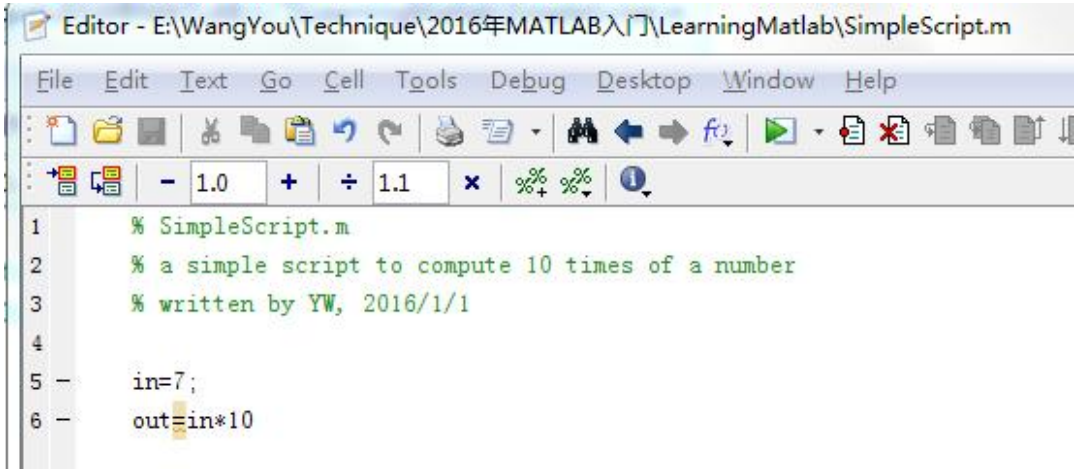
点击运行m文件中的命令语句

```
>> SimpleScript

out =

    50
```

在命令窗口输入m文件名

如果给定的数值不断改变。。。。

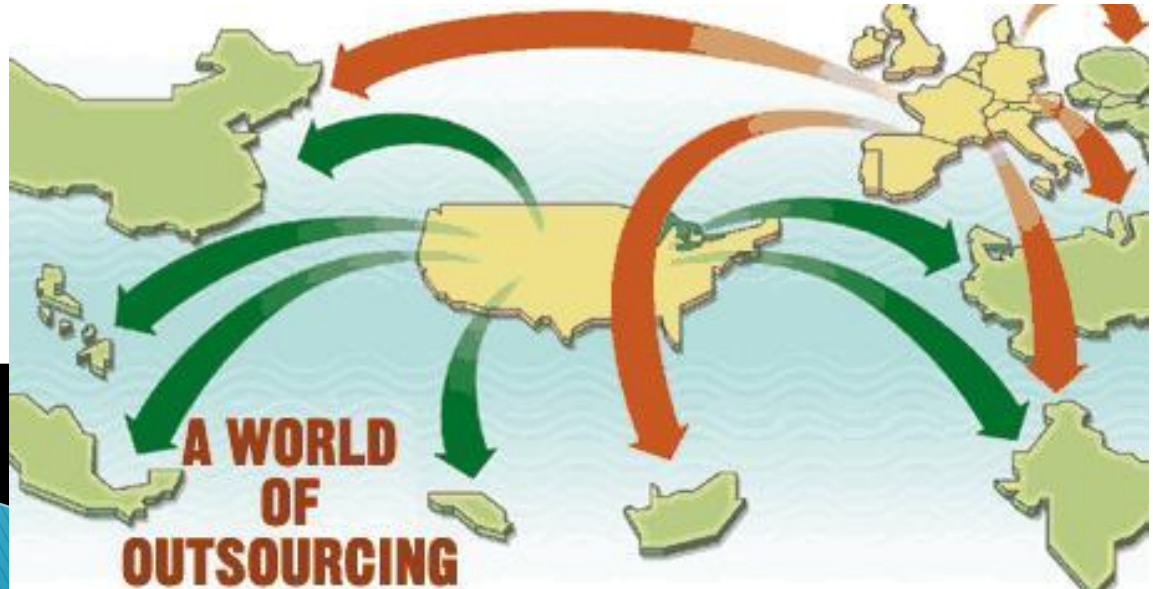

Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\SimpleScript.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
% SimpleScript.m
% a simple script to compute 10 times of a number
% written by YW, 2016/1/1
in=6;
out=in*10


Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\SimpleScript.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
% SimpleScript.m
% a simple script to compute 10 times of a number
% written by YW, 2016/1/1
in=7;
out=in*10

有没有更好地方法？

4.4 函数（function）

- 什么是函数？
  - 任务“外包”，不同任务交给不同函数完成
  - 提供给函数所需要的变量即可
  - 函数的整个运行是隐藏的
  - 函数将反馈给你想得到的变量



- 为什么要用函数？
  - 使程序的结构更清楚更容易懂
  - 使程序更短，只需调用函数的名字就可以
  - 节省时间，不用重复写同样的语句。

- 什么时候需要考虑建立函数呢？
  - 这些语句是否今后还可能再用到？
  - 在当前这个程序中，这些语句是否可能用到不止一次？
  - 不管这些语句有多短，都有必要建立函数！

4.4.1 创建函数

- 新建一个m文件，命名为SimpleFunction（函数文件），保存到LearningMatlab文件夹。
  - 把输入给函数的数值*10倍，得到输出的数值

