Lecture 12: Matlab 简介(一)

张伟平

Monday 30$^{\text{th}}$ November, 2009

Contents

Chapter 1
Introduction

MATLAB的名称源自 Matrix Laboratory, 它是一种科学计算软件, 专门以矩阵的形式处理数据. matlab 将高性能的数值计算和可视化集成在一起, 并提供了大量的内置函数, 从而被广泛地应用于科学计算 、控制系统、信息处理等领域的分析、仿真和设计工作，而且利用 matlab 产品的开放式结构，可以非常容易地对 matlab 的功能进行扩充, 从而在不断深化对问题认识的同时, 不断完善 matlab 产品以 提高产品自身的竞争能力.

Matlab特性:

➤ **数值计算的功能**: 矩阵的运算, 分析, 线性代数求解, 微分方程式, 稀疏矩阵, 特殊函数之处理, 傅利叶转换, 资料分析;

➤ **绘图功能**: 2D, 3D, 三维图形处理, 声音及动画处理.

➤ **程式语言功能**: 语言简单易学, 并把编辑,编译,连接,执行功能融为一体, 调试程序手段丰富.

➤ **绘图介面设计的功能**: 下拉式功能表之设计, 按钮设计, 滑鼠处理.

➤ **强大的工具盒**: 控制系统, 模糊逻辑, 影像处理, 频谱分析, 信号处理器, 统计, 偏微分方程, 类神经网路

➤ **扩充功能**: matlab compiler, matlabC 数学程式库。

1.1 GUI and Basic functions

启动Matlab后, 在菜单栏中: Desktop - Desktop Layout - Default 这样就选择了默认的桌面模式. 此时, 在屏幕上可以看到以下界面

✻ **Menu**: 包括File, Edit, View, Debug, Desktop, Window, Help等菜单.

✻ **Toolbar**: 工具栏, 就是菜单栏下面那些图标.

✻ **Current Directory**: 位于Toolbar 旁边的当前目录栏, 从这里可以直接输入你要跳转的目录, 其使用方法和Windows里面的”打开文件”窗口很像.

✻ **Current Directory** : 在左上方还有一个叫做Current Directory 的选项卡, 它和Workspace共用一个窗口, 作用也上面提到的一样, 只不过这里面可以直接显示出当前目录中的文件. 这大大方便了我们的操作.

✻ **Workspace**: 从这个窗口, 我们可以看到当前Matlab工作空间中变量信息. 点击选项卡上的字就可以切换Worksapce和Current Directory 选项卡了.

✻ **Command History**: 位于左下的窗口, 其中记录了你曾经在Matlab中输入的命令.

✻ **Start Button**: 就像Windows里面的开始按钮, 从这里你可以选择一个你想做的任务.

✻ **Command Window**: 位于右方的Command Window是最重要的窗口, 所有命令行都从这里输入.

以上是可以在屏幕上直接看到的窗口. 另外还有几个是你 Call 它们才会出来的.

✻ **Editor**: Matlab自带编辑器, 用来编辑m文件. 支持语法高亮, 设置断点. 比较好用.

✻ **Array Editor**: 用于显示, 编辑变量. 有点像Excel, 在workspace里双击相关变量能自动打开, 或使用命令*openvar('*变量名*')*打开.

1.1.1 Command Window

❑ **MATLAB中常用的几个命令**:

- **mkdir**:新建目录, 例如: mkdir test.
- **cd**和**cd ..** :进入目录; 回到当前目录的上一层目录.

- **mdir** :删除目录.
- **clc** :清屏, 此命令只打扫屏幕, 对已有变量等毫无影响.
- **edit** :用来打开默认的Editor 编辑 m 文件, 可以采用edit 文件名的方式来新 建/打开一个指定了名称的文件.
- **exit** :退出**MATLAB**.

❑ **MATLAB中默认的结果变量**: **ans**, 如果表达式没有名称, 则给出的结果 会自动存在**ans**中, 直到下一次没有名称的变量运算结束前, **ans**中所储存的值不变.

❑ **更改显示的精度**

要控制Matlab显示计算结果的模式(注意, 只是显示结果, 其机器内部存储值不受影响), 我们可以使用**Format**命令. 命令语法: **Format** *type*. 常用*type*参数有:

type	result	example
+	(正数)+, (负数)-, (零)blank	+
bank	银行格式, 元角分表示	3.14
compact	紧凑格式, 清除显示的空行,	`theta = pi/2`
	和 loose 相反	theta=
		1.5708
hex	十六进制	400921fb54442d18
long	双精度15位, 单精度8位	3.14159265358979
long e	15位有效数字的科学计数法表示	3.141592653589793e+00
long g	从long 和long e中自动选择最佳表示	3.14159265358979
loose	输出变量之间有空行	`theta = pi/2`
	和 compact 相反	theta=
		1.5708
rat	有理数表示	355/133
short	输出小数点后4位, 所有最多不超过7位.	3.1416
	大于1000的实数使用科学计数法表示	1.0012e+003
short e	5位有效数字的科学计数法表示	1.0012e+003
short g	从short和short e中自动选择最佳表示	3.1416

1.1.2 Command History

这个窗口存储了命令窗口里所有曾经使用的命令. 这个子窗口有如下这些用处:

运行单个命令 双击窗口中储存的命令, 该命令将再次被运行.

