第 1 章　MATLAB 概述

MATLAB 的名称源自 Matrix Laboratory，它是 MathWorks 公司开发的一种科学计算软件。MATLAB 将高性能的数值计算、符号计算和可视化集成在一起，并提供了大量的内置函数，从而被广泛地应用到科学计算、控制系统、信息处理等领域的分析、仿真和设计工作中。MATLAB 是国际公认的优秀数学应用软件之一。

本章介绍 MATLAB 的基础知识，包括 MATLAB 的发展历程、语言特点及 MATLAB 7.0 的新产品和新功能，帮助读者对 MATLAB 有一个整体的了解。

1.1　MATLAB 的发展历程

20 世纪 70 年代中期，数值计算成为工程技术和科研的有效手段之一。Cleve Moler 博士及其同事共同开发了基于 Fortran 语言的 LINPACK 和 EISPACK 函数库，以支持数值计算。这两个函数库代表了当时矩阵计算的最高水平，并为广大科技及工程人员所广泛使用。20 世纪 70 年代末期，Cleve Moler 博士发现在使用这两个函数库的过程中大量时间都被放在了接口程序设计上，为了减少工作量，Cleve Moler 亲自编写了 LINPACK 和 EISPACK 函数库的接口程序，并以 MATLAB 作为该接口程序的名字。

20 世纪 80 年代初期，Cleve Moler 与 John Little 等利用 C 语言开发了新一代的 MATLAB 语言，包括编译解释程序、图形功能的设计以及各类数学分析的子模块，并撰写了用户指南。1984 年，为了推广 MATLAB 在数值计算中的应用，Cleve Moler 与 John Little 等正式成立了 MathWorks 公司，从而把 MATLAB 推向市场，并开始了对 MATLAB 工具箱等的开发设计。

1993 年，MathWorks 公司推出了 MATLAB 4.0 版本，1995 年推出 MATLAB 4.2C 版本，1997 年推出了 MATLAB 5.x 版本（Release 11），2000 年推出 MATLAB 6 版本（Release 12），2003 年推出 MATLAB 6.5 版本（Release 13），最新版本是 2004 年 7 月推出的 7.0 版本（Release 14）。

新的版本集中了日常数学处理中的各种功能，包括高效的数值计算、矩阵运算、信号处理和图形生成等功能。在 MATLAB 环境下，用户可以集成地进行程序设计、数值计算、图形绘制、输入/输出、文件管理等各项操作。MATLAB 提供了一个人—机交互的数学系统环境。该系统的基本数据结构是矩阵，在生成矩阵对象时，不要求明确定义矩阵的维数。与利用 C 语言或 Fortran 语言进行数值计算的程序设计相比，利用 MATLAB 可以节省大量的编程时间。在美国的一些大学里，MATLAB 正在成为对数值线性代数，以及其他一些高等应用数学课程进行辅助教学的有益工具。在工程技术界，MATLAB 也被用来解决一些实际课题和数学模型问题。由于 MATLAB 的开放性和可移植性，自第 1 版发行以来，已有众多的科技工作者加入到 MATLAB 的开发队伍中，并为形成最新版本的 MATLAB 系统做出了巨大的贡献。

MATLAB 以商品形式出现后，仅短短几年，就以其良好的开放性和运行的可靠性，使

原先控制领域里的封闭式软件包（如英国的 UMIST，瑞典的 LUND 和 SIMNON，德国的 KEDDC）纷纷淘汰，而改以 MATLAB 为平台加以重建。在 20 世纪 90 年代，MATLAB 已经成为国际控制界公认的标准计算软件。到 90 年代初期，在国际上 30 多个数学类科技应用软件中，MATLAB 在数值计算方面独占鳌头，而 MatheMatica 和 Maple 则分居符号计算软件的第二和第三名。MathCAD 因其提供计算、图形、文字处理的统一环境而深受中学生欢迎。在欧美大学里，诸如应用代数、数理统计、自动控制、数字信号处理、模拟与数字通信、时间序列分析、动态系统仿真等课程的教科书都把 MATLAB 作为内容，MATLAB 是攻读学位的大学生、硕士生、博士生必须掌握的基本工具。因此，MATLAB 已经被确认为准确、可靠的科学计算标准软件。在许多国际一流学术刊物上（尤其是信息科学刊物），都可以看到 MATLAB 的应用。在设计研究单位和工业部门，MATLAB 被认定是进行高效算法研究和开发的首选软件工具。如美国 National Instruments 公司信号测量、分析软件 LabVIEW，Cadence 公司信号和通信分析设计软件 SPW 等，或者直接建筑在 MATLAB 之上，或者以 MATLAB 为主要支撑。又如 HP 公司的 VXI 硬件，TM 公司的 DSP，Gage 公司的各种硬卡、仪器等都接受 MATLAB 的支持。

