



大数据挖掘与统计学习

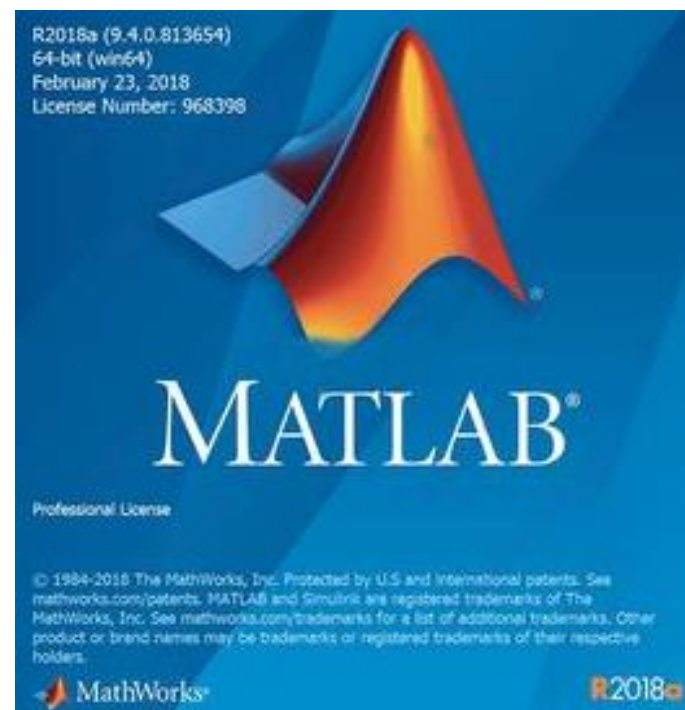
软件工程系
文化遗产数字化国家地方工程联合中心
可视化技术研究所
张海波
讲师/博士(后)

一、Matlab概述

1、MATLAB语言及其特点

MATLAB是“**MATrix LABoratory**”的缩写，它是由美国Mathworks公司于1984年推出的一种科学计算软件。与其它计算机语言相比，**MATLAB**有以下显著特点：

- ◆ 人机界面友好
- ◆ 强大而简易的作图功能
- ◆ 功能丰富，可扩展性强
- ◆ 超强的数值运算功能
- ◆ 实用的程序接口





2、MATLAB语言的工作环境

- ◆ MATLAB启动
- ◆ MATLAB命令窗口
- ◆ MATLAB工作空间
- ◆ 命令历史窗口
- ◆ 当前工作目录窗口
- ◆ MATLAB搜索路径
- ◆ MATLAB帮助系统

