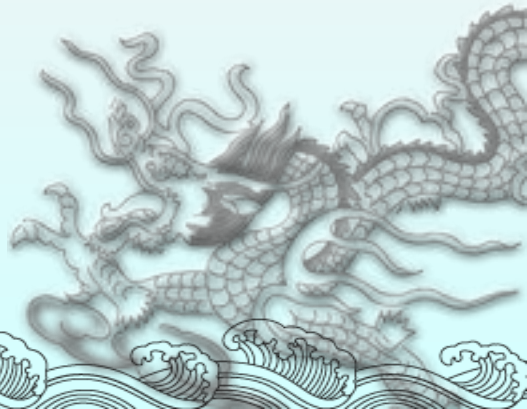
第一章 MATLAB语言概述

1.1 什么是MATLAB

- **MATLAB：MATrix LABoratory。1980年前后，Cleve Moler教授编写的Linpack 和Eispack的接口程序。**

1984年，MATLAB第1版(DOS版)
1992年，MATLAB4.0版
1997年，MATLAB 5.0版
1999年，MATLAB 5.3版
2000年，MATLAB 6.0版
2002年，MATLAB 6.5版
2004年，MATLAB 7.0版
2007年， MATLAB 7.5版
2010年， MATLAB 7.11版
2016年，MATLAB R2016b版



MATLAB

技术计算语言

全球数以百万计的工程师和科学家使用 MATLAB® 来分析和设计可改变世界的系统和产品。MATLAB 广泛应用于汽车主动安全系统、行星际宇宙飞船、健康监控设备、智能电网和 LTE 蜂窝网络。它用于机器学习、信号处理、图像处理、计算机视觉、通讯、计算金融学、控制设计、机器人学等等。

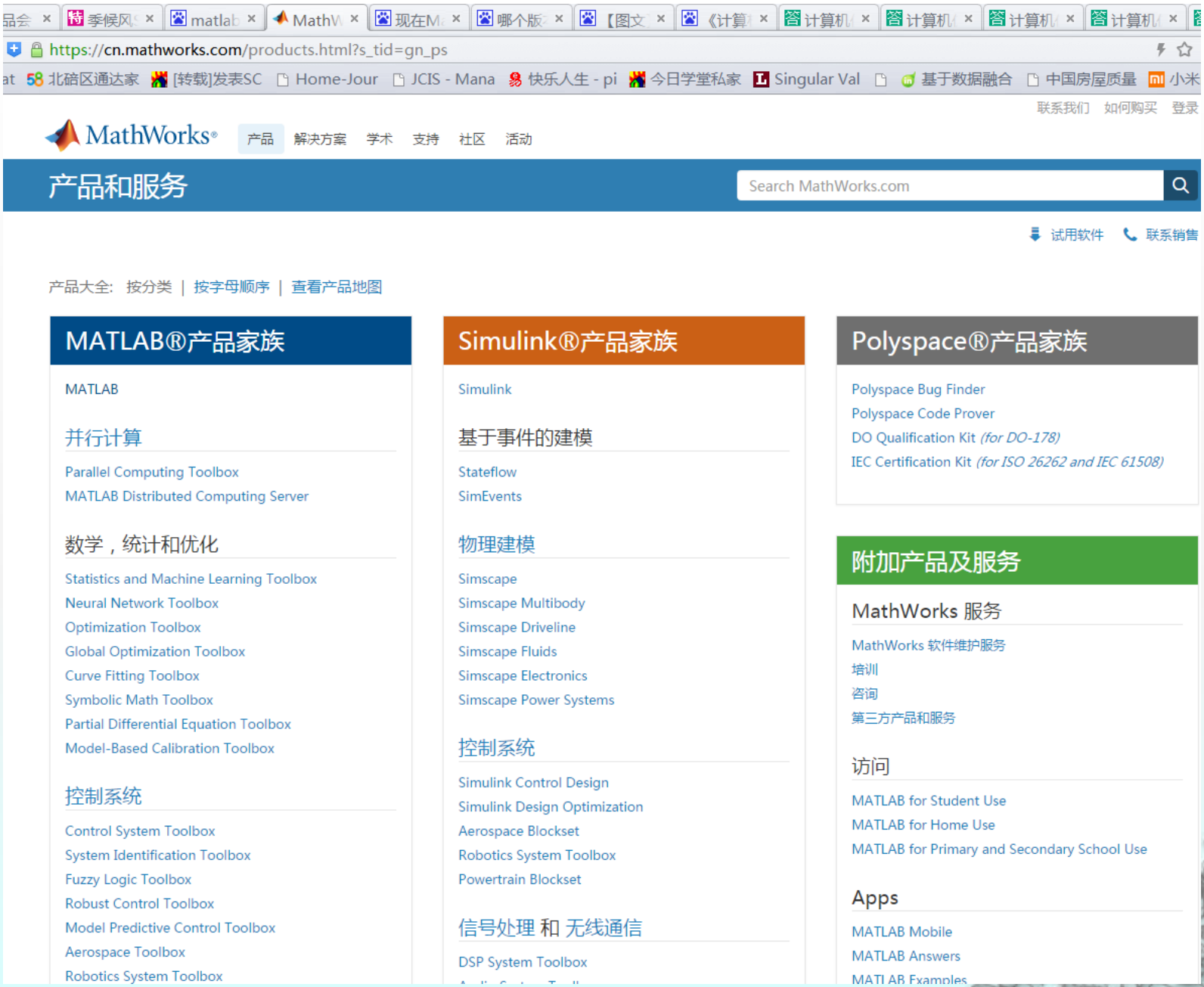
数学。图形。编程。

MATLAB 平台为解决工程和科学问题进行了优化。基于矩阵的 MATLAB 语言是世界上最自然的计算数学表示方法。内置图形使得可视化和洞察数据变得简单易行。大量的预制工具箱库可让您即刻开始使用对您的应用领域至关重要的算 法。桌面环境鼓励试验、探索和发现。这些 MATLAB 工具和功能全部经过严格测试，并为相互协同工作而定制。

扩展。集成。部署。

MATLAB 帮助您让想法超越桌面的限制。您可以对大型数据集运行分析，并扩展到集群和云。MATLAB 代码可以与其他语言集成，从而允许您将算法和应用程序部署在 Web、企业和生产系统内。

