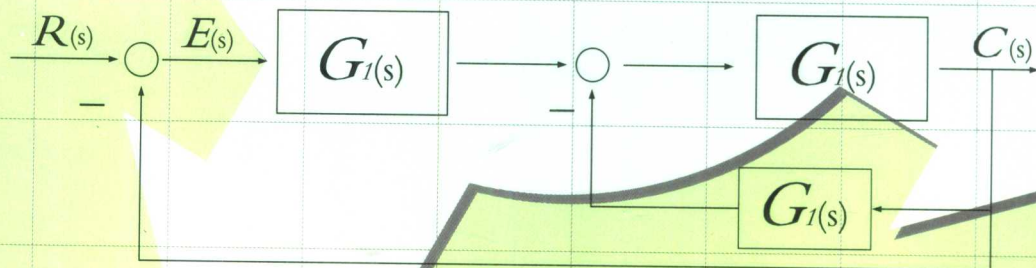


自动控制原理

实验指导

李秋红 叶志锋 徐爱民 编著



国防工业出版社

National Defense Industry Press

自动控制原理实验指导

李秋红 叶志锋 徐爱民 编著

国防工业出版社

· 北京 ·

图书在版编目(CIP)数据

自动控制原理实验指导/李秋红,叶志锋,徐爱民编
著. —北京:国防工业出版社,2007.5
ISBN 978-7-118-05094-3

I. 自… II. ①李…②叶…③徐… III. ①自动控制
理论—高等学校—教学参考资料②自动控制—实验—高
等学校—教学参考资料 IV. TP13

中国版本图书馆 CIP 数据核字(2007)第 040560 号

※

国防工业出版社 出版发行

(北京市海淀区紫竹院南路 23 号 邮政编码 100044)

北京市李史山胶印厂印刷

新华书店经售

*

开本 787×1092 1/16 印张 11 字数 274 千字

2007 年 5 月第 1 版第 1 次印刷 印数 1—4000 册 定价 24.00 元

(本书如有印装错误,我社负责调换)

国防书店:(010)68428422

发行邮购:(010)68414474

发行传真:(010)68411535

发行业务:(010)68472764

前 言

自动控制理论已经广泛应用于工业、农业、交通、航空、航天等各个领域,极大地提高了社会劳动生产率,改善了人们的生产、生活条件,丰富和提高了人民的生活水平。在现代社会生活中,自动化设备无所不在,自动控制理论已经与人们的生活密不可分。随着电子计算机的出现及其性能的日益提高,许多控制领域的复杂问题,都可以借助计算机来解决。特别是 1984 年 Mathworks 公司推出了高性能的数值计算和可视化软件——Matlab 后,使许多控制领域的问题可以借助 Matlab 轻松解决。

众所周知,Matlab 在控制界之所以能成为最受欢迎的语言,除了其自身在科学运算与可视化方面的优势外,最大的特点还在于其开放性。正因为这个原因,许多控制界的专家编写出各种各样的工具箱,这些工具箱现已成为各自领域的标准。

众多理工院校的机电类专业都把《自动控制原理》作为其必修课程之一。在多年的教学中,作者发现,广大学生最关注的问题就是:书本上学到的理论知识是否有用?如何应用?从自动控制理论教材来看,主要讲述基本理论和方法,学生在学习过程中着眼于理论分析。而在实际解决工程问题时,大多数都是借助 Matlab 这个工具来完成的,所以有必要在讲授课程的同时,锻炼学生应用所学知识、借助 Matlab 解决控制系统问题的能力。本书从这一点出发,紧密结合《自动控制原理》教材,介绍了如何借助 Matlab,应用所学理论知识和方法,进行基本的控制系统分析和设计。

本书区别于其他应用 Matlab 解决控制系统问题的书籍的重要之处在于:本书和《自动控制原理》教材结合紧密,涵盖了胡寿松主编的《自动控制原理》所有基础内容的 Matlab 求解,以大量实例演示基本控制理论和方法的应用,使读者能深刻理解和掌握所学的理论知识,同时提高实际动手解决问题的能力。