1.2 MATLAB 产品组成及语言特点

MATLAB 支持从概念设计、算法开发、模型仿真和实时实现的理想集成环境。无论进行科学研究还是工程应用，MATLAB 都是必不可少的模型和算法仿真工具。一般而言，MATLAB 的典型应用包括：

- 数据分析和可视化；
- 数值和符号计算；
- 建模、仿真和原型开发；
- 算法预设计与验证；
- 图形用户界面设计；
- MATLAB 应用与 C、C++、Java 以及 Web 集成；
- 图像和视频信号处理；
- 一些特殊的矩阵计算应用，例如自动控制理论、统计、数字信号处理（时间序列分析）等。

1.2.1 MATLAB 的主要产品构成

MATLAB 由一组面向具体应用的工具箱组成，包含了完整的函数集用来对数字图像、控制系统、小波分析和神经网络等特殊应用进行分析和设计。MATLAB 的工具箱是开放的，因此 MATLAB 的工具箱越来越多，功能也越来越强大。用户可以编写自己的工具箱，使用时与使用 MATLAB 提供的工具箱一样。因此，出现了越来越多的免费或商业 MATLAB 工具箱（参见附录 A）。

MATLAB 的主要产品构成如下。

1. MATLAB 集成开发环境

MATLAB 提供了一个集成的开发环境，方便用户开发自己的应用程序。它有一系列的工

具和功能体，其中大部分具有图形用户界面，包括桌面（Desktop）、命令窗口、历史窗口（History）、工作空间（Workspace）、文件和搜索路径等。

2. MATLAB 数学函数库

MATLAB 提供了强大的数学函数库，既包括最基本的矩阵运算函数，如矩阵求逆等，又包括一些特殊的数学函数，如贝塞尔函数等。数学函数库是 MATLAB 进行数据分析的基础。

3. MATLAB 图形用户接口

MATLAB 提供了图形用户接口函数，包括二维和三维图形显示、图像处理、动画和图形显示的高级命令。设计图形用户界面（GUI）的工具包括布局编辑器、排列工具、属性观察器和菜单编辑器等。

4. MATLAB 的专用领域工具箱

MATLAB 提供了一系列专用领域的工具箱，如模型仿真、神经网络、小波分析、信号处理、图像处理等，用于解决特定领域的工程问题。工具箱是开放和可扩展的，用户可以根据需要选择购买和选择安装需要的工具箱。

5. MATLAB Compiler

MATLAB Compiler（编译器）提供了将 MATLAB 语言编写的 M 文件自动转换为 C 或 C++格式文件的能力，支持用户进行独立应用开发。利用 MATLAB Compiler，用户可以快速地开发出功能强大的独立应用。

6. MATLAB Simulink

Simulink 是一个对动态系统进行建模、仿真和分析的软件包。它既可以仿真线性系统，又可以仿真非线性系统。它使得 MATLAB 的功能得到了进一步的扩展：

- 实现了可视化建模，在 Windows 环境下，用户可以通过简单的鼠标操作建立直观的系统模型，进行分析仿真。
- 实现了与 MATLAB 中 M 文件的数据共享，甚至可以与硬件实现实时信息交换。
- 将理论研究与工程实际有机地结合在一起。

7. Stateflow

与 Simulink 的模型框结合，描述复杂事件驱动系统的逻辑行为，驱动系统在不同的模块之间进行切换。

8. Real-Time Workshop

Real-Time Workshop 与 Stateflow 直接从 Simulink 模型与 Stateflow 框图中生成高效的可移植 C 代码或 Ada 代码。只需要简单的操作，用户无须烦琐的手工编程与调试就可以生成应用代码。

MATLAB 安装完后，在 MATLAB 命令符下输入 ver 命令，即可了解安装了哪些工具箱以及它们的版本。本书的 MATLAB 7.0 工具箱版本如下（仅列举了部分与混合编程相关的及