定义函数　　输出变量（output argument）

```
function out=SimpleFunction(in)
% a very simple function
% written by YW 2/2014

out=10*in;
```

输入变量（input argument）

m文件包括脚本m文件和函数m文件

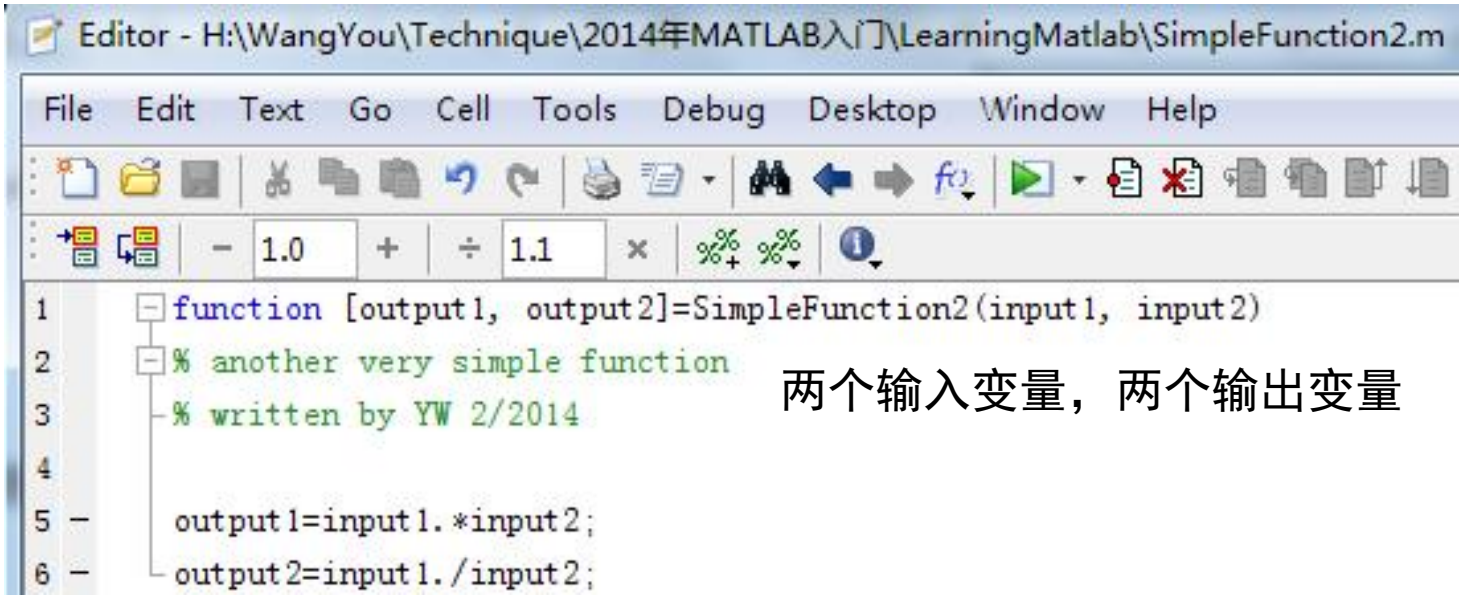
注意：
Octave可以用Ctrl+C来停止命令的运行

- 在命令窗口调用函数
  - 当前目录/路径（Current directory）必须是函数所在的文件夹
  - 函数输出/输入的变量（例如，my_out）不一定要与函数m文件中的对应变量（例如，out）一致。

```
>> my_out=SimpleFunction(3)
??? Undefined function or method 'SimpleFunction' for input arguments of type 'double'.

>> pwd

ans =

C:\Users\Administrator\Documents\MATLAB

>> cd H:\WangYou\Technique\2014年MATLAB入门\LearningMatlab
>> my_out=SimpleFunction(3)

my_out =

    30
```

```
>> inval=5;
my_out=SimpleFunction(inval);
my_out

my_out =

    50
```

- 新建一个SimpleFunction2的m文件
- 函数可以接收多个输入变量，也可以给出多个输出变量
  - 输入变量数目和输出变量数目多少无关，不需要相等



两个输入变量，两个输出变量

```
>> myinput1=3;
myinput2=4;
[myoutput1, myoutput2]=SimpleFunction2(myinput1, myinput2)

myoutput1 =

    12


myoutput2 =

    0.7500
```

```
>> vect1 = [1:5];
vect2  = [2:2:10];
[myoutput1, myoutput2]=SimpleFunction2(vect1, vect2)

myoutput1 =

     2     8    18    32    50


myoutput2 =

    0.5000    0.5000    0.5000    0.5000    0.5000
```

4.4.2其他有用的函数操作

- 对矩阵的数值进行标准化

```
function outMat=scaleMat(inMat, range)
% outMat=scaleMat(inMat, [range])
% Scales a matrix to a new range of values.
% Inputs:
%    inMat: unscaled matrix
%    range: 1*2 vector containing the low and high values for the scaled
%    matrix. Default is [0,1].
%
% Output: scaled matrix with minimum value range (1), and max value range
% (2).
%
% written by YW 2/2014
```