打开MATLAB

◆ 桌面快捷按钮

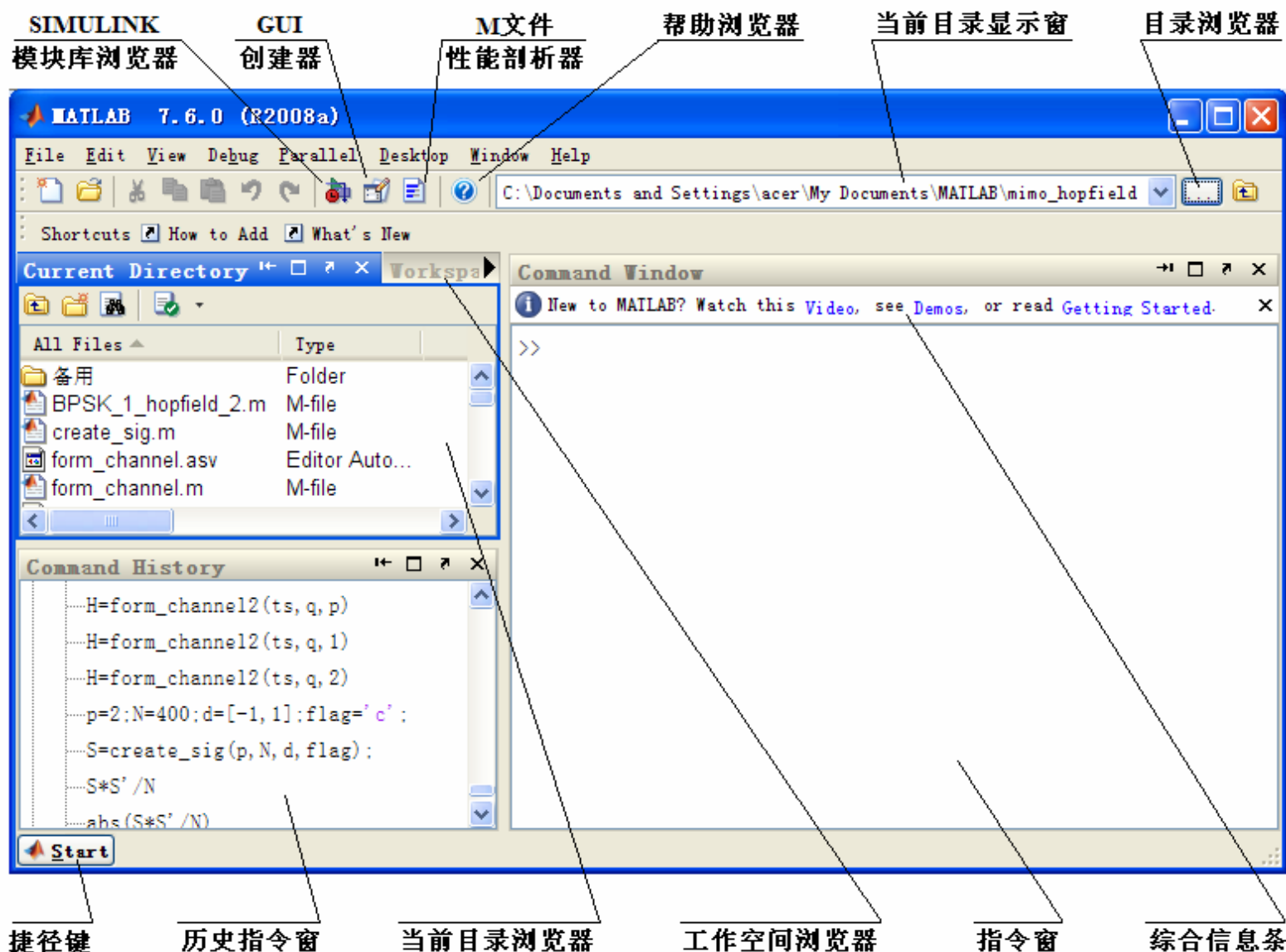


◆ 安装目录



matlab.exe

MATLAB界面



MATLAB 7.11.0 (R2010b)

File Edit View Debug Parallel Desktop Window Help

Current Folder: F:\Program Files\MATLAB\R2010b\bin

Shortcuts How to Add What's New

Current Folder

- bin
- worker.bat
- Untitled.m
- ProductRoots
- myvar.mat
- mw_mpiexec.bat
- msvc_modules_installer.pm
- mexutils.pm
- mexsetup.pm
- mexext.bat
- mex.pl
- mex.bat
- mcc.bat
- mbuild.bat
- matlab.exe
- matlab.bat
- license.txt
- lcdata.xsd
- lcdata.xml
- insttype.ini
- deploytool.bat
- win32
- util
- registry
- m3iregistry

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

restore previous default settings by following the steps outlined in

[Click here](#) if you do not want to see this message again.

```
>> a=pascal(3)
```

```
a =
```

```
1 1 1
```

```
1 2 3
```

```
1 3 6
```

```
>> b=[1 2 3;4 5 6;7 8 9]
```

```
b =
```

```
1 2 3
```

```
4 5 6
```

```
7 8 9
```

Workspace

Name	Value
a	[1,1,1;1,2,3;1,3,6]
b	[1,2,3;4,5,6;7,8,9]

Command History

- %-- 2013-05-28 20:17 --> clear
- %-- 2013-05-28 21:54 --> clc
- %-- 2013-06-30 18:40 --> a=pascal(3)
- %-- 2013-06-30 18:40 --> b=[1 2 3;4 5 6;7 8 9]

Start

MATLAB R2013a

HOME | PLOTS | APPS

New Script | New | Open | Find Files | Compare | Import Data | Save Workspace | New Variable | Open Variable | Clear Workspace | Analyze Code | Run and Time | Clear Commands | Simulink Library | Layout | Preferences | Set Path | Parallel | Help | Community | Request Support | Add-Ons

FILE | VARIABLE | CODE | SIMULINK | ENVIRONMENT | RESOURCES

Current Folder: F:\Program Files\MATLAB\R2013a\bin

Name
 m3registry
 registry
 util
 win32
 Balloon.wav
 deploytool.bat
 insttype.ini
 lcdata.xml
 lcdata.xsd
 lcdata_utf8.xml
 matlab.bat
 matlab.exe
 mbuild.bat
 mcc.bat
 MemShieldStarter.bat
 mex.bat
 mex.pl
 mexext.bat
 mexsetup.pm
 mexutils.pm
 msvc_modules_installer.pm
 mw_mpiexec.bat
 time.mp3
 worker.bat

Details

Ready

Command Window

New to MATLAB? Watch this [Video](#), see [Examples](#), or read [Getting Started](#).

Toolbox Path Cache read in 0.67 seconds.
 MATLAB Path initialized in 3.11 seconds.
 fx >>

Workspace

Name	Value	M
------	-------	---

Command History

```

-- 2014-05-17 18:36 --%
fn=input(' Enter WAV filename:',
Balloon
[x,fs,nb]=wavread(fn);
clear
clc
plot(data,'DisplayName','data')
HELP MEMORY
help MEMORY
-- 2014-06-15 10:24 --%
  
```

显示桌面

MATLAB R2013a

HOME PLOTS APPS

Search Documentation

No Variable Selected

Selection

Current Folder

- Name
- m3iregistry
- registry
- util
- win32
- Balloon.wav
- deploytool.bat
- insttype.ini
- lcdata.xml
- lcdata.xsd
- lcdata_utf8.xml
- matlab.bat
- matlab.exe
- mbuild.bat
- mcc.bat
- MemShieldStarter.bat
- mex.bat
- mex.pl
- mexext.bat
- mexsetup.pm
- mexutils.pm
- msvc_modules_installer
- mw_mpiexec.bat
- time.mp3
- worker.bat

Search

★ FAVORITES

- plot
- Plot as mult...
- Plot as mult...
- bar
- area
- pie
- hist
- contour
- surf
- mesh

MATLAB LINE PLOTS

- plot
- Plot as mult...
- Plot as mult...
- plotyy
- semilogx
- semilogy
- loglog
- area
- errorbar
- plot3
- comet