了解 MATLAB 的用途。


https://cn.mathworks.com/products.html?s_tid=gn_ps
MathWorks
产品 解决方案 学术 支持 社区 活动
联系我们 如何购买 登录
产品和服务
Search MathWorks.com
试用软件 联系销售
产品大全: 按分类 | 按字母顺序 | 查看产品地图
MATLAB®产品家族
MATLAB
并行计算
Parallel Computing Toolbox
MATLAB Distributed Computing Server
数学，统计和优化
Statistics and Machine Learning Toolbox
Neural Network Toolbox
Optimization Toolbox
Global Optimization Toolbox
Curve Fitting Toolbox
Symbolic Math Toolbox
Partial Differential Equation Toolbox
Model-Based Calibration Toolbox
控制系统
Control System Toolbox
System Identification Toolbox
Fuzzy Logic Toolbox
Robust Control Toolbox
Model Predictive Control Toolbox
Aerospace Toolbox
Robotics System Toolbox
Simulink®产品家族
Simulink
基于事件的建模
Stateflow
SimEvents
物理建模
Simscape
Simscape Multibody
Simscape Driveline
Simscape Fluids
Simscape Electronics
Simscape Power Systems
控制系统
Simulink Control Design
Simulink Design Optimization
Aerospace Blockset
Robotics System Toolbox
Powertrain Blockset
信号处理 和 无线通信
DSP System Toolbox
Polyspace®产品家族
Polyspace Bug Finder
Polyspace Code Prover
DO Qualification Kit (for DO-178)
IEC Certification Kit (for ISO 26262 and IEC 61508)
附加产品及服务
MathWorks 服务
MathWorks 软件维护服务
培训
咨询
第三方产品和服务
访问
MATLAB for Student Use
MATLAB for Home Use
MATLAB for Primary and Secondary School Use
Apps
MATLAB Mobile
MATLAB Answers
MATLAB Examples

试用软件

控制系统

Control System Toolbox
System Identification Toolbox
Fuzzy Logic Toolbox
Robust Control Toolbox
Model Predictive Control Toolbox
Aerospace Toolbox
Robotics System Toolbox

信号处理 和 无线通信

Signal Processing Toolbox
DSP System Toolbox
Audio System Toolbox
Communications System Toolbox
Wavelet Toolbox
RF Toolbox
Antenna Toolbox
Phased Array System Toolbox
LTE System Toolbox
WLAN System Toolbox

图像处理与计算机视觉

Image Processing Toolbox
Computer Vision System Toolbox
Vision HDL Toolbox
Image Acquisition Toolbox
Mapping Toolbox

测试和测量

Data Acquisition Toolbox
Instrument Control Toolbox
Image Acquisition Toolbox

Simulink Design Optimization
Aerospace Blockset
Robotics System Toolbox
Powertrain Blockset

信号处理 和 无线通信

DSP System Toolbox
Audio System Toolbox
Communications System Toolbox
Phased Array System Toolbox
SimRF
Computer Vision System Toolbox

代码生成

Simulink Coder
Embedded Coder
HDL Coder
Vision HDL Toolbox
Simulink PLC Coder
Fixed-Point Designer
DO Qualification Kit *(for DO-178)*
IEC Certification Kit *(for ISO 26262 and IEC 61508)*

实时仿真和测试

Simulink Real-Time
Simulink Desktop Real-Time

确认、验证和测试

Simulink Verification and Validation
Simulink Design Verifier
Simulink Test

MATLAB for Student Use
MATLAB for Home Use
MATLAB for Primary and Secondary School Us

Apps

MATLAB Mobile
MATLAB Answers
MATLAB Examples
MATLAB Academy
Hardware Support Packages
ThingSpeak
Cody
Cody Coursework
File Exchange

R2016b

MATLAB, Simulink和其它83种产品的相关更新

» 了解更多
马上下载

» 发现如何解决您的计算问题

Matlab 语言是一种以矩阵运算为基础的交互式程序语言。集成度高，实用方便，输入简洁，运算高效，内容丰富。

特点：

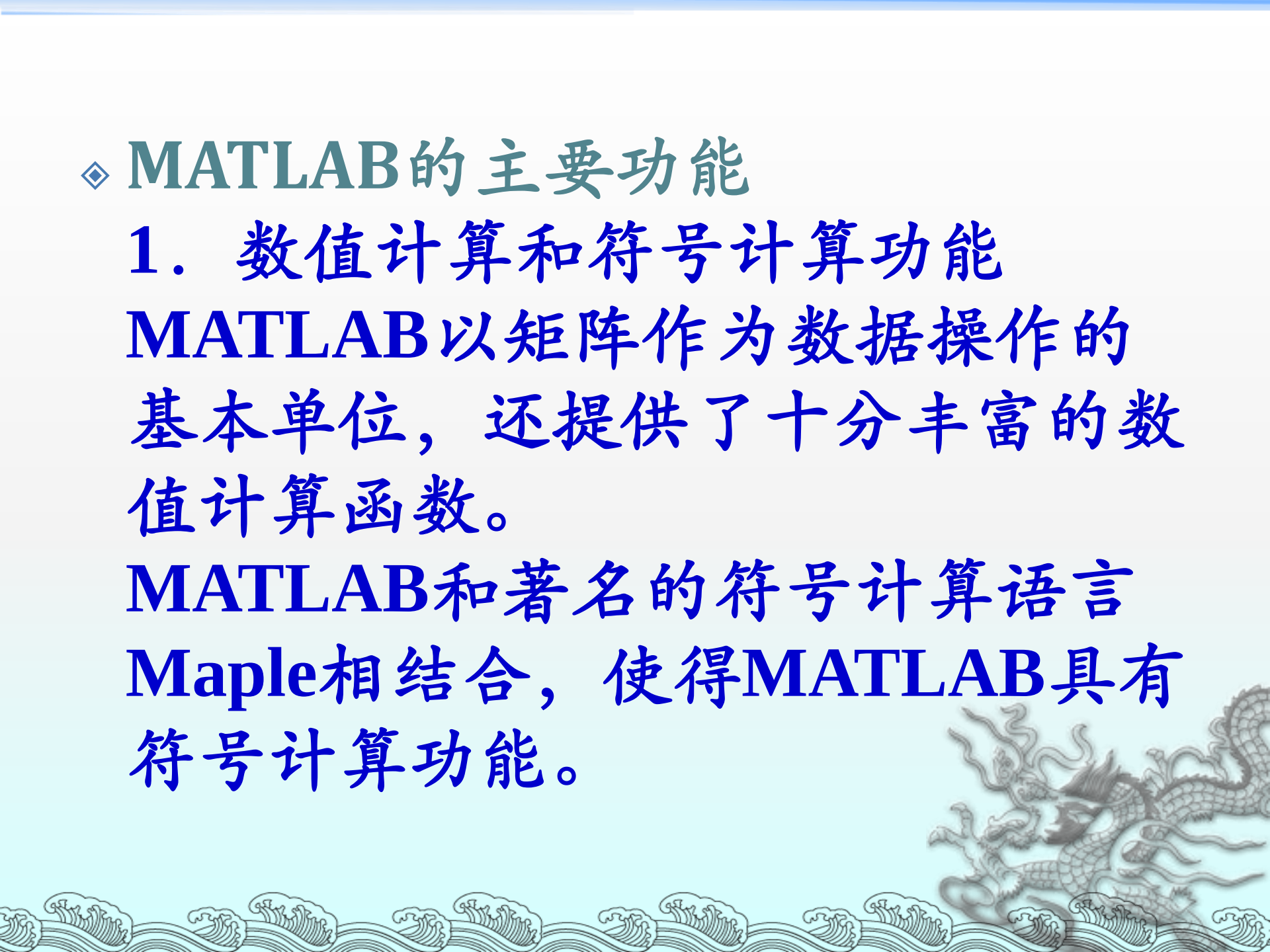
1. Matlab是一种解释性语言：输入算式即得结果，无编译
2. 变量的多功能性
3. 运算符号的多功能性
4. 语言规则与笔算式相似
5. 强大而又简易的作图功能
6. 智能程度高
7. 功能丰富，可扩展性强：基础部分+扩展部分