个别应用工具箱）：

```
>> ver
-----------------------------------------------------------------------------
MATLAB Version 7.0.0.19920（R14）
MATLAB License Number: 0
Operating System: Microsoft Windows 2000 Version 5.0（Build 2195: Service Pack 4）
Java VM Version: Java 1.4.2 with Sun Microsystems Inc. Java HotSpot（TM）Client VM
-----------------------------------------------------------------------------
MATLAB                                  Version 7.0      （R14）
Simulink                                Version 6.0      （R14）
Excel Link                              Version 2.2      （R14）
Image Processing Toolbox                Version 4.2      （R14）
MATLAB Builder for COM                  Version 1.1      （R14）
MATLAB Builder for Excel                Version 1.2      （R14）
MATLAB Compiler                         Version 4.0      （R14）
```

1.2.2 MATLAB 语言的特点

MATLAB 具有不同于其他语言如 Fortran、C 语言等的特点，被称为第四代计算机语言，又称为“草稿纸式”的语言。MATLAB 把工程技术人员从烦琐的程序代码中解放出来，可以快速地验证自己的模型和算法。概括起来，MATLAB 语言具有如下主要特点。

1. 方便的矩阵和数组运算

MATLAB 是以矩阵为基础的，可以方便地进行矩阵的算术运算、关系运算和逻辑运算等。MATLAB 有特殊矩阵专门的库函数，可以高效地求解诸如信号处理、控制、优化等问题。变量不需要预先定义，也不需要预先定义矩阵（包括数组）的维数。

2. 编程效率极高

MATLAB 是一种面向科学和工程计算的高级语言。它以矩阵运算为基础，极少的代码即可实现复杂的功能。例如求矩阵的秩，MATLAB 只需要一条语句 det( )，而 C 语言等则需要几十甚至上百条代码。

3. 易学易用，使用方便

MATLAB 易学易用，其函数名和表达更接近我们书写计算公式的思维表达方式，MATLAB 编写程序犹如在演草纸上排列公式与求解问题。MATLAB 是一种解释性语言，不需要专门的编译器。具体地说，MATLAB 运行时，可直接在命令行输入 MATLAB 语句，系统立即进行处理，完成编译、链接和运行的全过程。因此，MATLAB 语言不仅是一门语言，广义上是一种语言调试系统。

4. 可扩充性强

MATLAB 有着丰富的库函数，在进行复杂的数学运算时可以直接调用。用户可以根据需

要方便地编写和扩充新的函数库。为了充分利用 Fortran 和 C 语言资源，用户可以通过混合编程在 MATLAB 中调用 Fortran 和 C 语言的源程序，也可以在 C 语言和 Fortran 中使用 MATLAB 的数值计算功能。

5. 可移植性好

MATLAB 本身是用 C 语言编写的，而 C 语言的可移植性好。MATLAB 函数可以很方便地移植到 C 语言平台。除了内部函数以外，MATLAB 的绝大部分函数和工具箱函数都是公开的，可以用文本编辑器打开。

但是，MATLAB 作为一种解释性语言，与 C 语言等其他高级语言相比较，也存在着以下一些缺点：

（1）运行效率较低，执行相同功能的代码运行时间较长；

（2）M 文件为文本文件，文本编辑器可直接打开，不利于算法保密；

（3）访问硬件能力相对较差，图形用户界面功能也不够灵活。

1.3 MATLAB 7.0 的新功能和新产品

1.3.1 MATLAB 7.0 的新功能

MATLAB 的版本发展较快，自 20 世纪 80 年代诞生以来已发展到 7.0 版本。2004 年 MathWorks 推出了最新的 7.0 版本，它较以前版本有了较大的改进。MATLAB 7.0 的特点在于全新的桌面及各种不同领域的集成工具，以方便用户使用。MATLAB 7.0 的桌面如图 1-1 所示。