MATLAB STEM AND STAIR PLOTS

- stem
- stairs
- stem3

MATLAB BAR PLOTS

- bar
- barh
- Stacked ba...
- Stacked ba...
- hist
- pareto
- plotmatrix

MATLAB SCATTER PLOTS

Plots for Selection All plots

Catalog

Reuse Figure
New Figure

OPTIONS

Workspace

Name	Value	M

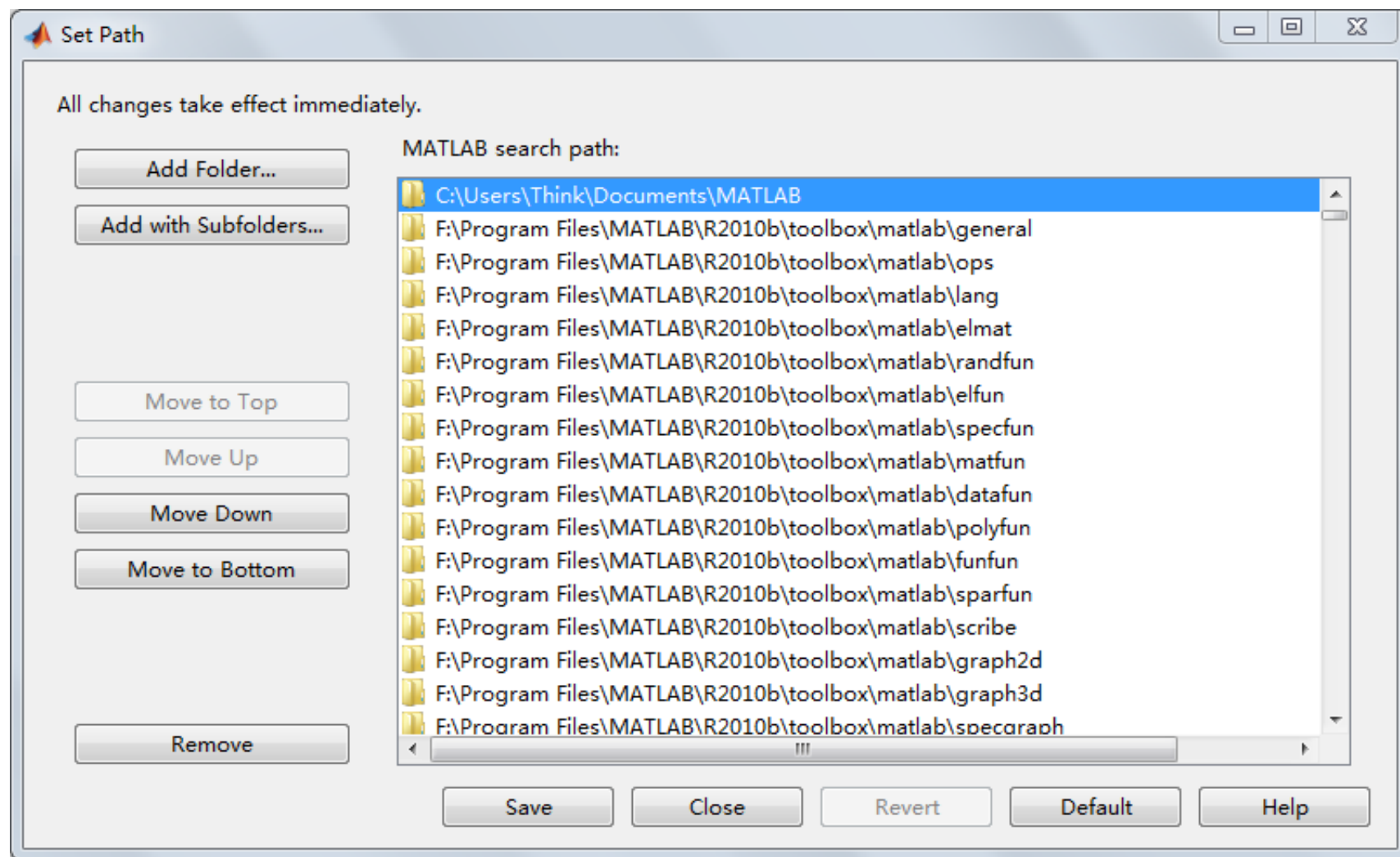
Command History

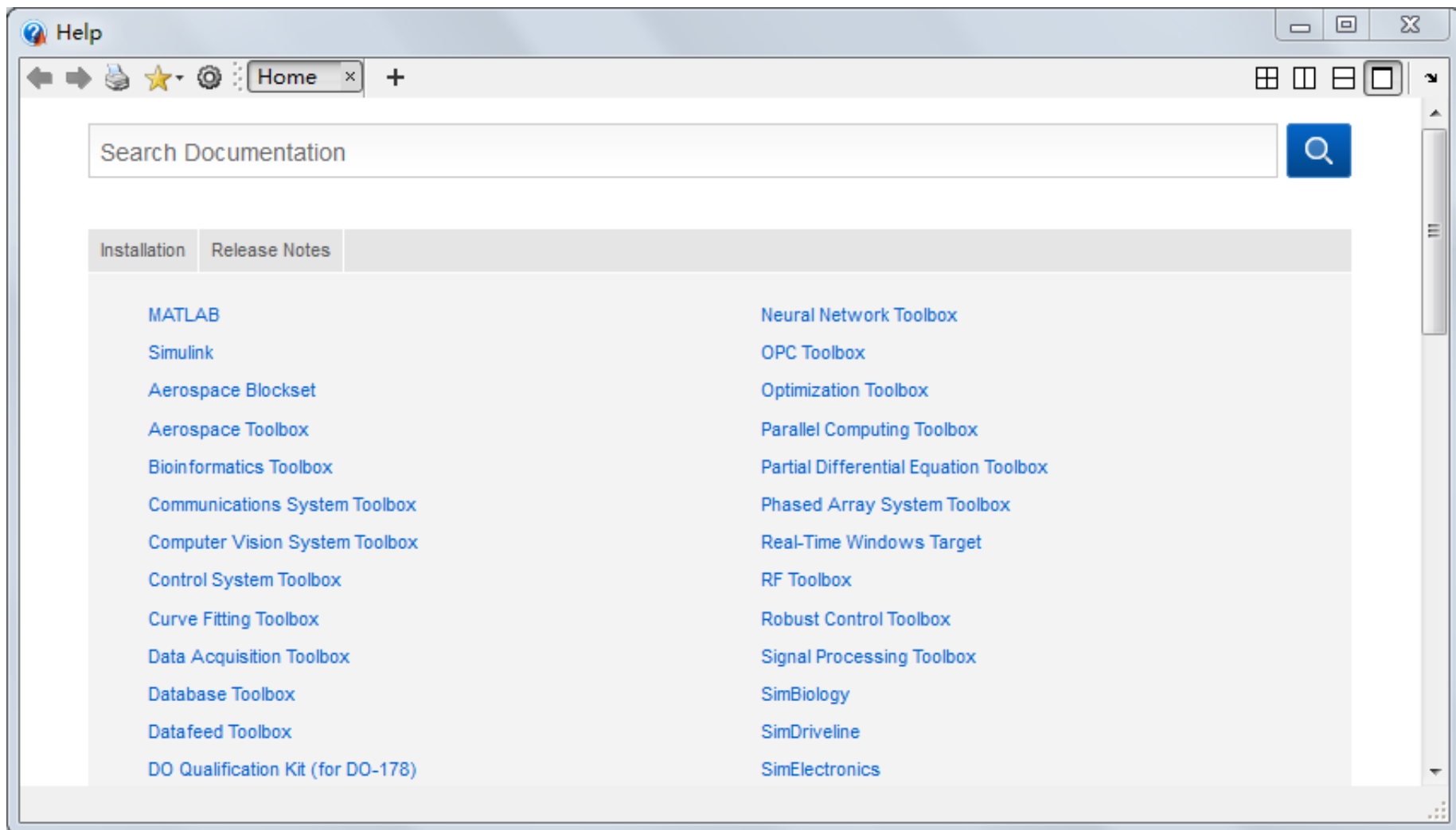
```

-- 2014-05-17 18:36 --%
fn=input(' Enter WAV filename:');
Balloon
[x,fs,nb]=wavread(fn);
clear
clc
plot(data,'DisplayName','data')
HELP MEMORY
help MEMORY
-- 2014-06-15 10:24 --%
  
```

Ready

选择菜单“file”→“set path”进入搜索路径管理窗口







◆矩阵运算

◆符号运算

◆关系运算和逻辑运算

输入矩阵A、B的值

$A = [1 \ 2 \ 3 \ 4; \ 5 \ 6 \ 7 \ 8; \ 9 \ 10 \ 11 \ 12; \ 13 \ 14 \ 15 \ 16]$

$B = [1, \text{sqrt}(25), 9, 13; \ 2, 6, 10, 7*2;$
 $3+\sin(\pi), 7, 11, 15; \ 4, \text{abs}(-8), 12, 16]$

A =				B =			
1	2	3	4	1	5	9	13
5	6	7	8	2	6	10	14
9	10	11	12	3	7	11	15
13	14	15	16	4	8	12	16

矩阵下标与子矩阵提取

$X=[1\ 2\ 3\ 0;5\ 6\ 0\ 8;9\ 0\ 11\ 12;0\ 14\ 15\ 16]$

X =

1	2	3	0
5	6	0	8
9	0	11	12
0	14	15	16

X(2, 3)