函数m文件的标题信息非常详细

range用[]是为了说明range这个变量不是必须给出，不给出时函数提供了默认值

```
% Deal with default values for range if none are provided
if ~exist ('range','var')
    range=[0,1];
end

% minimum value of input matrix
minVal = min (inMat(:));

% maximum value of input matrix
maxVal = max (inMat(:));

% scale the matrix to range between 0 and 1.
outMat = (inMat-minVal)/(maxVal-minVal);

% scale this new matrix to values in 'range'
outMat=(range(2)-range(1))*outMat+range(1);
```

判断是否存在输入变量range，如果没有输入变量，range=[0,1]

用min寻找输入变量inMat矩阵中的最小值

用max寻找输入变量inMat矩阵中的最小值

把inMat的数值转换为0-1之间的数值

把inMat的数值转换为range所设定范围的数值

- 如果输入变量没有给出，会使用默认的值

```
>> mat = [1,2;3,4]
scaleMat(mat,[0,12])

mat =

     1     2
     3     4


ans =

     0     4
     8    12
```

```
>> scaleMat(mat)

ans =

         0    0.3333
    0.6667    1.0000
```

- 能够使用默认值，是用exist命令实现的
- Exist可以检测某个变量是否存在（返回逻辑变量0/1）

```
% Deal with default values for range if none are provided
if ~exist ('range','var')
    range=[0,1];
end
```

```
>> help exist
 EXIST  Check if variables or functions are defined.
    EXIST('A') returns:
      0 if A does not exist
      1 if A is a variable in the workspace
      2 if A is an M-file on MATLAB's search path.  It also returns 2 when
          A is the full pathname to a file or when A is the name of an
          ordinary file on MATLAB's search path
      3 if A is a MEX-file on MATLAB's search path
      4 if A is a MDL-file on MATLAB's search path
      5 if A is a built-in MATLAB function
      6 if A is a P-file on MATLAB's search path
      7 if A is a directory
      8 if A is a Java class
```

```
>> clear all
x = pi;
exist('x','var')

ans =

     1

>> exist('y', 'var')

ans =

     0
```

- 在脚本m文件中调用函数。
  - 例如，在SineInAperture3的脚本m文件中，调用了MakeSineAperture函数，该函数可生成正弦光栅（grating），一种常用的视觉实验刺激

```
% SineInAperture3.m
% creates an image of apertured vertical sinusoids in a tilted array
% written by YW, 2/2014

% information about the sinusoids
clear all
close all
x=linspace(-pi, pi,100);
sf=[3 6 9 12]; % spatial freq in cycles per image
rad=2;

% creats a 100*100*length(sf) matrix containing two 2D apertured sinusoids
% of different spatial frequncies.
for s=1:length(sf)
    sinewave2D(:,:,s)= MakeSineAperture(x,sf(s),rad);
end
```

提前创建函数需要的输入变量x，sf，rad

调用函数

输出变量sinewave2D是一个三维矩阵，100*100*4

- SineInAperture3程序m文件-续前页

```
% initialize the tile matrix by filling it with zeros
imgsize=length(x);
ntiles=4;
sep=30;
tilesize=(ntiles*(imgsize+sep))+sep;
tilematrix=zeros(tilesize);
startpos=sep:sep+imgsize:length(tilematrix)-1;

for rtile=1:ntiles
    for ctile=1:ntiles
        tilematrix(startpos(rtile):startpos(rtile)+imgsize-1,startpos(ctile):startpos(ctile)+imgsize-1)=sinewave2D(:,:,rtile);
    end
end
imagesc(tilematrix);
colormap(gray(256));
axis off;
```