- **MATLAB的主要功能**

  1．数值计算和符号计算功能

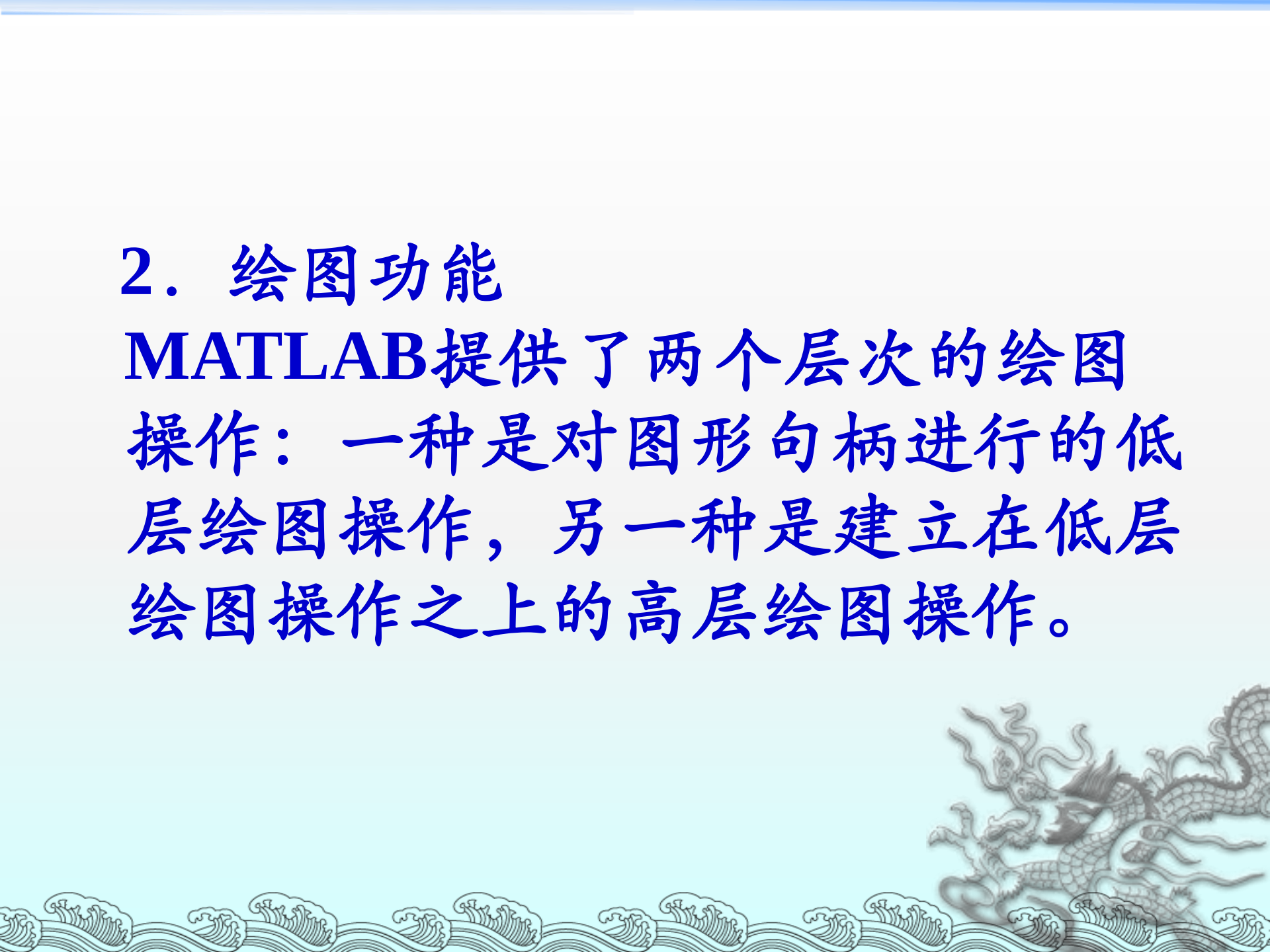
  MATLAB以矩阵作为数据操作的基本单位，还提供了十分丰富的数值计算函数。

  MATLAB和著名的符号计算语言Maple相结合，使得MATLAB具有符号计算功能。



2．绘图功能

MATLAB提供了两个层次的绘图操作：一种是对图形句柄进行的低层绘图操作，另一种是建立在低层绘图操作之上的高层绘图操作。

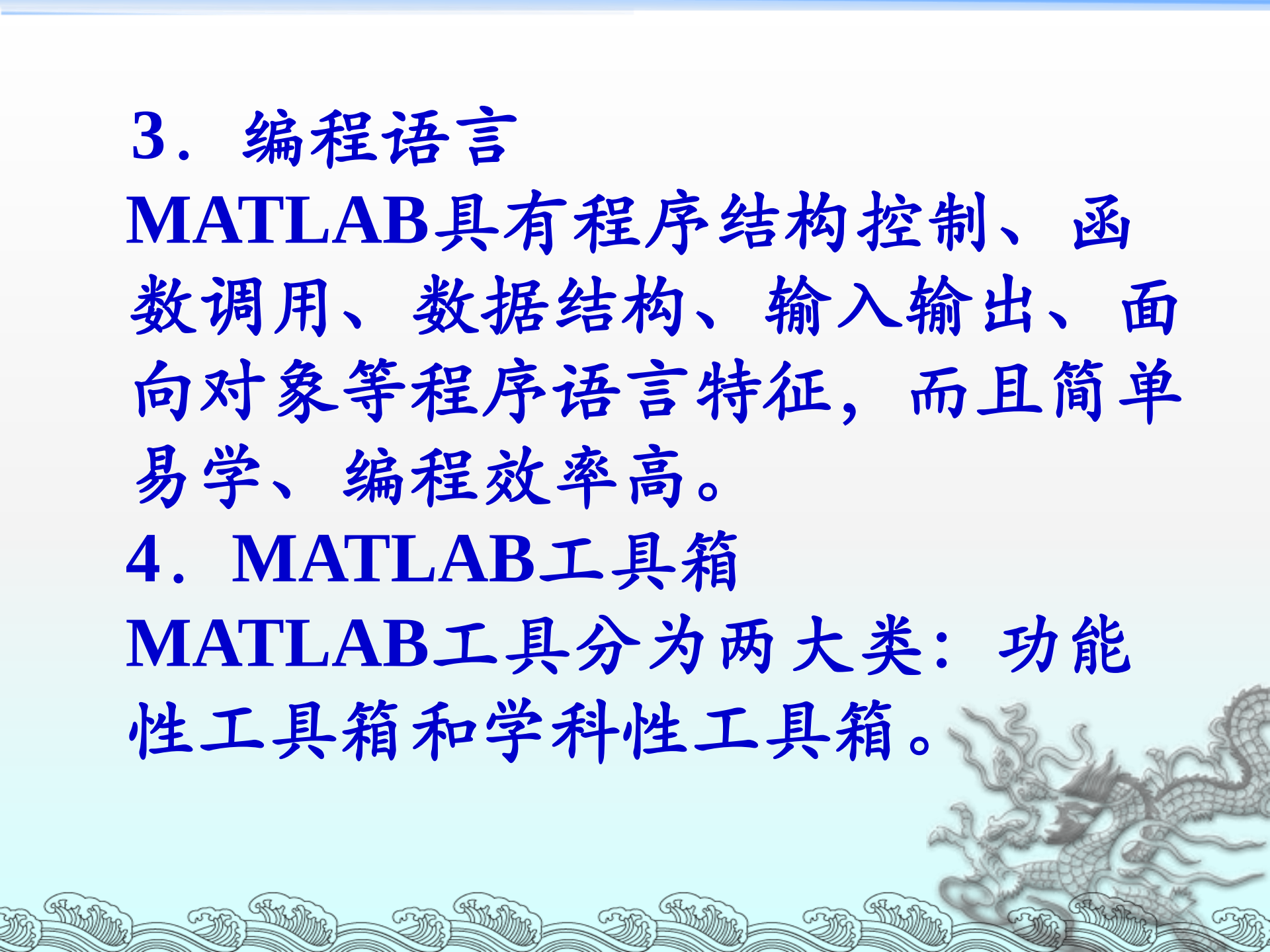


3．编程语言

MATLAB具有程序结构控制、函数调用、数据结构、输入输出、面向对象等程序语言特征，而且简单易学、编程效率高。

4．MATLAB工具箱

MATLAB工具分为两大类：功能性工具箱和学科性工具箱。



- 例如，用一个简单命令求解如下线性系统：

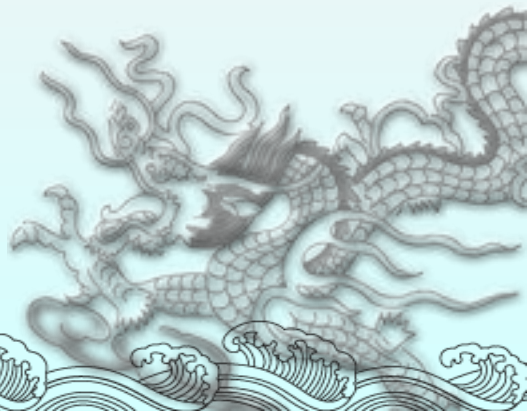