ans =

0

X(2, :)

ans =

5

6

0

8

X(2:3, 1:3)

ans =

5

6

0

9

0

11

X(2:end, 1)

ans =

5	6	0
9	0	11

X(:)

ans =

1	5	9	0	2	6	0	14	3	0	11	15	0	8	12
---	---	---	---	---	---	---	----	---	---	----	----	---	---	----

16

```
>>A=[1 2 3 4; 5 6 7 8; 9 10 11 12; 13 14 15 16]
```

```
A =
```

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

```
>>A(1,1)=0
```

```
A =
```

0	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



```
>>A(2,2)=A(1,2)+A(2,1)
```

```
A =
```

0	2	3	4
5	7	7	8
9	10	11	12
13	14	15	16

```
>>A(4,4)=cos(0)
```

```
A =
```

0	2	3	4
5	7	7	8
9	10	11	12
13	14	15	1



特殊矩阵生成

```
>> a=zeros(3)
```

```
a =
```

0	0	0
0	0	0
0	0	0

```
>> b=ones(3)
```

```
b =
```

1	1	1
1	1	1
1	1	1

```
>> c=rand(3)

c =

    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575

>> d=randn(3)

d =

    2.7694    0.7254   -0.2050
   -1.3499   -0.0631   -0.1241
    3.0349    0.7147    1.4897
```

数学函数

matlab中函数

功能： matlab中通过rand函数产生的是介于0到1(不包括1)之间的伪随机数。更多信息请见本词条[参考资料](#)或者在matlab命令窗口输入help rand获得帮助信息。