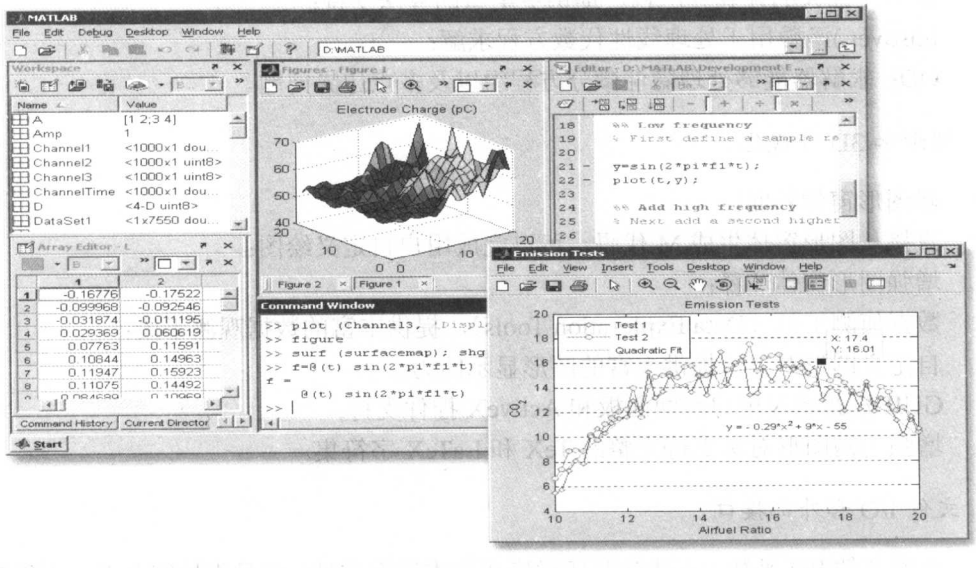

图 1-1 MATLAB 7.0 桌面

MATLAB 7.0 针对编程环境、代码效率、数据可视化、数学计算、文件 I/O 等方面进行升级，具体内容包括以下方面。

1. 开发环境

- 重新设计的桌面环境，针对多文档界面应用提供了简便的管理和访问方法，允许用户自定义桌面外观、创建常用命令的快捷方式；
- 增强数组编辑器（Array Editor）和工作空间浏览器（Workspace Brower）功能，用于数据的显示、编辑和处理；
- 在当前目录浏览器（Current Directory Browser）工具中，增加代码效率分析、覆盖度分析等功能；
- M-Lint 编码分析，辅助用户完成程序性能分析，提高程序执行效率；
- 增强 M 文件编辑器(M-Editor)，支持多种格式源代码文件可视化编辑，例如 C/C++、HTML、Java 等。

2. 编程

- 支持创建嵌套函数（Nested Function），提供更灵活的代码模块化方式；
- 匿名函数（Anonymous Function）功能，支持在命令行或者脚本文件中创建单行函数（Single Line Function）；
- 支持条件分支断点，可以在条件分支语句中进行程序中断调试；
- 模块化注释，支持为代码段注释。

3. 数学

- 支持整数算术运算；
- 支持单精度数据类型运算，包括基本算术运算、线性代数、 FFT 等；
- 使用更强大的计算算法包，提供更丰富的算法支持；
- linsove( )函数用于处理线性代数方程求解；
- ODE 求解器能够处理隐性微分方程组以及多点边界问题。

4. 图形和 3D 可视化

- 新图形窗体界面；
- 直接从图形窗体生成 M 代码，可以完成用户自定义绘图；
- 增强图形窗体注释；
- 数据侦测工具（Data Exploration Tools），提供丰富的数据观测手段；
- 自定义图形对象，提供丰富的图形显示能力；
- GUIDE 新增对用户界面面板和 ActiveX 控件支持；
- 增强句柄图形对象支持完整的 TeX 和 LaTeX 字符集。

5. 文件 I/O 和外部接口

- 新增文件 I/O 函数，支持读取任意格式文本数据文件，并且支持写入 Excel 和 HDF5 格式数据文件；
- 具有压缩功能的 MAT 文件格式，支持快速数据文件 I/O 能力；
- javaaddpath( )函数，无须重新启动 MATLAB 即完成 Java 类的加载、删除等功能；

- 支持 COM、服务器事件以及 VBS；
- 支持 SOAP，使用网络服务；
- FTP 对象，直接访问 FTP 服务器；
- 支持 Unicode 编码格式，增强 MAT 文件字符集。

6. 性能与系统平台支持

- JIT 加速器支持所有数值数据类型；
- Windows XP 系统下支持 3GB 内存访问。

1.3.2 MATLAB 升级及新增的模块

MATLAB 7.0 升级了 29 个产品模块，新增了 12 个产品模块。

1.3.2.1 模块主要特性总结

1. Control System Toolbox 6

Control System Toolbox 6 工具箱为动态系统闭环控制器设计与分析提供了强大的工具，主要新特性包括：

- 使用基于LAPACK和SLICOT算法引擎的强大数值计算引擎，提高计算速度与精度；
- 针对非稳态系统提供了更好的模型缩减算法；
- 新开发的模型分解命令。

2. Database Toolbox 3

Database Toolbox 3 支持在 MATLAB 环境中访问支持 ODBC/JDBC 标准的数据库，使用 Visual Query Builder 工具可以完成数据库的访问、修改等工作，这一过程不需工程师了解任何 SQL 语言，主要新特性包括：