$$3x1+ x2 - x3 = 3.6$$
$$x1+2x2+4x3 = 2.1$$
$$-x1+4x2+5x3 = -1.4$$

- 在MATLAB命令窗口输入：

```
A=[3 1 -1;1 2 4;-1 4 5]; b=[3.6;2.1;-1.4];
x=A\b
```

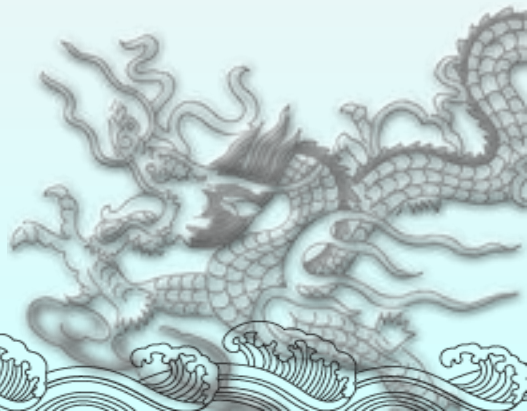
- 运行后的结果为：

```
x =1.4818
   -0.4606
   0.3848
```



例如，用简短命令计算并绘制在$0 \le x \le 6$范围内的$\sin(2x)$、$\sin(x^2)$、$(\sin(x))^2$

- 在MATLAB命令窗口输入：
- x=linspace(0,6)
- y1=sin(2*x),y2=sin(x.^2),y3=(sin(x)).^2;
- plot(x,y1,x, y2,x, y3)



