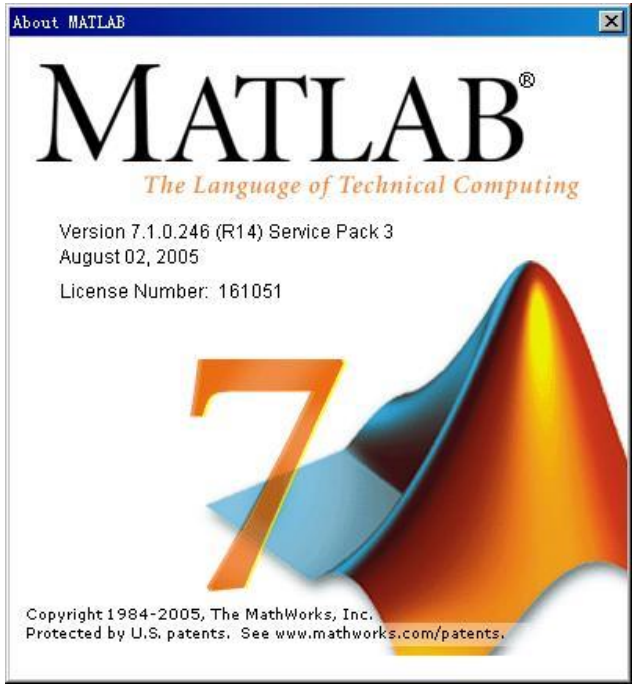
MATLAB软件初步

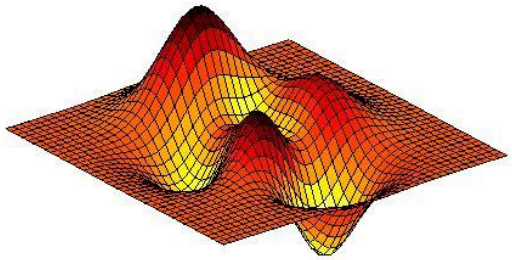


郭雨珍

2020.07.15

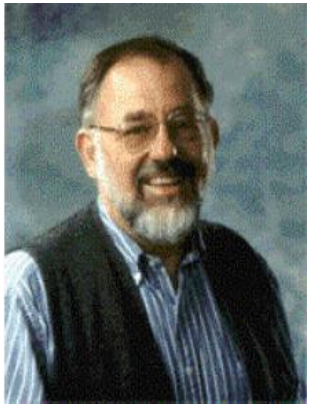
1 Matlab概述

- 前言
- Matlab软件概述
- Matlab的桌面环境及入门知识



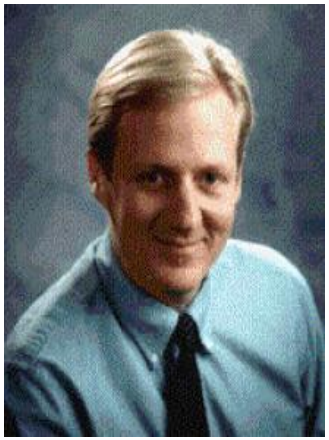
1.1 MATLAB的历史及影响

- 70年代中期，Cleve Moler博士及其同事在美国国家基金会的帮助下，开发了LINPACK和EISPACK的FORTRAN语言子程序库，这两个程序库代表了当时矩阵运算的最高水平。
- 到了70年代后期，身为美国新墨西哥州大学计算机系系主任的CIeve Moler，在给学生上线性代数课时，为了让学生能使用这两个子程序库，同时又不用在编程上花费过多的时间，开始着手用FORTRAN语言为学生编写使用LINPACK和EISPACK的接口程序，他将这个程序取名为MATLAB，其名称是由MATrix和 LABoratory（矩阵实验室）两个单词的前三个字母所合成。
- 在1978年，Malab就面世了。这个程序获得了很大的成功，受到了学生的广泛欢迎。在以后的几年里，Matlab在多所大学里作为教学辅助软件使用，并作为面向大众的免费软件广为流传。

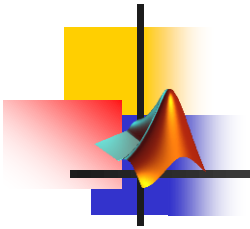


Cleve Moler

- 将MATLAB商品化的不是Cleve Moler,而是一个名叫Jack Little的人。当免费的MATLAB软件到Stanford大学，Jack Little正在该校主修控制，便接触到了当时MATLAB，直觉告诉他，这是一个具有巨大发展潜力的软件。因此他在毕业没多久，就开始用C语言重新编写了MATLAB的核心。在Moler的协助下，于1984年成立MathWorks公司，首次推出MATLAB商用版。在其商用版推出的初期，MATLAB就以其优秀的品质（高效的数据计算能力和开放的体系结构）占据了大部分数学计算软件的市场，原来应用于控制领域里的一些封闭式数学计算软件包（如英国的UMIST、瑞典的LUND和SIMNON、德国的KEDDC）就纷纷被淘汰或在MATLAB上重建。



Jack Little

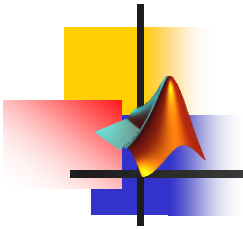


MATLAB就是这样经过了近30年的专门打造、20多年的千锤百炼，它以高性能的数组运算（包括矩阵运算）为基础，不仅实现了大多数数学算法的高效运行函数和数据可视化，而且提供了非常高效的计算机高级编程语言，在用户可参与的情况下，各种专业领域的工具箱不断开发和完善，MATLAB取得了巨大的成功，已广泛应用于科学研究、工程应用，用于数值计算分析、系统建模与仿真。

早在20世纪90年代初，欧美等发达国家的大学就将MATLAB列为一种必须掌握的编程语言。近几年来，国内的很多大学也将MATLAB列为了本科生必修课程。

与Maple、Mathematica数学计算软件相比，MATLAB以数值计算见长，而Maple等以符号运算见长，能给出解析解和任意精度解，而处理大量数据的能力远不如MATLAB。

MATLAB软件功能之强大、应用之广泛，已成为为21世纪最为重要的科学计算语言。可见学习掌握这一工具的重要性。



1.2 MATLAB产品的体系结构

实际上MATLAB
习使用MATLAB呢?

MATLAB产

- MATLA
- MAT
- MAT
- Simulink
- Simulink E
- Real-Time V
- Stateflow
- Stateflow Co

围绕着MATLAB这个计算核心，形成了诸多针对不同
被称为专用工具箱
的列表以及每个工具箱的使
档。MATLAB本身所提供
其他公司或研究单
的总数已有100多个，
果你有特别的应用
有相关的工具箱，
具箱了。

MATLAB是MATLAB产品家族的计算核心与基础，是集高性能数值计算与数据可视化于一体的高效编程语言。

包，
供许多
专用模块集，如Communication Blockset、DSP Blockset、SimPowerSystem Blockset、Signal Processing Blockset等，详见MATLAB在线帮助文档。

Real-T
Simulin
成实时

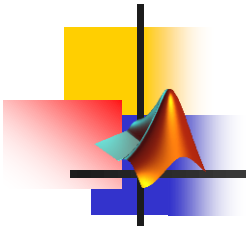
据
代码生

Stateflow
有限状态机理论针对复杂的事件驱动系统进行建模、仿真的工具。

成C语言程序代码的功能、
与C语言程序代码到VHDL
re Description Language，一
功能，可以看出，高级的
系统
或低级的芯片算法设计，都可用MATLAB、Simulink、Stateflow及相关的工具箱来完成。

Stateflow Coder是基于Stateflow状态图生成高效、优化的程序代码。

由这些模块产品之间的关系可以图1.1表示。



1.3 MATLAB编程语言的特点

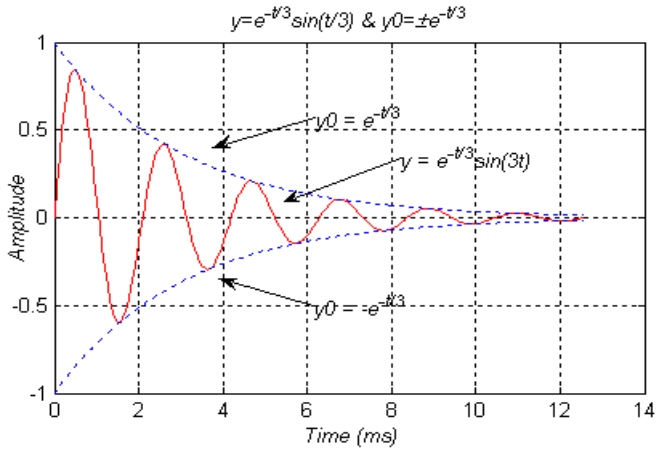
MATLAB语言主要有以下几个特点：

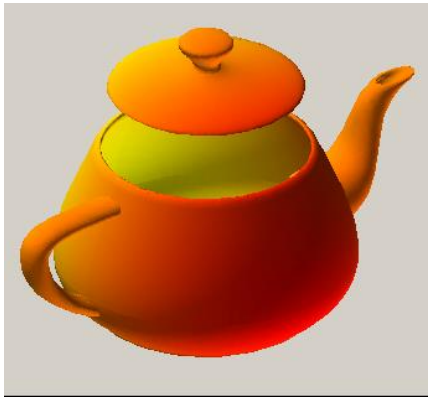
- **语法规则简单**。尤其内定的编程规则，与其他编程语言（如C、Fortran等）相比更接近于常规数学表示。对于数组变量的使用，不需类型声明，无需事先申请内存空间。
- **MATLAB基本的语言环境提供了数以千计的计算函数**，极大的提高了用户的编程效率。如，一个fft函数即可完成对指定数据的快速傅里叶变换，这一任务如果用C语言来编程实现的话，至少要用几十条C语言才能完成。
- **MATLAB是一种脚本式（scripted）的解释型语言**，无论是命令、函数或变量，只要在命令窗口的提示符下键入，并“回车（Enter）”，MATLAB都予以解释执行。
- **平台无关性（可移植性）**。MATLAB软件可以运行在很多不同的计算机系统平台上，如Windows Me/NT/2000/XP、很多不同版本的UNIX以及Linux。无论你在哪一个平台上编写的程序都可以运行在其它平台上，对于MATLAB数据文件也一样，是平台无关的。极大保护了用户的劳动、方便了用户。其绘图功能也是平台无关的。无论任何系统平台，只要MATLAB能够运行，其图形功能命令就能正常运行。

因此，MATLAB是一个简单易用、功能强大的高效编程语言。

总结：

- 功能强大
  - 数值运算优势
  - 符号运算优势(Maple)
  - 强大的2D、3D数据可视化功能
  - 许多具有算法自适应能力的功能函数





- 语言简单、内涵丰富
  - 语言及其书写形式非常接近于常规数学书写形式；
  - 其操作和功能函数指令就是常用的计算机和数学书上的一些简单英文单词表达的，如：help、clear等；
  - 完备的帮助系统，易学易用。
- 扩充能力、可开发能力较强
  - MATLAB完全成了一个开放的系统
  - 用户可以开发自己的工具箱
  - 可以方便地与Fortran、C等语言接口
- 编程易、效率高
  - Matlab以数组为基本计算单元
  - 具有大量的算法优化的功能函数

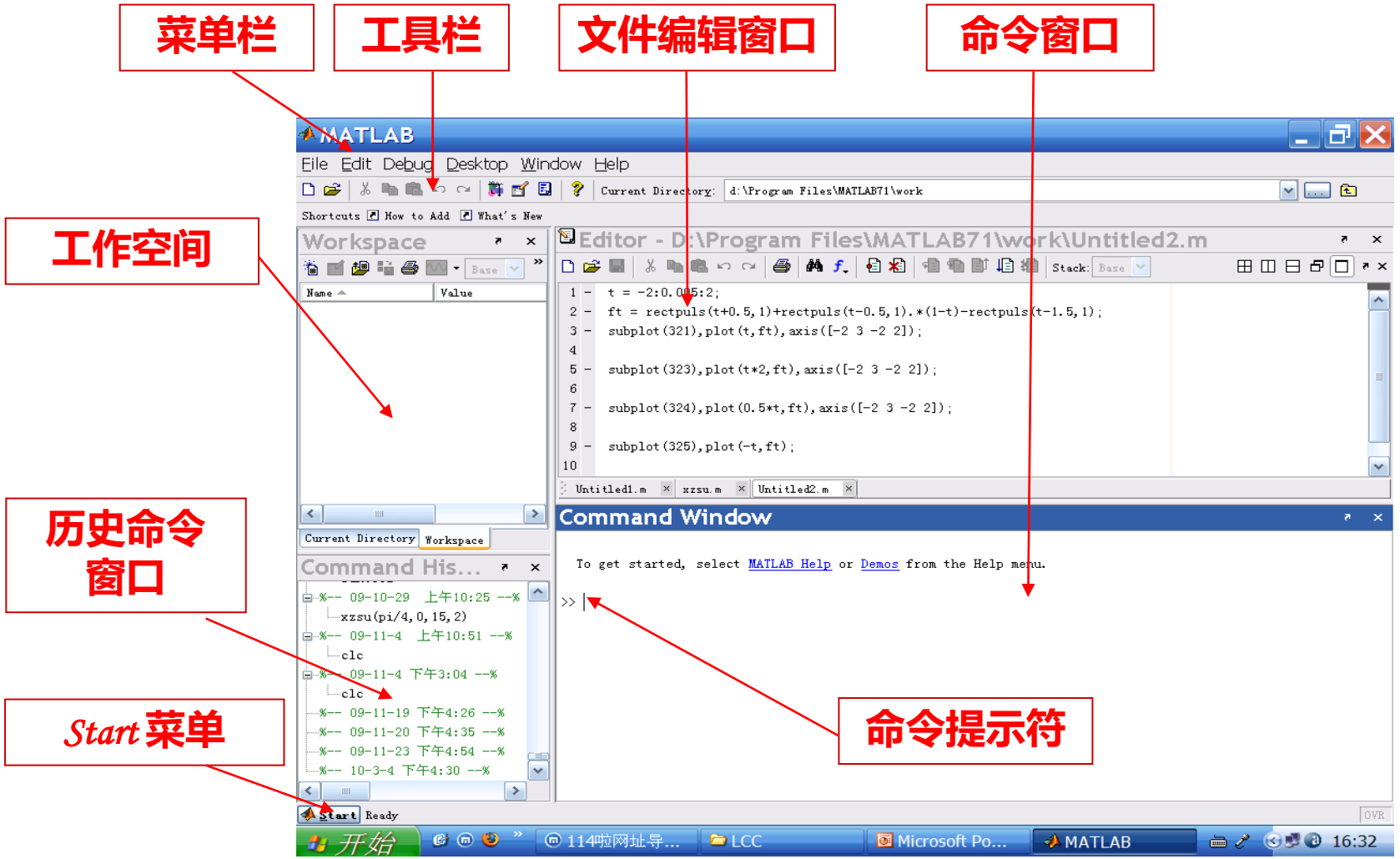
2 MATLAB的桌面环境及入门知识

- 启动与退出MATLAB
- 命令窗口及使用
- 数值表示、变量、表达式
- 命令历史窗口
- 工作空间
- 获取在线帮助

2.1 启动与退出MATLAB

- 启动MATLAB
  - 直接用鼠标双击桌面上MATLAB图标
  - 或Windows桌面的“开始”—〉“所有程序”—〉“MATLAB7.1”
- 退出MATLAB
  - 关闭MATLAB桌面
  - 在命令窗口执行quit或exit命令
- MATLAB缺省桌面（见下页）

2.1 启动与退出MATLAB （续）



2.2 命令窗口的使用

- 激活命令窗口。
- “>>”与闪烁的光标一起表明系统就绪，等待输入。
- 命令窗口脱离MATLAB桌面。

■ 简单计算

【例】计算 $\left[12+2\times(7-4)\right]\div 3^2$

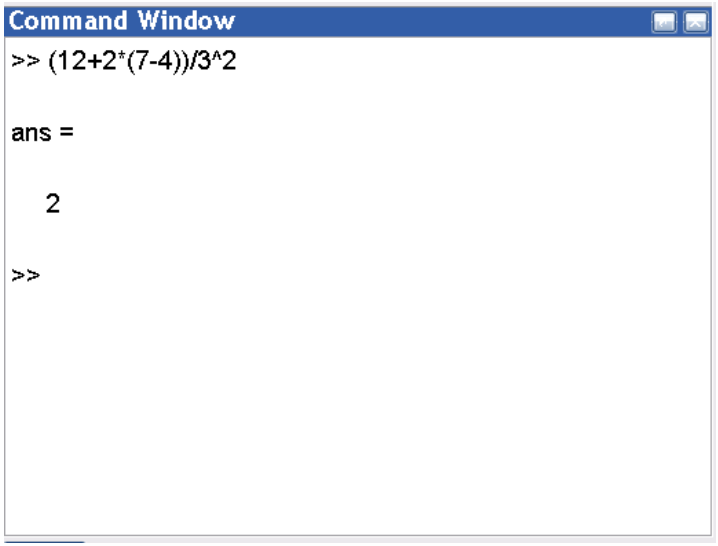
（1）在MATLAB命令窗口输入以下内容:

```
>>(12+2*(7-4))/3^2
```

（2）按【Enter】键，指令执行。

（3）返回的计算结果:

```
ans=
   2
```



2.2 命令窗口（续）

〖说明〗

- 在命令窗口【Enter】键提交命令执行。
- Matlab所用运算符（如+、-、^等）是各种计算程序中常见的。
- 计算结果中的“ans”是英文“answer”的一种缩写，其含义就是“运算答案”。ans是Matlab的一个预定义变量。

2.2 命令窗口（续）

■ 简单计算（续）

【例】计算sin(45°)

```
>>sin(45*pi/180)

ans=
    0.7071
```

- **Matalb**中正弦函数**sin**就是常见的正弦函数。
- 它的参数值是以“弧度”为单位的。
- **pi**也是**Matalb**的预定义变量。
- **pi=3.14159…**
- **Matlab**对字母大小写是敏感的。

【例】计算$\left(\sqrt{2e^{x+0.5}+1}\right)$的值，其中x

```
>>sqrt(2*exp(4.92+0.5)+1)
 ans=
    21.2781
```

- **Matalb**中开平方—**sqrt(x)**，是英文**square root**的缩写 。
- **Matalb**中指数函数**exp(x)**，常见的表达方式。

2.2 命令窗口 （续）

“clc”清除窗口显示内容的命令。

【例】计算 $y=\frac{2\sin(0.3\pi)}{1+\sqrt{5}}$ 的值。

```
>>y=2*sin(0.3*pi)/(1+sqrt(5))

y=
   0.5000
```

【例】计算 $y=\frac{2\cos(0.3\pi)}{1+\sqrt{5}}$ 的值。

```
>>y=2*cos(0.3*pi)/(1+sqrt(5))

y=
   0.3633
```

命令行编辑

- “↑”键调回已输入过命令。
- 修改。

2.2 命令窗口 （续）

【例】计算半径为5.2m的圆的周长和面积。

```
>>radius=5.2; %圆的半径
>>area=pi*5.2^2, circle_len=2*pi*5.2

area =
      84.9487
circle_len =
          32.6726
```

- 以上两例，命令行中用到了等号“=”。
- 计算结果不再赋给“**ans**”，而是赋给用户指定的变量**y**、**area**、**circle_len**。
- 无论是预定义变量还是用户自定义变量都被存储在系统的工作空间内，即系统定义的一个存储窗口变量的内存空间。
- **Who**、**whos**命令用来显示工作空间的变量
- **clear**命令用来清除工作空间的变量。

2.2 命令窗口 （续）

```
>>who
Your variables are:
ans         circle_len    y
area        radius
>>whos
Name          Size                Bytes    Class
  ans         1x1                     8    double array
  area        1x1                     8    double array
  circle_len    1x1                   8    double array
  radius      1x1                     8    double array
  y           1x1                     8    double array
Grand total is 5 elements using 40 bytes
```

2.2 命令窗口 （续）

```
>>clear y
>>who
Your variables are:
ans         circle_len
area        radius
>>clear ans area
>>whos
Your variables are:
Name          Size                Bytes  Class
 circle_len     1x1                    8  double array
 radius         1x1                    8  double array
Grand total is 2 elements using 16 bytes
```

2.2 命令窗口 （续）

- 数值显示格式设置
  - 缺省显示格式：简洁的短（short g）格式
  - 窗口命令及语法格式：format 显示格式关键字

    如：format long %15位数字显示
- 常见通用命令

命令	含义
clc	清除命令窗口的显示内容
clear	清除Matlab工作空间中保存的变量
who或whos	显示Matlab工作空间中的变量信息
dir	显示当前工作目录的文件和子目录清单
cd	显示或设置当前工作目录
type	显示指定m文件的内容
help或lookfor	获取在线帮助
quit或exit	关闭/推出MATALB

2.3 数值表示、变量及表达式

数值的记述

Matlab的数只采用习惯的十进制表示，可以带小数点和负号;其缺省的数据类型为双精度浮点型（double）。

例如：3　-10　0.001　1.3e10　1.256e-6

变量命令规则

- 变量名、函数名对字母的大小写是敏感的。如myVar与myvar表示两个不同的变量。
- 变量名第一个字母必须是英文字母。
- 变量名可以包含英文字母、下划线和数字。
- 变量名不能包含空格、标点。
- 变量名最多可包含63个字符（6.5及以后的版本）。

2.3 数值表示、变量及表达式（续）

Matlab预定义的变量

变量名	意义
ans	最近的计算结果的变量名
eps	MATLAB定义的正的极小值=2.2204e-16
pi	圆周率π
inf	∞值，无限大
i或j	虚数单元，sqrt(-1)
NaN	非数，0/0、∞/ ∞

〖说明〗

- 每当MATLAB启动完成，这些变量就被产生。
- MATLAB中，被0除不会引起程序中断，给出报警的同时用inf或NaN给出结果。
- 用户只能临时覆盖这些预定义变量的值，Clear或重启MATLAB可恢复其值。

2.3 数值表示、变量及表达式（续）

- Matlab预定义的变量

变量名	意义
nargin	**所用函数的输入变量数目**
nargout	**所用函数的输出变量数目**
realmin	**最小可用正实数**
realmax	**最大可用正实数**

2.3 数值表示、变量及表达式（续）

■ 运算符和表达式

运算	数学表达式	MATLAB运算符	MATLAB表达式
加	a+b	+	a+b
减	a-b	-	a-b
乘	axb	*	a*b
除	a/b或a\b	/或\	a/b或a\b
幂	a^b	^	a^b

〖说明〗

- Matlab用“\”和”/”分别表示“左除”和“右除”。对标量而言，两者没有区别。对矩阵产生不同影响。
- MATLAB表达式的书写规则与“手写方式”几乎完全相同。
- 表达式按与常规相同的优先级自左至右执行运算。
- 优先级：指数运算级别最高，乘除次之，加减最低。
- 括号改变运算的次序。

2.3 数值表示、变量及表达式（续）

- 复数及其运算
  - MATLAB中复数的表达：z=a+bi，其中a、b为实数。
  - MATLAB把复数作为一个整体，象计算实数一样计算复数。

【例】复数z1=3+4i，z2=1+2i, z3= $2e^{\frac{\pi}{6}i}$

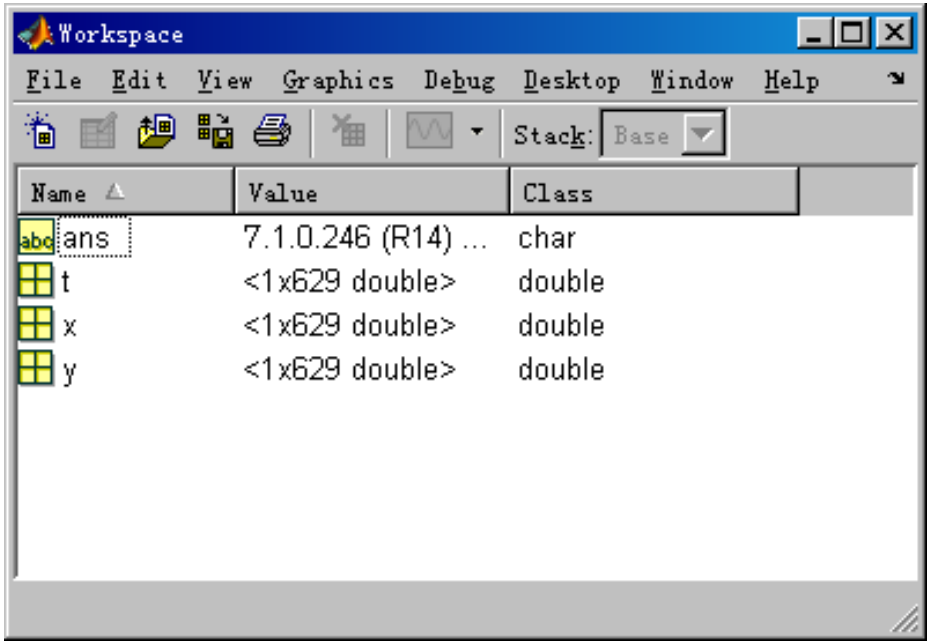
计算 $z=\frac{z_1 z_2}{z_3}$

```
>>z1=3+4*i, z2=1+2*i, z3=2*exp(i*pi/6), z=z1*z2/z3
>>z_real=real(z), z_image=imag(z),
>>z_angle=angle(z), z_length=abs(z),
```