本书共分 10 章。第一章介绍了 Matlab 应用的基本知识,包括后面用到的应用 Matlab 进行绘图的方法;第二章介绍了线性系统的数学模型,包括 Laplace 变换、结构图和信号流图的基本知识;第三章介绍了线性系统的时域分析方法,包括典型输入信号作用下系统响应的计算,系统动态和静态性能指标的计算;第四章介绍了线性系统的根轨迹的绘制和分析方法;第五章介绍了线性系统的频域分析,包括频率特性的物理意义、幅相曲线和对数频率特性曲线的绘制及稳定裕度的计算;第六章介绍了线性系统的校正方法,包括超前校正、滞后校正、超前-滞后校正、反馈校正及应用 Simulink 进行 PID 控制器设计的方法;第七章介绍了线性离散系统的分析与校正方法;第八章介绍了非线性控制系统分析方

法,包括相平面分析法和描述函数分析法;第九章介绍了状态空间形式数学模型的计算与分析方法,包括线性变换及可控性和可观测性等;第十章简单介绍了线性二次型最优控制系统的设计问题。

本书第一章至第六章由李秋红编写,第七章和第八章由徐爱民编写,第九章和第十章由叶志锋编写。

由于时间仓促,作者的学识水平有限,疏漏之处在所难免,欢迎广大读者给予批评和指正。

作者

目 录

第一章 Matlab 简介	1
1.1 Matlab 应用领域	1
1.2 基本 Matlab 操作	1
1.3 Matlab 编程基础	4
1.3.1 常量与变量	4
1.3.2 矩阵的输入方法	5
1.3.3 Matlab 基本运算	10
1.3.4 控制语句	11
1.4 图形功能	12
习题	15
第二章 线性系统的数学模型	17
2.1 Laplace 变换	17
2.1.1 Laplace 变换基本知识	17
2.1.2 应用 Matlab 求解 Laplace 变换与反变换	18
2.2 数学模型	22
2.2.1 数学模型基本知识	22
2.2.2 用 Matlab 表示数学模型	23
2.3 结构图简化	24
2.4 信号流图	29
习题	31
第三章 线性系统的时域分析	33
3.1 系统动态性能	33
3.1.1 系统动态性能基本知识	33
3.1.2 系统动态性能的 Matlab 求解	34
3.2 线性系统的稳定性与稳态误差	45
3.2.1 稳定性与稳态误差的基本知识	45
3.2.2 应用 Matlab 判断系统的稳定性	46
3.2.3 稳态误差的计算	50
习题	53
第四章 线性系统的根轨迹分析	56
4.1 根轨迹的基本知识	56
4.2 二阶系统的解析根轨迹图	57
4.3 根轨迹绘制与分析	58

习题	65
第五章 线性系统的频域分析	66
5.1 频率特性物理意义	66
5.2 Nyquist 曲线与 Nyquist 稳定性判据	72
5.3 Bode 图与稳定裕度	73
习题	79
第六章 系统校正	80
6.1 校正基本知识	80
6.2 串联超前校正	81
6.2.1 无源超前校正网络	81
6.2.2 超前校正装置设计	82
6.3 串联滞后校正	85
6.3.1 无源滞后校正网络	85
6.3.2 滞后校正装置设计	86
6.4 串联滞后—超前校正	88
6.4.1 无源滞后—超前校正网络	88
6.4.2 无源滞后—超前校正装置设计	89
6.5 测速—超前网络反馈校正	92
6.6 PID 控制	94
6.6.1 PID 控制原理	95
6.6.2 应用 Simulink 设置 PID 控制器参数	98
习题	102
第七章 线性离散系统的分析与校正	103
7.1 线性离散系统的基本概念	103
7.2 z 变换和 z 反变换	103
7.3 离散系统的数学模型	106
7.4 离散系统的稳定性分析	113
7.5 离散系统的动态性能分析	114
习题	117
第八章 非线性控制系统分析	119
8.1 非线性控制系统概述	119
8.2 相平面法	119
8.2.1 相平面法的基本概念	119
8.2.2 基于 Matlab/Simlink 的相平面法分析	120
8.3 描述函数法	123
8.3.1 描述函数法的基本概念	123
8.3.2 基于 Matlab 的描述函数法分析	125
习题	128
第九章 线性系统的状态空间分析	130