运行多个命令 可以按住Ctrl 或者Shift 键选择多个命令, 然后选择右键菜单 中的: Evaluate Selection

储存命令 用前一行中所用办法, 选中你想储存的命令, 然后选择右键菜单中: Create M-file

自定义快捷方式 同样,也是先选中命令, 再选择邮件菜单中的: Create Short-Cut, 之后可以通过点击自定义的按钮, 快捷地执行前面所选择地命令.

要清除这个窗口中历史命令数据, 也是通过右键菜单中地 Delete Selection/ Delete to Selection/ Clear Entire history 这些命令.

与历史命令有关的两个实用功能:

- 自动补齐功能

这个功能非常实用. 当你输入较长的命令行时, 可以按Tab键自动补齐. 能够补齐的要素包括:

- Matlab内部命令, 常量, 在Search path内的函数等.
- 当前工作区内已有变量名.
- 当前目录下已有的文件名, 目录名.
- 在Command History中记录在案的历史命令.

❑ Diary 功能

很实用的功能, 能记录你在Command Window中输入的所有命令, 以及大部分的结 果输出. 实用方法:

开始记录 语法: diary filename 将此语句之后的命令和输出结果记录在filename这个文件中. 如果不指定文件名, 将记录在一个叫做diary的文件里面.

中断记录 语法: diary off 此语句之后的命令和结果将不被记录.

继续记录 语法: diary on 在记录被暂停后, 继续开始. 之前的记录不会丢

失.

1.1.3 MatLab Help

❑ **help 命令名或函数名** 适用于知道函数或者命令名, 但是不知道具体的使用方法.

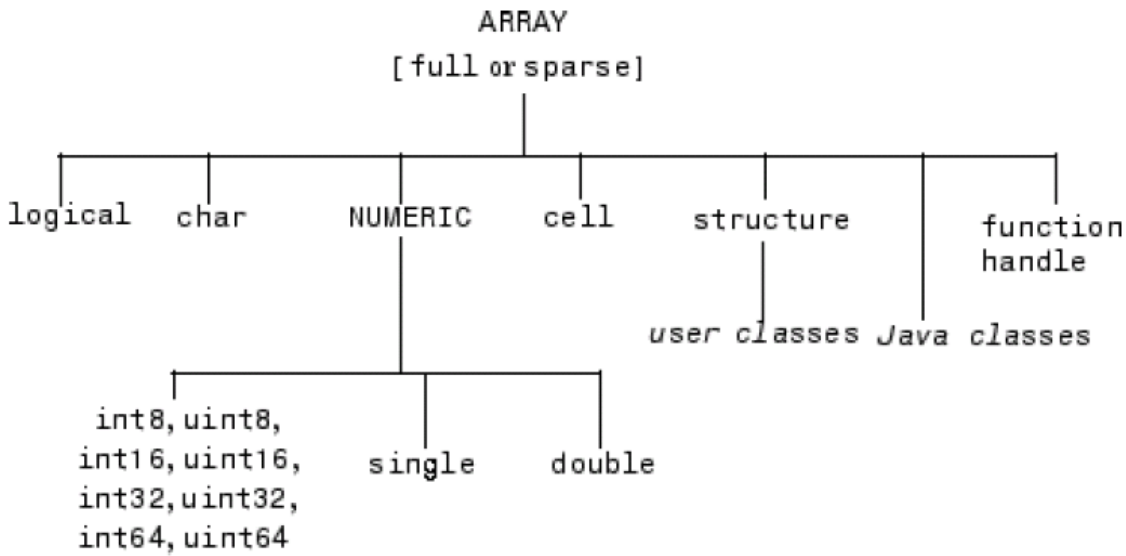
❑ **lookfor-大海捞针** 可以查找所有含有要查询关键字的文档, 检索会很慢!

❑ **doc-详细帮助** doc和help语法相同, 但是它会打开Matlab自带的网页浏览器, 显示更为详细的帮助.

❑ **demo-演示系统** 里面包含了很多Matlab使用实例, 而且新版Matlab中的demo还包括了flash. 启动方法, 可以在command window中直接输入: demo 命令.