用法：

- 1.rand(n)表示产生一个n×n的随机矩阵，n必须是整数.否则会报错。
- 2.rand(m,n)或rand([m n])产生m×n均匀分布的随机矩阵，元素取值在0.0~1.0。
- 3.X=rand(1,10);产生10个0~1的随机数。
- 4.Y = rand(size(A));产生一个与A同样大小的数组。
- 5.产生介于a到b之间的伪随机数。
- 6.rand('state',0)表示恢复到最初产生随机数的状态。
- 7.rand('state',sum(100'clock))定义随时间变化的初值。

示例：

randn

[编辑](#)[讨论](#)

randn (random normal distribution) 是一种产生标准正态分布的随机数或矩阵的函数，属于MATLAB函数。返回一个n*n的随机项的矩阵。如果n不是个数量，将返回错误信息。

外文名	randn	类别	编程
性质	计算机	属于	MATLAB函数
		英文含义	random normal distribution

目录

- 1 MATLAB函数randn简介
- 2 应用举例

MATLAB函数randn简介

[编辑](#)

用法：

Y = randn (n)

返回一个n*n的随机项的矩阵。如果n不是个数量，将返回错误信息。

Y = randn(m,n) 或 Y = randn([m n])

返回一个m*n的随机项矩阵。

Y = randn(m,n,p,...) 或 Y = randn([m n p...])

产生随机数组。



创建符号变量和符号表达式

```
syms x y real      %创建实数符号变量
z=x+i*y;           %创建z为复数符号变量
real(z)             %复数z的实部是实数x
ans =
      x
syms a b c x        %创建多个符号变量
f2=a*x^2+b*x+c      %创建符号表达式
f2 =
      a*x^2 + b*x + c
```



符号运算

```
>> A=sym(' [a, b; c, d]' );
```

```
>> B=sym(' [1 2; 3 4]' );
```

```
>> C=A+B
```

C =

```
[ a+1,  b+2]
```

```
[ c+3,  d+4]
```



$A = [-2, -1, 0, 0, 1, 2, 3]$

$L1 = \sim(A > 1)$ %判断A中，哪些元素不大于1

$L2 = (A > 0) \& (A < 2)$ %判断A中，哪些元素大于0且小于3

A =

-2	-1	0	0	1	2	3
----	----	---	---	---	---	---

L1 =

1	1	1	1	1	0	0
---	---	---	---	---	---	---

L2 =

0	0	0	0	1	0	0
---	---	---	---	---	---	---

3、MATLAB程序设计

Matlab通常使用命令驱动方式，当单行命令输入时，Matlab立即处理并显示结果，同时将运行说明和命令存入历史命令窗口。Matlab语句的磁盘文件称作M文件，因为这些文件名的末尾是.M形式。