如果**z**是复数，**abs(z)**是实部和虚部的平方和开根号，如果**z**是实数**abs(z)**是绝对值。

2.4 工作空间

- 查看工作空间内存变量，可以由who、whos 。
- 命名新变量。
- 修改变量名
- 删除变量
- 绘图
- 保存变量数据
- 装入数据



2.5 历史窗口

- 历史窗口：
  - 首先记录每次启动时间
  - 并记录在命令窗口输入命令，此次运行期间，输入的所有命令被记录为一组，并以此次启动时间为标志。
- 使用历史窗口：
  - 可以查看命令窗口输入过的命令或语句
  - 可以选择一条或多条命令执行拷贝、执行、创建M文件等。

要清除历史记录，可以选择Edit菜单中的Clear Command History 命令

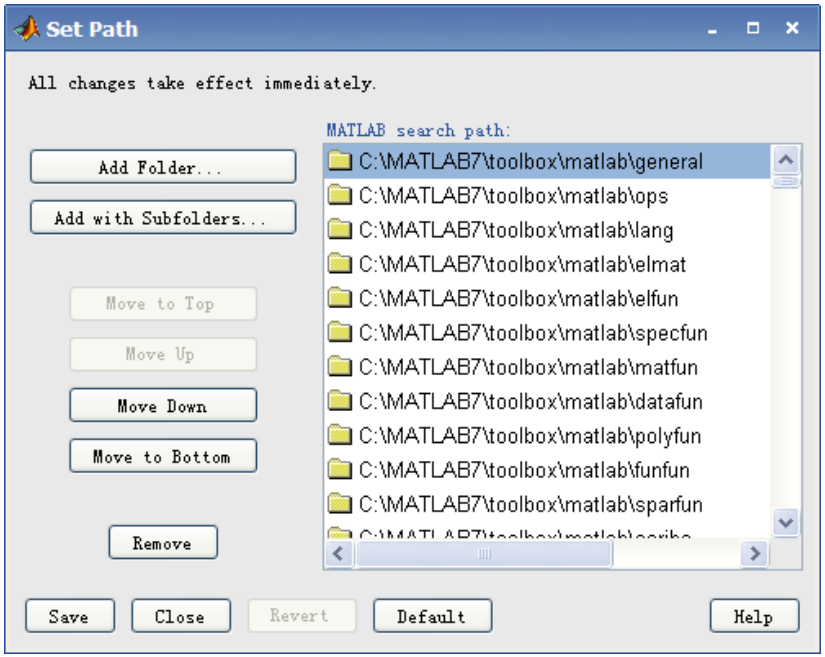
2.6 当前目录窗口和搜索路径

当前目录窗口： 指Matlab运行时的工作目录。

- 只有在当前目录和搜索路径下的文件、函数才可以被运行和调用。
- 如果没有特殊指明，数据文件也将存放在当前目录下；
- 用户可以将自己的工作目录设置成当前目录，从而使得所有操作都在当前目录中进行。

搜索路径：指Matlab执行过程中对变量、函数和文件进行搜索的路径。

- **在File菜单中选择Set Path命令或在命令窗口输入pathtool命令，出现搜索路径设置对话框：**



！修改完搜索路径后，需要进行保存。

2.7 获取在线帮助

- MATLAB提供的帮助信息有两类
  - 简单纯文本帮助信息
    - help
    - lookfor（条件比较宽松）例：inverse
  - 窗口式综合帮助信息（文字、公式、图形）
    - doc
    - helpwin

- help % 帮助总览
  help elfun % 关于基本函数的帮助信息
  help exp % 指数函数exp的详细信息

- lookfor % 根据完整或不完整的关键词搜索
  lookfor integral %查找有关积分的指令
  lookfor fourier %查找能进行傅里叶变换的指令

- doc %以超文本方式给出帮助信息
  doc eig %eig求矩阵的特征值和特征向量

3 数据的输入

简单矩阵的输入

1.直接输入：

矩阵的一行中的元素用空格或逗号分隔；矩阵行与行之间用分号“；”隔开，整个矩阵放在一个[]里。

```
A=[1,2,3;4,5,6;7,8,9]
```

2.矩阵的分行输入：

```
A=[1,2,3
   4,5,6
   7,8,9]
```

3 数据的输入(续)

- 特殊向量和特殊矩阵

1 特殊向量

t=[0:0.1:10]

%产生从1~10的行向量,元素之间间隔为0.1

t=linspace(n1,n2,n)

%产生n1和n2之间线性均匀分布的n个数

t=logspace(n1,n2,n)

%产生10^n1和10^n2之间按照对数距离等距产生n个数

3 数据的输入(续)

2 特殊矩阵

单位矩阵 **eye(m)**　　%得m*m的单位矩阵

eye(m,n)　%得到一个可允许的最大单位矩阵其余处补0

eye(size(a))　%得到与矩阵a同样大小的单位矩阵

所有元素为1的矩阵

ones(n), ones(size(a)), ones(m,n)

所有元素为0的矩阵

zeros(n), zeros(m,n)

3 数据的输入(续)

2 特殊矩阵

空矩阵 **q=[]**

%矩阵q在工作空间之中，但他的大小为0

a(:,3)=[]　%删除矩阵a的第3列

随机数矩阵

rand(m,n)　　%产生m*n矩阵，其中元素是服从[0,1]上的均匀分布的随机数

3 数据的输入(续)

normrnd(mu,sigma,m,n)　%产生m*n矩阵，其中的元素是服从均值为mu、标准差为sigma的正态分布的随机数。

exprnd(mu,m,n)　%产生m*n矩阵，其中的元素是服从均值为mu的指数分布的随机数。

poissrnd(mu,m,n)　%产生m*n矩阵，其中的元素是服从均值 为mu的泊松分布的随机数。

unifrnd(a,b,m,n)　%产生m*n矩阵，其中的元素是服从区间[a,b]上均匀分布的随机数。

3 数据的输入(续)

随机置换

```
randperm(n)        %产生1~n的一个随机全排列

perms([1:n])       %产生1~n的所有全排列
```

【功能演示-1】

求方程 $2x^5-3x^3+71x^2-9x+13=0$ 的全部根。

```
p = [2,0,-3,71,-9,13];%建立多项式系数向量
x = roots(p);  %求根
```

```
x =

  -3.4914
   1.6863 + 2.6947i
   1.6863 - 2.6947i
   0.0594 + 0.4251i
   0.0594 - 0.4251i
```

【功能演示-2】求解线性方程组

$$\begin{cases} 2x+3y-z=2 \\ 8x+2y+3z=4 \\ 45x+3y+9z=23 \end{cases}$$

```
a = [2,3,-1;8,2,3;45,3,9];%建立系数矩阵a
b = [2;4;23];%建立列向量b
x = inv(a)*b
```

```
x =
    0.5531
    0.2051
   -0.2784
```

符号计算

```
syms x y z %建立符号变量
[x,y,z]=solve(2*x+3*y-z-2,8*x+2*y+3*z-4,45*x+3*y+9*z-23)
```

```
x =
151/273
y =
8/39
z =
-76/273
```

【功能演示-3】求解定积分

求解 $I=\int_0^1 x\ln(1+x)dx$

```
quad('x.*log(1+x)',0,1)
```

```
ans =
0.250
```

或

```
syms x
nt(x*log(1+x),0,1)
ans =
1/4
```

【功能演示-4】多项式曲线拟合

- plot是画图函数
- polyval是求值函数
- polyfit用于多项式曲线拟合

注：

$y(x)=x^3-2x^2-5$　　*In MATLAB*　$y=[1\quad -2\quad 0\quad -5]$

p=polyfit(x,y,m)其中 x, y为已知数据点向量, 分别表示横,纵坐标, m为拟合多项式的次数, 结果返回m次拟合多项式系数, 从高次到低次存放在向量p中.

y0=polyval(p,x0)可求得多项式在x0处的值y0

【功能演示-4】多项式曲线拟合

考虑如下 x-y 一组实验数据：

x=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

y=[1.2, 3, 4, 4, 5, 4.7, 5, 5.2, 6, 7.2]

注：$y(x)=x^3-2x^2-5$ *In MATLAB* $y=[1 \quad -2 \quad 0 \quad -5]$

- 一次多项式拟合：

  p1 = polyfit(x,y,1) 

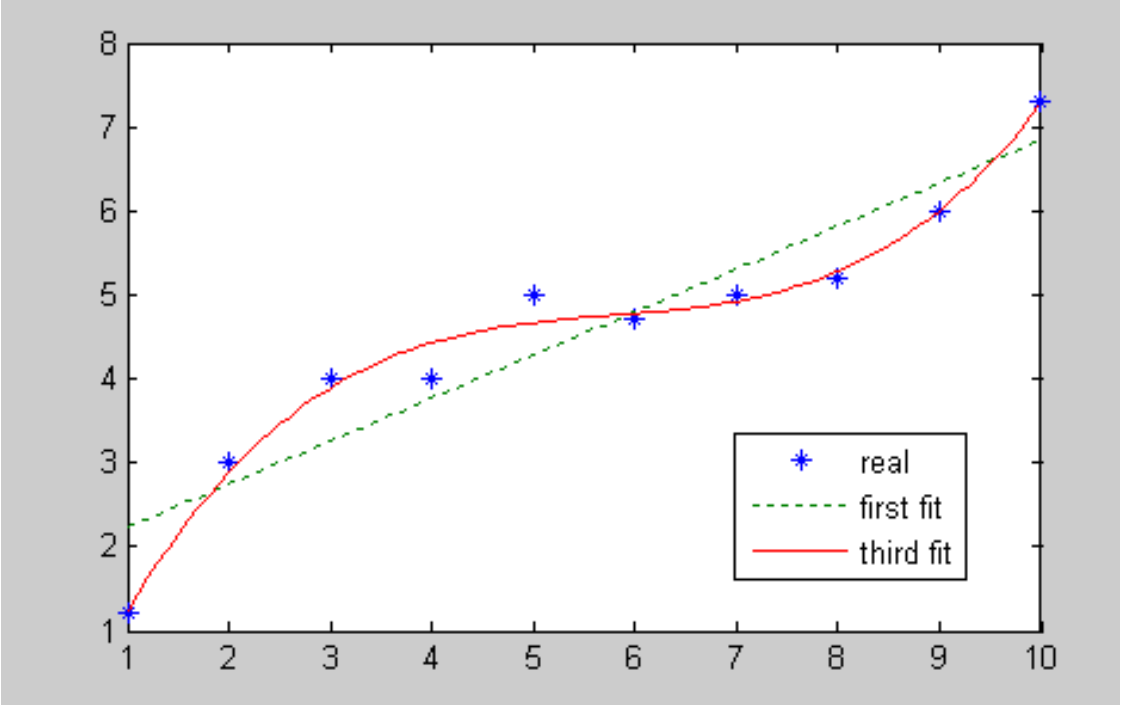
- 三次多项式拟合：

  p3 = polyfit(x,y,3) 

- plot 原始数据、一次拟合曲线和三次拟合曲线

```
x2=1:0.1:10;
y1=polyval(p1,x2)
y3=polyval(p3,x2)
plot( x, y, '*', x2, y1, ':', x2, y3)
```

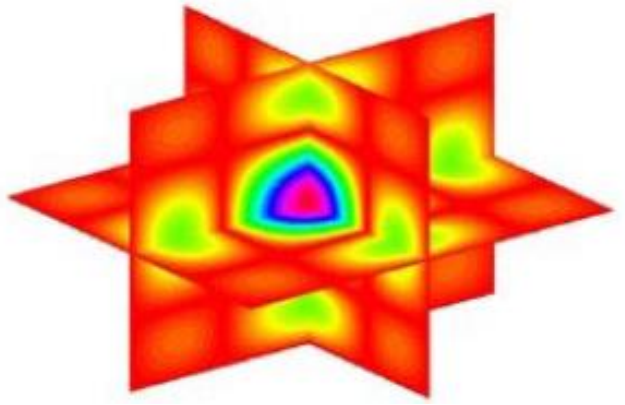
拟合曲线图



由图可见，三次拟合结果较好。

4 Matlab矩阵(数组)的表示

- 数组的概念
- 一维数组变量的创建
- 二维数组变量的创建
- 数组元素的标识与寻访
- 数组运算
- 多维数组



4.1 数组(array)的概念

数组定义:

按行(row)和列(column)顺序排列的实数或复数的有序集，被称为数组。

数组中的任何一个数都被称为这个数组的元素，由其所在的行和列标识，这个标识也称为数组元素的下标或索引。Matlab将标量视为1×1的数组。

对m行、n列的2维数组a:

计为m×n的数组a;

*行标识、列标识均从1开始;

行标识从上到下递增;

列标识从左到右递增。

a=

1	2	3	4	5
2	22	23	24	25
3	32	33	34	35
4	42	43	44	45

a(3, 4)=34 row is first

4.1 数组(array)的概念

- 数组的分类
  - 一维数组，也称为向量(vector) 。
    - 行向量(row vector)、列向量(column vector)。
  - 二维数组(矩阵matrix)。
  - 多维数组。

* 有效矩阵：每行元素的个数必须相同，每列元素的个数也必须相同。

4.1数组(array)的概念（续）

数组 (array)	大小(size)	
$a=\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}$	3×2	
$b=[1 \quad 2 \quad 3 \quad 4]$	1×4	行向量
$c=\begin{bmatrix}1\\ 2\\ 3\end{bmatrix}$	3×1	列向量

a(2,1)=3　　a(1,2)=2　　b(3)=3　　c(2)=2

4.2 创建一维数组变量

- 第一种方法：使用方括号“[ ]”操作符

【例2-1】创建数组(行向量)a=[1 3 pi 3+5i]

```
>>a=[1 3 pi 3+5*i]  %or a=[1, 3, pi, 3+5*i]
a=( 1.0000     3.0000     3.1416     3.0000 + 5.0000i)
```

所有的向量元素必须在操作符“[ ]”之内；

向量元素间用空格或英文的逗点“,”分开。

- 第二种方法：使用冒号“:”操作符

【例2-2】创建以1~10顺序排列整数为元素的行向量b。

```
>>b=1:10
b=1 2 3 4 5 6 7 8 9 10
```

4.2 创建一维数组变量（续）

【例2-3】键入并执行c=1:2:10和d=1:2:9

```
>> c=1:2:10
    c=1 3 5 7 9
>>d=1:2:9
    d= 1 3 5 7 9
```

利用冒号“:”操作符创建行向量的基本语法格式：

x=Start:Increment:End

- Start表示新向量x的第一个元素；
- 新向量x的最后一个元素不能大于End；
- Increment可正可负，若负，则必须Start>End；若正，则必须Start<End，否则创建的为空向量。
- 若Increment=1,则可简写为：x=Start:End。

4.2 创建一维数组变量（续）

- **第三种方法：利用函数linspace**

函数linspace的基本语法

x= linspace(x1, x2, n)

- 该函数生成一个由n个元素组成的行向量；
- x1为其第一个元素；
- x2为其最后一个元素；
- x1、x2之间元素的间隔=(x2-x1)/(n-1)。
- 如果忽略参数n，则系统默认生成100个元素的行向量。

【例2-4】键入并执行x= linspace(1,2,5)

x=1.0000　1.2500　1.5000　1.7500　2.0000

4.2 创建一维数组变量（续）

同学们可以在实验时察看x= linspace(1,2)执行结果。

- **第四种方法：利用函数logspace**

通过实验认识该函数的功能。

4.2 创建一维数组变量（续）

- 列向量的创建
  - 使用方括号“[ ]”操作符，使用分号“;”分割行。

【例2-5】键入并执行x= [1; 2; 3]

```
X=1
  2
  3
```

  - 使用冒号操作符

【例2-6】键入并执行x= (1:3)’　% “ ’ ”表示矩阵的转置

4.2 创建一维数组变量（续）

- 创建数组变量的一般方法
  - 创建变量的赋值语句的一般格式

    var=expression

    - var为变量名
    - expression为MATLAB合法表达式
      - 可以是单独的常数值或数值数组；
      - 也可以由常数值、其他变量（部分或全部）、数值数组和运算符（+、-等）构成。

【例2-7】键入并执行

```
a=[0 1+6]; b=[a 6 7]; c=[6 a 7]; d=[6 a 7 a];
```

4.2 创建一维数组变量（续）

- 一旦被创建，变量就被存储在工作空间，可以通过“Workspace”窗口或在“Command Window”执行“whos”命令察看。

  - 操作一：使用“Workspace”窗口察看变量
  - 操作二：使用“whos”命令察看变量

4.3 创建二维数组变量

- 第一种方法：使用方括号“[ ]”操作符

使用规则

- 数组元素必须在“[ ]”内键入；
- 行与行之间须用分号“;”间隔，也可以在分行处用回车键间隔；
- 行内元素用空格或逗号“,”间隔。

【例3-1】键入并执行a2=[1 2 3;4 5 6;7 8 9]

```
a2=
     1  2  3
     4  5  6
     7  8  9
```

【例3-2】键入并执行a2=[1:3;4:6;7:9] %结果同上

4.3 创建二维数组变量（续）

【例3-3】由向量构成二维数组。

```
>>a=[1 2 3]; b=[2 3 4];
>>c=[a;b];
>>c1=[a b];
```

- 第二种方法：函数方法

函数ones(生成全1矩阵)、zeros (生成全0矩阵) 、reshape

☞ “help elmat”获得基本的矩阵生成和操作函数列表

【例3-4】创建全1的3x3数组。

```
>>ones(3)
```

【例3-5】创建全1的3x4数组。

```
>>ones(3,4)
```

4.3 创建二维数组变量（续）

【例3-5】 reshape的使用演示

```
>>a=-4:4
    a=
      -4 -3 -2 -1 0 1 2 3 4
>>b=reshape(a, 3, 3)
    b=
    -4  -1  2
    -3   0  3
    -2   1  4
```

- 数组元素的排列顺序，从上到下按列排列，先排第一列，然后第二列，…
- 要求数组的元素总数不变。

4.4 数组元素的标识与寻访

- 数组元素的标识
  - “全下标（index）”标识

    经典数学教科书采用“全下标”标识法：每一维对应一个下标。

    - 如对于二维数组，用“行下标和列下标”标识数组的元素，a(2,3)就表示二维数组a的“第2行第3列”的元素。
    - 对于一维数组，用一个下标即可，b(2)表示一维数组b的第2个元素，无论b是行向量还是列向量。

4.4 数组元素的标识与寻访

- **“单下标”（linear index）标识**

  所谓“单下标”标识就是用一个下标来表明元素在数组的位置。

  - 对于二维数组，“单下标”编号：设想把二维数组的所有列，按先后顺序首尾相接排成“一维长列”，然后自上往下对元素位置执行编号。

- **两种“下标”标识的变换：sub2ind、ind2sub**

```
sub2ind   %将矩阵中指定元素的行列下标转换成存储的序列
ind2sub   %将线搜索中的序列转化成矩阵中元素的行列下标
```

4.4 数组元素的标识与寻访

【例4-1】单下标的使用

```
>>a=zeros(2, 5);
>>a(:)=-4:5
    a =
        -4    -2     0     2     4
        -3    -1     1     3     5
```

注意数组的排列顺序。

4.4 数组元素的标识与寻访

元素与子数组的寻访与赋值

【例4-3】一维数组元素与子数组的寻访与赋值

```
>>a=linspace(1,10,5)
    a =
       1.0000    3.2500    5.5000    7.7500   10.0000
>>a(3)      %寻访a的第3个元素
    ans =
       5.5000
>>a([1 2 5])  %寻访a的第1、2、5个元素组成的子数组
    ans =
       1.0000    3.2500   10.0000
```

4.4 数组元素的标识与寻访

```
>>a(1:3)   %寻访前3个元素组成的子数组
    ans =
       1.0000    3.2500    5.5000
>>a(3:-1:1)  %由前3个元素倒序构成的子数组
    ans =
       5.5000    3.2500    1.0000
>>a(3:end)
    ans =
       5.5000    7.7500   10.0000
>>a(3:end-1)
    ans =
       5.5000    7.7500
```

- %第3个及其后所有元素构成的子数组
- 函数end作为参数使用，返回最后一个元素的下标

4.4 数组元素的标识与寻访

```
>>a([1 2 3 5 5 3 2 1])
    ans =
        1.0000    3.2500    5.5000   10.0000   10.0000    5.5000
        3.2500    1.0000
```

- 数组元素可以被任意重复访问，构成长度大于原数组的新数组。

```
>>a(6)
    ??? Index exceeds matrix dimensions.
```

- 下标值超出了数组的维数，导致错误

```
>>a(2.1)
    ??? Subscript indices must either be real positive
        integers or logicals.
```

- 下标值只能取正整数或逻辑值

4.4 数组元素的标识与寻访

```
>>a(3)=0          %修改数组a的第3元素值为0
    a =
       1.0000    3.2500    0    7.7500   10.0000
>>a([2 5])=[1 1]
    a =
       1.0000    1.0000    0    7.7500   1.0000
```

- 可以修改指定数组元素的值
- 一次可以修改多个数组元素的值
- 要修改的数组元素的个数应与送入数组的元素个数相同

4.4 数组元素的标识与寻访

【例4-3】二维数组元素与子数组的寻访与赋值

```
>>a_2=zeros(2, 4) %创建2x4的全0数组
    a_2 =
        0    0    0    0
        0    0    0    0
>>a_2(:)=1:8
    a_2 =
        1    3    5    7
        2    4    6    8
>>a_2([2 5 8])   %单下标方式寻访多个元素
    ans =
        2    5    8
```

注意元素的排列顺序

4.4 数组元素的标识与寻访

```
>> a_2([2 5 8]) =[10 20 30]
    a_2 =
        1     3    20     7
       10     4     6    30
>>a_2(:,[2 3])=ones(2)  %双下标方式寻访并修改
    a_2 =
        1     1     1     7
       10     1     1    30
```

4.4 数组元素的标识与寻访

- 二维数组可以“单下标”方式或“全下标”方式访问、赋值；
- “单下标”方式赋值时，等号两边涉及的元素个数必须相等；
- “全下标”方式赋值时，等号右边数组的大小必须等于原数组中涉及元素构成的子数组的大小。

4.4 数组元素的标识与寻访

```
>>a_2(:,end)
    ans =
        7
       30
>>a_2(:,end-1)
    ans =
        1
        1
>>a_2(:, end:-1:3)
    ans =
        7     1
       30     1
```

```
>>a_2(end,:)
    ans =
       10     1     1    30
>>a_2(end,[2:4])
    ans =
        1     1    30
>>a_2 ([4 6])=6:7
    a_2 =
        1     1     1     7
       10     6     7    30
>>a_2(end,[2:end-1])
```

What is the result?