9.1 状态空间模型的线性变换及其标准形	130
9.1.1 状态空间表达式的线性变换	130
9.1.2 非奇异线性变换的不变特性	130
9.1.3 几种常用的线性变换	132
9.2 线性系统的可控性、可观性判定	135
9.2.1 线性系统的可控性	135
9.2.2 应用 Matlab 求解系统的可控性	136
9.2.3 线性系统的可观性及 Matlab 求解	140
9.2.4 连续系统的完全可控标准形和完全可观测标准形	142
9.2.5 在 Matlab 中求系统的可控标准形和可观测标准形	144
9.3 系统的反馈结构及状态观测器	146
9.3.1 系统输出反馈	146
9.3.2 系统的状态反馈与闭环系统的状态矩阵	147
9.3.3 系统的极点配置	147
9.3.4 系统的状态观测器	150
9.3.5 系统的状态降维观测器	153
9.4 李亚普诺夫稳定性	156
习题	160
第十章 线性二次型问题最优控制	162
10.1 最优控制基本概念	162
10.2 线性二次型问题的最优控制	163
习题	165
参考文献	166

第一章 Matlab 简介

1.1 Matlab 应用领域

Matlab 名字由 MATrix 和 LABoratory 的前 3 个字母组合而成。它是 Mathworks 公司于 1984 年推向市场的一套高性能的数值计算和可视化软件。它集数值分析、矩阵运算、信号处理和图形显示于一体。经过二十几年的发展，Matlab 已成为一个国际公认的最优秀的科技应用软件，其强大的扩展功能更是为各个工程领域提供了分析和设计的基础，成为国际控制界应用最广的首选计算机软件。同时也已成为国内外高等院校自动控制原理、高等数学、数值分析、信号处理及工程应用等课程的基本教学工具。

1.2 基本 Matlab 操作

在 Matlab 安装完成以后，一般会在桌面或快捷启动栏生成快捷键 ，最简单的打开方式就是双击该快捷键，也可使用你熟知的其他方式打开。Matlab 打开以后出现一个 Flash 界面，默认的情况下出现如图 1-1 所示的 Matlab 操作界面。在这个界面上平铺着 3 个最常用的窗口：指令窗口（Command Window）、历史指令窗口（Command History）、工作空间窗口（Workspace）。当前路径窗口（Current Directory）作为一个交互界面隐藏在工作空间窗口后面，只能看见窗口名，在这个窗口中列出了当前路径中的所有文件和文件夹，包括文件的类型和最后修改时间等信息。同时在快捷工具栏中也有一个“Current Directory”下拉列表框，其中列出了已经使用过的路径，用于当前路径的选

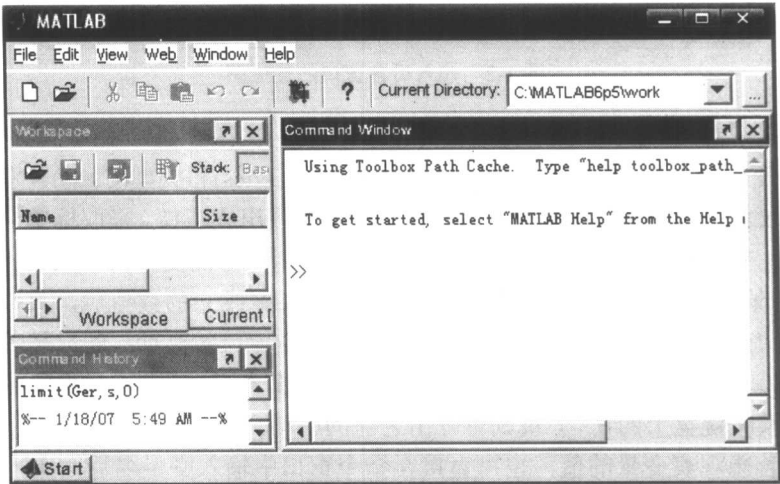


图 1-1 Matlab 操作界面

择。若希望在新的路径下进行工作，可以单击列表框后的“...”来设置。在窗口的左下角还有一个【Start】按钮，其作用和主菜单【View】的下拉子菜单【Launch Pad】相近。下面对各主要交互界面的功能加以简单介绍。