M文件有两种类型：命令（Script）文件

函数（function）文件



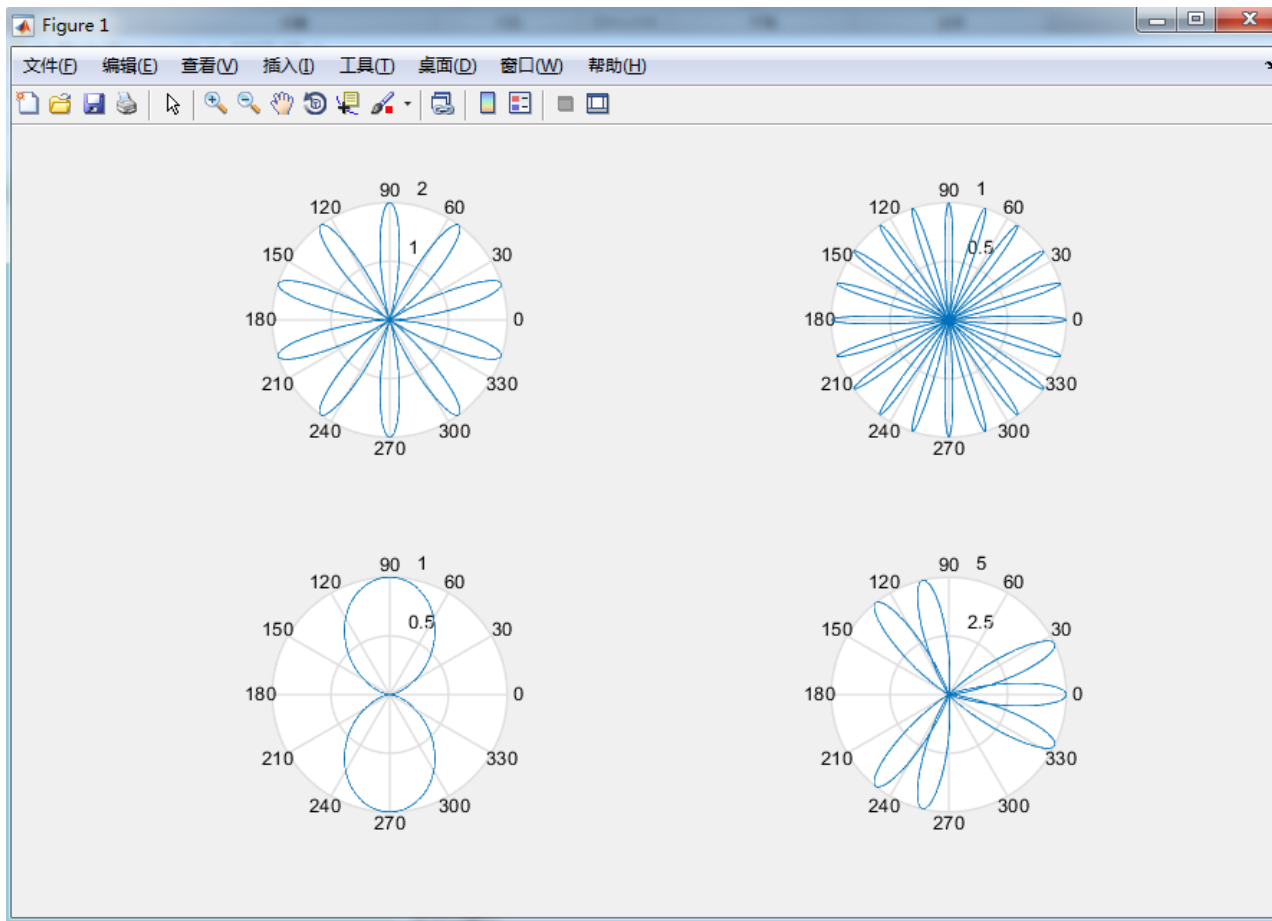
（Script）文件

第一类型M文件称为命令（Script）文件，特点如下：

- （1）最简单的M文件，它是一系列命令、语句的简单组合；
- （2）没有输入和输出参数；
- （3）顺序执行文件后变量是全局变量，保存在工作空间中；
- （4）可以直接运行。

例：%绘制花瓣

```
theta=-pi:0.01:pi;  
rho(1,:)=2*sin(5*theta).^2; rho(2,:)=cos(10*theta).^3;  
rho(3,:)=sin(theta).^2; rho(4,:)=5*cos(3.5*theta).^3;  
for k=1:4 subplot(2,2,k),polar(theta,rho(k,:))  
end
```



运行方式

- (1) 将所有命令复制粘贴到命令行窗口，按回车执行。
- (2) 在M文件编辑器中选择“Debug”，再选“Run”运行，或直接按“F5”运行程序。
- (3) 在命令行中键入文件名，再回车，注意不要加扩展名“.m”。



M文件命名时不要用纯数字，这样会导致错误的结果。若有一个名为“1.m”的M文件，运行后的结果只能是1。

function文件

函数文件的特点如下：

- (1) 以**function**为引导；
- (2) 可以接受输入、输出参数；
- (3) 内部变量为局部变量，运行完被释放。
- (4) 不能直接运行，必须调用。

函数定义行
function 函数名, 输入变量, 输出变量

```
function X=total (n)
%total 计算从1到n的n个数之和
%如果n比1小, 则提示错误。
if n<1
    error ('Input must be larger than 1');
end
k=1:n;
X=sum (k);
```

注释行: 在命令窗口键入
total后显示出来. 显示
续的若干个%右边的文

函数体: 包括函数的全部程序代码

函数文件编写完之后, 保存的文件名必须与函数名同名!



例 函数文件示例—average.m。

```
function y = average(x)
```

```
% AVERAGE 求向量元素的均值
```

```
% 语法:
```

```
% Y = average(X)
```

```
% 其中, X 是向量, Y为计算得到向量元素的均值
```

```
% 若输入参数为非向量则出错
```

```
% 代码行
```

```
[m, n] = size(x);
```

```
% 判断输入参数是否为向量
```

```
if (~((m == 1) | (n == 1)) | (m == 1 & n == 1))  
% 若输入参数不是向量，则出错  
    error('Input must be a vector')  
end  
% 计算向量元素的均值  
y = sum(x)/length(x);
```

在命令行中键入下面的指令运行例14的代码：

```
>> z = 1:99;  
>> y = average(z)  
y =  
    50
```



Matlab控制流

1 选择结构

当人们判断某一条件是否满足，根据判断的结果来选择不同的解决问题的方法时，就需要使用选择结构。和C语言类似，**MATLAB**的条件判断可使用if语句或者switch 语句。



if语句

if语句的基本语法结构有三种，分别如下：

(1) if 关系运算表达式

MATLAB语句

end

选择结构表示，当关系表达式结果为逻辑真时，执行MATLAB语句，可以是一个表达式，也可以是多个表达式。语句结尾处必须有关键字end。

(2) if 关系运算表达式

MATLAB语句A

else

MATLAB语句B

end

选择结构表示，当关系表达式结果为逻辑真时，执行语句A，否则执行语句B，语句B结尾必须具有关键字end。

(3) if 关系运算表达式a

MATLAB语句A

elseif 关系运算表达式b

MATLAB语句B

else 关系运算表达式c

end

这种选择结构可判断多条关系表达式结果，按照执行逻辑关系执行相应语句。

例 if语句的使用——if_exam.m。

```
clear all
    I=1;
    J=2;
if I == J
    A(I, J) = 2;
elseif abs(I-J) == 1
    A(I, J) = -1;
else
    A(I, J) = 0;
end
```



switch语句

另外一种构成选择结构的关键字就是switch。在处理实际问题的时候，往往要处理多个分支，这时如果使用if-else语句处理多分支结构往往使程序变得冗长，降低了程序可读性。switch语句就可以用于处理多分支选择，语法结构如下：



switch语句

另外一种构成选择结构的关键字就是switch。在处理实际问题的时候，往往要处理多个分支，这时如果使用if-else语句处理多分支结构往往使程序变得冗长，降低了程序可读性。switch语句就可以用于处理多分支选择，语法结构如下：

switch后的表达式可以是一个数值类型表达式或是一个数值类型的变量，当这个表达式的值同case后面的某一个常量表达式相等时，则执行case后面常量表达式后面的语句。

注意：MATLAB的switch和C语言不同。C语言case后面的语句必须包含类似break语句的流程控制语句，否则程序会依次执行符合条件的case语句后面的每一个case分支。但是在MATLAB中就不必如此，**程序仅仅执行符合条件的case分支。**