运行命令语句得到图形如图1.2所示

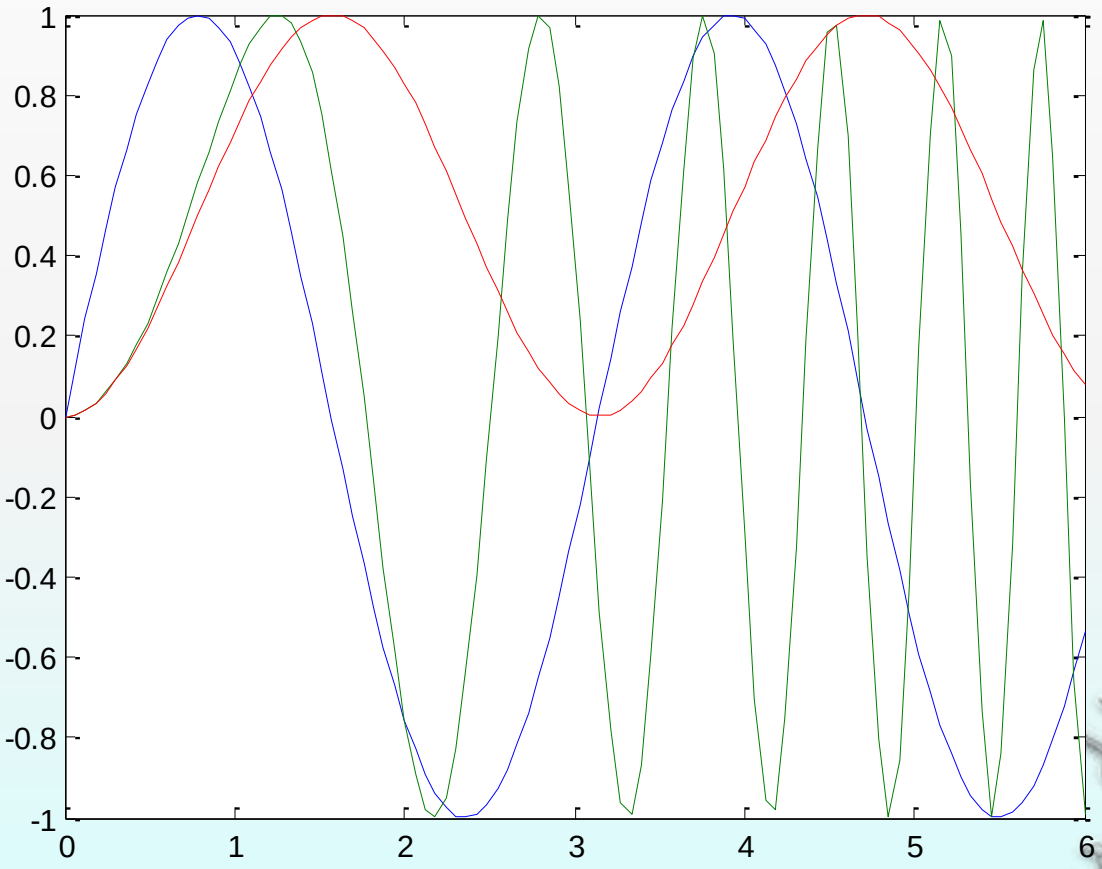


图1.2 函数sin(2x)，sin（x^2），$(\sin(x))^2$的图形

第二章 MATLAB 的基本语法

第2章 MATLAB的基本语法

2.1 变量及其赋值

2.1.1 标识符与数据格式

标识符是标志变量名、常量名、函数名和文件名的字符串的总称。在MATLAB中，变量和常量的标识符最长允许19个字符。字符包括全部的英文字母（大小写52个）、阿拉伯数字和下划线等符号，标识符中第一个字符必须是英文字母。

2.1.2 矩阵及其元素的赋值

赋值就是把数赋予代表常量或变量的标识符。在MATLAB中，变量都代表矩阵。列矢量可被当作只有一列的矩阵；行矢量也可被当作只有一个行的矩阵；标量应看作1×1阶的矩阵。赋值语句的一般形式为：

变量＝表达式（或数）

1．赋值要求

在输入矩阵时，应遵循以下规则：

- 整个矩阵的值应放在方括号中；
- 同一行中各元素之间以逗号“，”或空格分开；
- 不同行的元素以分号“；”隔开。

2．变量的元素的标注

在MATLAB中，变量的元素用圆括号“（　）”中的数字（也称为下标）来注明，一维矩阵（也称数组）中的元素用一个下标表示，二维矩阵由两个下标数构成，以逗号分开，对三维矩阵则由三个下标数构成。

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

a =

     1     2     3
     4     5     6
     7     8     9

>> a(1,3)

ans =

     3
```

```
>> size(a)

ans =

     3     3
```

```
>> a(4,4)=7

a =

     1     2     3     0
     4     5     6     0
     7     8     9     0
     0     0     0     7
```

```
>> size(a)

ans =

     4     4
```

3．赋值的技巧，冒号(:)

功能：产生数组、向量的下标或用于迭代。

说明：

冒号是MATLAB中最常用的操作符之一，它可用于建立向量、下标阵列和迭代，其格式如表2.3所示。利用冒号可从向量、矩阵和高维阵列中选取指定的行和列，其格式如表2.4所示。

表2.3 冒号使用格式(1)

格　式	功　能
j:k	等同于[j，j+1，…，k]
j:k	当 j>k 时为空
j:i:k	等同于[j，j+i，j+2i，…，k]
j:i:k	当 i>0 且 j>k，或者 i<0 且 j<k 时为空

表2.4 冒号使用格式(2)

格　式	功　　能
A(:，j)	取 A 的第 j 列
A(i，:)	取 A 的第 i 行
A(:，:)	等效于二维阵列，对矩阵而言，它等同于 A
A(j:k)	取出 A(j)，A(j+1)，…，A(k)元素
A(:，j:k)	取出 A 的从第 j 列到第 k 列的元素，即取出 A(:，j)，A(:，j+1)，…，A(:，k)
A(:，:，k)	取出三维阵列 A 的第 k 页
A(i，j，k，:)	取出四维阵列 A 中的向量，向量由 A(i，j，k，1)，A(i，j，k，2)，A(i，j，k，3)等元素组成
A(:)	将 A 的所有元素排成列向量

3．赋值技巧

(1) 利用冒号“：”给全行的元素赋值， t=j:i:k

(2) 利用行、列标注构成新的矩阵

```
>> b=1:2:10

b =

     1     3     5     7     9

>> c=1:10

c =

     1     2     3     4     5     6     7     8     9    10
```

利用冒号给全行的元素赋值

```
>> a

a =

     1     2     3     0
     4     5     6     0
     7     8     9     0
     0     0     0     7

>> a(4,:)=[5,3,2,1]

a =

     1     2     3     0
     4     5     6     0
     7     8     9     0
     5     3     2     1
```

利用行、列标注构成新的矩阵

```
a =

     1     2     3     0
     4     5     6     0
     7     8     9     0
     5     3     2     1

>> b

b =

     1     3     5     7     9

>> b=a([2,4],[1,3])

b =

     4     6
     5     2
```

4. 特殊矩阵和数组

基本矩阵和阵列	
eye	建立单位矩阵
ones	建立全 **1** 阵列
zeros	建立全 **0** 阵列
rand	建立均匀分布的随机数和阵列
randn	建立正态分布的随机数和阵列
linspace	建立线性间空向量
logspace	建立对数间空向量

(1)单位矩阵函数eye()

函数功能：产生对主角线元素为1，其它元素为0的单位矩阵。

eye()的调用格式如下：

A=eye(n)　　返回一个n×n阶单位矩阵；

A=eye(m , n)　返回一个m×n阶单位矩阵，或用A=eye([m , n])；

(2) zeros函数、ones函数、rand以及randn函数

A=zeros(n)　　返回一个n×n阶零矩阵；

A=zeros(m , n)　　返回一个m×n阶零矩阵；

(3) linspace函数和logspace函数

linspace函数的调用格式如下：

y=linspace(a,b) 产生一个行矢量y，该矢量把a和b间的数等分100份而得到。

y=linspace(a,b,n) 产生一个行矢量y，该矢量把a和b间的数等分n份而得到。

5. Matlab 内部特殊变量和常数

续表

特殊变量和常数	
ans	列出变量最近的值
pi	圆周率 π(=3.14159…)
i,j	虚数单位
NaN	非数值
Inf	无穷大数
realmax	最大的正浮点数
realmin	最小的正浮点数
nargin,nargout	函数变量数
varargin,varargout	传递或返回可变的变量数
eps	浮点数相对精度
computer	识别运行 MATLAB 的计算机
inputname	输入变量名
flops	统计浮点运算次数
时间和日期	

5. Matlab 内部特殊变量和常数

⑴变量ans：临时变量，通常指示当前的答案。

⑵常数eps：表示浮点相对精度；其值是从1.0到下一个最大浮点数之间的差值。变量值作为一些MATLAB函数计算的相对浮点精度，按IEEE标准，esp= 2^{-52}近似为2.2204e-016。

⑶常数realmax：表示最大正浮点数；任何大于该值的运算都溢出。在具有IEEE标准浮点格式的机器上，realmax略小于21024 ，近似为1.7977e+308。

⑷常数realmin：表示最小正浮点数；任何小于该值的运算都溢出。在具有IEEE标准浮点格式的机器上，realmin略小于2^{-1024}，近似为2.2251e-308。

⑸常数pi：表示圆周率π =3.1415926535897.....。

⑹常数Inf：代表正无穷大，一般被0除或溢出则产生无穷大结果。如2/0，2^10000均产生结果：Inf；而log(0)产生结果：-Inf。

⑺虚数单位i，j：表示复数虚部单位，相当于$\sqrt{-1}$。

⑻NaN：表示非数值。如当Inf-Inf，Inf/Inf， 0*Inf，0/0均产生该结果。

6．复数的赋值方式

MATLAB的每一个元素都可以是复数，实数是复数的特例。复数的虚数部分用i或j表示。对复数矩阵有两种赋值方法：

(1) 可将矩阵元逐个赋予复数