指令窗口默认处于 Matlab 界面的右侧，是进行各种 Matlab 指令操作的最主要窗口。在该窗口中，可输入各种 Matlab 指令、函数、表达式、变量等，并显示除图形外的所有运算结果。

历史指令窗口默认情况下处于 Matlab 界面的左下方。该窗口记录在“指令窗口”中已经运作过的指令、函数。

工作空间窗口记录着当前内存中的所有的变量。

例如在指令窗口输入：

```
x = 2 ;  
y = 3 ;  
z = x + y ;
```

Matlab 界面如图 1-2 所示。

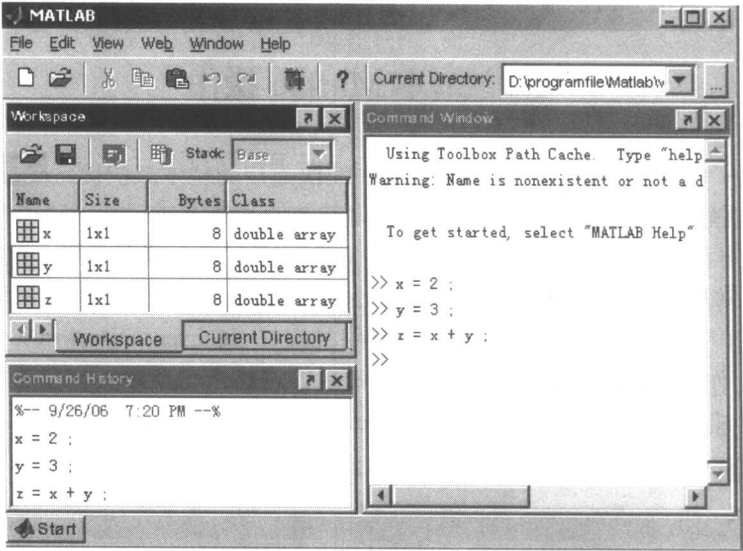


图 1-2 输入后各窗口的变化图

从图 1-2 中看出，在指令窗口的输入全部出现在历史指令窗口中，同时在工作空间窗口中出现了变量的名字(Name)、维数(Size)、字节数(Bytes)和类型(Class)。双击工作空间窗口中变量 x 和 y 所在的行，则出现如图 1-3 所示的弹出式窗口。采取标签页的形式显示出变量的值。Workspace 即为工作空间，是 Matlab 用来存储变量的一块内存空间。当 x 或 y 的数值改变，需重新计算 z 的数值时，可在输入 x 或 y 的数值后，直接双击历史指令窗口中的“z = x + y;”，则这一语句将出现在指令窗口并执行；若只想呼出这一语句而不想执行，或想修改后再执行，可按键盘上的【↑】，则可呼出指令窗口中最后输入的语句；若连续按键盘上的【↑】键则可呼出之前的语句，则可对呼出的语句进行修改。

同样，若想观察变量的值，也可直接在指令窗口中输入变量名后回车，注意：变量名后不能加分号，否则变量的数值将不会在窗口中显示出来。当然这个指令也会记录在

历史指令窗口中。指令窗口中输入“clc”可以清除指令窗口中的内容；输入“clear”可以清除工作空间窗口中的信息；输入“clf”可以清除当前活动图形窗口中的内容。在程序中输入这些指令也可以起到相同的作用。同样你也可以选择【Edit】菜单，则在出现的下拉菜单中，有【Clear Command Window】、【Clear Command History】、【Clear Workspace】等菜单项，分别执行清理指令窗口、清理历史指令和清空工作空间的功能。