- 支持 Java 语言的 SQL 对象 BINARY 和 OTHER；
- 支持脱离 Visual Query Builder 直接向 ODBC/JDBC 数据库写入数据；
- Visual Query Builder 支持结构和数值数组。

3. Filter Design Toolbox 3

Filter Design Toolbox 3 支持高级数字滤波器的设计、仿真与分析工作，提供了基本滤波器体系结构和设计方法，包括自适应滤波器和多速率滤波器，并且可以用于复杂的实时 DSP 系统开发，主要新特性包括：

- 扩展支持二阶 IIR 滤波器，提供滤波器的设计、重构、定标与量化、图形化分析的能力；
- 支持多速率滤波器的设计与分析，将单速率滤波器与多速率滤波器分析设计工作集成；
- 增强的定点滤波器仿真、分析和集成功能，支持 Signal Processing Toolbox；
- FIR 滤波器设计函数；
- 改进 FDATool 工具，包含了高级滤波器的设计方法、多速率滤波器设计功能，改进

了量化设计功能等；

- 支持通过 Filter Design HDL Coder 将滤波器设计结果进行 FPGA 仿真。

4. Instrument Control Toolbox 2

Instrument Control Toolbox 2 提供了与各种仪器设备进行数据通信的能力，该工具箱支持 GPIB、VISA、TCP/IP 以及 UDP 等通信方式。工程师可以在 MATLAB 环境下生成数据发送给相应的仪器设备或者从仪器设备中读取数据并且进行分析和可视化工作，主要的新特性包括：

- 支持 IVI、VXIPlugandPlay 等设备，MATLAB 的仪器驱动无须用户了解仪器特性即可开展工作；
- 新开发的图形界面工具 TMTool，用户管理设备驱动。

5. Mapping Toolbox 2

Mapping Toolbox 2 提供在 MATLAB 环境下进行地理信息显示、分析的函数以及相应的图形化界面工具，主要新特性包括：

- 支持标准 GIS 和地理信息数据文件格式；
- 支持 Transverse Mercator 项目和 PROJ.4 项目数据库。

6. MATLAB Compiler 4

MATLAB Compiler 4 能够将 MATLAB 的算法和应用程序文件转变成可以发布的独立可执行的应用程序， MATLAB Compiler 4 支持更多的 M 语言特性，主要新特性包括：

- 兼容 MATLAB 面向对象数据类型；
- 共享库函数开发环境，兼容 R14 和 R13 版本的程序开发；
- 增强 C++ 语言集成；
- 支持生成各种独立可执行的应用程序，包括 C/C++ 共享库、 COM 组件和 Excel Plug-in 等。

7. MATLAB Report Generator 2

使用 MATLAB Report Generator 2 可以根据用户开发的 MATLAB 应用程序自动创建各种文档报告，主要新特性包括：

- 重新设计的图形界面工具；
- 更快的文档生成速度；
- 支持生成 Adobe PDF 格式文档；
- 增加 MATLAB 组件，支持 Axes、句柄图形以及 MATLAB 属性表格。

8. Optimization Toolbox 3

Optimization Toolbox 3 提供了针对通用问题或者大规模优化问题处理的算法，支持线性规划、二次规划、非线性最小二乘法、非线性方程求解等功能，主要新特性包括：

- 二进制整数规划问题求解；
- 针对中等规模问题实现无约束优化算法函数 fminunc；

- 增加使用单纯形算法的线性规划函数 linprog。

9. Signal Processing Blockset 6（原来的 DSP Blockset）

Signal Processing Blockset 6 适用于扩展 Simulink，进行针对帧信号的数字信号处理系统设计、仿真与分析工作。它能够完成通信系统、音频/视频系统、数字控制器、雷达/声呐系统、消费类电子产品以及医疗器械等信号处理系统的开发工作，主要新特性包括：

- 音频与语音处理功能，包括 LPC to/from RC，G.711 Codec，CIC 等；
- 扩展的数字滤波器，包括 4 类浮点滤波器和 15 类定点结构滤波器；
- 增强的定点支持（需要 Simulink Fixed-Point）用于滤波器、信号统计模块等功能；
- 新开发的定点参数设置对话框用于设置模型的定点特性，例如字长、二进制小数位以及整数溢出等；
- 新改进的 Scope 模块，支持 Waterfall Scope。

10. Virtual Reality Toolbox 4

Virtual Reality Toolbox 4 可以将三维虚拟现实场景引入到 Simulink 模型中，用于 Simulink 模型运算结果的可视化显示工作，主要新特性包括：