4.4 数组元素的标识与寻访

【例4-4】 size、length函数

```
>>a=ones(4,6)*6
>>m=size(a)
>>len=length(a)
>>b=1:5;
>>length(b)
>>c=b'
>>length(c)
```

- size函数返回变量的大小，即变量数组的行列数
- length函数返回变量数组的最大维数

4.4 数组元素的标识与寻访

■ 双下标到单下标的转换

【例4-5】 sub2ind函数-双下标转换为单下标

```
>>A = [17 24 1 8; 2 22 7 14; 4 6 13 20];
>>A(:,:,2) = A - 10
>>A(2,1,2)
>>sub2ind(size(A),2,1,2)
>>A(14)
```

4.4 数组元素的标识与寻访

■ 单下标到双下标的转换

【例4-6】 ind2sub函数-双下标转换为单下标

```
>>b = zeros(3);
>>b(:) = 1:9
>>IND = [3 4 5 6]
>>[I,J] = ind2sub(size(b),IND)
```

4.5 多维数组

多维数组的定义

在 MATLAB 的数据类型中，向量可视为一维数组，矩阵可视为二维数组，

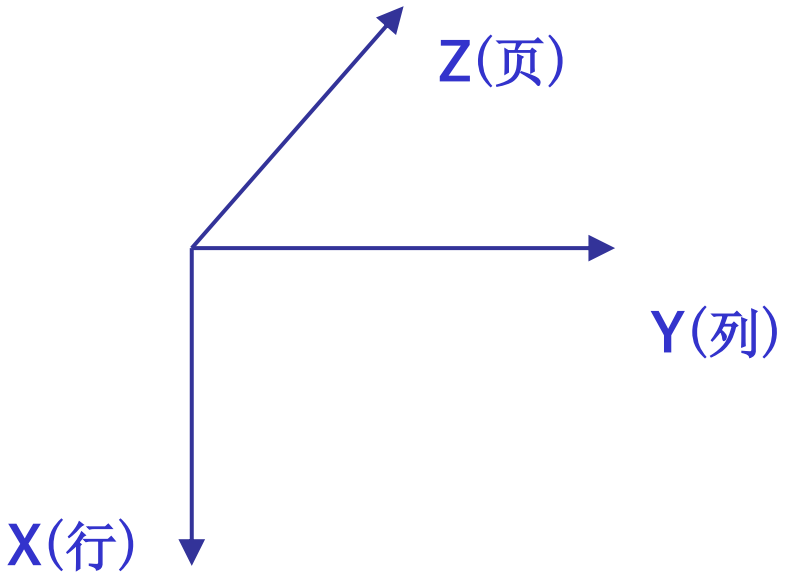
对于维数(Dimensions)超过2 的数组均可视为「多维数组」(Multidimesional Arrays，简称N-D Arrays)。

4.5 多维数组（续）

- 将两个二维（平面）数组叠在一起，就构成三维数组，**第三维称为「页」(Page)**，如下图所示：

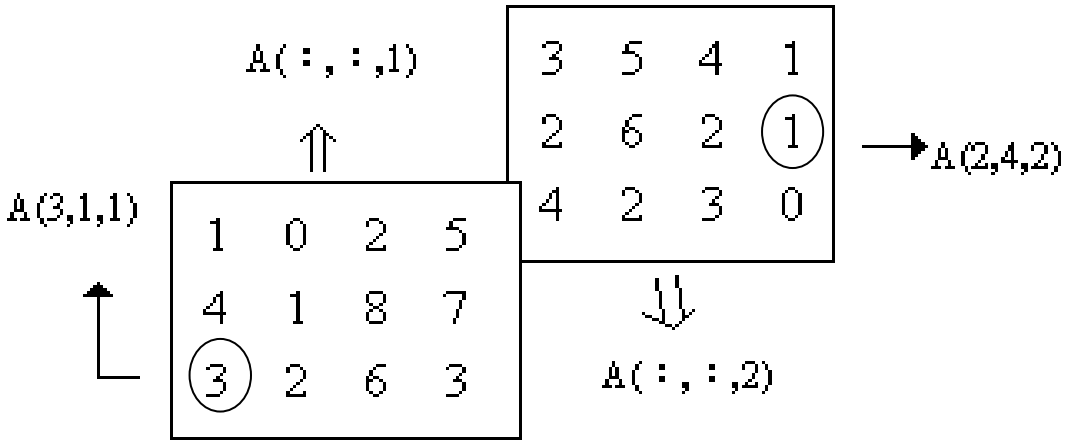
4.5 多维数组（续）

- 三维数组，可对应至一个X-Y-Z三维立体坐标，如下图所示：



4.5 多维数组（续）

- 三维数组元素的寻址：可以(行、列、页)来确定。
- 以维数为 3×4×2 的三维数组为例，其寻址方式如下图所示：



- 数组 A 是三维数组，其中 A(：,：,1)代表第一页的二维数组，A(：,：,2)代表第二页的二维数组。

4.5 多维数组（续）

- 多维数组的建立
  - 建立一个简单的多维数组，可直接由MATLAB 命令视窗内输入（使用“[ ]”操作符）
  - 例：由两个相同大小二维数组创建三维数组

```
A(:, :, 1) = [1 0 2 5; 4 1 8 7; 3 2 6 3];

A(:, :, 2) = [3 5 4 1; 2 6 2 1; 4 2 3 0]
```

```
A(:,:,1) =
     1     0     2     5
     4     1     8     7
     3     2     6     3

A(:,:,2) =
     3     5     4     1
     2     6     2     1
     4     2     3     0
```

4.5 多维数组（续）

执行命令：whos A，得到如下结果：

Name	Size	Bytes	Class
A	3x4x2	192	double array

Grand total is 24 elements using 192 bytes

4.6、数组的算术运算

- MATLAB数组支持线性代数中所有的矩阵运算。
- 建立特有的数组运算符，如：“.*”、“./”等。

4.6、数组的算术运算

MATLAB数组运算符列表

运算	运算符	含义说
加	+	相应元素相加
减	-	相应元素相减
乘	*	矩阵乘法
点乘	.*	相应元素相乘
幂	^	矩阵幂运算
点幂	.^	相应元素进行幂运算
左除或右除	\或/	矩阵左除或右除
左点除或右点除	.\或./	A的元素被B的对应元素除

4.6、数组的算术运算（续）

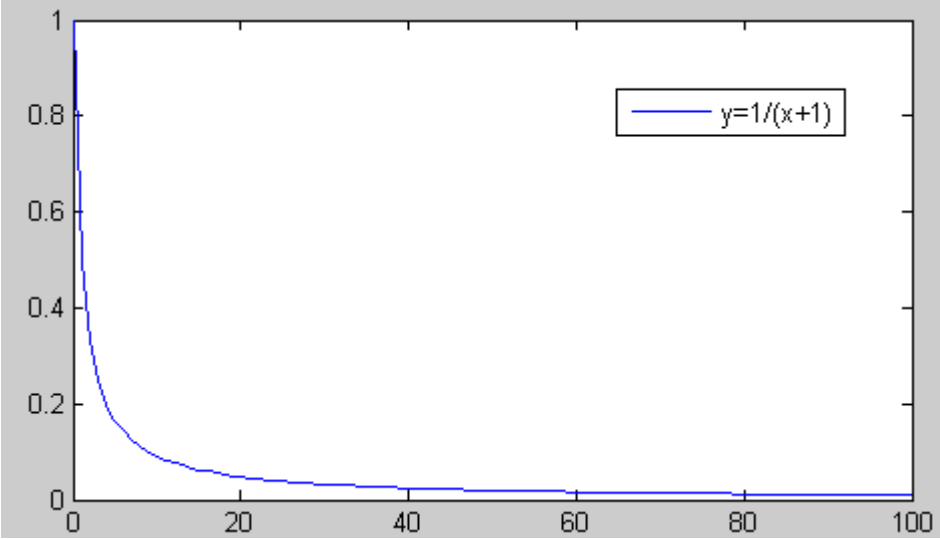
【例5-1】数组加减法

```
>>a=zeros(2, 3);
>>a(:)=1:6;
>>b=a+2.5
b =
    3.5000    5.5000    7.5000
    4.5000    6.5000    8.5000
>>c=b-a
   c =
    2.5000    2.5000    2.5000
    2.5000    2.5000    2.5000
```

4.6、数组的算术运算（续）

【例5-2】画出y=1/(x+1)的函数曲线，x∈[0, 100]。

```
x=0:100;
y=1./(x+1);
plot(x, y);
legend('y=1/(x+1)');
```



4.6、数组的算术运算（续）

【例5-2】生成一个信号：

x=sin(2*pi*t)+sin(4*pi*t)

```
t = [0:199]./100;      %采样时间点
x = sin(2*pi*t) + sin(4*pi*t); % 生成信号
plot(t,x);
legend('x = sin(2*pi*t) + sin(4*pi*t)');
```

4.6、数组的算术运算（续）

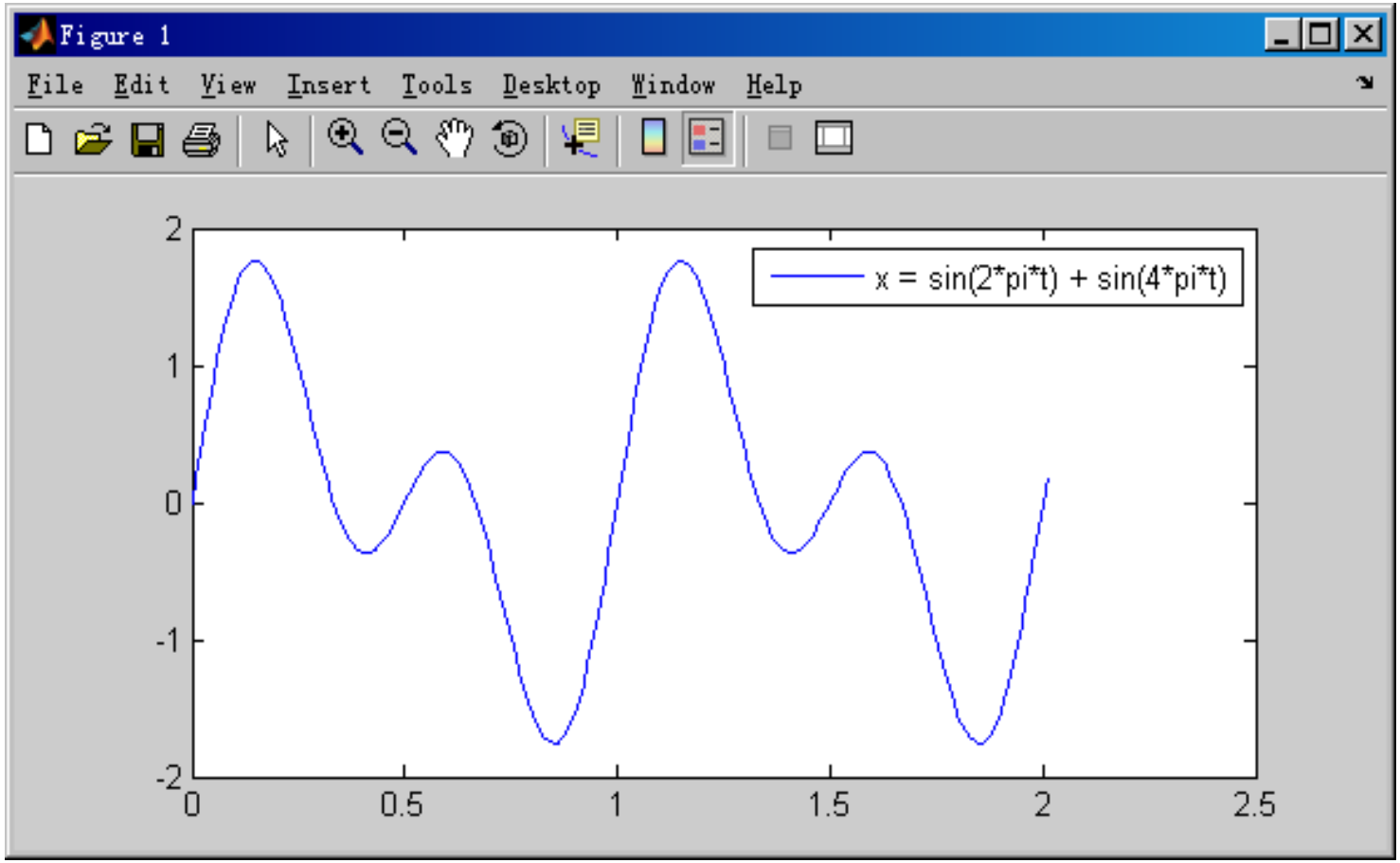


Figure 1
File Edit View Insert Tools Desktop Window Help
x = sin(2*pi*t) + sin(4*pi*t)
2
1
0
-1
-2
0
0.5
1
1.5
2
2.5

4.6、数组的算术运算（续）

【例5-2】点幂“.^”

举例

```
>>a=1:6
   a =
      1   2   3   4   5   6
>>b=reshape(a,2,3)
   b =
      1    3    5
      2    4    6
>>a=a.^2
   a =
      1   4   9   16   25   36
>>b=b.^2
   b =
      1    9    25
      4    16   36
```

4.7、关系运算

Matlab提供了6种关系运算符：

<、>、<=、>=、==、~=（不等于）

关系运算符的运算法则：

1、当两个标量进行比较时，直接比较两数大小。若关系成立，结果为1，否则为0。

2、当两个维数相等的矩阵进行比较时，其相应位置的元素按标量关系进行比较，并给出结果，形成一个维数与原来相同的0、1矩阵。

3、当一个标量与一个矩阵比较时，该标量与矩阵的各元素进行比较，结果形成一个与矩阵维数相等的0、1矩阵。

4.7、关系运算（续）

【例】建立5阶方阵A，判断其元素能否被3整除。

```
A = [24, 35, 13, 22, 63; 23, 39, 47, 80, 80; ...
   90, 41, 80, 29, 10; 45, 57, 85, 62, 21; 37, 19, 31, 88, 76]
A =
    24    35    13    22    63
    23    39    47    80    80
    90    41    80    29    10
    45    57    85    62    21
    37    19    31    88    76
P = rem(A,3)  %被3除，求余
P =
     1     0     0     0     1
     0     1     0     0     0
     1     0     0     0     0
     1     1     0     0     1
     0     0     0     0     0
```

4.8 逻辑运算

Matlab提供了3种逻辑运算符：

&（与）、|（或）、~（非）

逻辑运算符的运算法则：

1、在逻辑运算中，确认非零元素为真（1），零元素为假（0）。

2、当两个维数相等的矩阵进行比较时，其相应位置的元素按标量关系进行比较，并给出结果，形成一个维数与原来相同的0、1矩阵；

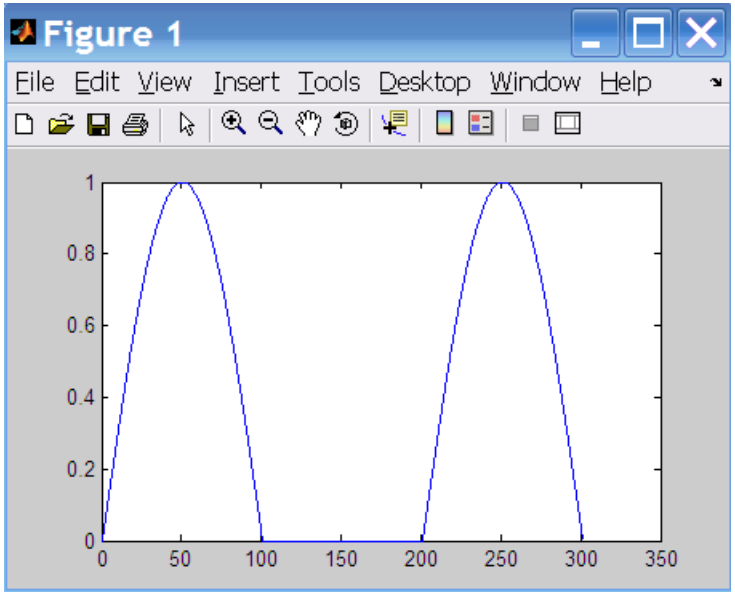
3、当一个标量与一个矩阵比较时，该标量与矩阵的各元素进行比较，结果形成一个与矩阵维数相等的0、1矩阵；

4、算术运算优先级最高，逻辑运算优先级最低。

4.8 逻辑运算

【例】在[0，3π]区间，求 $y=\sin x$ 的值。要求消去负半波，即(π,2π)区间内的函数值置零。

```
x = 0:pi/100:3*pi;
y = sin(x);
y1 = (y>=0).*y;  %消去负半波
plot(x,y1)
```



4.8 逻辑运算

【例】建立矩阵A，找出在[10，20]区间的元素的位置。

```
A = [4,15,-45,10,6;56,0,17,-45,0];
find(A>=10 & A<=20) %找到非零元素的位置
```

```
A =
4   15  -45   10    6
56   0   17  -45    0
ans =
3
6
7
```

5. 数据分析与统计

最大值和最小值

MATLAB提供的求数据序列的最大值和最小值的函数分别为**max**和**min**，两个函数的调用格式和操作过程类似。

1、求向量的最大值和最小值

求一个向量X的最大值的函数有两种调用格式，分别是：

(1) **y=max(X)**：返回向量X的最大值存入y，如果X中包含复数元素，则按模取最大值；

(2)**[y,I]=max(X)**：返回向量X的最大值存入y，最大值的序号存入I，如果X中包含复数元素，则按模取最大值。

求向量X的最小值的函数是**min(X)**，用法和**max(X)**完全相同。

【例】求向量的最大值

```
>>x=[-43,72,9,16,23,47];
>>y=max(x)     %求向量x中的最大值
y =
     72
>>[y,l]=max(x) %求向量x中的最大值及其该元素的位置
y =
    72
l =
    2
```

5. 数据分析与统计

2. 求矩阵的最大值和最小值

求矩阵A的最大值的函数有3种调用格式，分别是:

(1) max(A)：返回一个行向量，向量的第i个元素是矩阵A的第i列上的最大值;

(2) [Y,U]=max(A)：返回行向量Y和U，Y向量记录A的每列的最大值，U向量记录每列最大值的行号;

(3) max(A,[],dim)：dim取1或2。dim取1时，该函数和max(A)完全相同；dim取2时，该函数返回一个列向量，其第i个元素是A矩阵的第i行上的最大值。

求最小值的函数是min，其用法和max完全相同。

【例】求矩阵的最大值

```
>>x=[-43,72,9; 16,23,47];
>>y=max(x)     %求矩阵x中每列的最大值
y =
    16  72  47
>>[y,l]=max(x) %求矩阵x中每列的最大值及其该元素的位置
y =
   16  72  47
l =
    2  1  2
>>max(x, [],2) %求矩阵中每行的最大值
```

5. 数据分析与统计

■ 求和与求积

sum(X)：返回向量X各元素的和。

prod(X)：返回向量X各元素的乘积。

sum(A)：返回一个行向量，其第i个元素是A的第i列的元素和。

prod(A)：返回一个行向量，其第i个元素是A的第i列的元素乘积。

sum(A,dim)：当dim为1时，该函数等同于sum(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的各元素之和。

prod(A,dim)：当dim为1时，该函数等同于prod(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的各元素乘积。

5. 数据分析与统计

平均值与中值

求数据序列平均值的函数是**mean**，求数据序列中值的函数是**median**。两个函数的调用格式为：

mean(X)：返回向量X的算术平均值。

median(X)：返回向量X的中值。

mean(A)：返回一个行向量，其第i个元素是A的第i列的算术平均值。

5. 数据分析与统计

median(A)：返回一个行向量，其第i个元素是A的第i列的中值。

mean(A,dim)：当dim为1时，该函数等同于mean(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的算术平均值。

median(A,dim)：当dim为1时，该函数等同于median(A)；当dim为2时，返回一个列向量，其第i个元素是A的第i行的中值。

6 字符串（string、array of characters）

- 关于字符串
  - MATLAB处理字符(Characters)与字符串(Strings)的相关指令大部分都放在下列目录之中：

    {MATLAB根目录}\toolbox\matlab\strfun

  - 其中的「strfun」就是代表「String Functions」。若要查询与字符和字符串相关的指令，可在MATLAB 下输入：

    help strfun 或是 help strings

6.1 字符串的概念

- 字符（Characters）可以构成一个字符串（Strings），或字符数组(character array)。
- 一个字符串是被视为一个行向量（row vector）。
- 字符串中的每一个字符（含空格），以其ASCII 码的形式存放于行向量中，是该字符串变量的一个元素（element）。

6.1 字符串的概念

- Matlab 用「单引号」来界定一个字符串。
- 可以使用方括号“[ ]”直接连接多个字符串变量，得到一个新字符串变量。

【例3-2】命名字符串变量

```
str1 = 'I like MATLAB,';        % 建立字串变量 str1
str2 = ' JavaScript, and Perl!';        % 建立字串变量str2
str3 = [str1 str2]              % 直接连接str1及str2，以建立str3
```

```
str3 =
    I like MATLAB, JavaScript, and Perl!
```

6.1 字符串的概念

- 如要输入的字符串中有单引号，则由两个连续的单引号来表示。
- 若要计算字符串变量的长度（即组成字符串的个数），可用 **length** 指令。
- 【例3-2】含单引号字符串的输入

```
sentence = 'I''ve got a date!';
length(sentence)   % 计算字字符串sentence的长度
```

```
ans =     16
```

6.1 字符串的概念

- double 指令: 查看字符串变量的存储内容（即 ASCII 内码）
- char 指令: 将 ASCII 内码转换为字符串形式

【例3-2】：字符串与ASCII码的相互转换

```
sentence = 'I''ve got a date!';
sentenceAscii = double(sentence)   %查看 sentence 的 ASCII 码
sentence2 = char(sentenceAscii)          % 将 ASCII 码恢复成字符串形式
```

6.1 字符串的概念

```
sentenceAscii =

    73   39   118   101   32   103   111   116   32   97
    32   100  97    116   101   33

sentence2 =

    I've got a date!
```

6.1 字符串的概念

- **class** 或 **ischar** 指令: 可以用来判断一个变量的类型或它是否为字符串变量。

【例3-4】：判断一个变量是否为字符串变量。

```
chinese = '今日事，今日毕';
out1 = class(chinese)      % out1 的值是 “char”
x = chinese+1;
out2 = ischar(x)                % out2 的值是 0，代表 x 不是字符串变量
```

6.2. 一个字符数组变量存储多行字符串

- 第一种方法是使用二维字符数组（Two Dimensional Character Arrays）
- 必须先确认每个字符串（即每一行）的长度一样，否则就必须在短字符串结尾补齐空格

【例3-5】：多行字符串变量

```
departments = [‘ee  ’; ‘cs  ’; ‘econ’]
```

＊ 注意上述语句中空格字符的使用。

```
departments =
     ee
     cs
     econ
```

6.2 一个变量存储多个字符串 （续）

- 用char 指令存储多字符串

```
departments = char('ee', 'cs', 'econ')      % 注意空格及「,」的使用
```

得到结果和上例的一样;

- 从二维字符数组访问字符串时，切记要使用 deblank 指令来清除字符串尾部的空格字符

6.2 一个变量存储多个字符串（续）

【例3-5】：使用deblank命令清除字符串尾部空格

```
departments = char('ee', 'cs', 'econ');
dept1 = departments(1,:);    % (1,:)代表第一行的全部元素
dept2 = deblank(dept1);      % 使用 deblank 指令清除尾部的空格字符
len1 = length(dept1)         % 显示变量 dept1 的长度=4
len2 = length(dept2)         % 显示变量 dept2 的长度=2
```

6.3 字符串的操作

strcmp 指令: 用于比较字符串的内容的异同,不相等返回0,相等返回1

【例3-6】：字符串比较

```
str1 = 'today';
str2 = 'tomorrow';
str3 = 'today';
out1 = strcmp(str1, str2)      % 比较字符串 str1 和 str2
out1 = 0                       %表示字符串 str1 和 str2不同
out2 = strcmp(str1, str3)      % 比较字符串 str1 和 str3
out2 = 1                       %表示字符串 str1 和 str2相同
```

6.4 空数组（empty array）

- 有一维是0的数组即为空数组
- 空数组不占据存储空间
- 最简单的空数组：0 x 0的矩阵
- 复杂的空数组：0 x 5 or 10 x 0

例如：>>a=[]; b=ones(0,5);

察看空数组：>>a, b, c　% or whos a b c

※ 空数组并非全0数组

6.4 空数组（续）

- 数组维数的减小
  - 删除数组的某列和行

```
>>a = magic(4), a(:,2)=[]
```

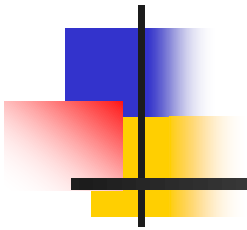
  - 删除(2-D、3-D)数组的单个元素
    - 使用“全下标”方式，不能删除单个元素

```
>>a(1, 2)=[]   %系统会警告信息
```

    - 使用“单下标”可以删除单个元素

```
>>a(2:4)=[]   %数组a将变为向量
```

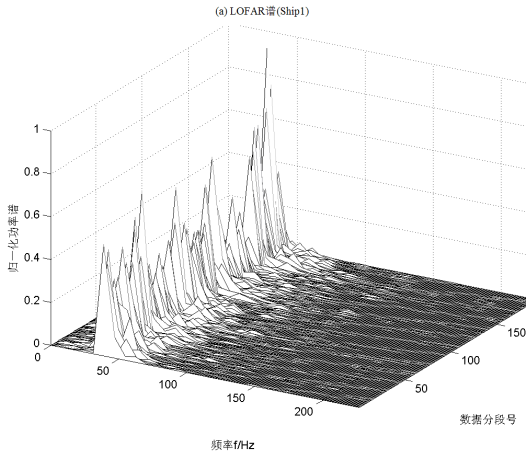
  - 使用“[]”同样可以减小字符数组的维数



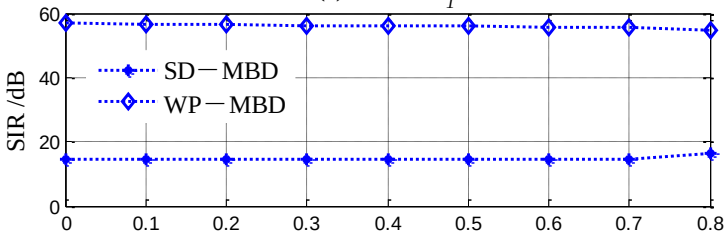
数据和函数的可视化

引言

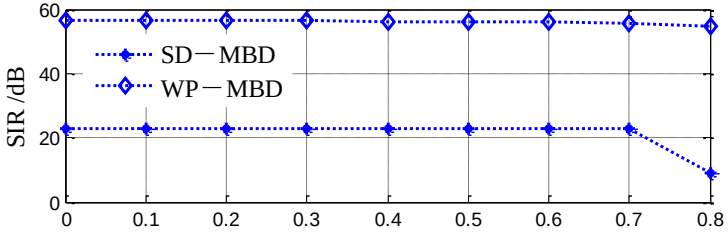
- 世界顶级的数值计算工具软件MATLAB具有极其强大的数据可视化功能，可制作具有出版质量图形。
- 在前面的课程中，已经使用了数据可视化命令plot。
- 详细介绍MATLAB这一部分的内容可以写一本书。
- 我们只能介绍MATLAB数据可视化的基础，2-D数据可视化、3-D数据可视化初步。
- 二维图形是将平面坐标上的数据点连接起来的平面图形。可以采用直角坐标系、对数坐标、极坐标等形式。数据点可以用向量或矩阵形式给出，类型可以是实型或复型。