例 switch结构使用示例——switch_examp.m。

```
clear all
algorithm = input('Enter an algorithm in quotes
(ode23, ode15s, etc:)', 's');
switch algorithm
case 'ode23'
    str = '2nd/3rd order';
case {'ode15s', 'ode23s'}
    str = 'stiff system';
otherwise
    str = 'other algorithm';
end
disp(str);
```




2. 循环结构

MATLAB中包含两种循环结构，一种是循环次数不确定的while循环，而另一种是循环次数确定的for循环。

while循环结构

while语句可用来实现“当”型的循环结构，形式如下：

```
while(表达式)
    MATLAB语句
end
```

当表达式为真时，循环执行由语句构成的循环体，特点是先判断循环条件，循环条件成立，即表达式运算结果为“真”，再执行循环体。循环体执行的语句可以是一句也可以是多句，在语句后必须使用关键字end作为循环结构的结束。

for循环结构

使用for语句构成循环是最灵活、简便的方法，使用for语句循环需要预先知道循环体执行的次数，这种循环一般叫作确定循环。for循环基本结构如下：

```
for index = start:increment:end
```

MATLAB语句

```
end
```

其中index的取值取决于start和end的值，通常使用等差的数列向量。

例 使用while语句求解

```
i = 1;  
sum = 0;  
while ( i <= 1000 )  
    sum = sum+i;  
    i = i+1;  
end  
str = ['计算结果为: ',num2str(sum)];  
disp(str)
```

例 使用for语句求解

```
sum = 0;  
for i = 1:1000  
    sum = sum+i;  
end  
str = ['计算结果为: ',num2str(sum)];  
disp(str)
```

3 break语句和continue语句

在循环结构中还有两条语句会影响程序的流程，就是break语句和continue语句，基本功能如下：

◆ break语句在执行循环体的时候强迫终止循环，即控制程序的流程使其提前退出循环，使用方法是

break;

◆ continue语句中断本次循环体运行，将程序流程跳转到判断循环条件的语句处，继续下一次循环，使用方法是

continue;

例8 break语句示例——break_example.m。

```
i = 0;j = 0;k = 0;
for i = 1:2
    for j = 1:2
        for k = 1:2
            if(k == 2)
                disp('退出循环');
                break;
            end
            str = sprintf('I = %d , J = %d , K = %d',i,j,k);
            disp(str);
        end
    end
end
disp('程序运行结束');
```



运行结果如下：

```
>> break_example
```

```
I = 1 , J = 1 , K = 1
```

退出循环

```
I = 1 , J = 2 , K = 1
```

退出循环

```
I = 2 , J = 1 , K = 1
```

退出循环

```
I = 2 , J = 2 , K = 1
```

退出循环

程序运行结束



例9 continue语句示例。

```
i = 0;
for i = 1:6
    if(i>3)
        continue
    else
        str = sprintf('I = %d',i);
        disp(str);
    end
end
str = sprintf('循环结束 I = %d',i);
disp(str);
```

运行结果如下：

```
>> continue_example
```

```
I = 1
```

```
I = 2
```

```
I = 3
```

循环结束 I = 6

continue语句终止当前循环，继续下一次循环运算，直到所有循环运算结束。

Matlab程序设计原则

- (1) %后面内容是程序注释，使程序更具可读性。
- (2) 在主程序开头用clear指令清除变量，在子程序中不要用clear。
- (3) 参数值要集中放在程序的开始部分。在语句后输入分号使结果不在屏幕上显示，以提高执行速度。
- (4) input指令可以用来输入一些临时数据；对于大量参数，则通过建立一个存储参数的子程序，在主程序中通过子程序名称来调用。
- (5) 程序尽量模块化。
- (6) 利用Debug进行程序的调试。
- (7) 设置好MATLAB的工作路径，以便程序运行。



Matlab文件相关操作

- ◆ 数据存储
- ◆ 数据导入
- ◆ 数据打开
- ◆ 底层文件输入输出

数据存储

save命令

功能：用以将**工作空间**中的变量保存到文件上。

格式一： **save**

将所有变量保存在“matlab.mat”的文件中，通过**load**命令来重新装入工作空间。

格式二： **save** 文件名 变量名

将指定变量保存在“文件名.mat”的文件中。

格式三： **save** 文件名 选项

使用“选项”指定ASCII文件格式，将变量保存到文件中。



数据导入

MATLAB中导入数据通常由函数load实现，用法如下：

- load: 如果matlab.mat文件存在，导入matlab.mat中的所有变量，如果不存在，则返回error。
- load filename: 将filename中全部变量导入到工作空间中。
- load filename X Y Z ...: 将filename中的变量X、Y、Z等导入到工作空间中，如果是MAT文件，在指定变量时可以使用通配符“*”。
- load -mat filename: 无论输入文件名是否包含有扩展名，将其以mat格式导入；如果指定文件不是MAT文件，则返回error。

例 将文件matlab.mat中的变量导入到工作区中。

解：首先应用命令whos -file查看该文件中的内容

:





```
>>whos -file matlab.mat
```

Name	Size	Bytes	Class
Attributes			
N	1x1	8	double
S	1x1	8	double

将该文件中的变量导入到工作空间中：

```
>> load matlab.mat
```

该命令执行后，可以在工作空间浏览器中看见这些变量，如图所示。

Workspace						
 Stack: Base  Select data to plot						
Name ▲	Value	Size	Bytes	Min	Max	
 N	141	1x1	8	141	141	
 S	1.9859	1x1	8	1.9859	1.9859	