ntiles设置每行有4个正弦光栅矩阵（每个矩阵大小为100*100）

sep设置每个正弦光栅之间的间隔

tilesize预设最终的tilematrix矩阵总大小

startpos设置每个正弦光栅矩阵‘放置’在tilematrix总矩阵中的起始位置

把正弦光栅‘放置’在tilematrix对应的位置＝给对应位置赋值

试一试：

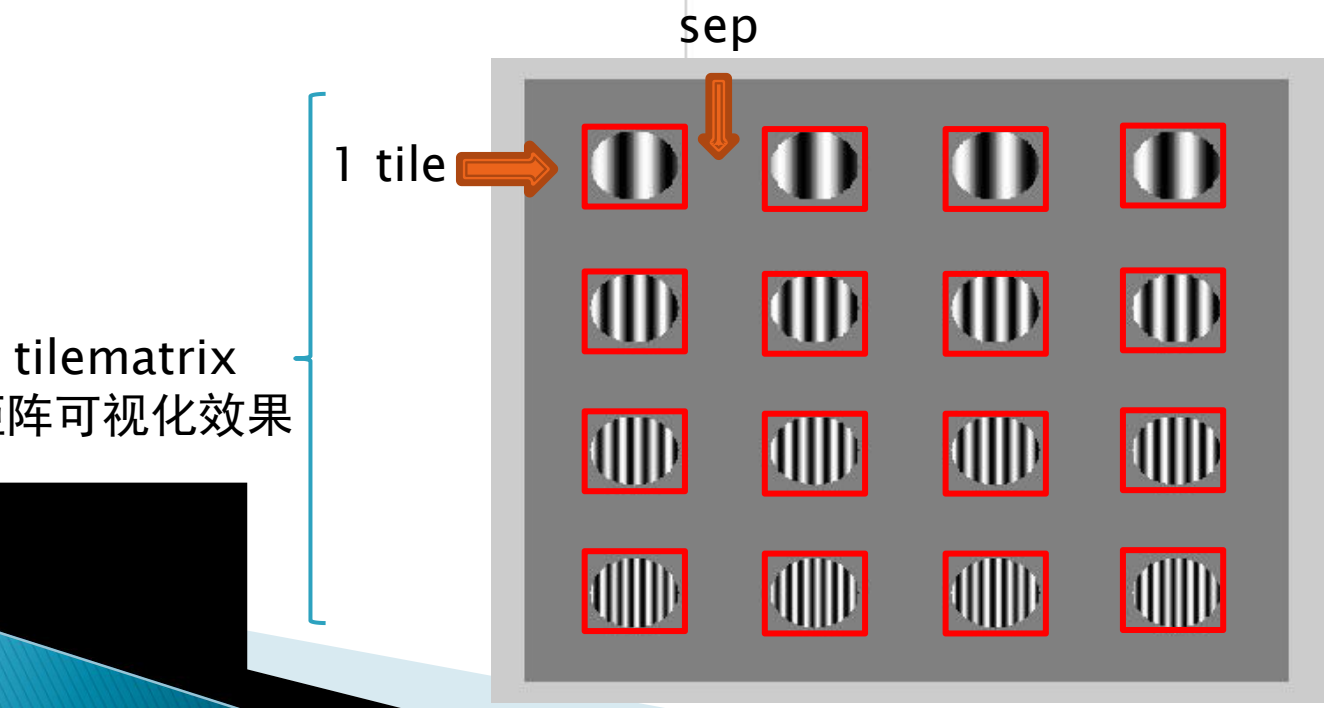
help image

help imagesc

help colormap

tilematrix

矩阵可视化效果



- MakeSineAperture函数，生成亮度以一定空间频率（spatial frequency, sf）正弦变化的具有一定大小(radius)的正弦光栅（sinusoidal grating）

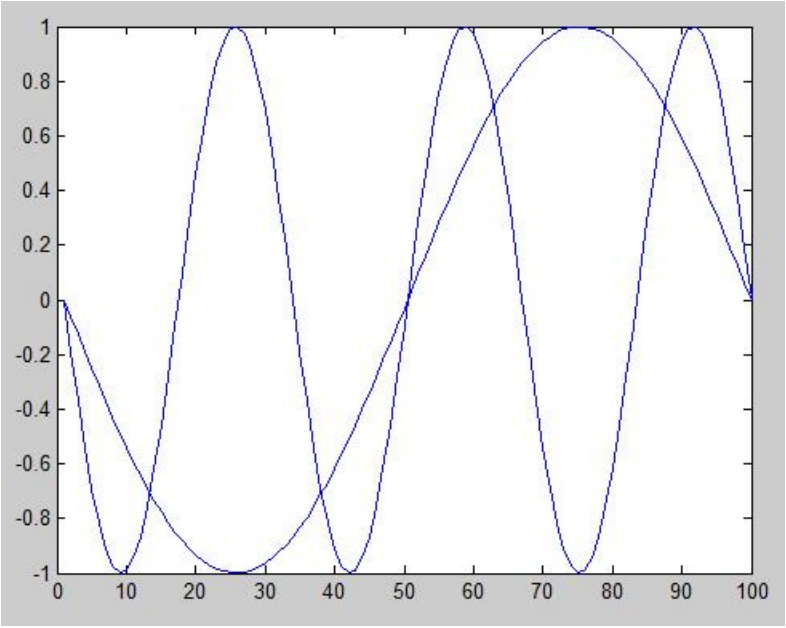
```
function sinewave2D = MakeSineAperture (xval, sf,radius)

% function that creates a vertical sinewave windowed by a circular aperture
%
% Inputs:
%   xval: for which the sinusoids is computed
%   sf:   the saptial frequecy of the sinusoid
%   radius: the radiusl of the aperture in pixels
% Outputs: a 2D sinusoidal grating image

onematrix=ones(size(xval));
sinewave=sin(xval*sf);
sinewave2D=(onematrix'*sinewave);
for r=1:length(xval)
    for c=1:length(xval)
        if xval(r).^2+xval(c).^2>radius^2
            sinewave2D(r,c)=0;
        end
    end
end
```

创建矩阵onematrix，所有元素的值均为1

通过计算得到正弦光栅矩阵sinewave2D

将超过以radius为半径的圆形范围外的位置都赋值为0

```
>> clear all;
close all;

xval=linspace(-pi, pi,100);
sf=3;
radius=2;

onematrix=ones(size(xval));
sinewave=sin(xval*sf)
```