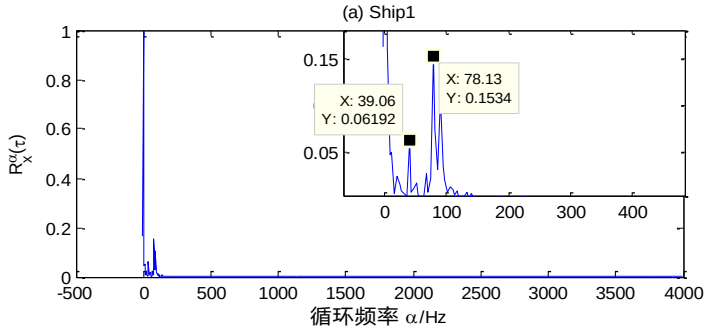
(a) 源信号 s_1

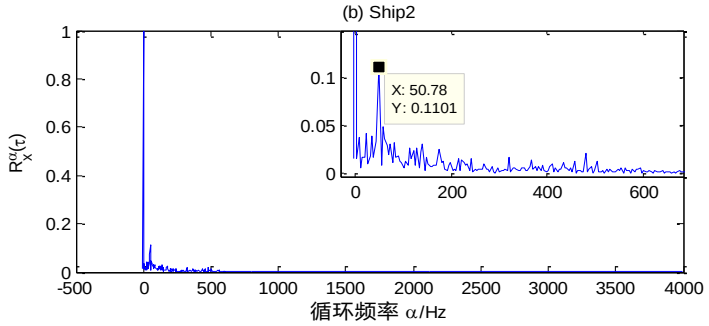


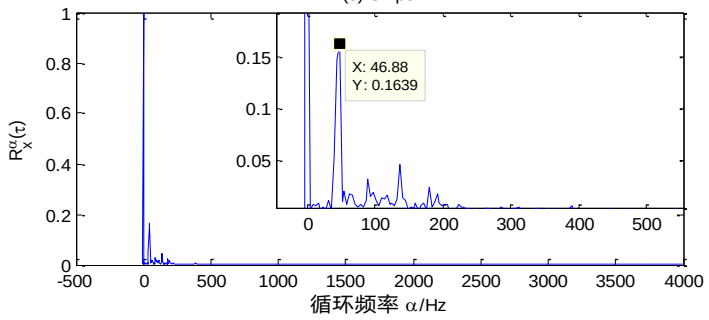
(b) 源信号 s_2

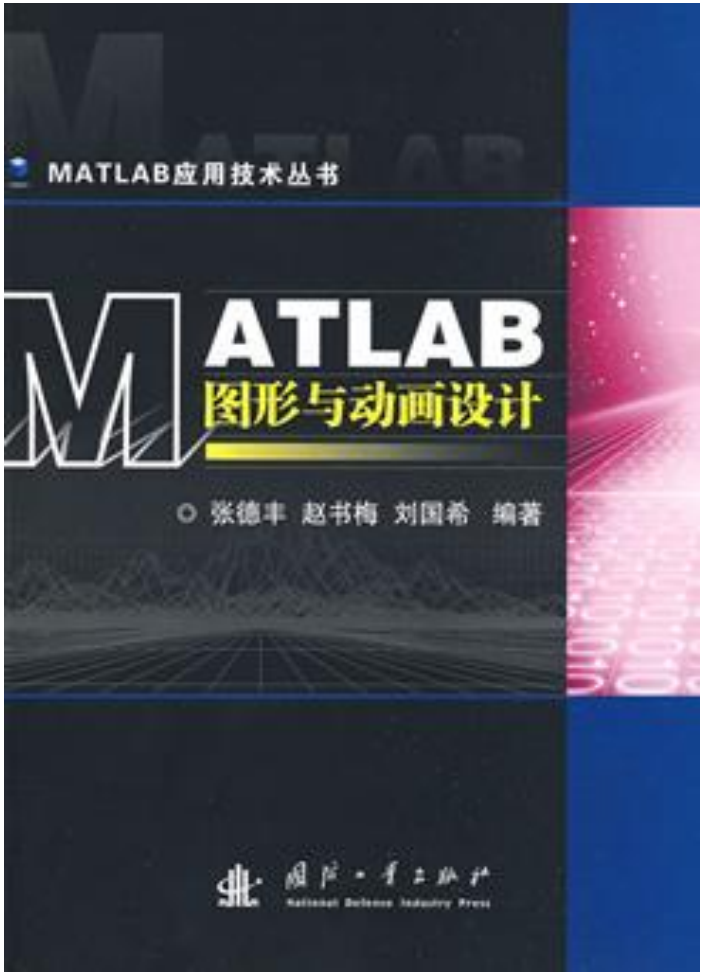


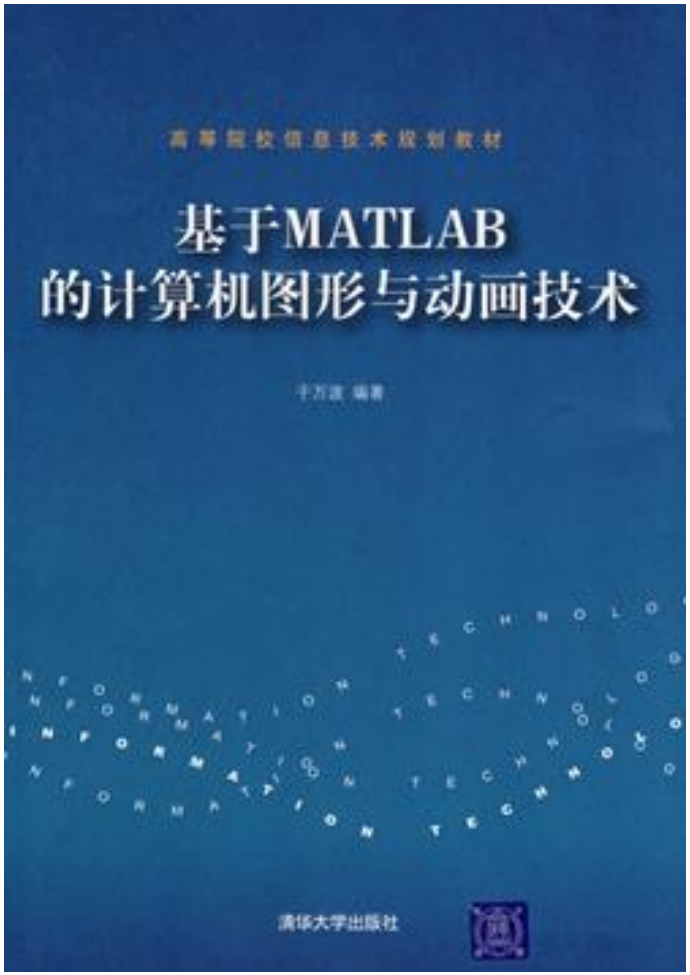
相似系数 ρ








MATLAB应用技术丛书
MATLAB
图形与动画设计
张德丰 赵书梅 刘国希 编著


基于MATLAB
的计算机图形与动画技术
清华大学出版社

二维曲线绘图的基本操作

- plot指令的基本调用格式

（1）plot(x)

- x为向量时，以该元素的下标为横坐标、元素值为纵坐标绘出曲线
- x为实数二维数组时，则按列绘制每列元素值相对其下标的曲线，曲线数等于x数组的列数。
- x为复数二维数组时，则按列分别以数组的实部和虚部为横、纵坐标绘制多条曲线

（2）plot(x, y)

- x、y为同维数组时，绘制以x、y元素为横纵坐标的曲线
- x为向量，y为二维数组、且其列数或行数等于x的元素数时，绘制多条不同颜色的曲线
- x为二维数组，y为向量时，情况与上相同，只是y仍为纵坐标。

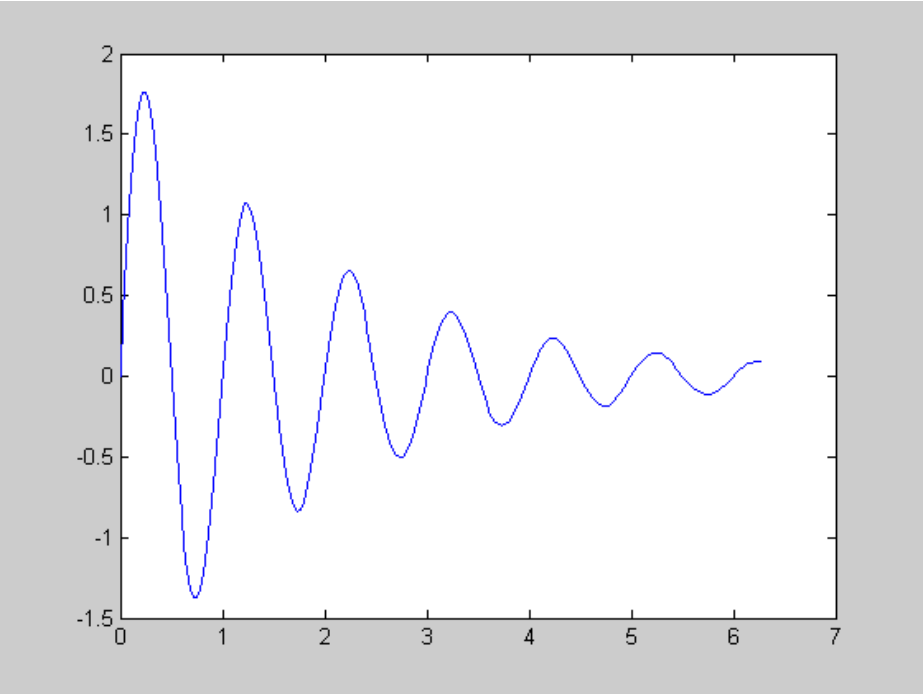
（3）plot(x1, y1, x2, y2, …)

- 绘制以x1为横坐标、y1为纵坐标的曲线1，以x2为横坐标、y2为纵坐标的曲线2，等等。
- 其中x为横坐标，y为纵坐标，绘制y=f(x)函数曲线。

例1使用直角坐标系

在[0，2π]区间内，绘制曲线 $y = 2e^{-0.5x}\sin(2\pi x)$

```
x = 0:pi/100:2*pi;
y = 2*exp(-0.5*x).*sin(2*pi*x);
plot(x,y)
```

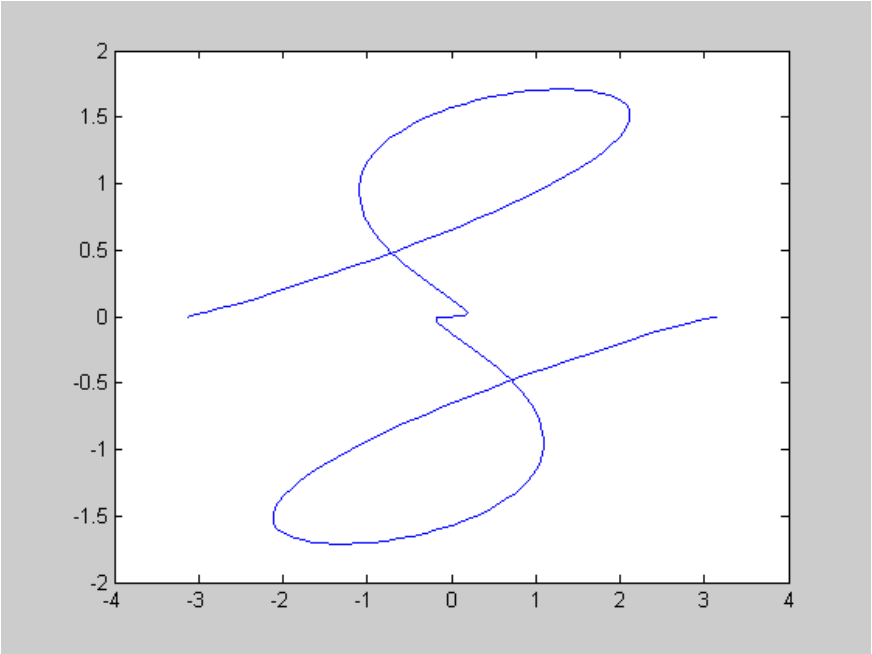


例2 使用参数方程绘制曲线

绘制曲线

$$\begin{cases} x = t\cos(3t) \\ y = t\sin^2 t \end{cases}, -\pi \le t \le \pi$$

```
t = -pi:pi/100:pi;
x = t.*cos(3*t);
y = t.*sin(t).^2;
plot(x,y)
```

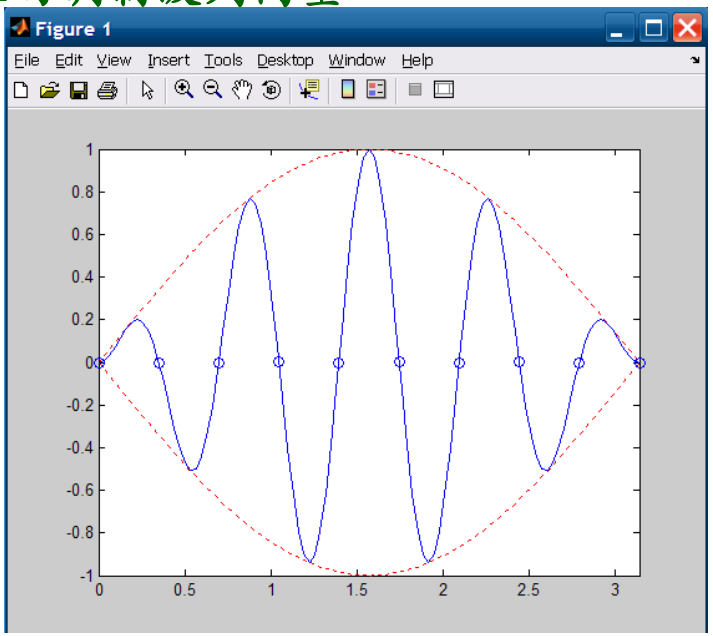


例3 绘制复杂曲线

用图形表示连续调制波形 及其包络线。

```
t=(0:pi/100:pi)';        %长度为101的时间采样列向量
y1=sin(t)*[1,-1];        %包络线函数值，是（101x2）的矩阵
y2=sin(t).*sin(9*t);     %长度为101的调制波列向量
t3=pi*(0:9)/9;

y3=sin(t3).*sin(9*t3);
plot(t,y1,'r:',t,y2,'b',t3,y3,'bo')
axis([0,pi,-1,1])
%控制轴的范围
```



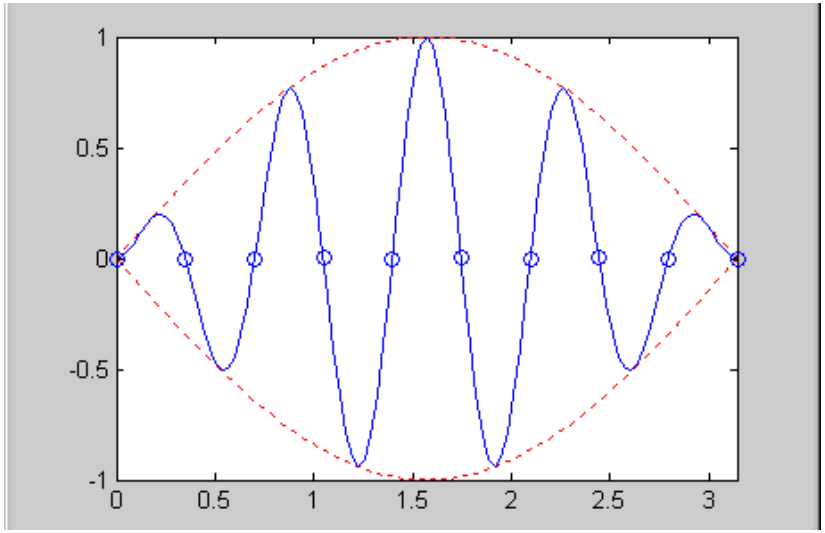
多次叠绘、双纵坐标和多子图

- 多次叠绘
  - 多次调用**plot**命令在一幅图上绘制多条曲线，需要**hold**指令的配合。
  - **hold on** 保持当前坐标轴和图形，并可以接受下一次绘制。
  - **hold off** 取消当前坐标轴和图形保持，这种状态下，调用**plot**绘制完全新的图形，不保留以前的坐标格式、曲线。

例 4 重绘曲线

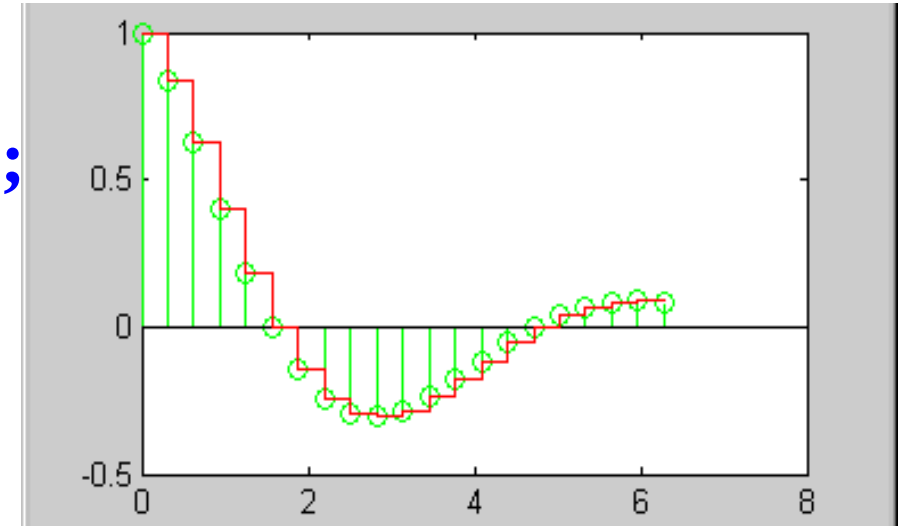
重绘波形$y = \sin(t)\sin(9t)$ 及其包络线。

```
t=(0:pi/100:pi)';  %长度为101的时间采样列向量
y1=sin(t)*[1,-1];  %包络线函数值，是（101x2）的矩阵
y2=sin(t).*sin(9*t);      %长度为101的调制波列向量
t3=pi*(0:9)/9;
y3=sin(t3).*sin(9*t3);
plot(t,y1,'r:')
hold on
plot(t,y2,'b')
plot(t3,y3,'bo')
axis([0,pi,-1,1])
hold off
```



例5 利用hold绘制离散信号的波形。

```
t=2*pi*(0:20)/20;
y=cos(t).*exp(-0.4*t);
stem(t,y,'g');
hold on;
stairs(t,y,'r');
hold off
```

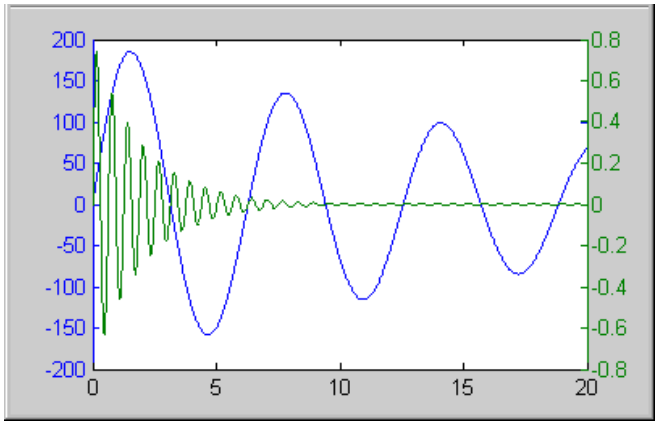


双纵坐标：plotyy指令

- plotyy指令调用格式：

plotyy(x1, y1, x2, y2)

x1-y1曲线y轴在左， x2-y2曲线y轴在右。



例6：

```
x = 0:0.01:20;
y1 = 200*exp(-0.05*x).*sin(x);
y2 = 0.8*exp(-0.5*x).*sin(10*x);
plotyy(x,y1,x,y2);
```

多子图

- MATLAB允许在同一图形窗口布置几幅独立的子图。具体指令：
  - subplot(m, n, k)

    使（mxn）幅子图中第k个子图成为当前图
  - subplot('postion', [left, bottom, width, height])

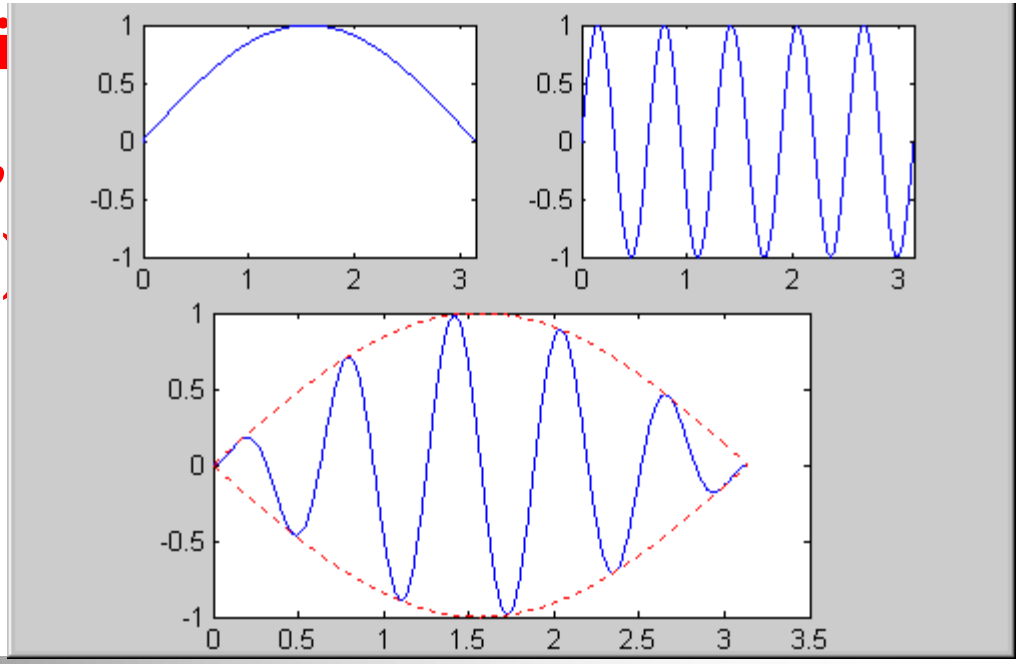
    在指定的位置上开辟子图，并成为当前图

说明：

- subplot(m, n, k)的含义：图形窗口包含（mxn）个子图，k为要指定的当前子图的编号。其编号原则： 左上方为第1子图，然后向右向下依次排序。该指令按缺省值分割子图区域。
- subplot('postion', [left, bottom, width, height])用于手工指定子图位置，指定位置的四元组采用归一化的标称单位，即认为整个图形窗口绘图区域的高、宽的取值范围都是[0， 1]，而左下角为（0,0）坐标。
- 产生的子图彼此独立。所有的绘图指令均可以在子图中使用。

例7 演示subplot指令对图形窗的分割

```
t=(pi*(0:1000)/1000)';
y1=sin(t);y2=sin(10*t);y12=sin(t).*sin(10*t);
subplot(2,2,1),plot(t,y1);axis([0,pi,-1,1])
subplot(2,2,2),plot(t,y2);axi
subplot('position',[0.2,0.05,
plot(t,y12,'b-',t,[y1,-y1],'r:'
```



绘制图形的辅助操作

曲线线形控制符

符号	-	:	-.	--
含义	实线	虚线	点划线	双划线

曲线颜色控制符

符号	b	g	r	c	m	y	k	w
含义	兰	绿	红	青	品红	黄	黑	白

曲线的色彩、线型和 数据点形

数据点型控制符

符号	含义	符号	含义
.	实心黑点	d	菱形符 diamond
+	十字符	h	六角星符 hexagram
*	八线符	o	空心圆圈
^	朝上三角符	p	五角星符 pentagram
<	朝左三角符	s	方块符 square
>	朝右三角符	x	叉字符
v	朝下三角符		