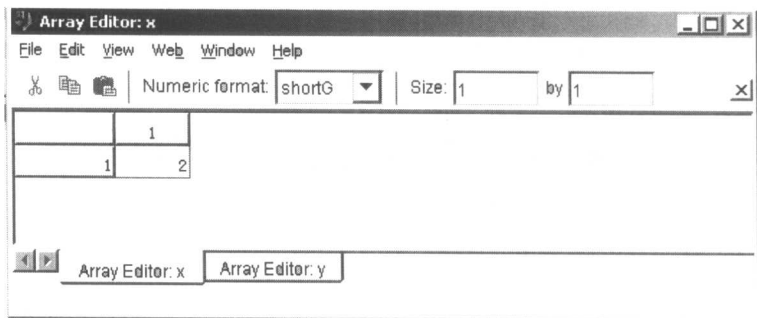


图 1-3 弹出式窗口图

对于简单的命令可以直接在指令窗口中输入相应的指令，但对于具有一定功能的较长的程序，直接输在指令窗口不便于以后的使用和修改，所以建议把它保存在文件中。只要单击图 1-1 中快捷工具栏中的“”图标，或者选择菜单项中的【File】→【New】→【M-File】，即可弹出程序编辑窗口，如图 1-4 所示。

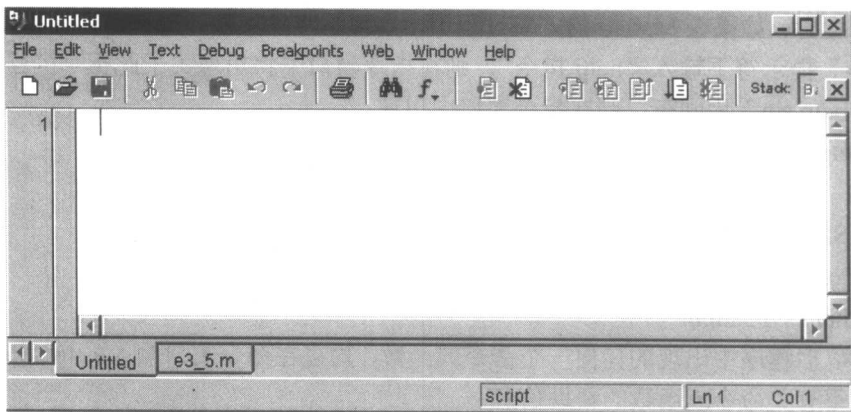


图 1-4 程序编辑窗口

可直接在图 1-4 所示窗口中编辑程序。对于同时打开的多个程序，采取标签页的形式进行程序之间的切换。未保存的文件以 Untitled 标识。文件的命名和变量一致，文件名的第一个字符必须是字母，其余的可由字母、数字和下划线组成，不能包含“#”、“*”等特殊符号，也不能使用表 1-1 列出的特殊字符。在编辑窗口的左下角的状态栏中，有提示信息，Ln 1、Col 1，表明当前光标所在的位置为第 1 行(Line)，第 1 列(Column)。程序的保存和其他基于 Windows 的窗口一样，在不特别指定扩展名的情况下，默认的扩展名为“.m”。程序的运行可以单击快捷工具栏中的“”图标，或按快捷键【F5】，也可以通过菜单项中的【Debug】→【Run】来执行程序，若第 1 次运行时程序未保存，会提示保存，以后每次运行时会自动保存程序。若程序不是保存在当前路径下，运行时会出现如

图 1-5 所示的操作提示信息, 包括: 改变当前路径; 将路径加到 Matlab 搜索路径的顶部; 将路径加到 Matlab 搜索路径的底部。通常选择“改变当前路径”, 使 Matlab 的当前路径变化到文件所在路径。程序的运行结果或出错信息会在指令窗口中显示。若想观察程序中某一变量的值, 只需将给变量赋值的语句后的分号去掉, 则变量的值将在指令窗口中显示; 同样也可在程序运行完成后, 直接在指令窗口输入变量名后, 回车, 即可得到其结果显示; 也可双击工作空间窗口中的变量名, 打开变量编辑窗口, 里面有当前变量的值, 此时可以改变当前变量的值, 在后面的操作中若不重新赋值, 则变量为改变后的数值。