Chapter 2
Data in MatLab

本章主要介绍Matlab中数据的类型, 组织形式, 存储方式, 及如何输入输出数据.



numeric-数值型 这是最大的一类, 通俗说来就是我们平常见到的数字, 其下面细分了好多类, 区别在于在计算机中储存的格式不同。

- int打头的那一串, 表示整型数, 就是整数, 后面的数字($X = 8, 16, 32, 64$)表示在计算机中用X个位的空间来存储这个数, 这样intX的数可以表示从$-2^X + 1$到$+2^X$之间的整数. 需要注意int64类型的数不能用来运算.
- uint打头的那一串, 和int打头的类似, 但是它叫做无符号整型数, 只表示正数, 一个intX的数表示范围是$0 \sim 2^X - 1$, 同样要注意uint64不能拿来运算.
- single 叫做单精度浮点数, 它用了32个位. Matlab中single可表示的数值范围如下, 超过则出错:

最小负数	最大负数	最小正数	最大正数
-3.40282e+038	-1.17549e-038	1.17549e-038	3.40282e+038

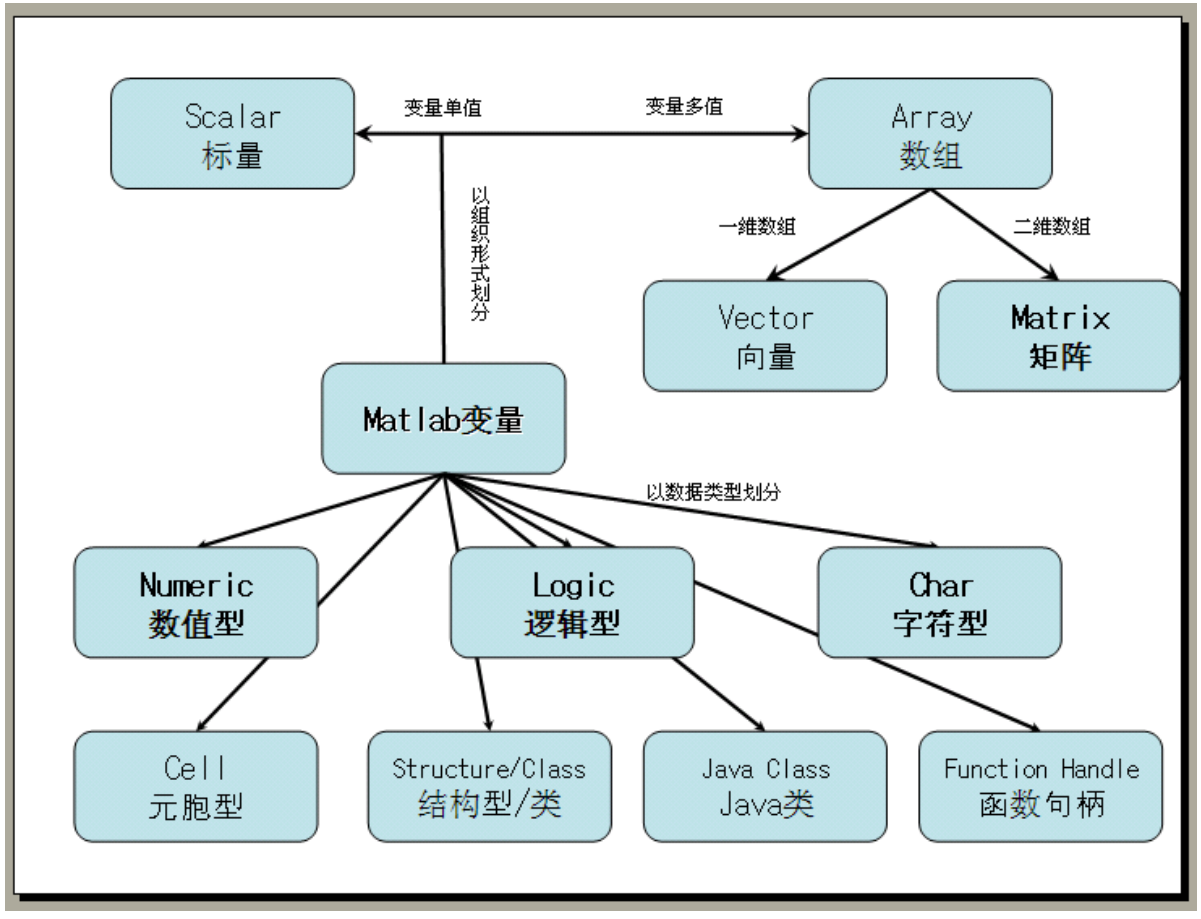
- double叫双精度浮点数, 它占用64个位, 如果你赋值时不指定变量的类型, 默认类型是double, 它能表示的数值范围如下:

最小负数	最大负数	最小正数	最大正数
-1.79769e+308	-2.22507e-308	2.22507e-308	1.79769e+308

char-字符型 顾名思义, 我们输入的字符也可以储存成为变量, 不过单个字符才是标量, 如果一串字符 - 例如一句话, 那就是字符型数组, 字符要用单引号括起来.

Sym-符号型 用*sym*或者*syms*可以把字符,表达式,方程,矩阵等定 义为数学符号, 它的运算结果为表达式.

其他细分的话, 还有逻辑型,cell-元胞型,structure-结构型,Java类, 函数句柄等等.