生成sf个周期的正弦数值
周期越多，光栅越密

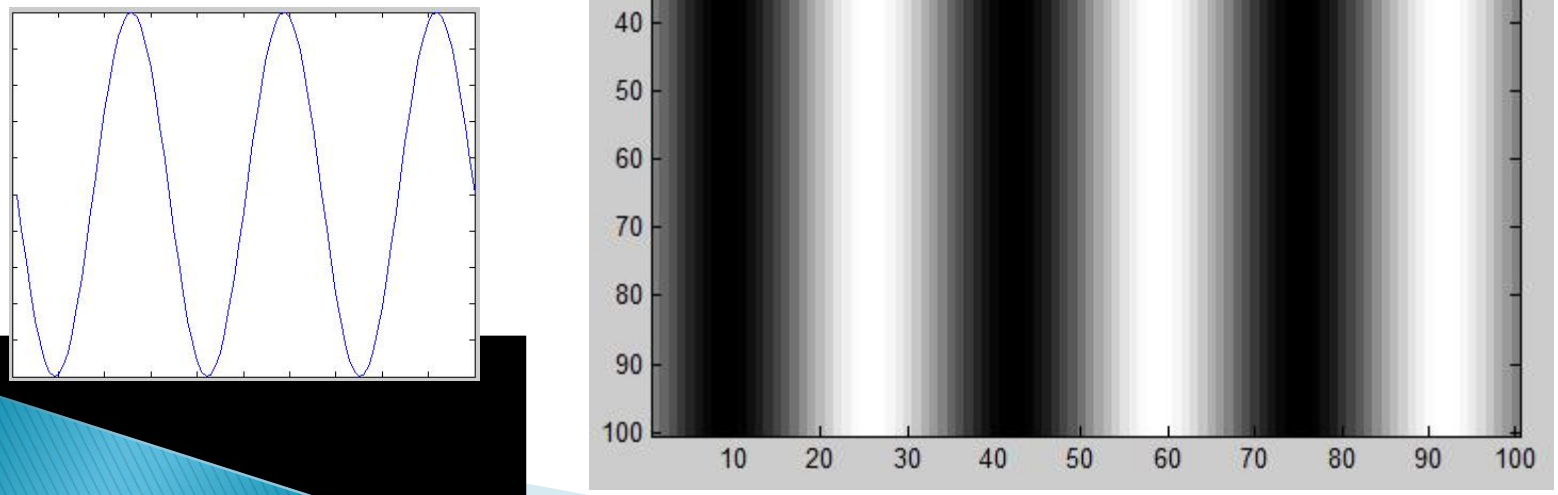
```
sinewave =

  Columns 1 through 22

   -0.0000   -0.1893   -0.3717   -0.5406   -0.6901   -0.8146   -0.9096   -0.9718   -0.9989   -0.9898   -0.9450   -0.8660   -0.7557   -0.6182   -0.4582   -0.2817   -0.0951    0.0951

  Columns 23 through 44

    0.8660    0.9450    0.9898    0.9989    0.9718    0.9096    0.8146    0.6901    0.5406    0.3717    0.1893    0.0000   -0.1893   -0.3717   -0.5406   -0.6901   -0.8146   -0.9096

  Columns 45 through 66

   -0.8660   -0.7557   -0.6182   -0.4582   -0.2817   -0.0951    0.0951    0.2817    0.4582    0.6182    0.7557    0.8660    0.9450    0.9898    0.9989    0.9718    0.9096    0.8146

  Columns 67 through 88

    0.0000   -0.1893   -0.3717   -0.5406   -0.6901   -0.8146   -0.9096   -0.9718   -0.9989   -0.9898   -0.9450   -0.8660   -0.7557   -0.6182   -0.4582   -0.2817   -0.0951    0.0951

  Columns 89 through 100

    0.8660    0.9450    0.9898    0.9989    0.9718    0.9096    0.8146    0.6901    0.5406    0.3717    0.1893    0.0000
```

```
>> sinewave2D=(onematrix'*sinewave)

sinewave2D =
```

生成行列相等的矩阵，每行的数值（从负到0到正）都按正弦周期变化

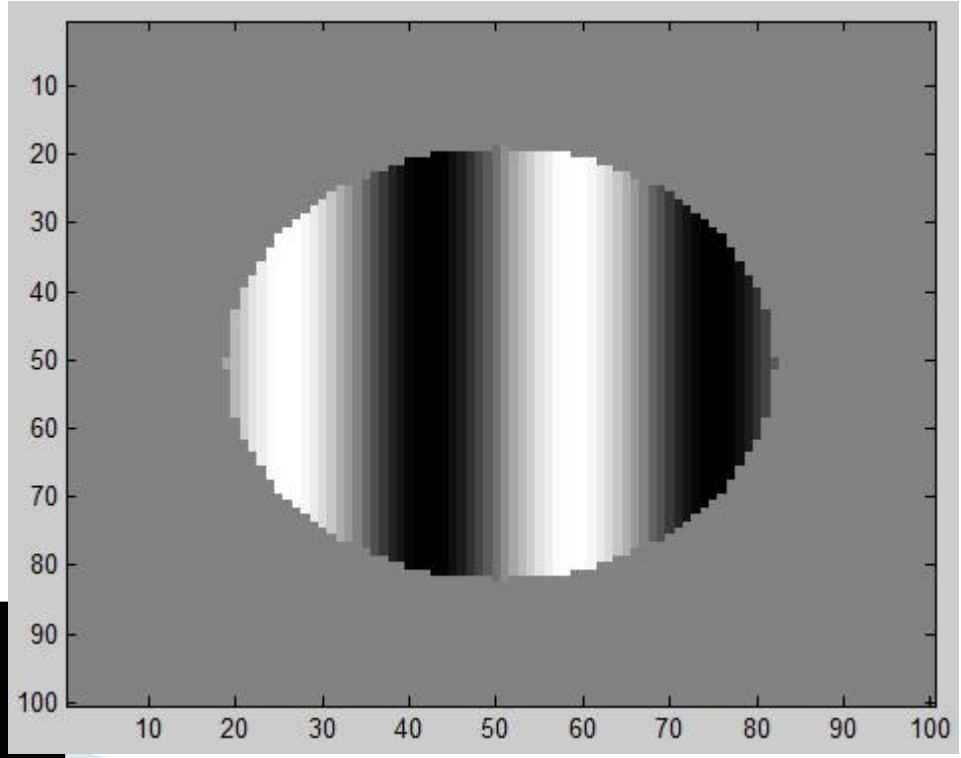
```
  Columns 1 through 22

   -0.0000   -0.1893   -0.3717   -0.5406   -0.6901   -0.8146   -0.9096
   -0.0000   -0.1893   -0.3717   -0.5406   -0.6901   -0.8146   -0.9096
```

```
>> imagesc(sinewave2D)
>> colormap(gray)
```



```
>> for r=1:length(xval)
    for c=1:length(xval)
        if xval(r).^2+xval(c).^2>radius^2
            sinewave2D(r,c)=0;
        end
    end
end
imagesc(sinewave2D);
colormap(gray);
```