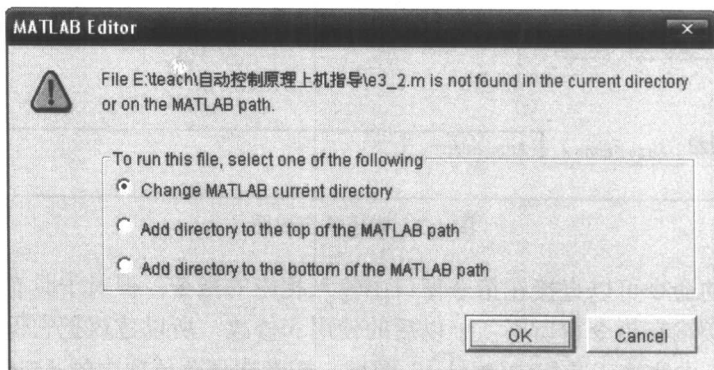


图 1-5 运行提示窗口

特别提醒: 如果在 Matlab 的安装过程中, 安装了帮助文件, 则只需在 Matlab 指令窗口中“>>”提示符下输入“help 指令”, 则在指令窗口中将出现关于此指令的帮助信息。如键入“help sin”后回车, 则在指令窗口中出现:

```
>> help sin
SIN      Sine.
        SIN(X) is the sine of the elements of X.
Overloaded methods
        help sym/sin.m
```

对于以后程序中出现的任何一个命令或函数, 均可在命令窗口输入 help 和命令或函数名来获得有关命令或函数的帮助信息。

1.3 Matlab 编程基础

1.3.1 常量与变量

Matlab 中使用的数据有常量与变量。常量有实数常量、复数常量和字符常量, 复数由实部和虚部组成。实数和复数常量可以用传统的十进制计数法表示, 也可以使用科学计数法表示。字符常量用单引号括起来, 如‘Good’, 字符常量也可用中文, 如‘输出结果为:’但其单引号必须在英文输入法下输入, 否则会报错。

Matlab 定义了两个复数变量: i 和 j。它们的值均为 $\sqrt{-1}$ 。如果 i 和 j 不重新赋值, 则

保留这种定义；如果想把变量 a 变成 $\sqrt{-1}$ ，则用 $a = \text{sqrt}(-1)$ 。

在 Matlab 中，虚数单位 $i = \text{sqrt}(-1)$ ，在工作空间内显示的值为：

$i = 0 + 1.0000i$

复数常量的生成可以利用如下语句：

$z = a + bi$;

或： $z = r * \exp(\theta * i)$ ，其中 r 是复数的模， θ 是复数幅角的弧度数。

2.3×10^5 在 Matlab 中用 “ $2.3 * 1e5$ ” 表示。 $1e8$ 则代表 10^8 。

在 Matlab 中，所有常量数值在内存的存储格式都是使用 IEEE 标准的 16 位长型格式，其数值的有效范围是 $10^{-308} \sim 10^{+308}$ 。

Matlab 里的变量无须事先定义。一种程序中的变量，以其名称在语句命令中第一次合法出现而定义。Matlab 是大小写敏感的语言，例如， Abc 和 ABC 是不同的变量。

Matlab 中设置了一些特殊的变量与常量，如表 1-1 所列。

表 1-1 Matlab 的特殊变量与常量

变 量 名	功 能 说 明
ANS (或 ans)	默认变量名，以应答最近一次操作运算结果
i 或 j	虚数单位，定义为 $\sqrt{-1}$
pi	圆周率
eps	浮点的相对误差
realmax	最大的正实数
realmin	最小的正实数
INF(或 inf)	代表无穷大
NaN (或 nan)	代表不定值 (即 0/0)
nargin	函数实际输入参数个数
nargout	函数实际输出参数个数

Matlab 中还可以设置一种特殊的变量——全局变量，只要在该变量前添加 Matlab 的关键字 “global” 就可将该变量设定为全局变量。全局变量必须在使用前声明，即这个声明必须放在主程序的首行。全局变量即作用在主程序里，又作用在调用的子程序里，用于数值的传递。