- 曲线的线形控制符、颜色控制符、数据点形控制符可以组合使用
- 其先后次序不影响绘图结果
- 也可以单独使用。

坐标、刻度和分格线控制

常用的坐标控制指令

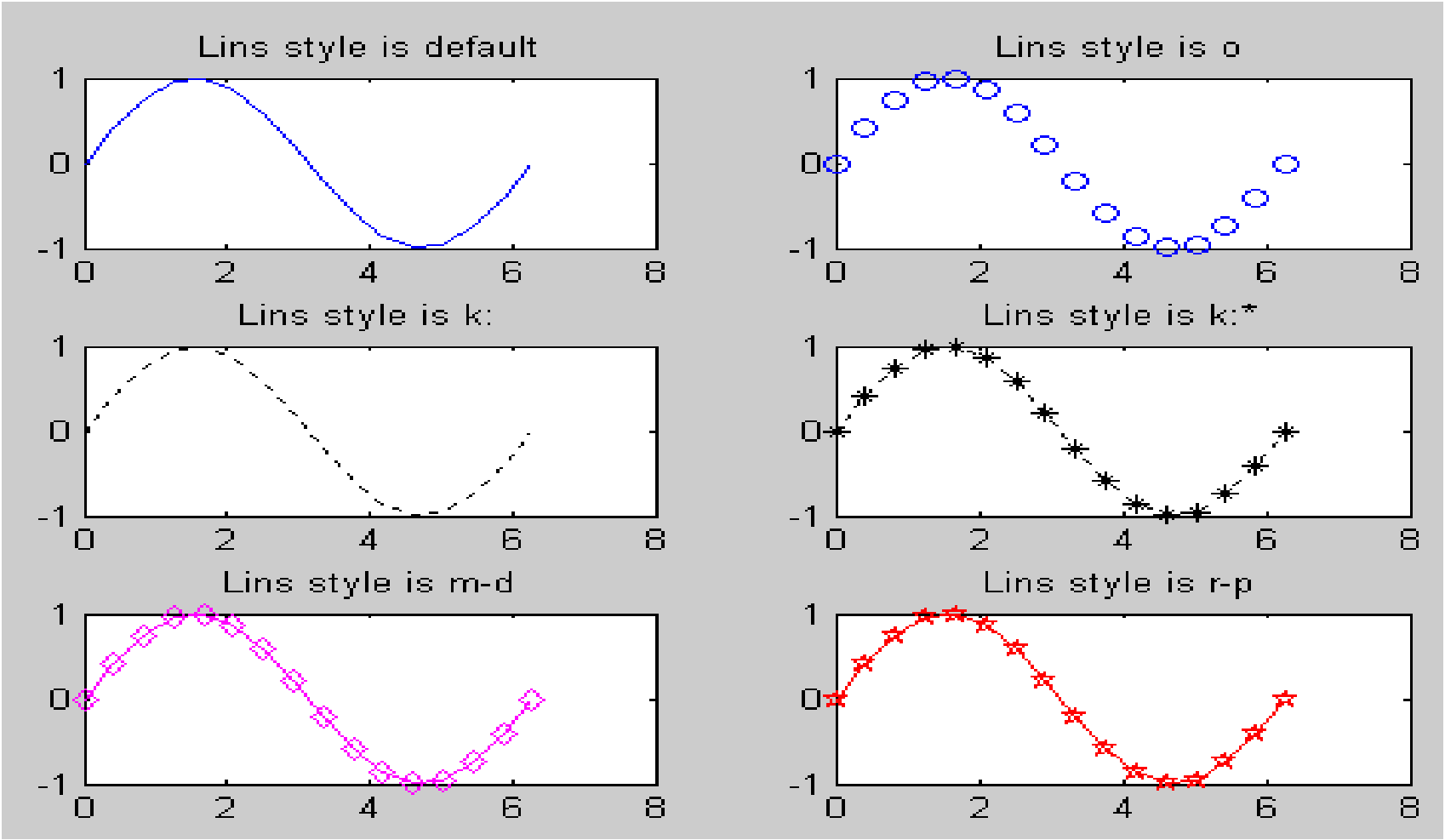
坐标轴控制方式、取向、范围		坐标轴的高宽比	
axis auto	使用缺省设置	axis equal	纵、横坐标等长刻度
axis manual	是当前坐标范围不变	axis image	纵、横坐标等长刻度,坐标框紧贴数据范围
axis off	不显示坐标轴	axis square	产生方形坐标系
axis on	显示坐标轴	axis normal	缺省矩形坐标系
axis ij	坐标原点在左上方		
axis xy	坐标原点在左下方		
axis (v) v=[x1, x2, y1, y2]	设定坐标范围		

以上只是常用的坐标控制指令，全部的坐标控制指令可使用 doc axis 命令可获得。

例8 绘制图形的辅助操作

Specify Line properties

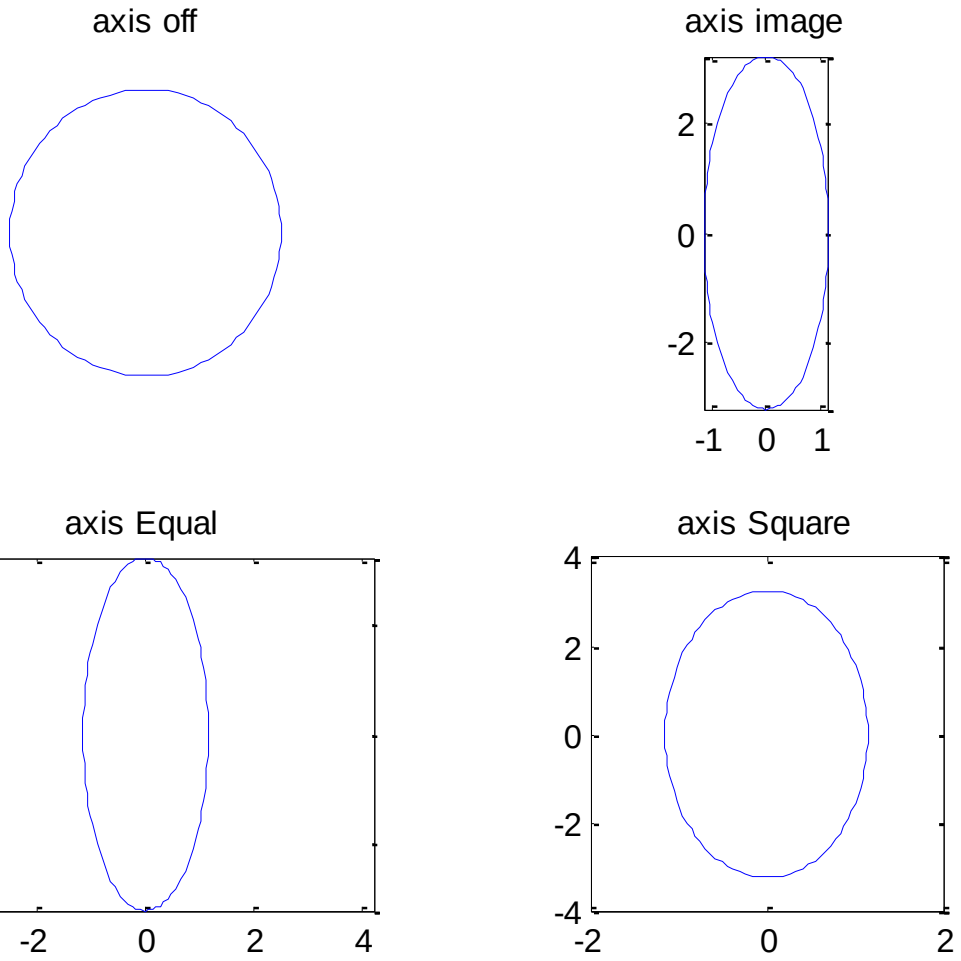
```
t=(0:15)*2*pi/15;
y=sin(t);
subplot(3,2,1), plot(t, y); title('Lins style is default')
subplot(3,2,2), plot(t, y, 'o'); title('Lins style is o')
subplot(3,2,3), plot(t, y, 'k:'); title('Lins style is k:')
subplot(3,2,4), plot(t, y, 'k:*'); title('Lins style is k:*')
subplot(3,2,5), plot(t, y, 'm-d'); title('Lins style is m-d')
subplot(3,2,6), plot(t, y, 'r-p'); title('Lins style is r-p')
```


Lins style is default
Lins style is o
Lins style is k:
Lins style is k:*
Lins style is m-d
Lins style is r-p
1
0
-1
0
2
4
6
8

例 9 坐标控制

绘制椭圆，长轴为3.25，短轴为1.15

```
t=0:2*pi/99:2*pi;
x=1.15*cos(t); y=3.25*sin(t);    % y为长轴，x为短轴
subplot(2,2,1); plot(x, y);
axis off
title('axis off');
subplot(2,2,2); plot(x,y);
axis image;
title('axis image');
subplot(2,2,3); plot(x,y);
axis equal;
title('axis Equal');
subplot(2,2,4); plot(x,y);
axis square;
title('axis Square');
```


axis off
axis image
2
0
-2
-1 0 1
axis Equal
2
0
-2
-4 -2 0 2 4
axis Square
4
2
0
-2
-4
-2 0 2

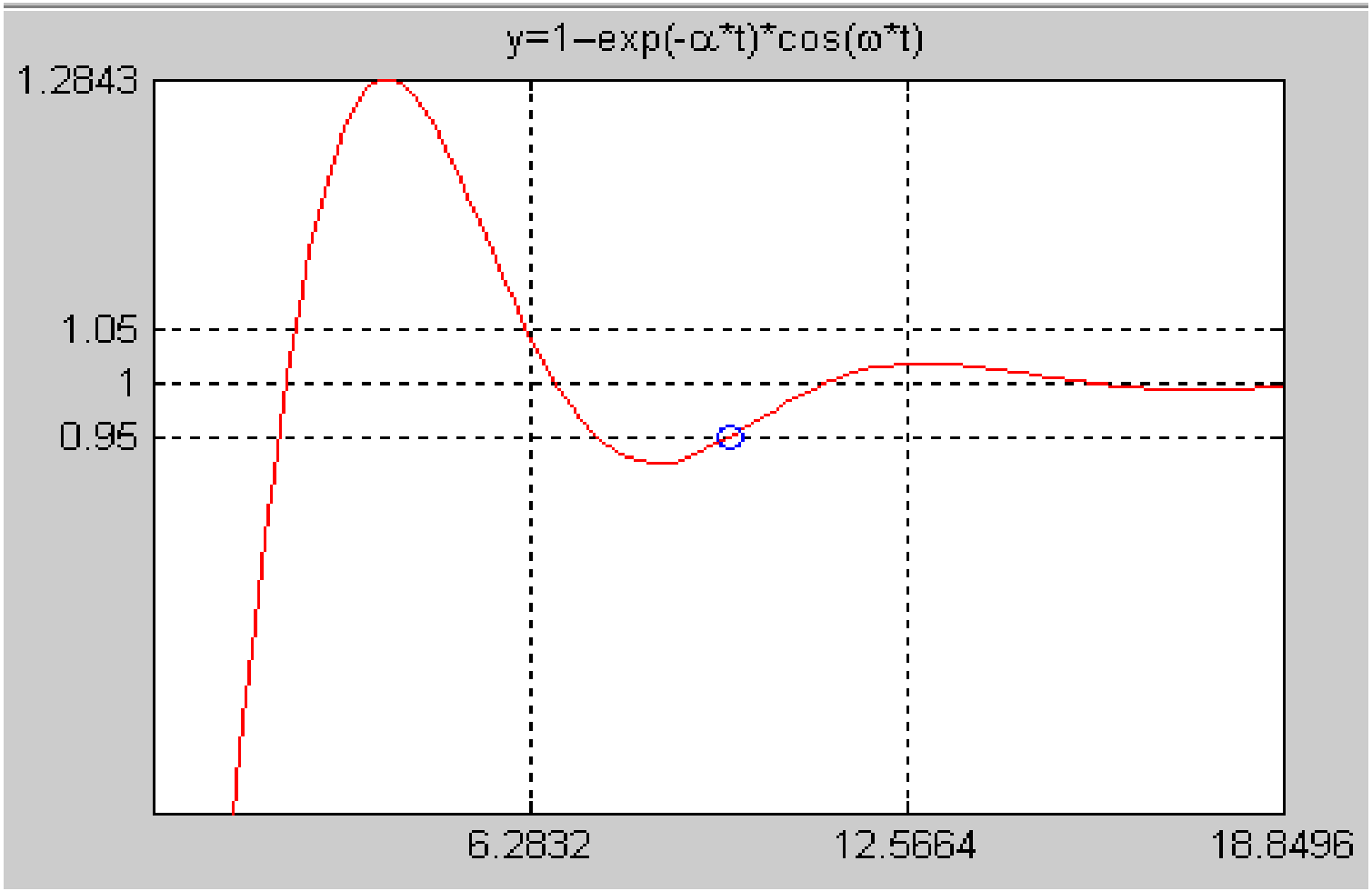
刻度、分格线和坐标框

- 分格线与grid指令
  - grid on　　画出分格线
  - grid off　　不画分格线
  - MATLAB的缺省设置是不画分格线；分格线的疏密取决于坐标刻度（改变坐标刻度，可改变分格线的疏密）。
- 坐标框
  - box on　　控制加边框线
  - box off　　控制不加边框线
- 刻度设置
  - 指令及格式：

    set(gca, 'xtick', xs, 'ytick', ys)
  - xs、ys可以使任何合法的实数向量，用于分别设置x、y轴的刻度。

例10绘制y=1-exp(-0.3*t).*cos(0.7*t)

```
t=6*pi*(0:100)/100;
y=1-exp(-0.3*t).*cos(0.7*t);
tt=t(find(abs(y-1)>0.05));
ts=max(tt);
plot(t,y,'r-');
grid on;
axis([0,6*pi,0.6,max(y)]);
title('y=1-exp(-\alpha*t)*cos(\omega*t)');
hold on;
plot(ts,0.95,'bo');
hold off;
set(gca,'xtick',[2*pi,4*pi,6*pi],'ytick',[0.95,1,1.05,max(y)]);
grid on;
```


y=1−exp(-α*t)*cos(ω*t)
1.2843
1.05
1
0.95
6.2832
12.5664
18.8496

图形标识

- 图形标识包括:
  - 图名（title）
  - 坐标轴名（xlabel、ylabel）
  - 图形文本注释（text）
  - 图例（legend）
- 简捷使用格式
  - title(s)　　% s为字符串变量或常量
  - xlabel(s)
  - ylabel(s)
  - legend(s)
  - text(x, y, s)　% 指定坐标（x, y）处加注文字

图形标识（续）

- 精细指令形式
  - 字体样式设置：

\fontname{arg} \arg \fontsize {arg} string

其中，String为要输出的字符串，其前面的均为属性控制，使用方法见下表。

	指令	arg取值	示例
指定字体	\fontname {arg}	Arial 宋体	'\fontname {'宋体'}
指定风格	\arg	Bf （黑体） It 斜体 1 Sl 斜体 2 Rm 正体	'\bf example'
指定大小	\fontsize {arg}	正整数（缺省 10P）	'\fontsize {16} Example1'

图形标识（续）

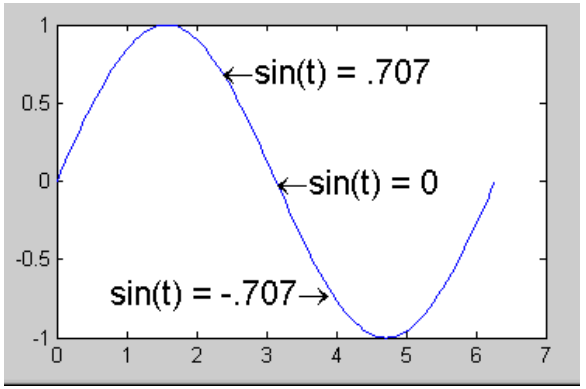
上下角标的控制

	指令	Arg取值	举例	
			示例指令	效果
上标	^{arg}	任何合法字符	'\exp^{-t}sin(t)'	$e^{-t}\sin(t)$
下标	_{arg}	任何合法字符	'U_{\alpha}'	U_{α}

希腊字母与特殊字符

Character Sequence	Symbol	Character Sequence	Symbol	Character Sequence	Symbol
\alpha	α	\upsilon	υ	\sim	~
\beta	β	\phi	ϕ	\leq	≤
\gamma	γ	\chi	χ	\infty	∞
\delta	δ	\psi	ψ	\clubsuit	♣
\epsilon	ε	\omega	ω	\diamondsuit	♦
\zeta	ζ	\Gamma	Γ	\heartsuit	♥

例11在正弦曲线上标注特殊值

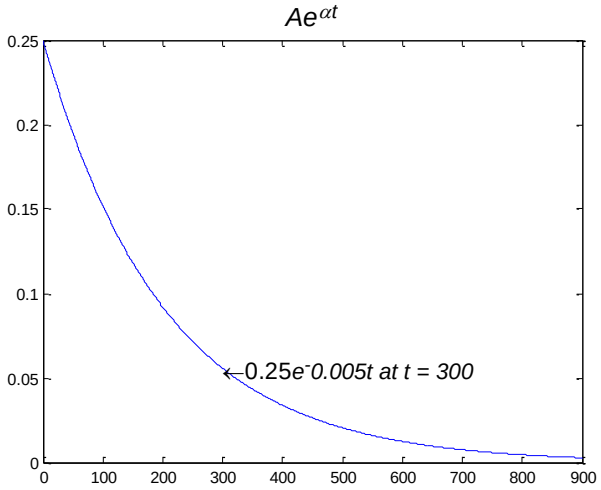



```
t=(0:100)/100*2*pi;
y=sin(t);
plot(t, y)
text(3*pi/4,sin(3*pi/4), '\fontsize{16}\leftarrowsin(t)
     = .707 ')
text(pi, sin(pi), '\fontsize{16}\leftarrowsin(t) = 0 ')
text(5*pi/4, sin(5*pi/4), '\fontsize{16}sin(t) = -
     .707\rightarrow',...
     'HorizontalAlignment','right')
```

其中，‘HorizontalAlignment’, ‘right’设置图形标识为水平右对齐

例12

```
t = 0:900;
plot(t,0.25*exp(-0.005*t))
title('\fontsize{16}\itAe^{\alphat}');
text(300,.25*exp(-0.005*300),...
    '\fontsize{14}\leftarrow0.25\ite^-0.005\itt at \itt =
      300');
```

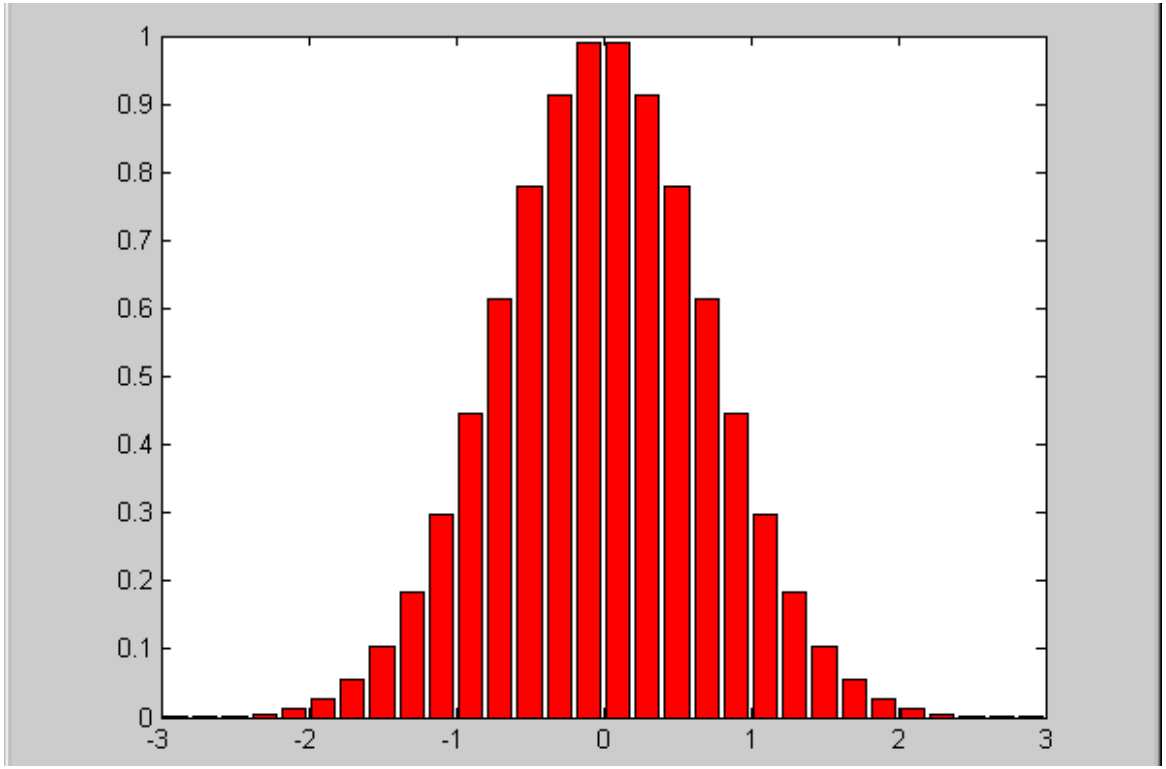


特殊图形

- 直方图(柱形图)bar
  - 垂直直方图
    - 累计式
    - 分组式
  - 水平直方图
    - 累计式
    - 分组式

例13

```
x = -2.9:0.2:2.9;
bar(x,exp(-x.*x),'r')
```



例14

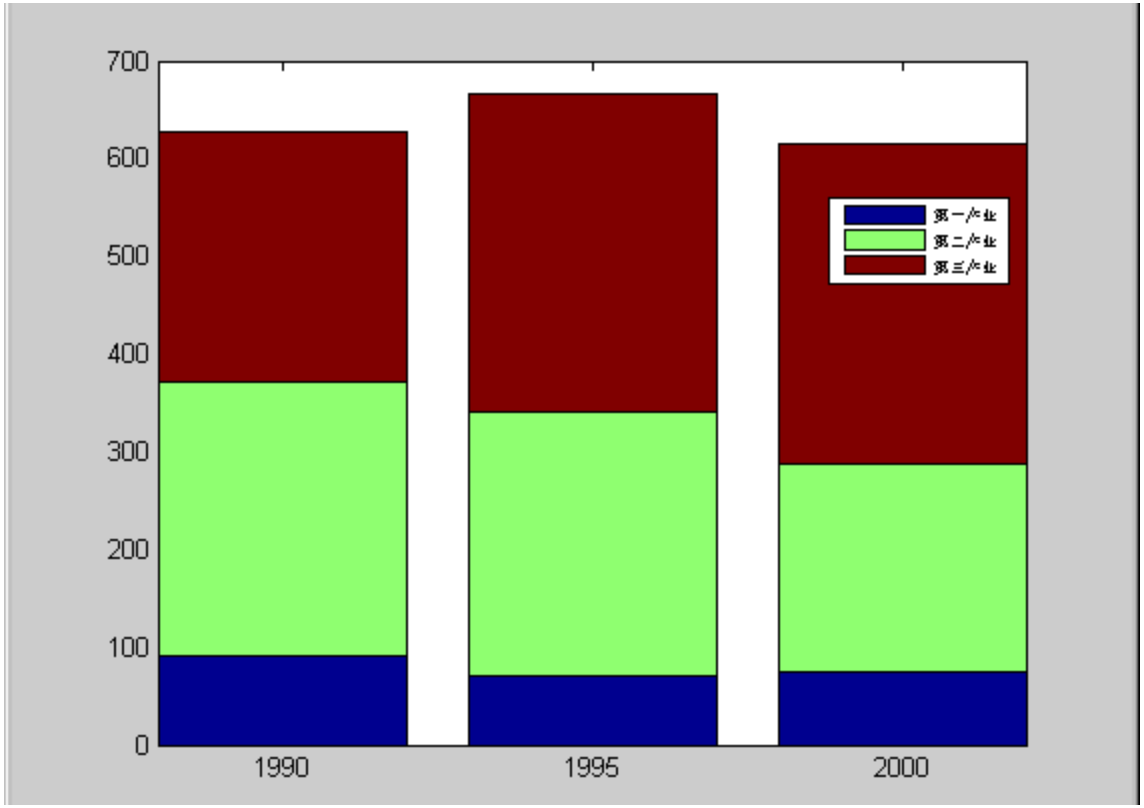
北京市从业人员统计

	1990年	1995年	2000年
第一产业	90.7	70.6	73.9（万人）
第二产业	281.6	271	214.6
第三产业	254.8	323.7	326.5