```
>> z=1+2i

z =

   1.0000 + 2.0000i

>> x=2+4*i

x =

   2.0000 + 4.0000i
```

```
>> z=[1+2i,3+4i;5+6i,7+8i]

z =

   1.0000 + 2.0000i   3.0000 + 4.0000i
   5.0000 + 6.0000i   7.0000 + 8.0000i
```

```
>> z=[1+2*i,3+4*i;5+6*i,7+8*i]

z =

   1.0000 + 2.0000i   3.0000 + 4.0000i
   5.0000 + 6.0000i   7.0000 + 8.0000i
```

(2) 将矩阵的实部和虚部分别赋值：

```
>> x=[1,3;5,7]+[2,4;6,8]*i

x =

   1.0000 + 2.0000i   3.0000 + 4.0000i
   5.0000 + 6.0000i   7.0000 + 8.0000i
```

```
>> i=2

i =

     2

>> x=[1,3;5,7]+[2,4;6,8]*i

x =

     5    11
    17    23
```

7．变量检查

在程序调试或变量的赋值过程，往往需要检查工作空间中的变量、变量的阶数以及变量赋值内容。在检查变量及其阶数等内容时，既可用工作空间窗口，也可在命令窗口使用**who**或**whos**命令来完成检查。当查看某变量的赋值情况，可在命令窗口直接键入该变量名回车即可。

```
>> who

Your variables are:

i   x   z

>> whos
  Name        Size                Bytes  Class     Attributes

  i           1x1                     8  double
  x           2x2                    32  double
  z           2x2                    64  double    complex
```

2.2 运算符与复数运算

算术运算符：进行数值计算。 **关系运算符：**比较两个操作数的大小。

逻辑运算符：进行逻辑运算。

算术操作符	
+ –* /\ ^ ' **kron** **:**	矩阵和阵列的算术运算 **Kronecker** 张量积 建立向量、阵列的下标或用于迭代
特殊字符	
(){}=' **. ... , ; % !**	特殊字符
关系操作符	
<> <= >= == ~=	关系操作运算
逻辑操作符	
& | ~ **xor**	逻辑操作运算 异或操作

1. 算术运算符+-* / \ ^ '

功能：矩阵和阵列的算术运算。

格式：

A+B　A-B　A*B　A.*B

A/B　A./B　A\B　A.\B

A^B　A.^B　A'　A.'

说明：

MATLAB定义了两种不同的算术运算：矩阵和阵列算术运算。矩阵算术运算由线性代数规则来定义，而阵列算术运算是元素对元素的运算，用句点来区分这两种运算。由于对加法、减法而言，这两种运算是相同的，因此不必使用 .+ 和 .-。

（1）A+B，A-B是最简单的运算；A，B应该具有相同的尺寸，当然如果其中之一为标量也是可以的，这时标量被看作是与另一个具有相同尺寸的等元素矩阵。

（2） C=A*B完成矩阵A、B的线性代数积，即

$$C(i,j)=\sum_{k=1}^{n}A(i,*)B(k,j)$$

当A，B中任一个为标量时，直接将它乘到另一个矩阵的每个元素中。

（3）C=A.*B是A，B对应元素相乘，即

$$C(i，j)=A(i，j)B(i，j)$$

（4）A/B完成矩阵右除，即它相当于A*inv(B)。

（5）A\B完成矩阵左除，即它相当于inv(A)*B 。

(6) 矩阵元素右除“A./B”与左除“A.\B”

矩阵元素右除“A./B”表示矩阵元素A(i,j)/B(i,j)；矩阵元素左除“A./B”表示矩阵B(i,j)/A(i,j)，因此，A和B必须大小相同，或者其中之一为标量。

(7) 矩阵幂“ ^ ”：X ^ p

如果p为标量，表示X的p次幂；如果X为标量，而p为矩阵，X ^ p用特征值和特征特征向量表示X的矩阵p次幂。输入必须为标量和方阵。

(8) 矩阵元素幂“. ^ ”：A. ^ B

A. ^ B表示矩阵元素A（i,j）的B(i,j)次幂，A与B必须大小相同，或者其中之一为标量。

(9) 矩阵转置“ ’ ”

A’表示矩阵A的线性代数转置。对于复矩阵，表示复共轭转置。

(10) 非共轭转置“ .’ ”

A.’表示非共轭转置；对于复矩阵，不包括共轭。

(11)冒号操作符“ : ”

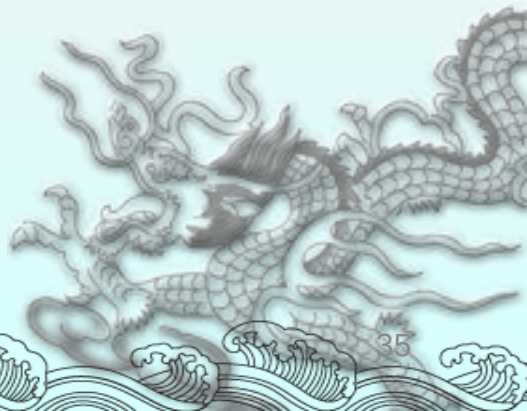
冒号是一个非常有用的操作符；可以产生向量、数组下标以及for循环。

4．特殊字符[]()\{\} = '．　…, ；　% ！

功能：特殊字符。

格式：

[]()\{\} = '．　…, ；　% ！



说明：

(1) [ ] 用于形成向量和矩阵，例如[6.9 2.5 0]为向量，[1 2 3；4 5 6]为矩阵，其中分号用于结束第一行。[]内还可以采用矩阵和向量，例如只要A、B、C的维数适当，就可利用D=[A B；C]产生更大的矩阵。

A=[]表示产生空矩阵A，A(m, :)=[]表示从A中删去第m行，A(:，n)=[]表示从A中删去第n列

[A1，A2，A3，…]=function_name(…)表示利用指定函数产生多个矩阵变量A1,A2，A3…。

(2) {}(花括号)用于单元阵列的赋值，例如A(2,1)={[1 2 3; 4 5 6]}，A{2, 2}=('str')。

(3) ()(括号)通常用于一般的算术表达式，指示优先运算；它还用于表示函数变量、向量下标和矩阵下标等。

- 例如X(3)表示X的第三个元素，X([1 2 3])为X的前三个元素。
- 如果X有n个元素，则X(n:-1:1)可将它们反序输出；
- 当V有m个元素，W有n个元素时，则A(V，W)可形成m×n矩阵，