- 支持动画显示结果的录制；
- 支持向量、矩阵数据输入，用于控制场景动画；
- 支持从 Simulink 模型中控制动画显示速率；
- 改进的三维场景观测器（Viewer），用于模型的创建和观察；
- 支持 USB Space Mouse , Space Traveler 运动控制输入设备，以及力回馈操纵杆输入设备。

1.3.2.2 升级产品模块及新模块

MATLAB 7.0 升级的 29 个产品模块和新增的 12 个产品模块列举见表 1-1 至表 1-3。

表 1-1 主要升级模块

模 块 名 称	模 块 名 称
Communications Blockset	Nonlinear Control Design Blockset （更名为 Simulink Response Optimization）
Communications Toolbox	Optimization Toolbox
Control System Toolbox	Real-Time Workshop
Database Toolbox	Real-Time Workshop Embedded Coder
DSP Blockset （更名为 Signal Processing Blockset）	Signal Processing Blockset
Embedded Target for Motorola MPC555	Simulink Fixed Point
Embedded Target for TI C6000 DSP	Simulink Report Generator
Filter Design Toolbox	Simulink Response Optimization
Financial Derivatives Toolbox	Stateflow

续表

模 块 名 称	模 块 名 称
Fixed-Point Blockset （更名为 Simulink Fixed Point）	Stateflow Coder
Instrument Control Toolbox	Statistics Toolbox
Mapping Toolbox	System Identification Toolbox
MATLAB Compiler	Virtual Reality Toolbox
MATLAB Report Generator	Wavelet Toolbox
Model Predictive Control Toolbox	

表 1-2　新产品模块

模 块 名 称	模 块 名 称
Bioinformatics Toolbox	OPC Toolbox
Embedded Target for TI C2000 DSP	RF Blockset
Filter Design HDL Coder	RF Toolbox
Fixed-Point Toolbox	Simulink Control Design
Genetic Algorithm and Direct Search Toolbox	Simulink Parameter Estimation
Link for ModelSim	Simulink Verification and Validation

表 1-3　其他升级模块

模 块 名 称	模 块 名 称
Aerospace Blockset 1.6	MATLAB Link for Code Composer Studio 1.3.1
Curve Fitting Toolbox 1.1.1	MATLAB Builder for COM 1.1（formerly named MATLAB COM Builder）
Data Acquisition Toolbox 2.3	MATLAB Builder for Excel 1.2（formerly named MATLAB Excel Builder）
Datafeed Toolbox 1.5	Model-Based Calibration Toolbox 2.1
Dials & Gauges Blockset 1.2	Mu-Analysis and Synthesis Toolbox 3.0.8
Embedded Target for Infineon C166 Microcontrollers 1.1	Neural Network Toolbox 4.0.3
Embedded Target for Motorola HC12 1.1	Partial Differential Equation Toolbox 1.0.5
Embedded Target for OSEK/VDX 1.1	Real-Time Windows Target 2.5
Excel Link 2.2	Robust Control Toolbox 2.0.10
Extended Symbolic Math Toolbox 3.1	Signal Processing Toolbox 6.2
Financial Time Series Toolbox 2.1	SimMechanics 2.2
Financial Toolbox 2.4	SimPowerSystems 3.1
Fixed-Income Toolbox 1.0.1	Simulink Accelerator 6
Fuzzy Logic Toolbox 2.1.3	Symbolic Math Toolbox 3.1

续表

模 块 名 称	模 块 名 称
GARCH Toolbox 2.0.1	xPC Target 2.5
Image Acquisition Toolbox 1.5	xPC Target Embedded Option 2.5
LMI Control Toolbox 1.0.1	xPC TargetBox 2.5

1.4 小　　结

本章介绍了 MATLAB 语言的基础知识，包括 MATLAB 的发展历程、语言特点及最新版本 MATLAB 7.0 的新产品和新功能。通过本章的学习，读者对 MATLAB 可以有一个整体的了解。

第 2 章　MATLAB 程序设计及代码优化

MATLAB 易学易用，其函数名和表达很接近我们书写计算公式的思维表达方式，用 MATLAB 编写程序犹如在演草纸上排列公式与求解问题。MathWorks 公司将 MATLAB 称为第四代编程语言，足见其简捷。同其他的程序设计语言一样，MATLAB 语言提供了流控制、函数、数据结构、输入/输出功能以及面向对象的程序设计方法。

2.1　MATLAB 的表达式和变量

2.1.1　表达式

表达式和变量是 MATLAB 的基础。MATLAB 采用的是表达式语言，用户输入的语句由 MATLAB 系统解释运行。MATLAB 的语句由表达式和变量组成。MATLAB 语句有两种常见的形式：