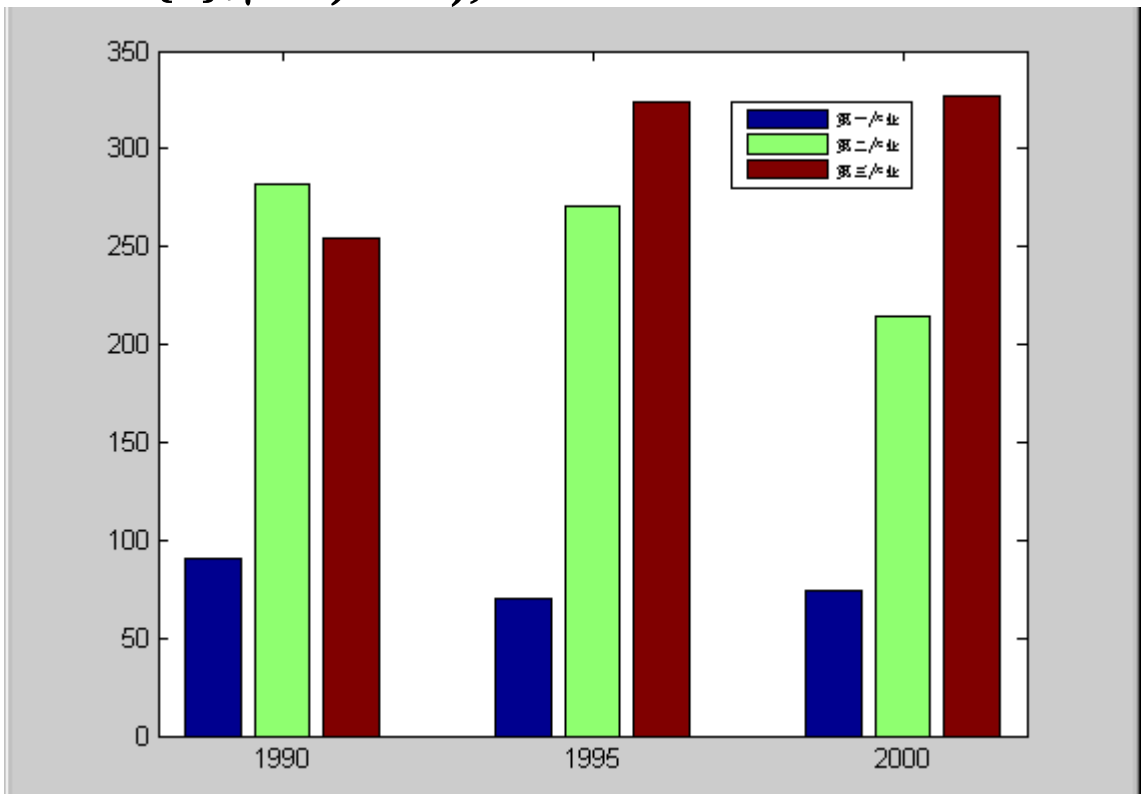
执行以下语句：

```
year=[1990 1995 2000];
people=[90.7 281.6 254.8; 70.6 271 323.7; 73.9 214.6 326.5];
bar(year, people, 'stack');  %累计式直方图
legend('\fontsize{6}第一产业', '\fontsize{6}第二产业', '\fontsize{6}第三
    产业');
```

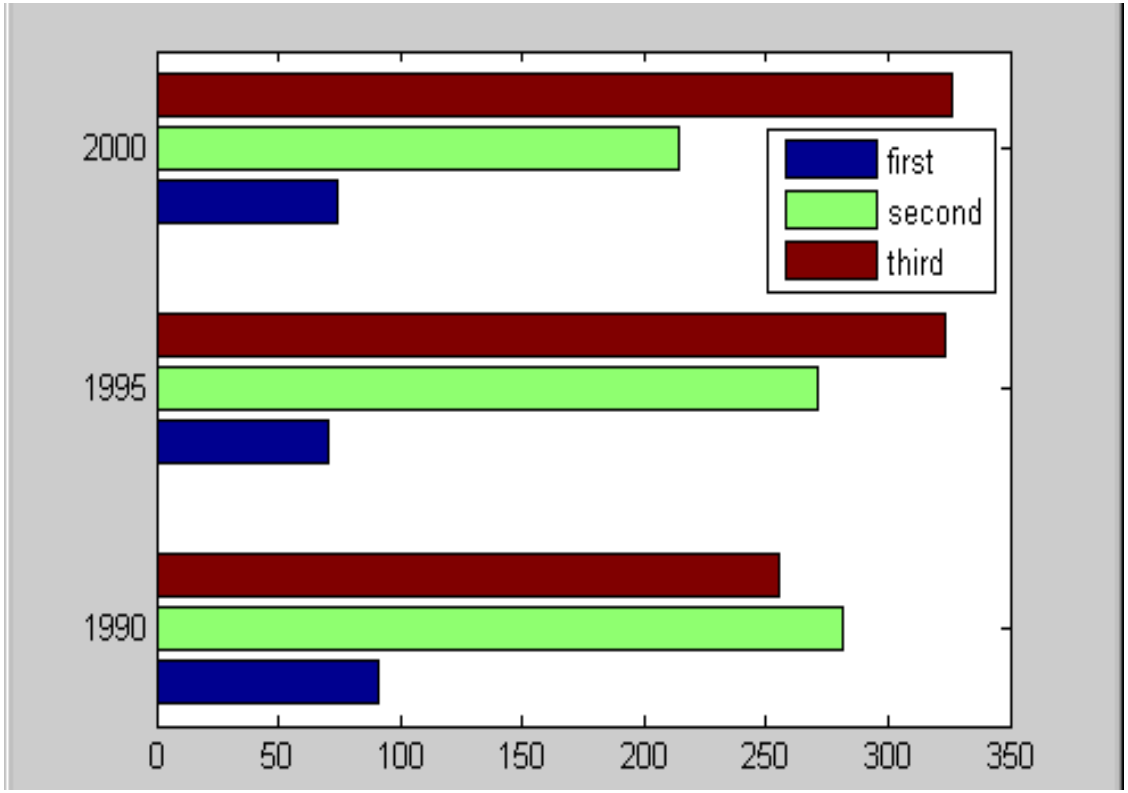
累计式直方图stack



```
bar(year, people, 'group');   % 分组式直方图
legend('\fontsize{6}第一产业','\fontsize{6}第二产业',
       '\fontsize{6}第三产业');
```

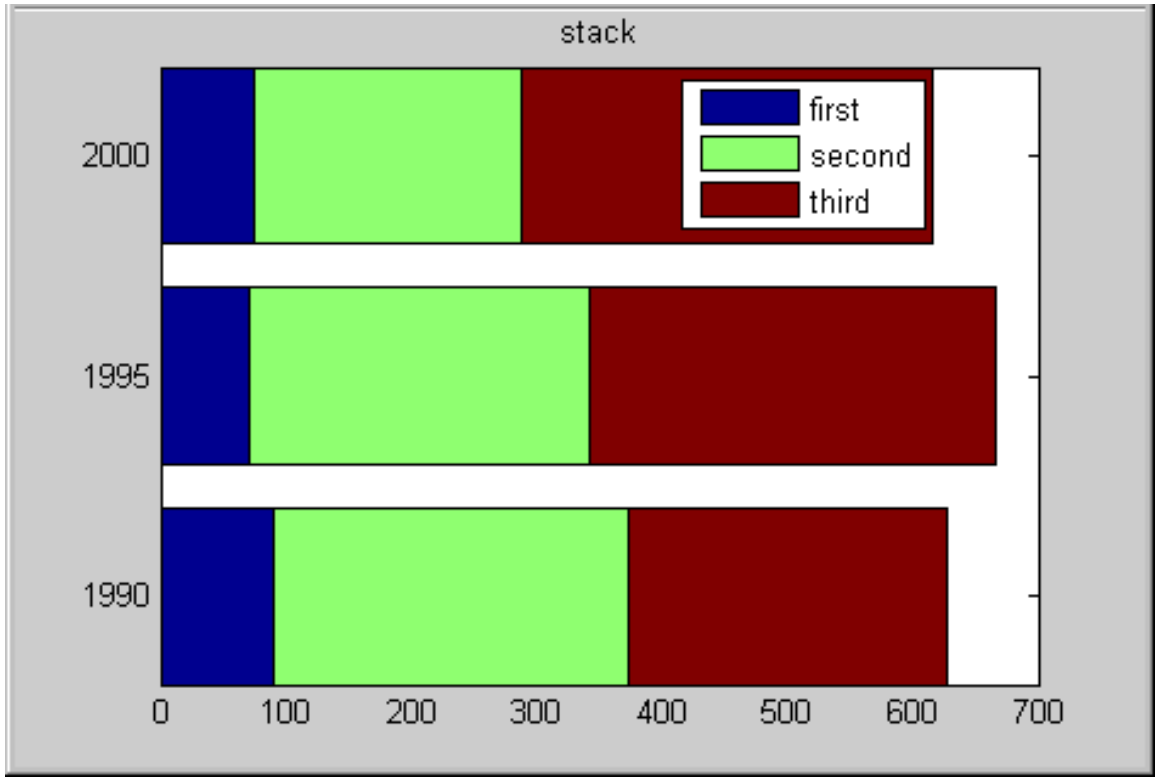


```
barh(year, people, ‘group’);    % 分组式直方图
legend(‘\fontsize{6}first’, ‘\fontsize{6}second’, ‘\fontsize{6}third');
```



barh(year, people, ‘stack’);　% 累积式直方图

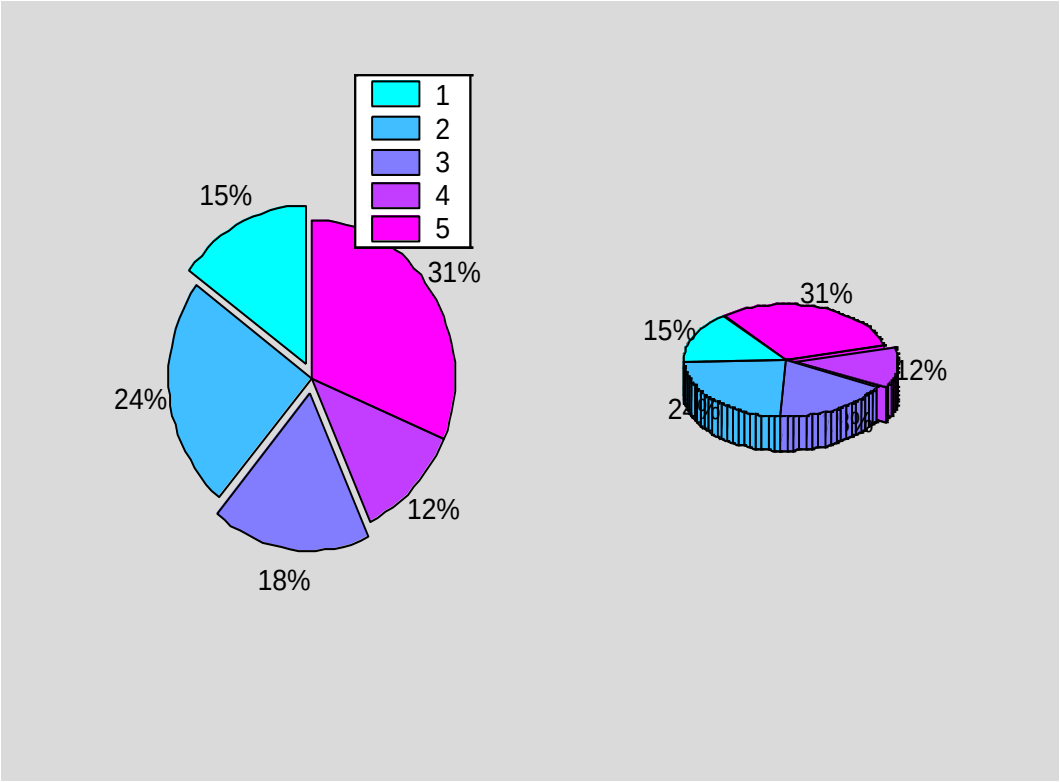
legend(‘\fontsize{6} first’, ‘\fontsize{6}second’, ‘\fontsize{6}third');



饼图指令pie

- 饼图指令pie 用来表示各元素占总和的百分数。该指令第二输入变量是与第一变量同长的0-1向量，1使对应扇块突出。

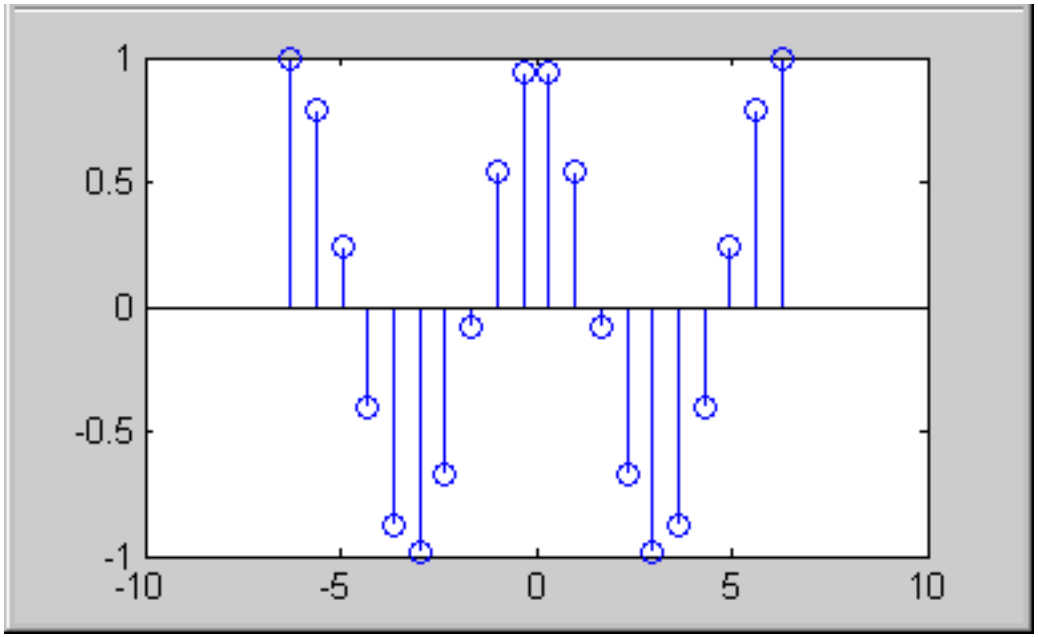
```
a=[1,1.6,1.2,0.8,2.1];
subplot(1,2,1),pie(a,[1 0 1 0 0]),
legend({'1','2','3','4','5'})
subplot(1,2,2), b=int8(a==min(a))
pie3(a,b)
colormap(cool)
```


1
2
3
4
5
15%
31%
24%
12%
18%
31%
15%
12%

离散杆图stem

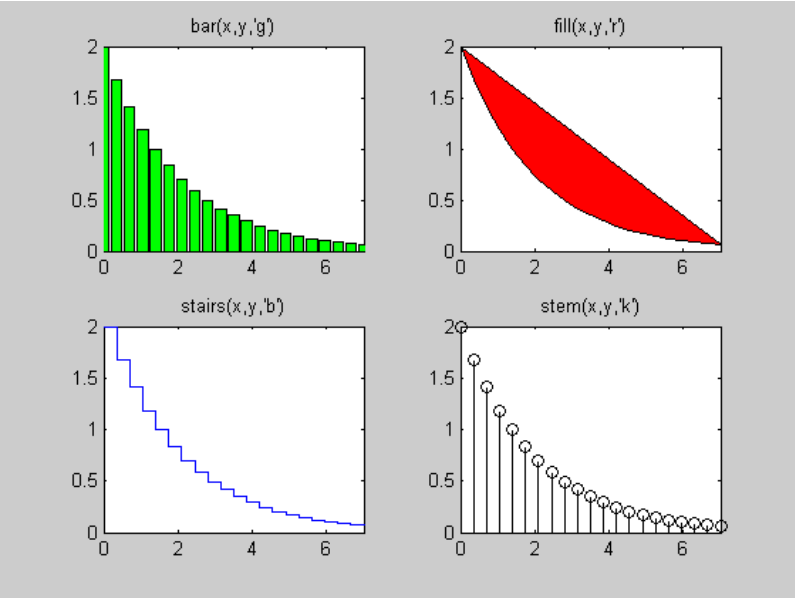
余弦波的采样信号图

```
t = linspace(-2*pi,2*pi,20);
h = stem(t,cos(t));
```



例15分别以条形图、填充图、阶梯图和杆图形式绘图

```
x = 0:0.35:7;
y = 2*exp(-0.5*x);
subplot(221);bar(x,y,'g');
title('bar(x,y,''g'')');axis([0,7,0,2]);
subplot(222);fill(x,y,'r');
title('fill(x,y,''r'')');axis([0,7,0,2]);
subplot(223);stairs(x,y,'b');
title('stairs(x,y,''b'')');axis([0,7,0,2]);
subplot(224);stem(x,y,'k');
title('stem(x,y,''k'')');axis([0,7,0,2]);
```



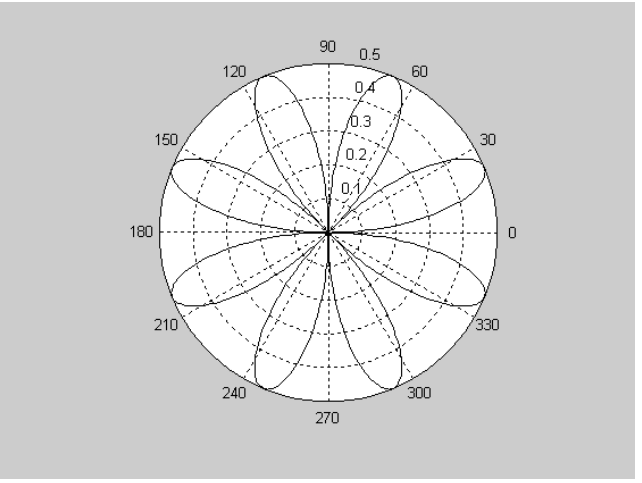
例16 极坐标图

polar函数用来绘制极坐标图，其调用格式为：

polar(theta,rho,选项)

例：绘制ρ=sin(2θ)cos(2θ)的图形

```
theta = 0:0.01:2*pi;
rho = sin(2*theta).*cos(2*theta);
polar(theta,rho,'k');
```

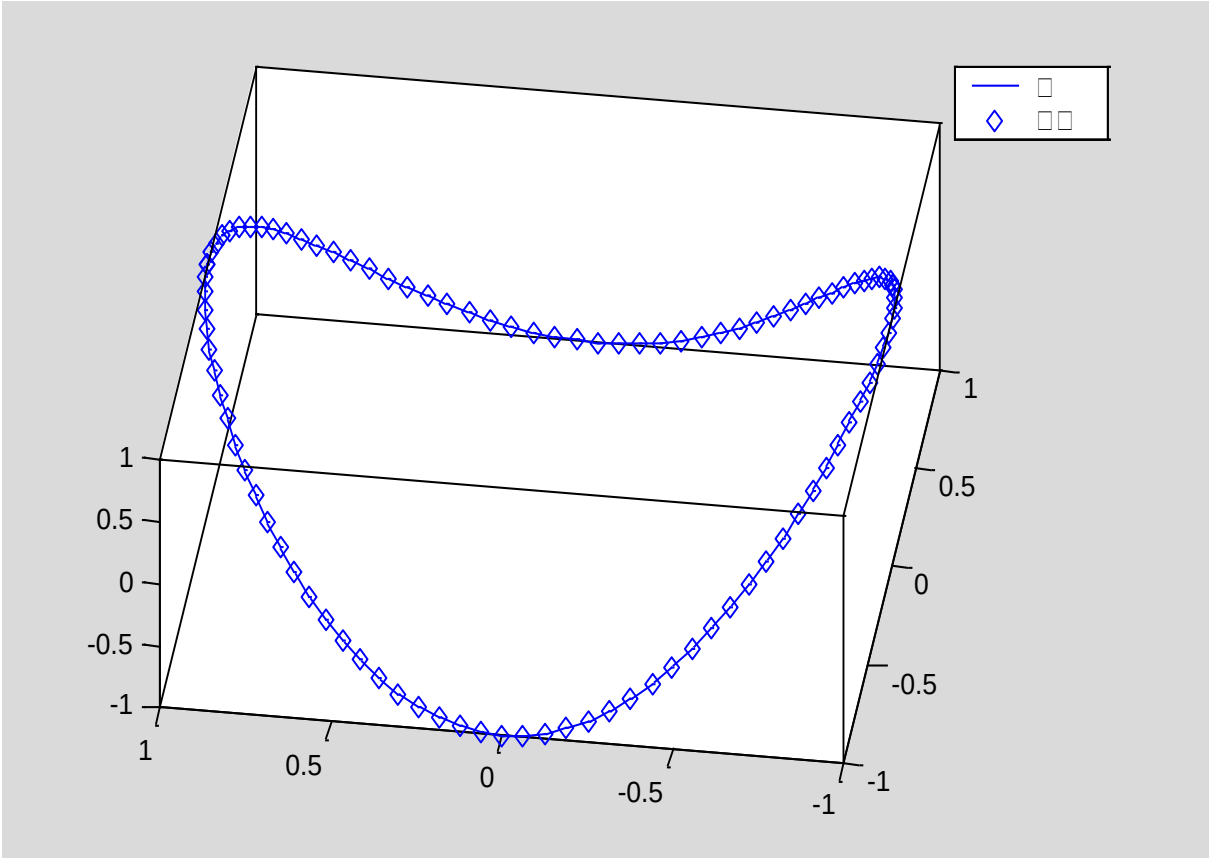


三维绘图的基本操作

- 三维线图指令plot3
  - 三维绘图指令中，plot3最易于理解，它的使用格式与plot十分相似，只是对应第3 维空间的参量。

```
t=(0:0.02:2)*pi;
x=sin(t);
y=cos(t);
z=cos(2*t);
plot3(x,y,z,'b-',x,y,z,'bd');
view([-82,58]);
box on
legend('链','宝石')
```

三维线图绘制结果



三维网线图（mesh）和曲面图（surf）

画函数z=f(x,y)所代表的三维空间曲面，需要做以下的数据准备工作：

- 确定自变量的取值范围和取值间隔。
  - x=x1:dx:x2;
  - y=y1:dy:y2;
- 构成x-y平面上的自变量采样“格点”矩阵。
  - 利用MATLAB指令meshgrid产生“格点”矩阵
  - [xa, ya]=meshgrid(x,y) ;
- 计算函数在自变量采样“格点”上的函数值，即z=f(x,y)。

- 网线图、曲面图绘制。

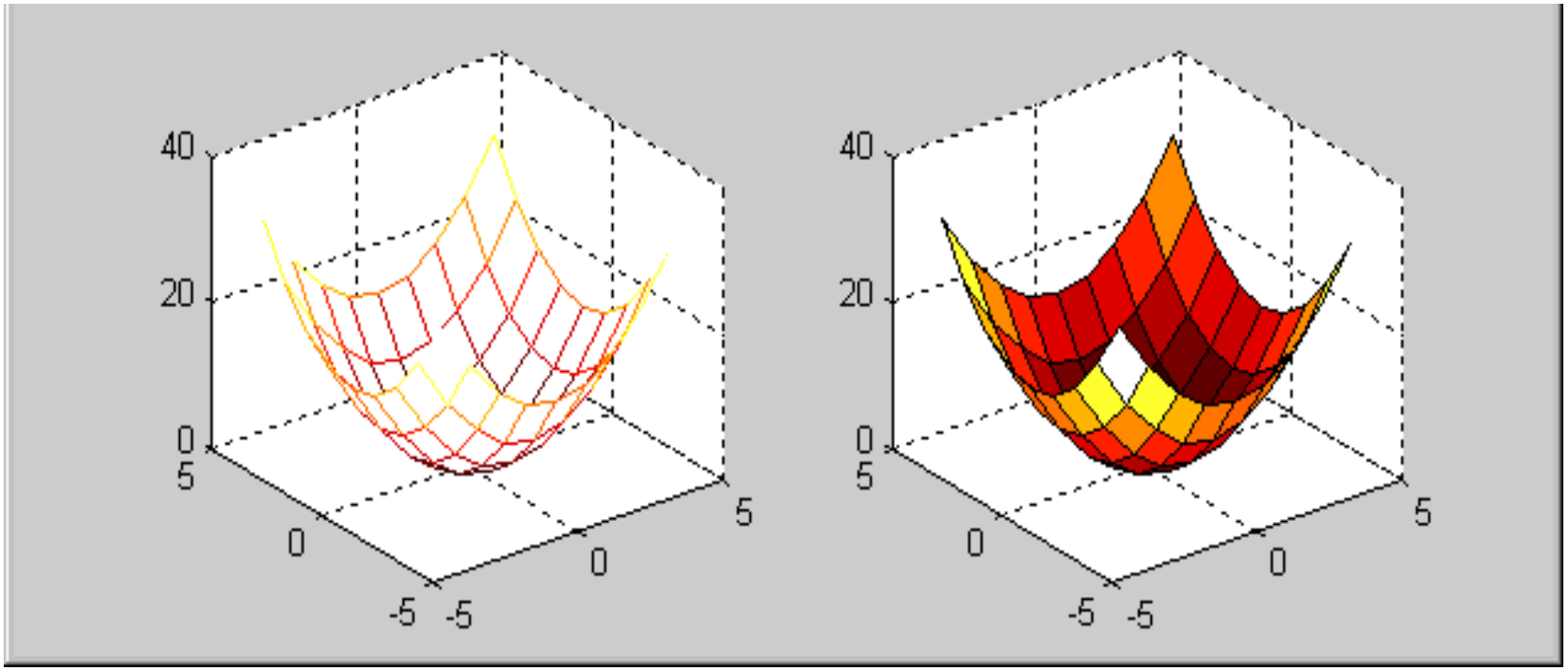
例17

绘制函数z=x^2+y^2的曲面

```
x=-4:4;y=x;
[x,y]=meshgrid(x,y);          %生成 x-y 坐标“格点”矩阵
z=x.^2+y.^2;                  %计算格点上的函数值
subplot(1,2,1), mesh(x,y,z);  %三维网格图

subplot(1,2,2), surf(x,y,z);  %三维曲面图
colormap(hot);
```