1.3.2 矩阵的输入方法

Matlab 最初是作为矩阵运算而开发的软件，Matlab 以复数矩阵为基本运算单元，既可以对它进行整体性处理，也可以对它的某个或某些元素进行处理。矩阵有多种输入方法：

1) 利用函数生成。

$\text{ones}(n, m)$ 生成 $n \times m$ 矩阵，矩阵元素全部为 1，若默认为 m ，则生成 $n \times n$ 的元素

全 1 矩阵;

`zeros( n, m)` 生成一 $n \times m$ 矩阵, 矩阵元素全部为 0, 若默认为 m , 则生成 $n \times n$ 的元素全 0 矩阵;

`eye( n, m)` 生成一 $n \times m$ 矩阵, 矩阵对角元素全部为 1, 若默认为 m , 则生成 $n \times n$ 的对角元素全 1 矩阵;

`rand( n, m)` 生成一 $n \times m$ 矩阵, 矩阵元素为 0~1 之间的随机数, 若默认为 m , 则生成 $n \times n$ 的随机矩阵;

`diag( X )`, X 为一向量, 函数返回以 X 中向量为对角元素的方阵。

例 1-1 在 Matlab 命令窗口中输入以下语句, 观察其输出。

```
A = ones( 2 )
B = zeros( 2 )
C = rand( 2 )
D = eye( 2 )
E = ones( 2, 3)
F = zeros(2, 3)
G = rand(2, 3)
H = eye( 2, 3)
I = diag( [ 1 2 3 ])
```

则输出结果为:

```
A =
    1    1
    1    1

B =
    0    0
    0    0

C =
    0.6154    0.9218
    0.7919    0.7382

D =
    1    0
    0    1

E =
    1    1    1
    1    1    1

F =
    0    0    0
    0    0    0

G =
    0.1763    0.9355    0.4103
```

```

0.4057    0.9169    0.8936
H =
     1     0     0
     0     1     0
I =
     1     0     0
     0     2     0
     0     0     3

```

2) 直接在表达式中列出元素。

下面是常用的几种方式。

例 1-2 在 Matlab 中输入下列矩阵。

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad B = [10 \quad 20 \quad 30], \quad C = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad D = [1 \quad 3 \quad 5 \quad 7 \quad 9]$$

输入:

```
A = [ 1, 2; 3, 4 ], B = [10 20 30], C = [1; 2; 3]
```

```
D = 1 : 2 : 10
```

输出:

```
A =
```

```

     1     2
     3     4

```

```
B =
```

```
10    20    30
```

```
C =
```

```

     1
     2
     3

```

```
D =
```

```
1     3     5     7     9
```

说明: 矩阵 D 在赋值时意为从 1 到 10, 每隔 2 取一个数值。若间隔为 1 时可省略 1 和冒号, 即为: $D = 1 : 10$, 间隔可为小数。从上例中可以看出, 当直接列出表达式元素给矩阵赋值时, 元素用中括号括起来, 用分号表示矩阵的换行, 逗号或空格作为矩阵元素之间的分隔符。

3) 建立在文件中。

如果矩阵过大, 可以用文件来保存, 文件可命名为 `x.m`、`x.txt`、`x.dat` 等。其中 `x` 为矩阵名, “.” 后面的为文件的扩展名。对于需要用 Matlab 进行处理的数据也可以保存成相应的数据文件。可用 `load` 指令加载数据。假设采用 C++ 语言生成了一数据文件, 命名为 `out.txt`, 则可以用 `load` 命令将其作为矩阵装入到 Matlab 中。`out.txt` 文件内容如图 1-6

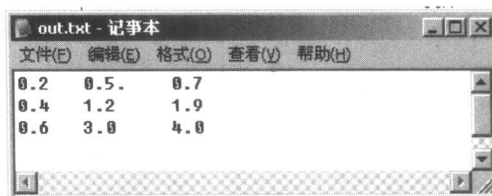


图 1-6 out.txt 数据文件内容

所示，使用指令：

```
load out.txt
```

变量 out 会出现在工作窗口，如图 1-7 所示。out 成为一个数组元素已知的数组。数组中元素即为 out.txt 文件中的数据。双击工作空间窗口中的 out，或在指令窗口输入 out 后回车，均可检查变量 out 的值。