在以矩阵中心点为中心，radius为半径的圆形区域之外的地方都为0，在圆形区域以内则保留原有的数值

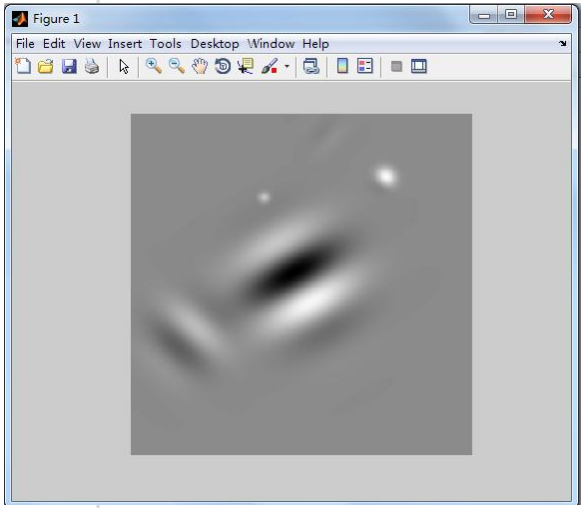


- 在脚本m文件中调用函数。
  - 例如，UseinsertGabor脚本m文件中，调用了insertGabor函数

```
params.n=400; % size of image (pixels)
img=zeros (params.n); % start with a blank image

% pick random parameters for each Gabor
for i=1:5
    params.center = 2*pi*(rand(1,2)-0.5);
    params.orientation = pi*rand(1); % radians (pi/4 = 45 degrees)
    params.width =rand (1); % 1/e half params.width of Gaussian
    params.sf =12*rand(1) ; % sf of sinewave carrier cycles/image
    params.phase = 2*pi*rand(1); % spatial params.phase of sinewave carrier (radians)
    params.contrast =rand(1); % contrast from 0 to 1
    img = insertGabor(params, img);
end

% show the image using the usual commmands
figure(1)
clf

% scale Gabor between 0 and 255
img = scaleMat(img, [0, 255]);

image (img);
axis equal
axis off
colormap(gray(256));
```

利用结构变量params把函数的多个输入（n，center，orientation，width，phase，contrast）等汇总在1个变量中

```m
function outImg = insertGabor (params, inImg)
% an example to use structure variable to organize input arguments
% written by YW, 2/2014

% Use meshgrid to define matrices x and y that range from -pi to pi;
[x,y] = meshgrid (linspace (-pi, pi, params.n));

if ~exist('inImg', 'var')
    inImg = zeros (params.n);
end

% Creat an oriented 'ramp' matrix as a linear combination of x and y. For
% example, when parms.orientation =0, cos=1 and sin=0 so ramp=x. When
% params. orientation is pi/2, then cos =0 and sin=1 so ramp=y.
ramp = cos (params.orientation)* (x-params.center(1))+sin(params.orientation)*(y-params.center(2));

% Sinusoidal carrier is a sinusoid on the matrix 'ramp'
sinusoid = params.contrast*sin(params.sf*ramp-params.phase);

% Gaussian envelope
Gaussian = exp(-((x-params.center(1)).^2+(y-params.center(2)).^2)/params.width^2);

% A gabor is the product of the sinusoid and the Gaussian
Gabor = sinusoid.*Gaussian;
outImg = inImg+Gabor;
```



- 其他设置输入参数的方法：采用input从命令窗口输入

Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\SineInAperture3Input.m

File Edit Text Go Cell Tools Debug Desktop Window Help

```
1   % SineInAperture3Input.m
2   % creates an image of apertured vertical sinusoids in a tilted array
3   % written by YW, 2/2014
4
5   % information about the sinusoids
6   clear all
7   close all
8   x=linspace(-pi, pi,100);
9   sf=[3 6 9 12]; % spatial freq in cycles per image
10  % rad=2;
11
12  % creats a 100*100*length(sf) matrix containing two 2D apertured sinusoids
13  % of different spatial frequncies.
14  for s=1:length(sf)
15      sinewave2D(:,:,s)= MakeSineApertureInput(x,sf(s));
16  end
```