2.1 Manipulating data

这么多数据类型里面, 数值型数据是最复杂的－那么多种类; 一般而言不必考

虑太多关于这么多种数值类型中选择哪种 类型的问题, 甚至乎, 根本就不需要了解除了double之外的其它数值类型. 在Matlab中, 如果你不特别说明, 所有数值变量都被当作double类型, 对于这种类型要掌握以下内容

判断

一个变量是否数值类型:

* **isnumeric**(变量名): 判断一个变量是否是数值类型;

* **isfloat**(变量名): 判断一个变量是否是浮点型(含单精度和双精度浮点),

* **isa**(变量名,’double’): 判断一个变量是否是双精度浮点数,

eps

由于硬件软件限制, 计算机中, 我们无法精确表示所有的数, 事实上, 计算机能存储的数不是连续的, 而是间隔了很短距离的离散数值, 存储时, Matlab会寻找与要存储数值最接近的那个可行存储值, 然后舍入为该值进 行存储, 这就造成了所谓的误差. 例如在Matlab中计算这个式子: $1-3*(4/3-1)$ 我们口算都知道结果是0, 但是由于存储误差等原因, 在Matlab中结果是$2.2204e-016$.

所以在Matlab中有一个特殊的变量: *eps*, 如果两个数之间差别不大于*eps*我们可以认为它们相等.

Inf

表示无穷大的特殊变量, 计算1/0, 以及超过所能表示的最大的值时都会得到这个结果. 可用*isfinite*(变量名)来判断是否时无穷大.

NaN

表示“不是数”的特殊变量, 计算0/0会得到此结果, 可以用*isnan*(变量名)来判断某个变量是否是“不是数”, 画图时常用它来“抹去”多余区 域.

2.1.1 Creating Objects

变量的名称是以字母开头, 后接字母, 数字,下划线等字符串. 变量名区分大小写. 变量赋值的语句为: 变量=表达式, 在表达式后可以跟注释内容, 以%开始.

MatLab中有一些由系统本身定义的变量, 包括: ans, eps, pi, i,j, inf, Inf, NaN, nan, nargin(函数输入参数个数), nargout(函数输出参数个数),

realmax(最大的正实数), realmin(最小的正实数), lasterr(最新的错误信息), lastwarn(最新的警告信息). 在命名变量时,应尽量避免使用这些系统定义的变量名.

数组的创建和访问

❑ 直接赋值

遵循以下原则:

➢ 矩阵的所有元素必须置于方括号[]内. 空的方括号则表示空.

➢ 矩阵一行内的元素使用空格或者逗号分开, 行之间使用分号或按回车键分开.

➢ 如果某行内容太多, 可以使用续行号..., 按回车后继续输入. 续行号 不能加在矩阵一行中的两个元素之间.

➢ 矩阵元素可以是实数,负数, 向量,矩阵等数值量或其变量名, 每行中的元素个数要相同.

➢ 输完矩阵内容后逗号, 或按回车键, 则显示出创建矩阵的内容; 若在其后加分号, 则按回车键后不显示该矩阵内容.

➢ 同一行内输入几个命令时, 需用逗号或者分号间隔.

例: 一维行数组

```
a=[1 2 3 4], b=[4,3,2,1]
```

↑Example

↓Example

一维列数组

```
c=[1;2;3;4];
```

↑Example

↓Example

二维数组

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

↑Example

↓Example

当数值比较多时, 我们可以用符号 ... 或者\ 将下一行续在上一行语句之后

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

↑Example

↓Example

❑ 特殊函数赋值

Matlab中有一些函数可以用来生成特殊的数组. 由于标量也是特殊数组, 所以这些 函数也可以用来生成特殊标量.

全零数组

zeros(行数，列数) 生成相应维度的全部是零的数组.

↑Code

↓Code

全一数组

ones(行数，列数) 生成对应维度的全部为1的数组

↑Code

↓Code

单位矩阵

eye(行数，列数) 生成对应维度的对角元为1的数组，*eye*(10)，*eye*(2,3)等.

↑Code

↓Code

随机数组

rand(行数，列数) 生成在[0,1]之间均匀分布的对应维度的数组

randn(行数，列数) 生成在标准正态分布的对应维度的数组

↑Code

↓Code

随机排列数组

randperm(n) 生成一个一维的将从1到n整数随机排列的数组.

↑Code

↓Code

等间隔赋值

linspace(起始值,终值,取样数量) 将在“起始值”和“终 值”之间等距离取“取样数量”个的数值，组成一个一维数组.