```
V=[1 2 3]
```

```
W=[1 2]
```

```
a =

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

```
>> a(V,W)

ans =

     1     2
     4     5
     7     8
```

(4) = (等号)用于表示赋值，如B=A表示将A的元素赋给B。

(5) == (双等号)表示关系相等操作符，用于关系表达式。

(6) ' (撇)表示矩阵转置。X'表示X的复共轭转置，X.'表示X的非共轭转置。

(7) ' ' (引号)用于表示字符向量，其值为相应字符的ASCII码。

(8) . (点)可表示小数点，如3.14；. (点)还可以表示元素对元素运算，它与其它算术运算符结合，构成阵列运算操作符；另外，. (点)还可以表示域访问，例如A为一种结构，field为A的域，则可使用A.field和A(i).field来访问。

(9) .. (两个点)在cd命令中用于表示父目录。

(10) ... (三个点)用于行末表示续行。

(11) ， (逗号)用于分隔矩阵下标和函数变量，也用于分隔多语句行中的语句。

(12) ； (分号)在方括号内用于指示行末，在语句或表达式后使用，表示抑制输出结果。

(13) %(百分号)表示注释信息，指示逻辑行的结束，在%之后的任意文本都作为说明。

(14) ！(感叹号)指示其后的内容为操作系统命令。

2.2.2 关系操作符

关系运算是指两个元素之间数值的比较。MATLAB所提供的关系操作符如表所示。

关系操作符	<	<=	>	>=	==	~=
说明	小于	小于或等于	大于	大于或等于	等于	不等于

关系比较结果只有两种可能，即1或0。1表示关系式这“真”，即关系式正确；0表示该关系为“假”，即它不成立。

2.2.3 逻辑操作符

逻辑操作符	功能描述
&	与：当两个操作数为真时，结果为真，其它为假。
|	或：当两个操作数至少有一个为真时，结果为真。
~	非：这是一个单目运算符，它只有一个操作数。操作数为真，结果为假；操作数为假，结果为真。

通常逻辑变量只能取0(假)和1(真)两个值。逻辑量的基本运算除“与(&)”、“或(|)”和“非(~)”外，有时也包括“异或(xor)”，不过“异或”可以用3种基本运算组合而成。两个逻辑量经过这4种逻辑运算后的输出仍然是逻辑量。

5．关系操作符< > <= >= == ~=

功能：关系操作运算。

格式：

A>B　　A>=B

A<B　　A<=B

A==B　　A~=B

说明：

关系操作符可完成两个阵列之间元素对元素的比较，其结果为同维数的阵列。当关系成立时相应的元素置为逻辑真(1)，否则置为逻辑假(0)。

操作符<、<=、>、>=只用于操作数的实部比较，而==，~=用于比较实部和虚部。

关系操作符的优先级介于逻辑操作符和算术操作符之间。

6．逻辑操作符 & | ~

功能：逻辑操作运算。

格式：

A&B　　A|B　　~A

说明：

&、|、~分别表示逻辑与、或、非，它们都是按元操作的。0表示逻辑假(F)，任何非零值表示逻辑真(T)。

逻辑操作的优先级最低，因此要优先计算算术和关系操作符。逻辑操作之间的优先次序为非(~)优先级最高，与(&)和或(|)优先级相同。

7．xor

功能：异或操作。

格式：

C=xor(A,B)

说明：

C=xor(A,B)完成阵列A和B对应的元素的异或操作。例如：

```
a=[0  0  pi  eps];
b=[0 -2  4  1.2];
c=xor(a, b)
c=
 0   1   0   0
```

2.2.4 其它关系与逻辑函数

xor(x,y)	异或运算。x或y非零(真)返回1，x和y都是零(假)或都是非零(真)返回0。
any(x)	如果在一个向量x中，任何元素是非零，返回1；矩阵x中的每一列有非零元素，返回1。
all(x)	如果在一个向量x中，所有元素非零，返回1；矩阵x中的每一列所有元素非零，返回1。

```
>> a=[0.53 0.47 0.81 0.3 -2.1 -0.9]

a =

    0.5300    0.4700    0.8100    0.3000   -2.1000   -0.9000

>> b=a>0.5

b =

     1     0     1     0     0     0

>> all(a>0.5)

ans =

     0
```

- 参考书目：
- 计算机仿真——基于matlab的电子信息类课程（第三版）

2.3 流 程 控 制

流程控制语句可改变程序执行的流程，MATLAB的流程控制语句有以下四类：

- if，else，elseif，end构成**条件转移**语句。
- switch，case，otherwise，end构成**情况切换**语句。
- while，end构成**不定次重复的循环**语句。
- for，end构成**指定次重复的循环**语句。

2.3.1 条件语句

最简单的条件语句是仅由if和end组成的语句，它可根据逻辑表达式的值选择是否执行。例如：

```
a=input('a=?'); %通过键盘对变量a赋值
if  rem(a,2)==0；%判断所给变量a被2除的余数是否为0
   disp('a is even') ；%显示
   b=a/2;
end
```

这一段程序完成当a为偶数时，b=a/2；否则不作任何处理。执行end后面的语句。

if语句可嵌套使用，多层嵌套可完成复杂的设计任务。

三个分支

```
If 表达式 1
    语句组 A
    elseif 表达式2
    语句组B
else
    语句组C
end
```

◆ 首先判断表达式1，若为真，执行A；

利用else和elseif可进一步给出条件，从而构成复杂的条件语句。

执行完后跳出该选择结构，继续执行end后面的语句；

◆ 如果表达式1为假，则跳过A，判断2，如果为真，则执行B；

◆ 如果表达式2为假，则执行C

```
n=input(‘n=?’);
if  n<0
        disp('n is negative.');
elseif  rem(n,2)==0
       disp('n is even ')
else
    disp(‘n is odd);
end
```