注释radius

函数的输入参数删掉radius

Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\Mak

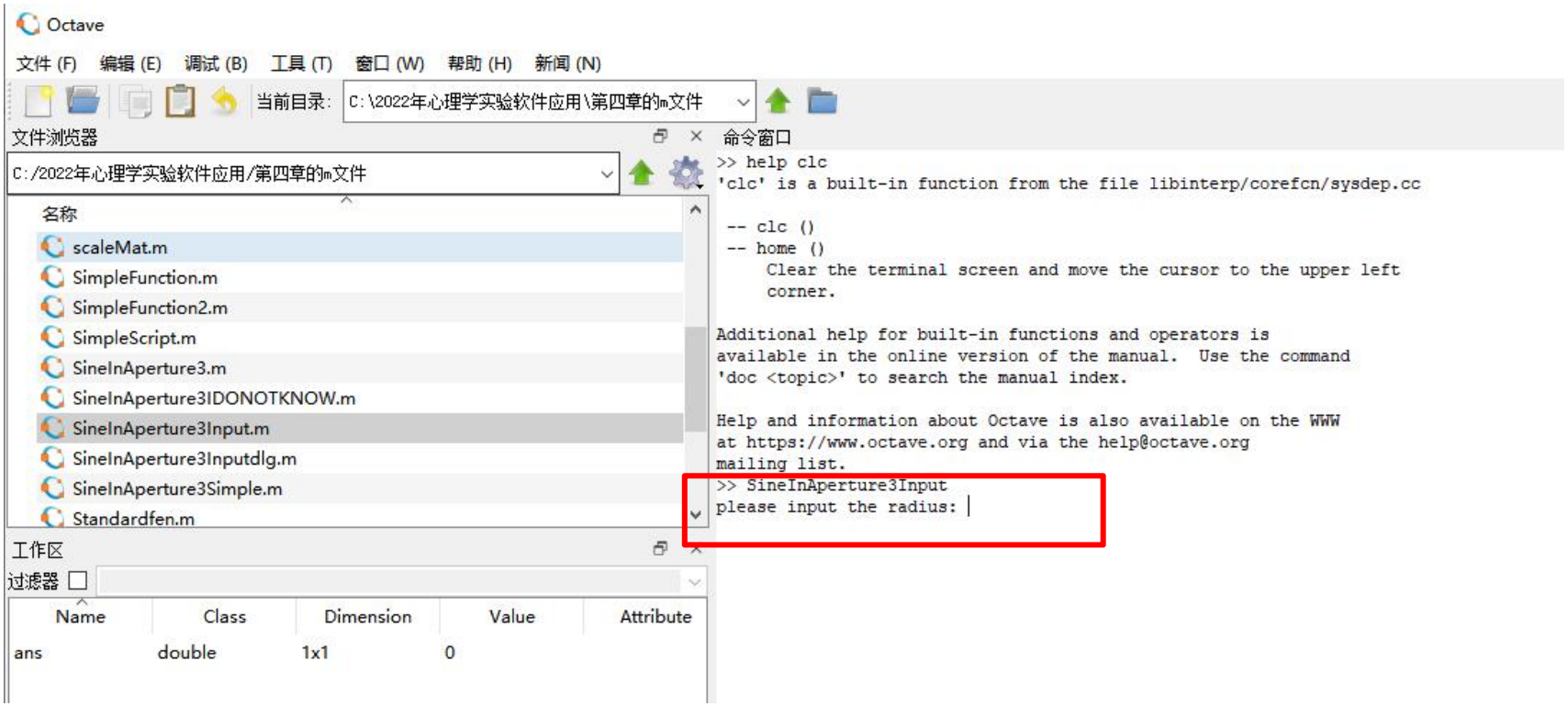
File Edit Text Go Cell Tools Debug Desktop Window Help

```
1   function sinewave2D = MakeSineApertureInput (xval, sf)
2
3   % function that creates a vertical sinewave windowed by
4   %
5   % Inputs:
6   %   xval: for which the sinusoids is computed
7   %   sf:   the saptial frequecy of the sinusoid
8   %   radius: the radiusl of the aperture in pixels
9   % Outputs: a 2D sinusoidal grating image
10
11  radius=input('please input the radius: ');
12  if isempty(radius)
13      radius=3;
14  end
15
16  onematrix=ones(size(xval));
17  sinewave=sin(xval*sf);
18  sinewave2D=(onematrix'*sinewave);
19  for r=1:length(xval)
20      for c=1:length(xval)
21          if xval(r).^2+xval(c).^2>radius^2
22              sinewave2D(r,c)=0;
23          end
24      end
25  end
```