↑Code ↓Code

对数等间隔赋值

logspace(起始值,终值,取样数量) 将生成的数组中每 个元素取以10为底的对数后得到的新数组恰好相当于 *linspace*(起始 值,终值,取样数量)

↑Code ↓Code

幻方阵

magic(维度) 产生相应维度的幻方阵(每行,每列,对角线元素和都相等)

↑Code ↓Code

- 循环赋值, 函数:

  格式: 起始值:间隔:终值 这个语句将产生一个一维行数组, 其最初值为

“起始值”, 其 后每一个值是前一个值加上跳跃间隔, 直到到达或最接近终值. 举例说明:

* $a = 1 : 2 : 11$ 将效果等同于$a = [1, 3, 5, 7, 9, 11]$

* $b = 1 : 4$ 当“跳跃间隔”等于1时可以省略. 此语句等同$b = [1, 2, 3, 4]$

* $c = 5 : -1 : 3$ “跳跃间隔”也可以负数, 此时注意要求起始值大于终值, c为空数组. 此数组等同于: $b = [5, 4, 3]$

数组大小

有关于数组大小的函数, 有

length

语法: *length*(数组名), 得到数组中最长的那个维度的长度, 例如, 一个5行3列元素的数组, length返回的值是5, 对于一维数组, 其返回值就是这个维度的长度.

size

语法: *size*(数组名), 得到数组的每个维度的长度, 其返回值是其元素按照维度次序排列的行向量, 5行3列的二维数组size函数返回[5,3].

ndims

语法: *ndims*(数组名), 通过这个函数可以得知数组一共有多少维.

numel

语法: *numel*(数组名), 数组中一共有多少个元素.

数组元素的访问

- 单个元素的访问

  比如

  c=[1,2,3,4]　则访问第一个元素　c(1)

  d=[1,2,3;4,5,6;7,8,9]　则访问第一行, 第三列元素为　d(1,3)

  ↑Example

  ↓Example

- 多个元素的访问 比如

↑Example

```
d=[1,2,3;4,5,6;7,8,9]
d(:,1), d(1:2,2:3), d(1:end,2:3), d([1,3],[1,2]),d(d(:,1)>4,:)
```

↓Example

在Matlab系统中, 无论多少维的数组, 其内部存储方式都是以该数组最后一维为顺序, 在每个数组里按照由第一列至最后一列的顺序存储. 因此, 利用此规律可以使用单指标访问数组元素. 比如一个$3 \times 3 \times 3$的三维数组, 我们想使用单指标访问其中某个元素. 注意到其存储顺序

↑Example

```
a=1:27 b=zeros(3,3,3) b(:)=a
```

↓Example

因此可以通过形如$b(6)$, $b(7)$这样的单指标寻址可以得到第$6, 7$这两个元素.

2.1.2 Operators

Matlab 中的运算符如下:

运算符					
数学运算		比较运算		逻辑运算	
+	加法	<	小于	~x	逻辑非
-	减法	>	大于	x&y	逐元素逻辑与
*	乘法	<=	小于	x&&y	同上(x真时才运行y)
/	除法	>=	大于	x|y	逐元素逻辑或
^	乘方	==	等于	x||y	同上(x真时才运行y)
		~=	不等于	xor(x, y)	异或

矩阵运算的简单例子

Example

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

```
>> A + A'

ans =

    32     8    11    17
     8    20    17    23
    11    17    14    26
    17    23    26     2

>> A'*A

ans =

   378   212   206   360
   212   370   368   206
   206   368   370   212
   360   206   212   378
```

```
>> d = det(A)

>> d

d =

     0

>> rref(A)

ans =


     1     0     0     1

     0     1     0    -3

     0     0     1     3

     0     0     0     0

>> inv(A)

Warning: Matrix is close to singular or badly scaled.

         Results may be inaccurate. RCOND = 9.796086e-018.
```

```
ans =

  1.0e+015 *

    0.1251    0.3753   -0.3753   -0.1251
   -0.3753   -1.1259    1.1259    0.3753
    0.3753    1.1259   -1.1259   -0.3753
   -0.1251   -0.3753    0.3753    0.1251
>> eig(A)
ans =

   34.0000
    8.0000
```

```
     0.0000
    -8.0000
>> A/2
ans =

    8.0000    1.5000    1.0000    6.5000
    2.5000    5.0000    5.5000    4.0000
    4.5000    3.0000    3.5000    6.0000
    2.0000    7.5000    7.0000    0.5000
>> A^2
>> poly(A)  %det(A-lambda I)多项式的系数
ans =

  1.0e+003 *
```

```
    0.0010    -0.0340    -0.0640     2.1760    -0.0000
```

↓Example

对数组的运算, 则运算符和一些常用运算函数为

矩阵的共轭转置	`A'`
矩阵的转置(不共轭)	`A.'`
矩阵 AB:	mtimes(A,B) 或者 A*B
对应元素相乘	times(A,B) 或者 A.*B
矩阵的幂次	mpower(A,B) 或者 `A^B`
逐元素的幂次	power(A,B) 或者 `A.^B`
矩阵的左除	mldivide(A,B) 或者 `A\B` 相当于求解AX=B
矩阵的右除	mrdivide(A,B) 或者 `A/B` 相当于求解XA=B
逐元素左除	ldivide(A,B) 或者 `A.\B`

逐元素右除	rdivide(A,B) 或者 `A./B`
取对角元	diag(A)
矩阵的行列式值	det(A)
矩阵或向量的模	norm(A)
矩阵的秩	rank(A)
矩阵的迹	trace(A)
矩阵的条件数	cond(A), condeig(A)
Cholesky 分解	chol(A)
矩阵的逆	inv(A)
线性方程组	linsolve(A,b)
LU 分解	lu(A)
Moore-Penrose 广义逆	pinv(A)
QR分解	qr(A)
矩阵的特征根(向量)	eig(A)

矩阵的最大特征根(向量)	eigs(A)
奇异值分解	svd(A)
求奇异值和向量	svds(A)

一些特定的, 或者高级函数可以使用**help specfun**或者**help elmat**来查看.