接下来用户可以访问这些变量。

```
>> N
```

```
N =
```

```
141
```


MATLAB中，另一个导入数据的常用函数为importdata，该函数的用法如下：

importdata('filename')，将filename中的数据导入到工作空间中；

A = importdata('filename')，将filename中的数据导入到工作空间中，并保存为变量A；

importdata('filename','delimiter')，将filename中的数据导入到工作空间中，以delimiter指定的符号作为分隔符。

例 从文件中导入数据。

```
>> imported_data = importdata('matlab.mat')
```

输出结果如下

```
imported_data =  
    S: 1.9859  
    N: 141
```

与load函数不同，importdata将文件中的数据以结构体的方式导入到工作空间中。

数据打开

MATLAB中可以使用open命令打开各种格式的文件，MATLAB自动根据文件的扩展名选择相应的编辑器。

注意：

`open('filename.mat')`将filename.mat以结构体的方式打开在工作空间中

`load('filename.mat')`将文件中的变量导入到工作空间中，如果需要访问其中的内容，需要以不同的格式进行。



例 open与load的比较。

输入命令

```
>> clear
```

```
>> A = magic(3)
```

```
>> B = rand(3);
```

```
>> save
```

数据保存到matlab.mat，使用load命令



```
>> clear
```

```
>> load('matlab.mat')
```

```
>> A
```

A =

8	1	6
3	5	7
4	9	2

```
>> B
```

B =

0.8147	0.9134	0.2785
0.9058	0.6324	0.5469
0.1270	0.0975	0.9575



然后使用open命令

```
>> clear
```

```
>> open(' matlab.mat' )
```

```
ans =
```

```
    A: [3x3 double]
```

```
    B: [3x3 double]
```

```
>> struc1=ans;
```

```
>> struc1.A
```

```
ans =
```

```
      8      1      6
```

```
      3      5      7
```

```
      4      9      2
```

```
>> struc1.B
```

```
ans =
```

```
    0.8147    0.9134    0.2785
```

```
    0.9058    0.6324    0.5469
```

```
    0.1270    0.0975    0.9575
```



底层文件输入与输出

1. fopen

使用fopen函数打开或创建文件，fopen函数的调用格式为：

`fid=fopen (filename, 'option')`

说明：fid用于存储文件句柄值，如果返回的句柄值大于0，则说明文件打开成功。文件名用字符串形式，表示待打开的数据文件。‘option’为打开方式，选项如下：

‘r’：打开文件进行读操作（缺省）；

‘w’：删除已存在内容或生成新文件，进行写操作；

‘a’：打开已存在文件或生成并打开新文件，进行写操作，在文件末尾添加数据；

另外，在字符串后添加“t”，如‘rt’，则将该文件以文本方式打开；如果添加的是“b”，则以二进制格式打开。

2. fclose

文件在进行完读、写等操作后，应及时关闭，以免数据丢失。关闭文件用fclose函数，调用格式为：

`sta=fclose(fid)`

说明：该函数关闭fid所表示的文件。sta表示关闭文件操作的返回代码，若关闭成功，返回0，否则返回-1。如果要关闭所有已打开的文件用fclose('all')。

1. fread

fread函数可以读取二进制文件的数据，并将数据存入矩阵。其调用格式为：

[A, COUNT]=fread(fid, size, precision)

说明：其中A是用于存放读取数据的矩阵、COUNT是返回所读取的数据元素个数、fid为文件句柄、size为可选项，若不选用则读取整个文件内容；若选用则它的值可以是下列值：N（读取N个元素到一个列向量）、inf（读取整个文件）、[M, N]（读数据到M×N的矩阵中，数据按列存放）。precision用于控制所写数据的精度。常用的数据精度有：char、uchar、int、long、float、double等。缺省数据精度为uchar，即无符号字符格式。



2. fwrite

功能：向文件中写入二进制数据。

COUNT=fwrite (fid, A, precision)

将A中元素写入指定文件，将其值转换为指定精度。

例 将一个二进制矩阵存入磁盘文件中。

```
>> a=[1 2 3 4 5 6 7 8 9];
```

```
>> fid=fopen('dtest.bin','wb')%以二进制写入方式打开文件
```

```
fid =
```

```
3 %其值大于0，表示打开成功
```

```
>> fwrite(fid,a,'double')
```

```
ans =
```

```
9 %表示写入了9个数据
```

```
>> fclose(fid)
```

```
ans =
```

```
0 %表示关闭成功
```

1. fscanf

fscanf函数可以读取文本文件的内容，并按指定格式存入矩阵。其调用格式为：

$$[A, COUNT]=fscanf(fid, format, size)$$

说明：从由fid所指定文件中读入所有数据，并根据format进行转换，并返回给矩阵A，'格式'字符串指定被读入数据的格式。size为可选项，决定矩阵A中数据的排列形式，。

2. fprintf

fprintf函数可以将数据按指定格式写入到文本文件中。其调用格式为：

$$fprintf(fid, format, A)$$

说明：fid为文件句柄，指定要写入数据的文件，format是用来控制所写数据格式的格式符，A用来存放数据。

例 创建一个字符矩阵并存入磁盘，再读出赋值给另一个矩阵。

```
a='string';  
fid=fopen('dchar1.txt','w');  
fprintf(fid,'%s',a);  
fclose(fid);  
fid1=fopen('dchar1.txt','rt');  
b=fscanf(fid1,'%s')
```

输出结果如下

b =

string

程序将矩阵a的值赋值给了矩阵b。