函数的输入参数删掉radius

采用input输入

- 其他设置输入参数的方法：采用input从命令窗口输入
  - 在’please input the radius’后面输入数字，代表radius



- 其他设置输入参数的方法：采用inputdlg从对话框输入

Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\SineInAperture3Input.m

File Edit Text Go Cell Tools Debug Desktop Window Help

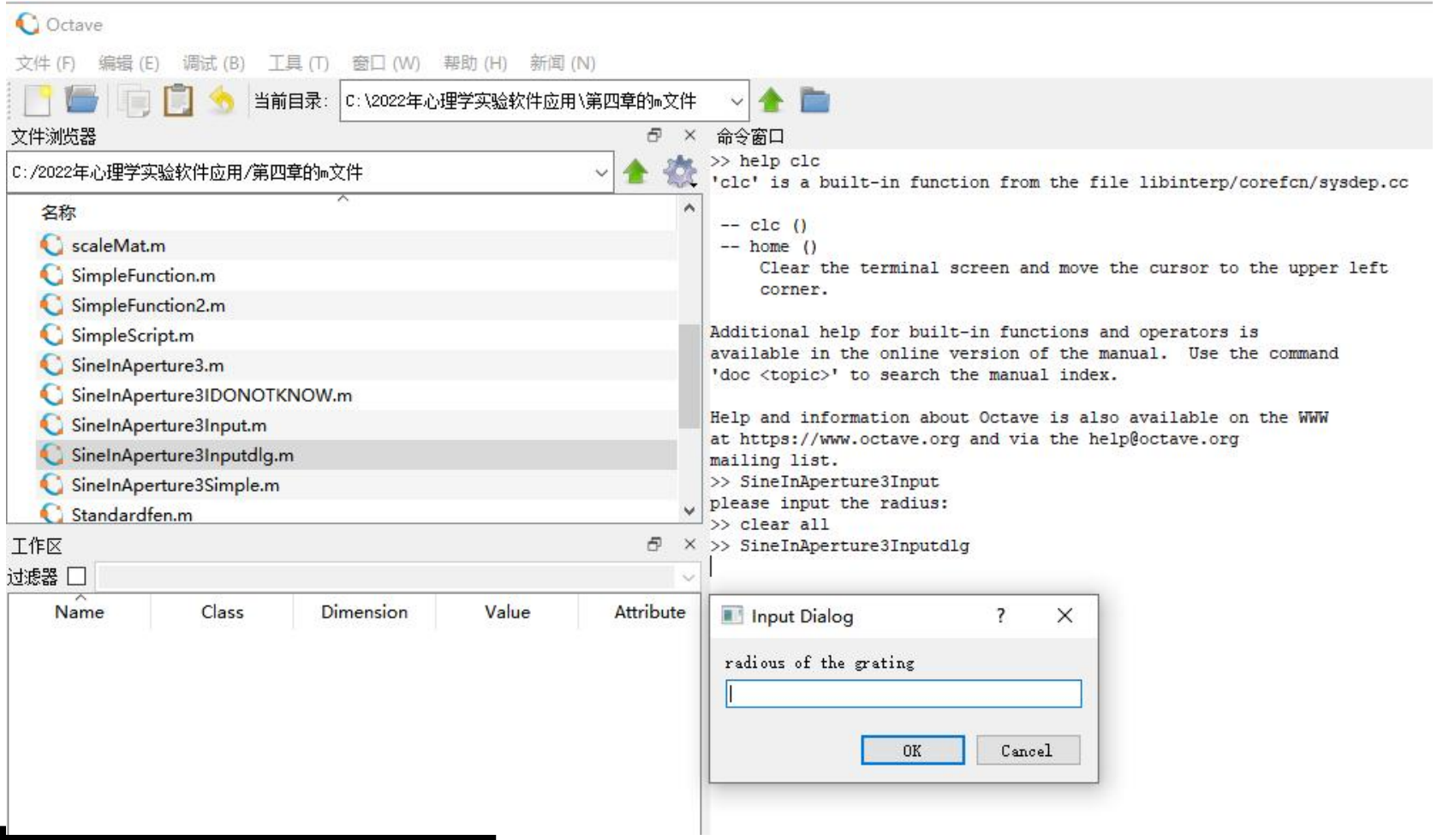
```
1   % SineInAperture3Input.m
2   % creates an image of apertured vertical sinusoids in a tilted array
3   % written by YW, 2/2014
4
5   % information about the sinusoids
6   clear all
7   close all
8   x=linspace(-pi, pi,100);
9   sf=[3 6 9 12]; % spatial freq in cycles per image
10  % rad=2;
11
12  % creats a 100*100*length(sf) matrix containing two 2D apertured sinusoids
13  % of different spatial frequncies.
14  for s=1:length(sf)
15      sinewave2D(:,:,s)= MakeSineApertureInput(x,sf(s));
16  end
```

Editor - E:\WangYou\Technique\2016年MATLAB入门\LearningMatlab\Ma

File Edit Text Go Cell Tools Debug Desktop Window He

```
1   function sinewave2D = MakeSineApertureInputdlg(xval, s
2
3   % function that creates a vertical sinewave windowed b
4   %
5   % Inputs:
6   %   xval: for which the sinusoids is computed
7   %   sf:   the saptial frequecy of the sinusoid
8   %   radius: the radiusl of the aperture in pixels
9   % Outputs: a 2D sinusoidal grating image
10
11  radius=inputdlg({'radious of the grating'});
12  radius=str2double(radius);
13
14  onematrix=ones(size(xval));
15  sinewave=sin(xval*sf);
16  sinewave2D=(onematrix'*sinewave);
17  for r=1:length(xval)
18      for c=1:length(xval)
19          if xval(r).^2+xval(c).^2>radius^2
20              sinewave2D(r,c)=0;
21          end
22      end
23  end
24
```

- 其他设置输入参数的方法：采用inputdlg从对话框输入



开始PRACTICE啦！