- 表达式；
- 变量＝表达式。

在第一种形式中，如果不指定将表达式的值赋给某个变量，表达式会将运算产生的结果自动地赋值给名为 ans 的变量，并显示在屏幕上。变量 ans 是一个默认的变量名，它会在以后的类似操作中被自动覆盖。

对于第二种形式，等号右边的表达式计算产生的结果赋值给等号左边的变量，并且放入内存中。

2.1.2　变量

变量是任何程序设计语言的基本要素。与常规的程序设计语言不同，MATLAB 语言的变量不需要事先声明，也不需要指定变量类型，MATLAB 会自动依据所赋予变量的值或对变量所进行的操作来识别变量的类型。赋值过程中如果赋值变量已经存在，将用新值代替旧值，新值类型代替旧值类型。在 MATLAB 中变量的命名应遵循如下规则：

- 变量名区分大小写；
- 变量名长度不超过 31 个字符；
- 变量名以字母开头，可以由字母、数字、下划线组成，但不能使用标点。

MATLAB 语言中的变量也存在变量域的问题。在未加特殊说明的情况下，MATLAB 语言将所识别的一切变量视为局部变量。

2.2　细胞数组与结构数组

2.2.1　细胞数组

用类似矩阵的记号将复杂的数据结构纳入一个变量之下。与矩阵中的圆括号表示下标类

似，细胞数组由大括号表示下标。

```
>> B={1,'Anne cat', 18, [100, 80, 75; 77, 60, 92; 67, 28, 90; 100, 89, 78]}
 B =   [1]    'Anne cat'    [18]    [4x3 double]
```

访问单元数组应该由大括号进行，如第 4 单元中的元素可以由下面的语句得出：

```
>> B{4}
ans = 100   80   75
      77    60   92
      67    28   90
      100   89   78
```

2.2.2 结构数组

MATLAB 的结构体有点像 C 语言的结构体数据结构。每个成员变量用点号表示，如 A.p 表示 A 变量的 p 成员变量。获得该成员比 C 更直观，仍用 A.p 访问，而不用 A->p。用下面的语句可以建立一个小型的数据库。

```
>> student_rec.number=1;
student_rec.name='Alan Shearer';
student_rec.height=180;
student_rec.test=[100, 80, 75; 77, 60, 92; 67, 28, 90; 100, 89, 78];
>> student_rec
student_rec =
number: 1
name: 'Alan Shearer'
height: 180
test: [4x3 double]
```

删除成员变量可以由 rmfield( )函数进行，添加成员变量可以直接用赋值语句即可。另外，数据读取可以由 setfield( )和 getfield( )函数完成。

2.3 类 与 对 象

本节讲解如何创建类来增加 MATLAB 的数据类型。通过创建类和对象，可以增加 MATLAB 的数据类型和操作方法。类定义了一种数据结构以及可用于此种数据的函数和运算符等。对象是类的实例。类与对象是 MATLAB 5.* 开始引入的数据结构。在 MATLAB 手册中定义了一个很好的类——多项式类。事实上，在实际工具箱设计中，用到了很多的类，例如在控制系统工具箱中定义了 LTI（线性时不变系统）类，并在此基础上定义了其子类：传递函数类 TF，状态方程类 SS，零极点类 ZPK 和频率响应类 FR。

下面通过一个例子来介绍类的构造。在 MATLAB 语言使用手册中给出了一个很有代表性的例子：多项式类的建立问题。假设我们想为多项式建立一个单独的类，重新定义加、减、乘及乘方等运算，并定义其显示方式。那么建立一个类至少应该执行下面的步骤：

（1）首先应该选定一个恰当的名字，例如这里的多项式类可选择为 polynom。以这个名字建立一个子目录，目录的名字前加@。对本例来说，即应该在当前的工作目录下建立

@polynom 子目录，而这个目录无须在 MATLAB 路径下再指定。

（2）编写类的构造函数，函数名应该和类同名。在定义类方法的子目录中，必须有一个称为构造程序的 M 文件。构造程序通过初始化类的数据结构和给类分配标号来定义类。

下面是多项式类的构造函数 polynom(a)的内容：

```
function p = polynom(a)
%POLYNOM Polynomial class constructor.
%    p = POLYNOM(v) creates a polynomial object from the vector v,
%    containing the coefficients of descending powers of x.
if nargin == 0
    p.c = [];
    p = class(p,'polynom');
elseif isa(a,'polynom')
    p = a;
else
    p.c = a(:).';
    p = class(p,'polynom');
end
```