其他一些常用运算函数为

arrayfun	对数组每个元素应用函数	sort	从小到大排序
cross	向量的叉乘	ceil	向上取整
cumprod	累积乘积	fix	向0的方向取整
cumsum	累积求和	floor	向下取整
dot	向量内积	mod	模
kron	Kronecker乘积	round	四舍五入取整
prod	数组元素乘积	tril	矩阵的上三角部分
sum	数组元素求和	triu	矩阵的下三角部分
find	找出满足条件的元素的脚标		

matlab中的基本数学函数可以用 **help elfun**来查看.

Matlab中可以将复数当作普通数值进行操作, 生成复数的方法: f=1+2i, 即可, 注意如果已有a=1,b=3, 从已有变量生成复数记得加乘号 c=a+b*i .

更多的函数请参考Matlab支持中心的函数手册

Chapter 3
Graphics

Matlab拥有强大的图形绘制能力, 其尤其擅长科学计算中的图形绘制. 绘图的基本命令比较简单如: plot, plot3等, 但是, 如果需要绘制复杂的图形, 如: feather, mesh, surf等, 或者需要按照特殊的要求修饰图形, 如涉及: 某曲线粗细设置, 在图形上标记字符, 同时画多个图形等实际任务, 可能就需要掌握Matlab中底层的绘图命令, 甚至图形句柄操作.

3.1 Use plotting tools

在使用matlab图形工具绘图时, 要先建立相应的变量. 比如我们想画函数$f(x) = x^3$在区间[-1,1]上的图形, 则先建立x, y两个变量

↑Example

```
x = -1:.1:1;
y = x.^3;
```

↓Example

然后使用命令**plottools**打开matlab图形工具窗口. 在变量框内同时选择x, y, 在右键菜单中选择 *plot(x,y)*, 即可以画出$f(x)$的图像. 选择图形后, 还可以对图形作进一步的修饰, 比如 添加坐标标示, 题目, 线的类型等等.

使用matlab绘图工具得到的图形, 可以使用菜单里的*show M-code*来查看生成图形的代码.

3.2 Use the command interface

3.2.1 Basic plots

函数**PLOT(X,Y,S)** 可以用于创建一个新的图形. 其中S为一串字符, 表示线的类型,颜色等. 可选的值有

b	蓝色	.	点	-	solid
g	绿色	o	圆	:	dotted

r	红色	x	x标示	-.	dashdot
c	青绿色	+	加号	--	dashed
m	洋红色	*	星	(none)	no line
y	黄色	s	方块		
k	黑色	d	钻石		
w	白色	v	三角形 (down)		
		^	三角形 (up)		
		<	三角形 (left)		
		>	三角形 (right)		
		p	五角星		
		h	六角星		

例如, 生成如下图形

↑Example

```
x = 0:pi/100:2*pi;
y = sin(x);
plot(x,y,'b-')
```

↓Example

如果我们想同时将多个函数绘制在一个图形上, 如

↑Example

```
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-.25);
y3 = sin(x-.5);
plot(x,y,'b-',x,y2,'r-.',x,y3,'k-')
```

↓Example

即, 函数*plot*的使用可以是

```
plot(x1,y1,s1,x2,y2,s2,...)
```

↑Code

↓Code

使用*plot*函数会在当前打开的图像窗口绘图, 如果没有打开图像窗口, 则自动打开一个 图像窗口(figure 1, 也可以使用命令*figure*打开图像窗口), 后续所有的绘图都作用在此图像窗口. 若不想绘制在同一个图像窗口里, 则可以使用**figure(n)**来打开第n个图像窗口. 如果想 重置当前图像窗口的内容, 则可以使用**clf**命令.

也可以使用函数**plot3**绘制三维图形. 比如绘制曲线$y = xsin(x)cos(x), z = xcos^2x, x \in [0, 20]$, 则

↑Example

```
x=0:0.05:20; y=x.*sin(x).*cos(x); z=x.*cos(x).*cos(x);
plot3(x,y,z,'r.'),grid,box,axis equal
```

↓Example

3.2.2 Adding Plots to an Existing Graph

有些绘图函数, 如*contour*函数, 在执行时会将当前图像窗口的内容重置, 因此如果 要使用此类函数向已经存在的图形上添加图时, 要先使用命令**hold on**, 然后再执行绘图函数, 结束后可以使用**hold off**来解除锁定. 如

↑Example

```
[x,y,z] = peaks;
pcolor(x,y,z)
```

```
shading interp
hold on
contour(x,y,z,20,'k')
hold off
```