当逻辑表达式不是标量时，则只有当矩阵的所有值为非零时，条件才满足，因此如果X为矩阵，则

```
if  X
  statements
end
```

等效于

```
if  all(X(:))
  statements
end
```

当逻辑表达式为空阵列时，则表示为FALSE，例如，当A为空阵列，则语句

```
if  A
    Statements1
else
    Statements2
end
```

执行Statements2。

2.3.2 情况切换语句 (Switch)

switch语句可根据表达式的不同取值执行不同的语句，这相当于多条if语句的嵌套使用。例如，根据varl变量的取值{−1，0，1}，分别执行相应的语句，可输入

```
switch varl
  case -1
    disp('varl is negative one.')
  case 0
    disp('varl is zero.')
  case 1
    disp('varl is positive one.')
  otherwise
    disp('varl is other value.')
end
```

但这里只有当**varl=-1、0、1**时才执行相应的语句，而所有其它的值都执行**otherwise**中的语句。在**if**语句中，我们可设定＞、＜、≥、≤这样的关系，但**switch**中只能采用相等的关系，这一点是两者的区别。

在switch的case语句中还可以采用多个数值，例如：

```
switch var2
  case {-2, -1}
     disp('var2 is negative one or two.')
  case 0
     disp('var2 is zero.')
  case {1,2,3}
     disp('var2 is positive one, two, or three.')
  otherwise
      disp('var2 is othervalue.')
end
```

这时，当**var2=-2**或**var2=-1**时，均执行第一个**case**之后的语句。

2.3.3 指定次重复循环语句 (for)

for语句可完成指定次重复的循环，这是广泛应用的语句。

```
for index=初值：增量：终值
        语句组 A
end
```

例如，为求n!，我们可循环n次，每次求出k!=(k-1)!×k。MATLAB程序为:

```
r=1;
for k=1:20
   r=r*k;
end
disp(r)
```

执行后结果为(20!)

```
2.4329e+18
```

使用for语句需要注意：

1. 不能通过在循环体内重新赋值给循环变量来终止for循环，只能使用break语句。

例如：

```
for n=1:7
    x(n)=sin(n*pi/10);
    n=7;  (无效)
end
```

改为：

```
for n=1:7
    x(n)=n+2;
    if n==5
        break;    (当执行到5的时候跳出来)
    end
end
```

使用for语句需要注意：

2. 为提高处理的速度，for 循环（while）被执行前，应预先分配数组。

例如：

```
x=zeros (1,10)
for n=1:10
    x(n)=sin(n*pi/10);
end
```

否则，每一次循环要花费时间对x分配更多的内存。

使用for语句需要注意：

3. 当有一个等效的数组方法来解答给定的问题时，应避免for循环。

例如前面的例子可以改写为：

```
>> n=1:10;
>> x=n+2;
```

执行更快！

4. For 循环可以按需嵌套，从而构成多重循环。

例如，利用rand函数产生10个随机数，然后利用嵌套for循环进行从大到小排序。

```
x=fix(100*rand(1,10)) ;  %向零靠拢
disp(x)
n=length(x);
for i=1:n−1
    for j=n: −1:i+1
    if x(j)>x(j-1)
       y=x(j); x(j)=x(j-1); x(j-1)=y;
    end
  end
end
disp(x)
```

执行后得到排序前和排序后的结果：

19 68 30 54 15 69 37 86 85 59

86 85 69 68 59 54 37 30 19 15

for循环中可利用break语句来终止for循环，如上例中加上交换标志(flag)，当一次内循环中没有找到一个单元需要交换时，说明排序工作已经结束，从而可以结束外循环。MATLAB程序为：

```
x=fix(100*rand(1,10)) ;
disp(x)
n=length(x);
```

```
for i=1:n-1
  flag=-1
  for j=n: -1:i+1
    if x(j)>x(j-1)
    y=x(j) ; x(j)=x(j-1); x(j-1)=y; flag=0 ;
    end
  end
  if flag, break, end
end
disp(x), disp(['循环次数为', num2str(i)])
```

执行后得

79　95　52　88　17　97　27　25　87　73

97　95　88　87　79　73　52　27　25　17

循环次数为6，这说明完成这10个数的排序只进行了6次内循环。

2.3.4 不定次重复循环语句 （while）

while语句可完成不定次重复的循环，它与for语句不同，每次循环前要判别其条件，如果条件为真或非零值，则继续循环，否则结束循环。当条件是一表达式时，其值必定会受到循环语句的影响。

基本调用格式：

```
while  表达式
       语句组A
end
```

例如，利用while 语句实现计算累加：

```
i=0;   %对累加器i变量清零
 while i<10;
      i=i+1      %计数累加
end
```

2.4.1 交互输入/输出命令

◆ Input函数是MATLAB中用于输入参数的常用函数，它可自带提示信息，例如：

```
f=input('frequency is')
```

执行时显示

```
frequency is
```

这时可输入频率值f。又如，要求输入方法的选择：

```
m=input('methods\n1---linear\n2---bilinear\n3---others\n')
```

执行时显示：

methods

1 --- linear

2 --- bilinear

3 --- others

这时可输入方法选择(1，2或3)。

当直接输入字符串变量时, 则应在input中指定's'项，如

m=input('methods:','s')

执行时显示

methods:

这时用户可直接输入方法的名称，如输入bilinear，这时m为字符串变量“bilinear”。

◆ 键盘控制

一般情况下，我们可利用debug命令对M函数文件进行调试，然而，利用keyboard函数也可以进行简单的调试。

在M文件的适当位置加上keyboard命令，MATLAB执行该命令时，会将控制权交给键盘，这时用户可检查当前局部工作空间中变量的内容，也可对变量值进行修改，或者直接输入新的变量(可使用MATLAB的任何命令建立)。利用return命令可退出键盘控制状态，MATLAB继续执行后续程序。

◆菜单输入 menu

利用menu函数可显示出输入菜单，用户只需利用鼠标点击菜单中的按钮，就可以完成输入操作。当然，输入的值为菜单选项的序号，因此编写程序时应加以变换。

格式：k=menu(‘title’,‘选项1’，‘选项2’，‘选项3’)

例如，要输入颜色的字符串变量scolor，它可取red、green、blue、yellow和black，则可输入

```
s=menu('color selection','red','green','blue','yellow','black')
switch s
    case 1, scolor='red';
    case 2, scolor='green';
    case 3, scolor='blue';
    case 4, scolor='yellow';
    case 5, scolor='black';
    otherwise disp('Error!')
  end
scolor
```

执行时可显示出如图2.2所示的菜单，假设在菜单中选择第4个按钮(yellow)，则可得到

```
scolor=
        yellow
```

显示命令 disp

调用格式：disp(变量名)

按格式要求输出变量命令sprintf

调用格式：sprint (显示格式，变量)

与C语言的含义一样

例：>>sprintf(‘%d’, round(pi))

ans=3

>>sprintf(‘%s’, ‘hello’)

ans= hello