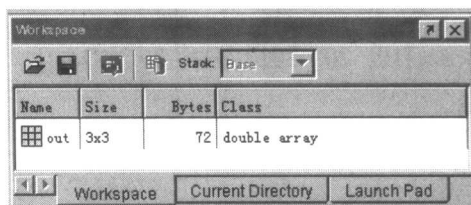


图 1-7 Workspace 窗口

注意：out.txt 文件中不能有字母和变量，只能是数据，数据之间用空格分开，也可以用逗号分隔，但要按照矩阵的行、列摆放。

4) 将小矩阵扩充成大矩阵，或用“:”从大矩阵中提取出小矩阵。

例 1-3 对于上面的 out 矩阵，演示矩阵的扩充和截取、保存。

程序名：e1_3.m

输入：

```
clear
```

```
clc
```

```
load out.txt
```

```
out1 = [ out ; [1.2 4.0 5.0] ]
```

```
out2 = out ( 1 : 2 , 2 : 3 )
```

```
out3 = out ( : , 2 )
```

```
out4 = out ( 3 , : )
```

```
out5 = [ out ( 2 , 1 : 2 ) 3 4 ; out ( 3 , 1 : 2 ) 4 5 ]
```

```
outx = [ 3 4 ; 4 5 ];
```

```
out6 = [ out ( 2 : 3 , 1 : 2 ) ; outx ]
```

```
out7 = [ out ( 2 : 3 , 1 : 2 ) zeros(2); zeros(2) outx ]
```

```
save out_1.txt out1 -ascii
```

```
save out_2.txt out2 -ascii
```

```
save out_34.txt out3 out4 -ascii
```

输出：

```

out1 =
    0.2000    0.5000    0.7000
    0.4000    1.2000    1.9000
    0.6000    3.0000    4.0000
    1.2000    4.0000    5.0000
out2 =
    0.5000    0.7000
    1.2000    1.9000
out3 =
    0.5000
    1.2000
    3.0000
out4 =
    0.6000    3.0000    4.0000
out5 =
    0.4000    1.2000    3.0000    4.0000
    0.6000    3.0000    4.0000    5.0000
out6 =
    0.4000    1.2000
    0.6000    3.0000
    3.0000    4.0000
    4.0000    5.0000
out7=
    0.4000    1.2000         0         0
    0.6000    3.0000         0         0
         0         0    3.0000    4.0000
         0         0    4.0000    5.0000

```

由例 1-3 可见，矩阵扩充时，原矩阵作为新矩阵的一个组成部分，被包含在中括号内，如 out1；取原矩阵的一部分作为新矩阵时，直接用小括号和下标来进行截取，如 out2、out3、out4。矩阵中 out(m,n) 指代矩阵中第 m 行第 n 列的元素，行标和列标与 C 语言不同，均从 1 开始。行标或列标用 “i:j” 表示时，则取 i 到 j 行或列的元素。若行标或列标用 “:” 表示时，则表明取所有的行或列。如 out2 即由数组 out 中的 1 到 2 行的 2 到 3 列组成。out3 为数组 out 第 2 列的所有行，out4 为数组 out 第 3 行的所有列。而 out5 是取原 out 矩阵的一部分：第 2、3 行的第 1、2 列，并在新矩阵 out5 的第 1 行中增加元素 3 和 4，第 2 行中增加元素 4 和 5。在 out6 矩阵中，先将要增加的数据赋值在 outx 矩阵中。矩阵 out7 将 outx 矩阵和取出的矩阵组成对角阵，其余元素为零。函数 zeros(k)，返回 k×k 的零矩阵。输出的同时当前路径下生成了 out_1.txt、out_2.txt、out_34.txt 文件。其中 out_34.txt 部分内容如图 1-8 所示。由图 1-8 可见，保存到文件中的数据均以科学计数法表示，数据后面的黑色方块代表换行符。