↓Example

3.2.3 Multiple Plots in One Figure

如果想在同一的图像窗口内绘制多个子图, 则可以使用

```
subplot(m,n,p)
```

↑Code
↓Code

生成$m \times n$个子图形. 其中p表示第p个子图形. 如

↑Example

```
t = 0:pi/10:2*pi;
[X,Y,Z] = cylinder(4*cos(t));
```

```
subplot(2,2,1); mesh(X)
subplot(2,2,2); mesh(Y)
subplot(2,2,3); mesh(Z)
subplot(2,2,4); mesh(X,Y,Z)
```

↓Example

3.2.4 Controlling the Axes

matlab默认使用数据的最大值和最小值来确定坐标轴. 也可以自己更改, 使用命令

```
2D图:  axis([xmin xmax ymin ymax]) 3D图:  axis([xmin xmax ymin ymax
zmin zmax])
```

↑Code

↓Code

可以使用命令**axis auto**恢复坐标轴的自动生成.

还可以使用比如

```
axis square
```

↑Code
↓Code

使图形区域为正方形. 使用 还可以使用比如

```
axis equal
```

↑Code
↓Code

使得坐标轴的刻度相同. 而命令**axis auto normal**重置坐标轴为自动模式. 另外, 可以使用**axis on, axis off, grid on, grid off**使 坐标轴和网格可见或者隐藏.

3.2.5 Axis Labels and Titles

控制图形标题和坐标标签, 以及向图形添加文本的命令是**title**, **xlabel**, **ylabel**, **zlabel** 和**text**. 例如

↑Example

```
t = -pi:pi/100:pi;
y = sin(t);
plot(t,y)
axis([-pi pi -1 1])
xlabel('-\pi \leq {\it t} \leq \pi')
ylabel('sin(t)')
title('Graph of the sine function')
text(1,-1/3,'{\it Note the odd symmetry.}')
```

↓Example

注意这个例子说明matlab可以使用LaTeX命令生成一些符号.

另外, 可以使用**legend**函数向图形添加legend. 添加Legend可以通过图像窗口菜单来完成, 也可是使用命令形式完成. 如下图:

↑Example

```
x = -pi:pi/20:pi;
plot(x,cos(x),'-ro',x,sin(x),'-.b')
legend('cos','sin',2);
```

↓Example

3.3 Mesh and Surface Plots

在matlab中, 绘制三维网格图或者曲面图, 使用函数**mesh**以及**surf**. 为显示函数$z = f(x, y)$, 需要先生成矩阵X和Y, 其元素为重复的行和列. 然后使用X, Y来 求函数的值Z. 函数**meshgrid**可以由一个向量, 或者两个向量来生成矩阵X和Y.

比如我们来绘制函数$sin(r)/r$, 其中$r = \sqrt{x^2 + y^2}$. 为避免出现0/0情形, 我们给r加上一个值eps.

↑Example

```
[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2) + eps;
Z = sin(R)./R;
mesh(X,Y,Z,'EdgeColor','black')
```

↓Example

如果要使用绘图工具, 则可以创建完R, Z后, 启动绘图工具: **plottools**, 选择变量及函数, 或者添加数据完成.

曲面图类似于网格图, 区别是使用颜色对网格图中的每个矩形配色. 每一个的颜色由Z的值和**colormap** 来决定. 如

↑Example

```
surf(X,Y,Z)
colormap hsv
colorbar
```

↓Example

还可以使用函数**alpha**来控制曲面图的透明度. 比如

↑Example

```
surf(X,Y,Z)
colormap hsv
alpha(0.2)
```

↓Example

3.4 Creating Specialized Plots

- Bar plot: 函数**bar**

  我们看一些例子

↑Example

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

↓Example

以及

↑Example

```
Y = round(rand(5,3)*10);
subplot(2,2,1)
bar(Y,'group')
title 'Group'
subplot(2,2,2)
bar(Y,'stack')
title 'Stack'
```

```
subplot(2,2,3)
barh(Y,'stack')
title 'Stack'
subplot(2,2,4)
bar(Y,1.5)
title 'Width = 1.5'
```

↓Example

❑ Pie Charts: 函数 **pie**

考虑三种产品五年的销售记录

↑Example

```
X = [19.3 22.1 51.6;
     34.2 70.3 82.4;
     61.4 82.9 90.8;
     50.5 54.9 59.1;
     29.4 36.3 47.0];

x = sum(X);  % 求出每列的和
```

```
pie(x)
% 使用函数pie的explode参数，以突出销售量最大的那部分
explode = zeros(size(x));
[c,offset] = max(x);
explode(offset) = 1;
pie(x,explode); colormap summer
```

↓Example

然后可以使用图形工具来添加标识. 也可以使用图形句柄以命令形式添加. 这将在下一讲中学习.

- Histograms: 函数**hist**

↑Example

```
Y = randn(10000,1);
hist(Y)
Y = randn(10000,3);
hist(Y)
```

↓Example

- Discrete Data Graphs: 函数 **stem**, **stairs**

绘制函数$f(t) = e^{-\alpha t} cos(\beta t)$.