函数z=x^2+y^2的曲面的绘制结果



图像文件的读写与图像显示

- **imread指令**

  读取图像文件（BMP, GIF , PNG, JPEG, and TIFF）

- **imshow指令**

  显示图像

- **imwrite指令**

  保存图像

- **例18：读取图像文件**

```
img1=imread(‘mudan.jpg');    % Load image data
img2=imread('eight.tif');
whos  img1 img2
```

Name	Size	Bytes	Class
img1	750x553x3	1244250	uint8 array
img2	242x308	74536	uint8 array

显示图像:

```
imshow(img1); % Display image
```

简单图像处理

```
lighter = 2 * img1;

subplot(1,2,1);
imshow(img1);
title('Original');  % Display image

subplot(1,2,2);
imshow(lighter);
title('Lighter');    % Display image
```

图像处理前后的比较

Original

Lighter

- 保存图像

```
>> imwrite(lighter, 'mysaved.jpg')
```

- 查看保存结果

```
>> dir mysaved.*
mysaved.jpg
```

彩色图像转换为灰度图像

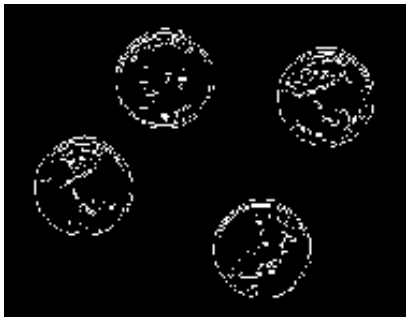
```
black = rgb2gray(img1);
imshow(black)
```

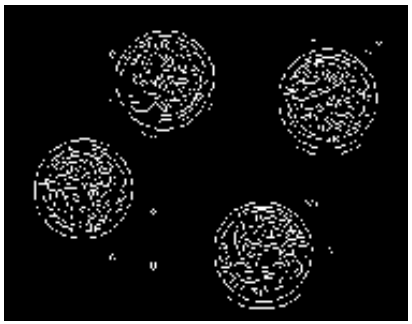
图像的缩放

```
zoom on
```

图像特征提取

```
imag_edge1 = edge(img2,‘sobel’);        %sobel边缘提取算法
subplot(121),imshow(imag_edge1)
imag_edge2 = edge(img2,'canny');        %canny边缘提取算法
subplot(122),imshow(imag_edge2)
```





Matlab命令的执行方式

- 交互式命令执行方式（命令窗口）

  逐条输入，逐条执行，操作简单、直观，但速度慢，执行过程不能保留。

- M文件的程序执行方式

  将命令编成程序存储在一个文件中（M文件），依次运行文件中的命令，可以重复进行。

- Matlab程序设计有传统高级语言的特征，又有自己独特的特点，可以利用数据结构的特点，使程序结构简单，编程效率高。

M文件的分类

- 用Matlab语言编写的程序，称为M文件。

  是由若干Matlab命令组合在一起构成的，
  它可以完成某些操作，也可以实现某种算法。

- M文件根据调用方式的不同分为两类：

  命令文件（Script File）

  函数文件（Function File）

- 它们的扩展名都是.m

命令文件和函数文件的区别

- 命令文件没有输入参数，也不返回输出参数；函数文件可以带输入参数，也可以返回输出参数。
- 命令文件对工作空间中的变量进行操作，文件中所有命令的执行结果也返回工作空间中；函数文件中定义的变量为局部变量，当函数文件执行完毕时，这些变量也被清除。
- 命令文件可以直接运行；函数文件不能直接运行，要以函数调用的方式来调用它。

例1 建立文件将变量a、b的值互换

命令文件：

```
clear;
a = 1:10;
b = [11,12,13,14;15,16,17,18];
c = a; a = b; b = c;
a
b
```

将文件保存为exch，并在命令窗口执行。

执行结果：

```
a =
   11   12   13   14
   15   16   17   18
b =
    1    2    3    4    5    6    7    8    9   10
```

函数文件

fexch.m

```
function [a,b] = exch(a,b)
c = a; a = b; b = c;
```

然后在命令窗口调用该函数文件：

```
clear;
x = 1:10;
y = [11,12,13,14;15,16,17,18];
[x,y] = fexch(x,y)
```

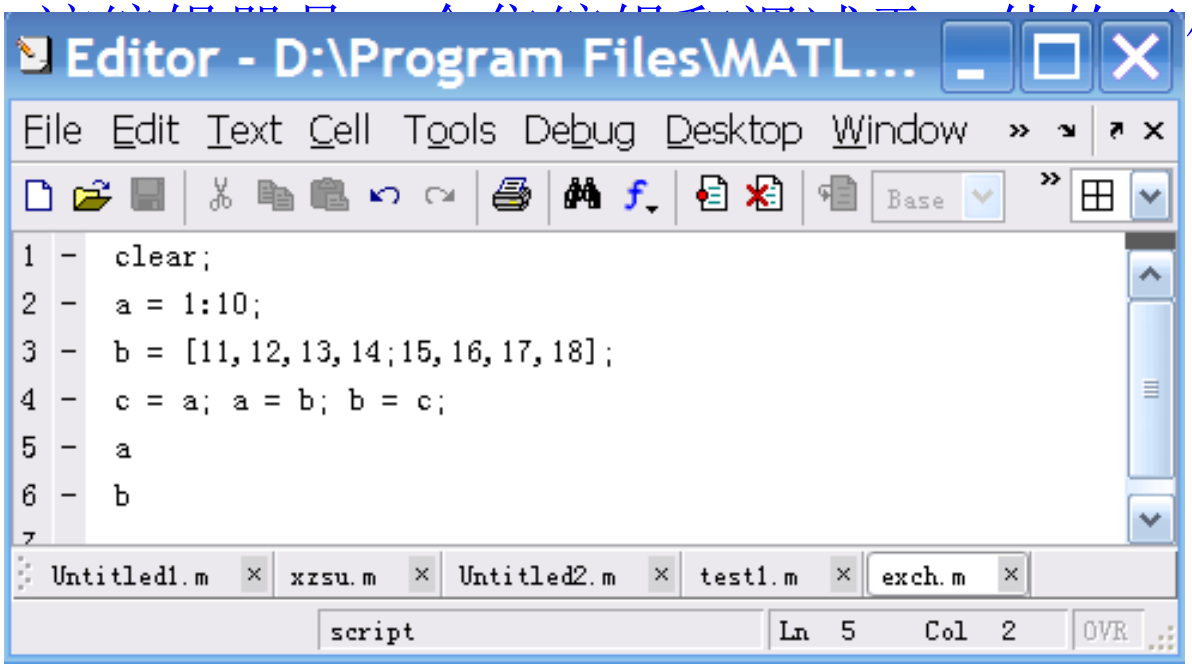
输出结果为：

```
a =
    11    12    13    14
    15    16    17    18
b =
     1     2     3     4     5     6     7     8     9    10
```

函数参数a，b，c未保留在工作空间中，x，y保留在工作空间中。

M文件的建立与打开

M文件是一个文本文件，可以用任何编辑程序来建立和编辑，一般最常用的是使用Matlab提供的文本编辑器。



作环境。

程序控制结构

- 顺序结构
- 选择结构
- 循环结构

任何复杂的程序都可以由这3种基本结构构成。

顺序结构

- 顺序结构是指按照程序中语句的排列顺序依次执行，直到程序的最后一个语句。（最简单的一种程序）

1、数据的输入

从键盘输入数据，则可以使用input函数来进行，调用格式为：

A = input（提示信息，选项）；

其中提示信息为一个字符串，用于提示用户输入数据。

顺序结构

- 例如：从键盘输入A矩阵，可以采用下面的命令来完成

  A = input（‘输入A矩阵’）；

  如果在input函数调用时采用’s’选项，则允许用户输入一个字符串。

  例：xm = input(‘What”s your name?’,’s’);

顺序结构

2、数据的输出

命令窗口输出函数主要有disp函数，其调用格式为：

disp(输出项）

其中输出项既可以为字符串，也可以为矩阵。

顺序结构

例如：

```
A = 'Hello, Tom';
disp(A)
```

输出为：Hello, Tom

又如：

```
A = [1,2,3;4,5,6;7,8,9];
disp(A)
```

输出为：

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

%disp函数输出格式更紧凑

顺序结构

例 求一元二次方程 $ax^2+bx+c=0$ 的根。

由于Matlab能进行复数运算，所以不需要判断方程的判别式，可直接根据求根公式求根。

程序如下：

```
a = input('a=?');
b = input('b=?');
c = input('c=?');
d = b*b-4*a*c;
x = [(-b+sqrt(d))/(2*a),(-b-sqrt(d))/(2*a)];
disp(['x1=',num2str(x(1)),',x2=',num2str(x(2))]);
```

程序输出为：

```
a=?4
b=?78
c=?54
x1=-0.7188,x2=-18.7812
```

顺序结构

3、程序的暂停

程序执行过程中暂停，可用pause函数，其调用格式为：

pause（延迟描述）

如果省略延迟时间，直接使用pause，则将暂停程序，直到

用户按任一键后程序继续执行。

若要强行中止程序的运行可按Ctrl+C键。

选择结构

选择结构是根据给定的条件成立或不成立，分别执行不同的语句。

Matlab用于实现选择结构的语句有if语句，switch语句和try语句。

1. if语句

在Matlab中，if语句有3种格式。

(1)单分支if语句

语句格式：

```
if 条件
    语句组
end
```

选择结构

例如：当x是整数矩阵时，输出x的值

```
if fix(x)==x
    disp(x);
end
```

选择结构

(2)双分支if语句

语句格式:

```
if 条件
    语句组 1
else
    语句组  2
end
```

当条件成立时，执行语句组1，否则执行语句组2，然后再执行if语句的后续语句。

选择结构

例 计算分段函数：

$$y=\begin{cases}\cos(x+1)+\sqrt{x^2+1}, x=10\\ x\sqrt{x+\sqrt{x}}, x\neq 10\end{cases}$$

程序如下：

```
x = input('请输入x的值：  ');
if x == 10
   y = cos(x+1)+sqrt(x*x+1);
else
   y = x*sqrt(x+sqrt(x));
end
y
```

也可以用单分支if语句来实现：

```
x = input('请输入x的值：  ');
y = cos(x+1)+sqrt(x*x+1);
if x~=10
    y = x*sqrt(x+sqrt(x));
end
y
```

选择结构

(3)多分支if语句

语句格式：

```
if 条件1
    语句组 1
elseif  条件2
    语句组  2
…
elseif  条件m
    语句组  m
else
    语句组n
end
```

大小写字母的置换

输入一个字符，若为大写字母，则输出其对应的小写字母；若为小写字母，则输出其对应的大写字母；若为数字字符则输出其对应的数值，若为其他字符则原样输出。

程序如下：

```
c = input('请输入一个字符'， 's');
if c >='A' & c<='Z'
    disp(setstr(abs(c) + abs('a')-abs('A')));
elseif c>='a' & c<='z'
    disp(setstr(abs(c)- abs('a') + abs('A')));
elseif c>='0' & c<='9'
    disp(abs(c)-abs('0'));
else
    disp(c);
end
```

setstr函数可以得到ASCII码

选择结构

2、switch语句

switch语句根据表达式的取值不同，分别执行不同的语句，其语句格式：

```
switch 表达式
case 表达式1
        语句组1
case 表达式2
        语句组2
…
case 表达式m
        语句组m
otherwise
        语句组 n
end
```

switch子句后面的表达式应为一个标量或一个字符串；case子句后面的表达式不仅可以为一个标量或一个字符串，还可以为一个元胞矩阵。

例

某商场对顾客所购买的商品实行打折销售，标准如下：

price<200	没有折扣
200<=price<500	3%折扣
500<=price<1000	5%折扣
1000<=price<2500	8%折扣
2500<=price<5000	10%折扣
5000<=price	14%折扣

输入所售商品的价格，求其实际销售价格。

请同学们试着编程实现。

程序如下

```
price = input('请输入商品价格');
switch fix（price/100)
    case{0,1}                    %价格小于200
        rate = 0;
    case{2,3,4}
        rate = 3/100;            %价格大于等于200但小于500
    case num2cell(5:9)
        rate = 5/100;            %价格大于等于500但小于1000
     case num2cell(10:24)
        rate = 8/100;            %价格大于等于1000但小于2500
     case num2cell(25:49)
        rate = 10/100;           %价格大于等于2500但小于5000
      otherwise
         rate = 14/100;          %价格大于等于5000
end
price = price*(1-rate)       %输出商品实际销售价格
```

num2cell函数是将数值矩阵转化为单元矩阵。

选择结构

3. try语句

try语句是一种试探性执行语句，其语句格式为：

```
try
    语句组1
catch
    语句组2
end
```

try语句先试探性执行语句组1，如果在执行过程中出现错误，则将

错误信息赋给保留的lasterr变量，并转去执行语句组2.

例

矩阵乘法运算要求两矩阵的维数相容，否则会出错。先求两矩阵的

乘积，若出错则自动转去求两矩阵的点乘。

程序如下：

```
A = [1,2,3;4,5,6];
B = [7,8,9;10,11,12];
try
    C = A*B;
catch
    C = A.*B;
end
C
lasterr        %显示出错原因
```

```
C =
     7    16    27
    40    55    72
ans =
Error using ==> mtimes
Inner matrix dimensions must agree.
```

循环结构

循环是指按照给定的条件，重复执行指定的语句，Matlab提供了两种实现循环结构的语句：for语句和while语句。

1、for语句

for语句的格式为：

```
for 循环变量 =表达式1：表达式2：表达式3
    循环体语句
end
```

其中表达式1的值为循环变量的初值，表达式2的值为步长，表达式3的值为循环变量的终值。步长为1时，表达式2可以省略。

例

已知 $y=\frac{1}{1^2}+\frac{1}{2^2}+\frac{1}{3^2}+\cdots+\frac{1}{n^2}$ ，当n=100时，求y的值。

程序如下：

```
y = 0;n = 100;
for i=1:n
    y = y+1/i/i;
end
y
```

输出结果为：

```
y =
    1.6350
```

利用Matlab的特点，常用向量运算来代替

循环操作，程序可以如下：

```
n = 100;
i = 1:n;
f = 1./i.^2;
y = sum(f)
```

循环结构

2、while语句

while语句的一般格式为：

```
while条件
    循环体语句
end
```

其执行过程为：

若条件成立，则执行循环体语句，执行后再判断条件是否成立，

如果不成立则跳出循环。

例

从键盘输入若干个数，当输入0时结束输入，求这些数的平均值和它们的和。

```
sum = 0;
n = 0;
x = input('Enter a number(end in 0):');
while(x~=0)
    sum = sum+x;
    n = n+1;
    x = input('Enter a number(end in 0):');
end
if(n>0)
    sum
    mean = sum/n
end
```

输出结果为：

```
Enter a number(end in 0):67
Enter a number(end in 0):89
Enter a number(end in 0):93
Enter a number(end in 0):70
Enter a number(end in 0):0
sum =
  319
mean =
  79.7500
```

循环结构

3、break语句和continur语句

它们一般与if语句配合使用。

break语句用于终止循环的执行。

当在循环体内执行到该语句时，程序将跳出循环，继续执行循环语句的

下一语句。

continue语句控制跳过循环体中的某些语句。

当在循环体内执行到该语句时，程序将跳过循环体中所有剩下的语句，

继续下一次循环。

例

求[100,200]之间第一个能被21整除的整数

程序如下：

```
for n = 100:200
    if rem(n,21)~=0;
       continue
    end
    break
end
n
```

程序输出结果为：

```
n =
     105
```

函数文件

函数文件是另一种形式的M文件，每一个函数文件都定义一个函数。

Matlab提供的标准函数大部分是由函数文件定义的。

函数文件的基本结构

函数文件由function语句引导，其基本结构为：

function 输出形参表 = 函数名（输入形参表）

注释说明部分

函数体语句

其中，以function开头的一行为引导行，表示该M文件是一个函数文件。

当输出形参多于一个时，应该用方括号括起来。

说明：

1. 关于函数文件名

函数文件名通常由函数名再加上扩展名.m组成。

当函数文件名与函数名不同时，Matlab将忽略函数名而确认文件名，因此调用时使用函数文件名。

2. 关于注释说明部分

注释说明包括3部分：

① 紧随引导行之后以%开头的第一注释行。

这一行一般包括大写的函数文件名和函数功能简要描述，供lookfor 关键词查询和help在线帮助时使用。

② 第一注释行及之后连续的注释行。

通常包括函数输入/输出参数的含义及调用格式说明等信息，构成全部在线帮助文本。

说明：

③ 与在线帮助文本相隔一空行的注释行。

包括函数文件编写和修改的信息，如作者和版本等。

3、关于return语句

如果在函数文件中插入了return语句，则执行到该语句就结束函数的执行，流程转至调用该函数的位置。通常也不使用return语句。

例5.10 编写函数文件，求半径为r的圆的面积和周长。

函数文件如下：

```
function [s,p] = fcircle(r)
% FCIRCLE calculate the area and perimeter of a circle of radii r
% r       圆半径
% s       圆面积
% p       圆周长

%2006年2月30日编
s = pi*r*r;
p = 2*pi*r;
```

说明：

将以上函数文件以文件名fcircle.m保存，然后在命令窗口调用。

```
[s,p] = fcircle(10)
```

输出结果是：

```
 s =
     314.1593
 p =
     62.8319
```

采用help命令或lookfor命令可以显示出注释说明部分的内容。

```
 help fcircle
```

屏幕显示

```
 FCIRCLE calculate the area and perimeter of a circle of radii r
  r       圆半径
  s       圆面积
  p       圆周长
```

函数调用

函数调用的一般格式是：

[输出实参表] = 函数名(输入实参表）

注意：函数调用时，各实参出现的顺序、个数，应与函数定义时相同。

例5.11 利用函数文件，实现直角坐标(x,y)与极坐标(ρ,θ)之间的转换。

函数文件：tran.m:

```
function [rho,theta] = tran(x,y)
rho = sqrt(x*x+y*y);
theta = atan(y/x);
```

调用tran.m的命令文件main1.m:

```
x = input('please input x=:');
y = input('please input y=:');
[rho,the] = tran(x,y);
rho
the
```

函数的嵌套调用

在Matlab中，函数可以嵌套调用，即一个函数可以调用别的函数。

一个函数调用自身称为函数的递归调用。

例5.12 利用函数的递归调用，求n！。

n！本身就是以递归的形式定义的：

$$n! = \begin{cases} 1, & n \le 1 \\ n(n-1)!, & n > 1 \end{cases}$$

显然，求n！需要求(n-1)!，这时可采用递归调用。

函数如下：

```
function f = factor(n)
if n<=1
   f = 1;
else
   f = factor(n-1)*n;  %递归调用求(n-1)!
end
```

函数的嵌套调用

在命令文件中调用该函数文件，求 s = 1!+2!+3!+4!+5!。

```
 s = 0;
for i = 1:5
      s = s + factor(i);
end
s
```

在命令窗口运行命令文件，结果如下：

```
 s =
      153
```

函数参数的可调性

Matlab在函数调用上有一个与一般高级语言不同之处：

函数所传递参数数目的可调性，即参数的数量可以改变。

在调用函数时，Matlab用两个预定义变量nargin和nargout分别记录调用该函数时的输入实参和输出实参的个数。

例5.13 nargin用法示例

函数文件examp.m:

```
function fout = charray(a,b,c)
if nargin == 1
    fout = a;end
if nargin == 2
    fout = a+b;end
if nargin == 3
    fout = (a*b*c)/2;
end
```

命令文件：

```
x = [1:3];
y = [1;2;3];
examp(x)
examp(x,y')
examp(x,y,3)
```

全局变量与局部变量

Matlab中，函数文件中的变量是局部变量。

如在若干函数中，都把某一变量定义为全局变量，那么这些函数将共用这个变量。

全局变量的作用域是整个Matlab的工作空间，所有函数都可以对它进行存取和修改。

全局变量用global命令定义，格式为： global 变量名

例5.13 全局变量应用示例。

先建立函数文件wadd.m，该函数将输入的参数加权相加：

```
function f = wadd(x,y)
global ALPHA BETA
f = ALPHA*x + BETA*y;
```

在命令窗口中输入：

```
global ALPHA BETA
ALPHA = 1;
BETA = 2;
s = wadd(1,2)
```

输出为：

```
s =
     5
```

程序调试

程序调试是程序设计的重要环节，也是程序设计人员必须掌握的重要技能。

Matlab提供了相应的程序调试功能，即可以通过文本编辑器对程序进行调试，又可以在命令窗口结合具体的命令进行。一般说来，应用程序的错误有两类，一类是语法错误，另一类是运行时的错误。 语法错误，给出相应的错误信息，并标出错误在程序中的行号。例如：

输入下列程序：

```
A = 87;
B = 9.3;
C = A+*B;
```

系统将给出错误信息：

```
??? Error: File: Untitled1.m Line: 3 Column: 7
Unexpected MATLAB operator.
```