↑Example

```
alpha = .02; beta = .5; t = 0:4:200;
y = exp(-alpha*t).*cos(beta*t);
plot(t,y)
stem(t,y) % alternative
xlabel('Time in \mu secs')
ylabel('Magnitude')
t = 0:10;
y = exp(-alpha*t).*cos(beta*t);
stairs(t,y)
hold on
plot(t,y,'--*')
hold off
label = 'Stairstep plot of e^{-(\alpha*t)} cos\beta*t';
text(3,1,label,'FontSize',14)
xlabel('t = 0:10','FontSize',14)
axis([0 10 -1.2 1.2])
```

↓Example

❑ Contour Plots: 函数 **contour**

↑Example

```
[X,Y,Z] = peaks;
[C,h]=contour(X,Y,Z,10)
clabel(C,h)
title({'Contour Labeled Using','clabel(C,h)'})
```

↓Example

3.5 Advanced plotting

函数绘图

数据绘图指令**plot**在绘制函数图形时, 需要先对函数进行分点取值, 因此非常麻烦. 函数 绘图指令**fplot**则可以避免这一点. 语法

fplot('fun',lims,'S',tol)

❖ 用单引号界定的输入参数fun, 是解析函数字符串表达式, 内联函数或者M-函数文件名.

❖ lim规定了绘图区间,用矩阵表示是lim=[a,b,c,d], 表示自变量x和函数y的取值范围分别是$x \in [a,b]$和$y \in [c,d]$.

❖ S表示线条的类型,颜色等.

❖ tol规定函数取值的相对误差, 经常省略, 默认值为2e-3.

例如在同一幅图中绘制函数$y = e^x, y = 3sin(x), y = x^2$三条函数曲线, $x \in [-\pi, \pi]$.

```
fplot('[exp(x),3*sin(x),x^2]',pi*[-1,1,-1,1])
```

↑Example

↓Example

隐函数(Implicit function) 绘图

matlab 提供了函数**ezplot**,**ezmesh**,**ezsurf**, **ezcontour**等函数来绘制隐函数的图形. 例如对隐函数

$$x^2 - y^4 = 0$$

```
ezplot('x^2-y^4')
ezplot('x^2-y^4',[0,10,0,10])
```

↑Example

↓Example

对曲面

$$\begin{aligned} x &= e^{-s}cos(t) \\ y &= e^{-s}sin(t) \quad 0 \le s \le 8, 0 \le t \le 5\pi \\ z &= t \end{aligned}$$

```
ezsurf('exp(-s)*cos(t)','exp(-s)*sin(t)','t',[0,8,0,5*pi])
```

↑Example

↓Example

图形对象与图形句柄

matlab将构成图形的各个基本要素成为图形对象. 这些对象包括计算机屏幕, 图形窗口(figure), 用户菜单(Uimenu),坐标轴(Axes),用户控件(Uicontrol),曲

线(Line),曲面(Surface), 文字(Text),图像(Image),光源(Light),区域块(Patch),方框(Rectangle)等.

Matlab在创建每个对象时, 都会给该对象分配唯一的一个值, 成为图形对象句柄(handle). 可以使用函数**set**和**get**, 以及图形对象的句柄来操作图形对象的各个属性.

set(handle,属性名1, 属性名1值, 属性名2, 属性名2值,...)

如

↑Example

```
x = 1:10;
y = x.^3;
h = plot(x,y);
set(h,'Color','red','LineWidth',3)
```

↓Example

get(handle,属性名)

如果不指定属性名, 则返回所有属性.

↑Example

```
get(h)
get(h,'Color')
get(0,'format')
get(0,'ScreenSize')
```

↓Example

最后一个是查看当前数字显示格式. 0是计算机屏幕这一对象的句柄值.

复杂一些的例子

↑Example

```
[x,y] = meshgrid([-2:.4:2]);
Z = x.*exp(-x.^2-y.^2);
fh = figure('Position',[350 275 400 300],'Color','w');
ah = axes('Color',[.8 .8 .8],'XTick',[-2 -1 0 1 2],...
          'YTick',[-2 -1 0 1 2]);
sh = surface('XData',x,'YData',y,'ZData',Z,...
             'FaceColor',get(ah,'Color')+.1,...
             'EdgeColor','k','Marker','o',...
             'MarkerFaceColor',[.5 1 .85]);
view(3)
```

↓Example

函数**findobj**可以通过查找特定值得到图形对象的句柄值. 比如对下图, 我们想将 文字标识$sin(t) = 0.707$移动到点$[3 * pi/4, sin(3 * pi/4)]$

↑Example

```
x=linspace(0,2*pi);y=sin(x);plot(x,y);
text(pi/4,sin(pi/4),'\leftarrowsin(t) = .707');
text(pi,sin(pi),' \leftarrowsin(t)=0');
text(1.8+pi/4,sin(pi+pi/4),'sin(t)=-.707\rightarrow');

text_handle = findobj('String','\leftarrowsin(t) = .707');
set(text_handle,'Position',[3*pi/4,sin(3*pi/4),0]);
```

↓Example