通过分析Matlab给出的错误信息，不难排查程序中的语法错误。

程序调试概述

Matlab对程序逻辑错误时无能为力的，不会给出任何提示信息。

可以通过

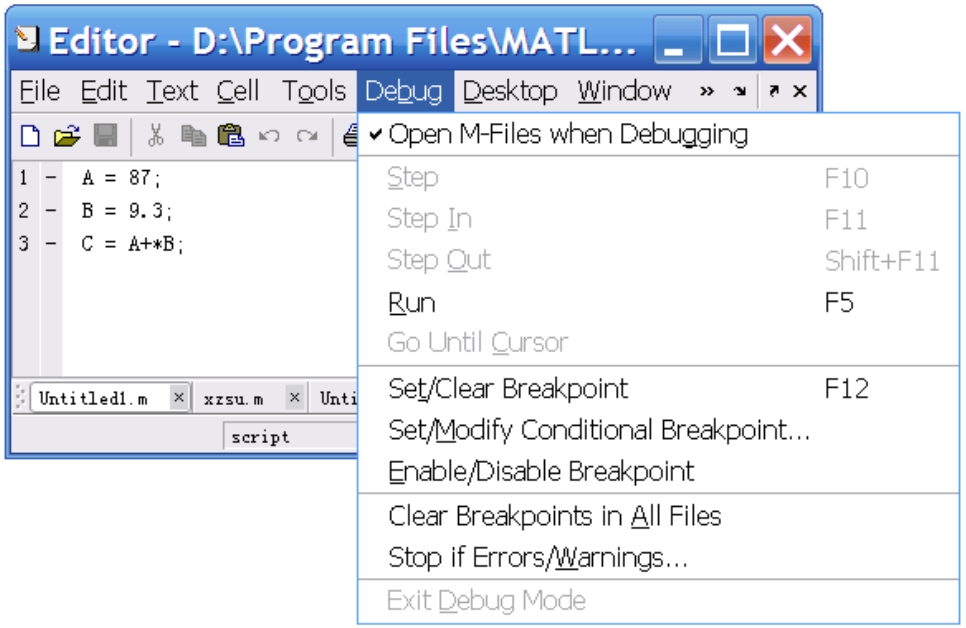
调试手段来发现。

采取的方法如下：

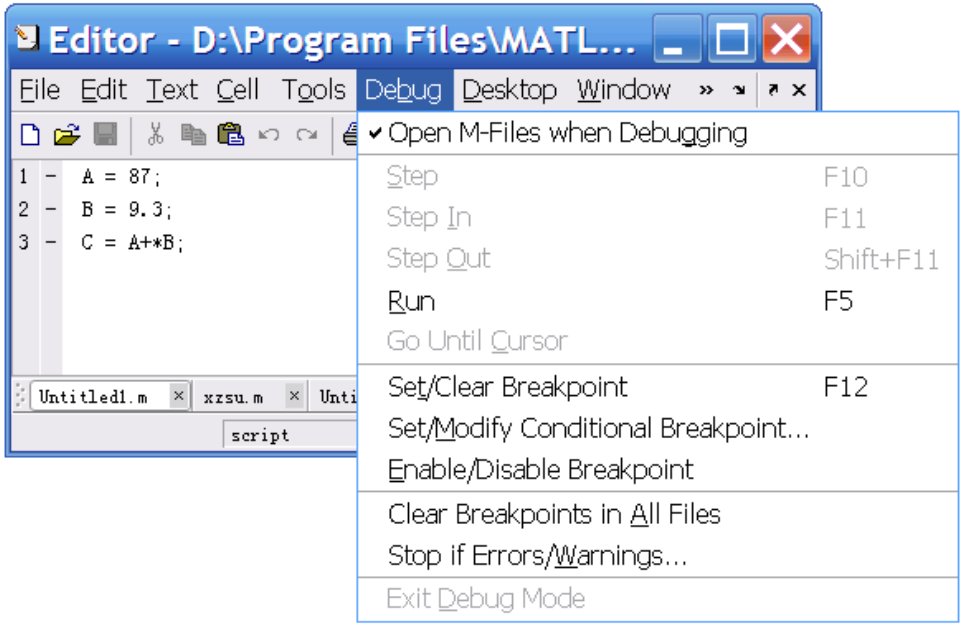
① 将程序的一些主要中间[illegible]误的区段。

② 使用Matlab的调试菜单[illegible]现程序调试。

③ 或使用命令方式来实现



Matlab调试菜单



1、控制单步运行

step：单步运行，不进入函数；

step in: 单步运行，进入函数；

step out：停止单步运行；

save and run：存储文件并开始运行。

2. 断点操作

stop if error/warnings: 在程序

执行出现错误或警告时，停止

程序运行，进入调试状态。

程序的调试

有一个求水仙花数的程序ex.m，试设置断点来控制程序执行。

调试步骤如下：

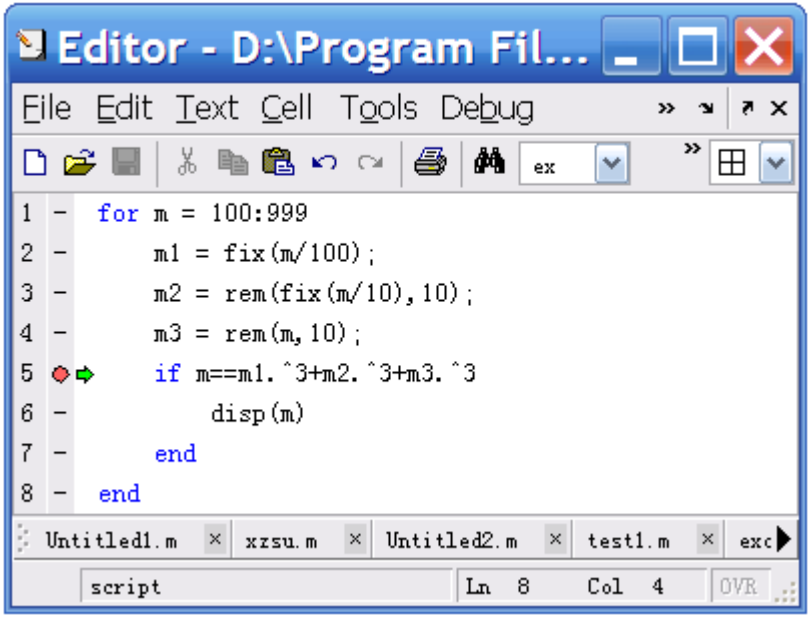
① 在if语句处设置断点：将插入点移至if语句所在行，选择Debug菜单的Set/Clear Breakpoint命令，在该行前面有一个红色圆点，程序运行时将在断点处暂停。

② 运行程序，检查中间结果。在命令窗口输入命令：ex 在窗口命令的K>>后输入变量名，检查变量的值。可以分析判断程序 的正确性。

③ 选择Debug菜单中的Continue命令，程序继续运行，在断点处又暂停，再检查 变量的值，一直到发现问题为止。

程序的调试

④ 切换工作空间，结束对程序的调试。打开编辑窗口中的Stack下拉列表框，从　　中选择Base，即将工作空间切换到主工作空间。然后选择Debug菜单中的Set Clear Breakpoint 命令清楚已设置的断点，在选择Continue命令，去除白色箭头，完成调试。



Matlab矩阵分析与处理

特殊矩阵

常见的特殊矩阵有零矩阵、幺矩阵、单位矩阵等，这类特殊矩阵在应用中具有通用性。

1、通用的特殊矩阵

常用的产生通用特殊矩阵的函数有：

zeros：产生全0矩阵（零矩阵）。

ones：产生全1矩阵（幺矩阵）。

eye：产生单位矩阵。

rand：产生0~1间均匀分布的随机矩阵。

randn：产生均值为0，方差为1的标准正态分布随机矩阵。

产生(0,1)区间均匀分布随机矩阵使用rand函数 产生均值为0，方差为1的标准正态分布随机矩阵使用randn函数

例 建立随机矩阵：

① 在区间[20,50]内均匀分布的5阶随机矩阵。
② 均值为0.6，方差为0.1的5阶正态分布随机矩阵。
命令如下：

```
x = 20+(50-20)*rand(5)
```

```
x =
   48.5039   42.8629   38.4630   32.1712   21.7367
   26.9342   33.6940   43.7581   48.0641   30.5860
   38.2053   20.5551   47.6544   47.5071   44.3950
   34.5795   44.6422   42.1462   32.3081   20.2958

   46.7390   33.3411   25.2880   46.8095   24.1667
```

```
y = 0.6 + sqrt(0.1)*randn(5)
```

矩阵结构变换

1、对角阵与三角阵

只有对角线上有非零元素的矩阵称为对角矩阵，在研究矩阵时，有时候需要将矩阵的对角线上的元素提取出来形成一个列向量，有时也需要用一个向量构造一个对角阵。

（1）提取矩阵的对角线元素函数：diag

例如：

```
A = [1,2,3;4,5,6];
D = diag(A)
D =
     1
     5
```

diag函数还有一种形式：diag(A,k)提取第k条对角线的元素。

例如：

```
D1 = diag(A,1)
D =
     2
     6
```

矩阵结构变换

(2)构造对角矩阵

如果V是一个m个元素的向量，diag(V)将产生一个m×m对角矩阵，其主对角线元素即为向量V的元素。

例如：

```
diag([1,2,-1,4])
 ans =
    1    0    0    0
    0    2    0    0
    0    0   -1    0
    0    0    0    4
```

例如：

```
diag(1:3,-1)
ans =
    0    0    0    0
    1    0    0    0
    0    2    0    0
    0    0    3    0
```

例

建立一个5×5矩阵A，然后将A的第一行元素乘以1，第二行乘以2，…第五行乘以5。

解：用一个对角矩阵左乘一个矩阵时，相当于用对角阵的第一个元素乘以该矩阵的第一行，依次类推。

命令如下：

```
A = ones(5);
D = diag(1:5);
D * A
```

```
ans =
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4
     5     5     5     5     5
```

矩阵求逆与线性方程组求解

矩阵的逆

对于一个方阵A，如果存在一个与其同阶的方阵B，使得：

$\boldsymbol{A}\cdot\boldsymbol{B}=\boldsymbol{B}\cdot\boldsymbol{A}=\boldsymbol{I}$ (I为单位矩阵）则称B为A的逆矩阵，当然，A也是B的逆矩阵。求方阵A的逆矩阵可调用函数inv(A)。

例5.18 求方阵A的逆矩阵，且验证。

```
A = [1,-1,1;5,-4,3;2,1,1];
B = inv(A);
A*B

ans =
   1.0000         0         0
  -0.0000    1.0000         0
  -0.0000         0    1.0000
```

用矩阵求逆方法求解线性方程组

将包含n个未知数，由n个方程构成的线性方程组表示为：

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

其矩阵表示形式为：

$$Ax = b$$

其中：

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

因此：

$$x = A^{-1}b$$

例 用求逆矩阵A的方法解线性方程组

$$\begin{cases} x+2y+3z=5 \\ x+4y+9z=-2 \\ x+8y+27z=6 \end{cases}$$

命令如下：

```
A = [1,2,3;1,4,9;1,8,27];
b = [5,-2,6]';
x = inv(A)*b   %x = A\b
x =
   23.0000
  -14.5000
    3.6667
```

~~也可以运用左除运算符求解。~~

矩阵行列式值

把一个方程看做一个行列式，并按行列式的规则求值，称为行列式的值。

在Matlab中，使用函数det(A)得到。

例如：

```
A = rand(5)
A =
    0.9501    0.7621    0.6154    0.4057    0.0579
    0.2311    0.4565    0.7919    0.9355    0.3529
    0.6068    0.0185    0.9218    0.9169    0.8132
    0.4860    0.8214    0.7382    0.4103    0.0099
    0.8913    0.4447    0.1763    0.8936    0.1389
B = det(A)
B =
   -0.0071
```

多项式(polynomial)

- 多项式的MATLAB表达
  - 多项式由一个行向量表示
    - 该向量元素是该多项式的系数
    - 且按降幂次序排列

  如：多项式$x^4-12x^3+25x+116$由行向量：

  p=[1 -12 0 25 116]表示。

  注意，必须包括具有零系数的项。
- 求解多项式的根？

  roots指令

多项式(polynomial)

- 举例：求解多项式$x^4-12x^3+25x+116$的根

```
>>p=[1  -12  0  25  116]
p =
    1   -12    0   25   116
>>r=roots(p)
r =
  11.7473
   2.7028
  -1.2251 + 1.4672i
  -1.2251 - 1.4672i
```

- MATLAB按惯例规定，多项式是行向量，根是列向量

多项式(polynomial)

- 已知多项式的根，求解多项式？
  - 能！
  - 使用poly指令
- 举例：由上例所得的根求其多项式

```
>> pp=poly(r)
pp =
    1.0000  -12.0000   -0.0000   25.0000
     116.0000
```

即：$x^4-12x^3+25x+116$

多项式(polynomial)

- 多项式的乘法(conv指令)
- 举例：多项式$a(x)=x^3+2x^2+3x+4$

  和$b(x)= x^3+4x^2+9x+16$的乘积。

```
>> a=[1  2  3  4] ;  b=[1  4  9  16];
>> c=conv(a, b)
c =
     1     6    20    50    75    84    64
```

  - 两个以上的多项式的乘法需要重复使用conv.

多项式(polynomial)

- 多项式加法: MATLAB没有提供进行加法运算的函数。
- 如果两个多项式向量大小相同，标准的数组加法有效。

把多项式a(x)与上面给出的b(x)相加。

```
>> d=a+b
d =
     2     6    12    20
```

结果: $d(x)= 2x^3+6x^2+12x+20$

- 当两个多项式阶次不同，低阶的多项式必须用首零填补，使其与高阶多项式有同样的阶次。

考虑上面多项式c和d相加:

```
>>  e=c+[0  0  0  d]
e =
     1     6    20    52    81    96    84
```

结果: $d(x)= x^6+6x^5+20x^4+52x^3+81x^2+96x+84$

多项式(polynomial)

- 问题：编写一个多项式加法运算的函数文件
- 明确需求
  - 定义一个函数，需要有两个输入参数（比如：p1、p2），一个输出参数p_out。
    - p1、p2表示两个待计算的多项式。
    - p_out表示两个多项式的求和结果
  - 函数的内部处理
    - 如p1、p2两参数大小相等，则直接相加: p_out=p1+p2
    - 如p1、p2两参数大小不等：
      - If length(p1)>length(p2)
        - P2前面要补0元素，使p1、p2两参数大小相等
      - 否则
        - P1前面要补0元素，使p1、p2两参数大小相等

多项式(polynomial)

- 函数文件的编写

```
function p_out=poly_sum(p1, p2)
% calculate the sum of two polynomials

if length(p1)==length(p2)
    p_out=p1+p2;
elseif length(p1)>length(p2)
    p2=[zeros(1,length(p1)-length(p2)), p2];
    p_out=p1+p2;
else
    p1=[zeros(1,length(p2)-length(p1)), p1];
    p_out=p1+p2;
end
```

多项式(polynomial)

- 多项式的除法(deconv)
- 举例说明：

$c(x)=x^6+6x^5+20x^4+50x^3+75x^2+84x+64$

除以$b(x)= x^3+4x^2+9x+16$

```
>> c=[1    6    20    50    75    84    64];
>> b=[1  4  9  16];
>>  [q , r]=deconv(c , b)
q =
    1    2    3    4
r =
    0    0    0    0    0    0    0
```

多项式(polynomial)

- 多项式的导数 (polyder)
- 举例：求$b(x)= x^3+4x^2+9x+16$的导数。

```
>> b=[1    4    9    16];
>> d=polyder(b)
d =
     3     8     9
```

结果为： $3x^2+8x+9$

另外两种形式为：

p = polyder(P,Q)：求P*Q的导函数

[p,q] = polyder(P,Q)：求P/Q的导函数，导数分子存入p，分母存入q

多项式(polynomial)

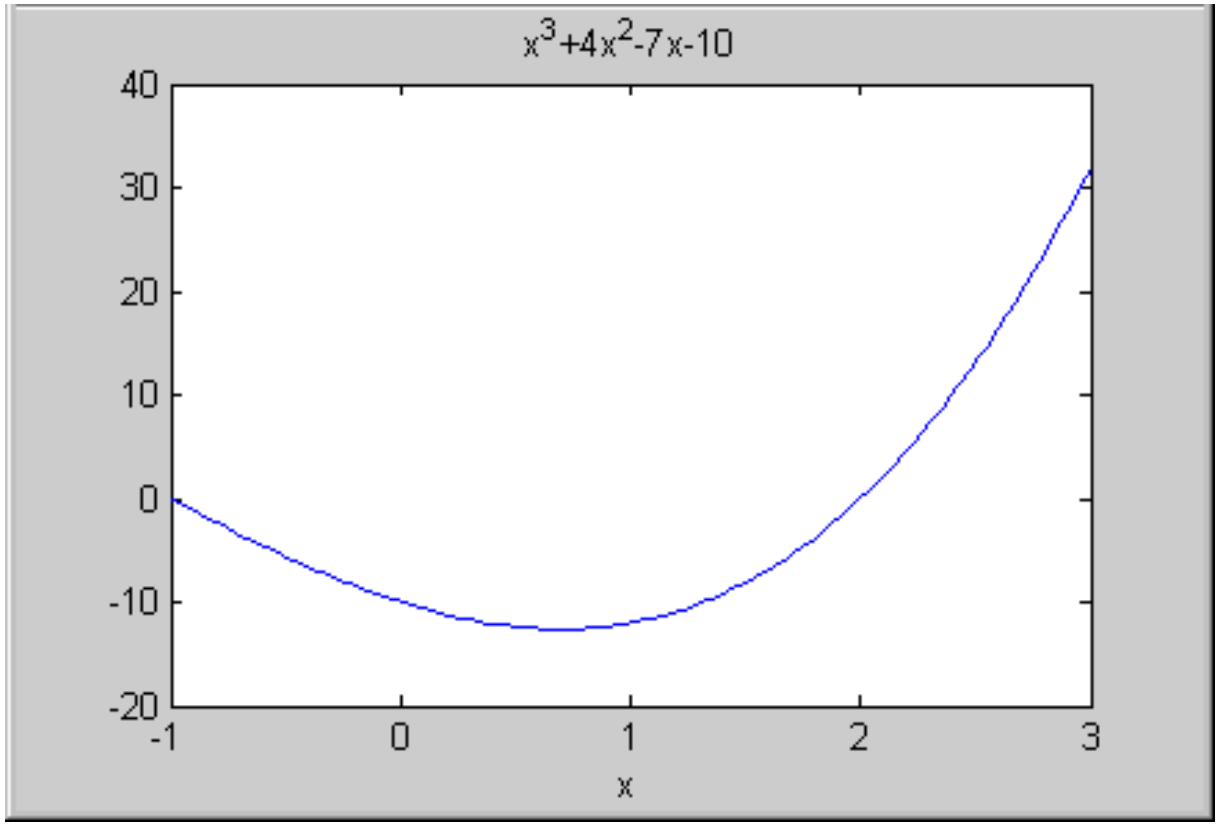
- 多项式的估值(polyval)
- 举例：绘制$p(x) = x^3+4x^2-7x-10$在[-1, 3]段上的曲线。

```
x=linspace(-1, 3);  % choose 100 data points between -1and 3.
p=[1  4  -7  -10];
v=polyval(p, x);

plot(x, v);
title('x^{3}+4x^{2}-7x-10');
xlabel('x')
```

多项式(polynomial)

p(x) = $x^3+4x^2-7x-10$在[-1, 3]段上的曲线:



函数的数值导数

- 导数定义为：

$$\frac{dy}{dx}=\lim_{h\to 0}\frac{f(x+h)-f(x)}{(x+h)-(x)}$$

- 则y=f(x)的导数可近似为：

$$\frac{dy}{dx}\approx\frac{f(x+h)-f(x)}{(x+h)-(x)}$$

这里h>0

它是y的有限差分除以x的有限差分。

- **MATLAB中没有直接提供数值导数的函数，只有计算向前差分的函数diff，其调用格式为：**

DX = diff(X)　计算向量X的向前差分

DX = diff(X，n)　计算向量X的n阶向前差分

例题

设x由[0,2π]间均匀分布的10个点组成，求sinx的1-3阶差分。

命令如下：

```
X = linspace(0,2*pi,10);
Y = sin(X);
DY = diff(Y)
D2Y = diff(Y,2)
D3Y = diff(Y,3)
```

```
DY =
    0.6428    0.3420   -0.1188   -0.5240   -0.6840   -0.5240   -0.1188
       0.3420    0.6428
D2Y =
   -0.3008   -0.4608   -0.4052   -0.1600    0.1600    0.4052    0.4608
       0.3008
D3Y =
   -0.1600   -0.0556    0.2452    0.3201    0.2452    0.0556   -0.1600
```

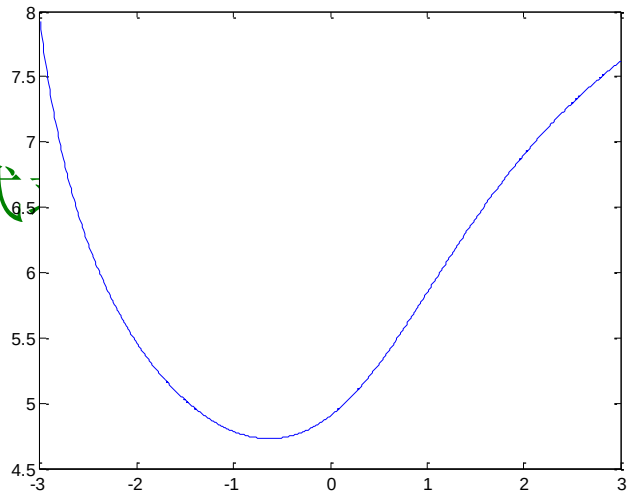
函数的数值导数（续）

例：设 $f(x)=\sqrt{x^3+2x^2-x+12}+\sqrt[6]{x+5}+5x+2$

在[-3,3]区间内以0.01为步长求数值导数。并画出导函数图像。

程序如下：

```
f = inline('sqrt(x.^3+2*x.^2-x+12)+(x+5).^(1/6)+5*x+2'); %内联函数
x = -3:0.01:3;
dx = diff(f([x,3.01]))/0.01;  %根据定义式求
plot(x,dx)
```



数值积分

一元函数的数值积分

- 常用积分指令：quad和quadl。

  一般说来，quadl比quad更有效。

- 具体调用格式如下：

  q = quadl(fun,a,b)

  q = quadl(fun,a,b,tol)

  q = quadl(fun,a,b,tol,trace)

  [q,fcnt] = quadl(fun,a,b,...)

  - 输入量fun为被积函数的句柄。
  - 输入量a, b分别是积分的下限、和上限，都必须是确定的数值;
  - 前3个输入参数是调用积分指令所必须的，其他可以缺省;
  - 输入量tol是一个标量，控制绝对误差;
  - 输入量trace为非0值时，将随积分的进程逐点画出被积分函数;
  - 输出参数fcnt返回函数的执行次数。

  Note：quad的调用格式与quadl相同

数值积分（续）

- 举例：求定积分 $I=\int_0^1 e^{-x^2}dx$

MATLAB指令quad和quadl求积分

```
>>fun=inline('exp(-x.*x)','x');  %数组乘符号.*的采用是必须的
>>Isim=quad(fun,0,1),  I8=quadl(fun,0,1)
    Isim =  0.7468
    I8 =  0.7468
```

数值积分（续）

- 举例：求解定积分 $I=\int_0^1\sqrt{\ln\frac{1}{x}}dx$

用quad指令求积分

```
>>ff=inline('sqrt(log(1./x))','x');
>>Isim=quad(ff,0,1)
Warning: Divide by zero.
> In inlineeval at 13
In inline.subsref at 25
In quad at 63
Isim = 0.8862
```

元素排序

Matlab中对向量X排序的函数是sort(X), 函数返回一个对X中的元素按升序排列的新向量。

sort函数也可以对矩阵A的各列（或行）重新排序，其调用格式为： [Y,I] = sort(A,dim)

dim=1,按列排序；dim=2,按行排序，Y是排序后的矩阵，I记录Y中的元素在A中的位置。

例：对下列矩阵做各种排序。

$$A=\begin{bmatrix} 1 & -8 & 5 \\ 4 & 12 & 6 \\ 13 & 7 & -13 \end{bmatrix}$$

命令如下：

```
A = [1,-8,5;4,12,6;13,7,-13];
 sort(A)
 ans =
 1    -8   -13
 4     7     5
13    12     6
-sort(-A,2)  %对A的每行按降序排列
ans =
5      1    -8
12     6     4
13     7   -13
```

数据插值

在工程测量和科学实验中，所得到的数据通常是离散的，要得到这些离散点以外的其他点的数值，就需要根据已知的数据进行插值。

插值函数一般由线性函数、多项式、样条函数或这些函数的分段函数充当。

一维数据插值：被插值函数有一个单变量。

采用的方法有：线性方法、最近方法、三次样条和三次插值。

在Matlab中实现这些插值的函数是interp1，其调用格式如下：

Y1 = interp1(X,Y,X1,method）

函数根据X，Y的值，计算函数在X1处的值。

X,Y是两个等长的已知向量，分别描述采样点和样本值；

X1是一个向量或标量，描述欲插值的点；

Y1是一个与X1等长的插值结果。

method是插值方法，允许的取值为：

数据插值

（1） ‘linear’：线性插值。默认的插值方式。它是把插值点靠近的两个数据点用直线连接，然后在直线上选取对应插值点的数据。

（2） ‘nearest’：最近点插值。根据已知插值点与已知数据点的远近程度进行插值。插值点优先选择较近的数据点进行插值。

（3） ‘cubic’：3次多项式插值。根据已知数据求出一个3次多项式，然后根据该多项式进行插值。

（4） ‘spline’：3次样条插值。指在每个分段内构造一个3次多项式，使其满足插值条件外，在各节点处具有光滑的条件。

例：给出概率积分数据表如下，用不同的插值方法计算f(0.472)。

x	0.46	0.47	0.48	0.49
f(x)	0.4846555	0.4937542	0.5027498	0.5116683

命令如下：

```
x = 0.46:0.01:0.49;
f = [0.4846555,0.4937542,0.5027498,0.5116683];
format long
interp1(x,f,0.472)
ans =
 0.49555332000000
interp1(x,f,0.472,'nearest')
ans =
0.49375420000000
interp1(x,f,0.472,'spline')
ans =
0.49556073600000
interp1(x,f,0.472,'cubic')
ans =
0.49556111971206
```

其中，3次样条和3次多项式的插值结果优于最近点插值方法和线性插值方法，但插值方法的好坏依赖于被插值函数，没有一种对所有函数都是最好的插值方法。

曲线拟合

数值插值要求逼近函数在采样点与被逼近函数相等，但由于测量误差，所获得的数据不一定准确，如果强求逼近显然不够合理。

曲线拟合不要求逼近函数通过各采样点，但要尽量的接近这些点，使误差在某种意义上达到最小。

曲线拟合的实现：

在matlab中，用polyfit函数来求得最小二乘拟合多项式的系数，再用polyval函数按所得的多项式计算所给出点上的函数近似值。

polyfit函数的调用格式为：

[P,S] = polyfit(X,Y,m)

函数根据采样点X和采样点函数值Y，产生一个m次多项式P及其在采样点的误差向量S。其中X、Y是两个等长的向量，P是一个长度为m+1的向量，P的元素是多项式系数。

polyval函数的功能是按多项式的系数计算x点多项式的值。

例：用一个三次多项式在区间[0,2π]内逼近函数sinx。

在给定区间内，均匀的选择20个采样点，并计算采样点的函数值

然后利用3次多项式逼近

命令如下：

```
x = linspace(0,2*pi,20);
y = sin(x);
p = polyfit(x,y,3)
y1 = polyval(p,x)
plot(x,y,':o',x,y1,'-*')
legend('sin(x)','fit')
```