可以看出，当采用无输入参数的格式调用函数 polynom( )时，构造程序将为该对象生成一个模板，通常情况下属性值为空；当输入参数为多项式时，构造程序将原封不动地返回该多项式；当输入参数不是多项式时，构造程序将把输入参数变形为行向量并赋给属性 c，最后用函数 class( )加上标号“polynom”，表明该对象为多项式。

一般来说，类的构造函数都采用类似 polynom(a)的框架，大致包含以下几个部分：

- 如果不给输入变量，则建立一个空的多项式；
- 如果输入变量 a 已经为多项式类，则将它直接传送给输出变量 p；
- 如果 a 为向量，则将此向量变换成行向量，再构造成一个多项式对象；
- 该结构的属性赋值；
- 用函数 class( )加标号。

（3）如果想正确地显示新定义的类，则必需首先定义 display( ) 函数，并对新定义的类重新定义其基本运算。对多项式来说，可以如下定义有关的函数：要改变显示函数的定义，则需在此目录下重新建立一个新函数 display( )。这种重新定义函数的方法又称为函数的重载。显示函数可以如下地重载定义：

```
function display(p)
% POLYNOM/DISPLAY Command window display of a polynom
disp(' ');
disp([inputname(1),' = '])
disp(' ');
disp(['     ' char(p)])
disp(' ');
```

从上面的定义可见，显示函数要求重载定义 char( ) 函数，用于把多项式转换成可显示的字符串。该函数的定义为：

```
function s = char(p)
% POLYNOM/CHAR
% CHAR(p) is the string representation of p.c
if all(p.c == 0)
    s = '0';
else
    d = length(p.c) - 1;
    s = [];
    for a = p.c;
        if a ~= 0;
            if ~isempty(s)
                if a > 0
                    s = [s ' + '];
                else
                    s = [s ' - '];
                    a = -a;
                end
            end
            if a ~= 1 | d == 0
                s = [s num2str(a)];
                if d > 0
                    s = [s '*'];
                end
            end
            if d >= 2
                s = [s 'x^' int2str(d)];
            elseif d == 1
                s = [s 'x'];
            end
        end
        d = d - 1;
    end
end
```

仔细研究此函数，可以发现，该函数能自动地按照多项式显示的格式构造字符串。比如，多项式各项用加减号连接，系数与算子之间用乘号连接，而算子的指数由 ^ 表示。再配以显示函数，则可以将此多项式以字符串的形式显示出来。

➢ 双精度处理

双精度转换函数的重载定义是很简单的。

```
function c = double(p)
```

```
% POLYNOM/DOUBLE  Convert polynom object to coefficient vector.
%    c = DOUBLE(p) converts a polynomial object to the vector c
%    containing the coefficients of descending powers of x.
c = p.c;
```

➢ 加运算

两个多项式相加，只需将其对应项系数相加即可。这样，加法运算的重载定义可由下面的函数实现。注意，这里要对 plus( ) 函数进行重载定义。

```
function r = plus(p,q)
% POLYNOM/PLUS  Implement p + q for polynoms.
p = polynom(p);
q = polynom(q);
k = length(q.c) - length(p.c);
r = polynom([zeros(1,k) p.c] + [zeros(1,-k) q.c]);
```

同理，还可以重载定义多项式的减法运算：

```
function r = minus(p,q)
% POLYNOM/MINUS Implement p - q for polynoms.
p = polynom(p);
q = polynom(q);
k = length(q.c) - length(p.c);
r = polynom([zeros(1,k) p.c] - [zeros(1,-k) q.c]);
```

➢ 乘法运算

多项式的乘法实际上可以表示为系数向量的卷积，可以由 conv( )函数直接获得，故可以如下重载定义多项式的乘法运算。

```
function r = mtimes(p,q)
% POLYNOM/MTIMES    Implement p * q for polynoms.
p = polynom(p);
q = polynom(q);
r = polynom(conv(p.c,q.c));
```

➢ 乘方运算

多项式的乘方运算只限于正整数乘方的运算，其 n 次方相当于将该多项式自乘 n 次。若 n=0，则结果为 1。这样我们就可以重载定义多项式的乘方运算为：

```
function p=mpower(a,n)
if n>=0, n=floor(n); a=polynom(a); p=1;
if n>=1,
for i=1:n, p=p*a; end
end
else, error('Power should be a non-negative integer.')
end
```

（4）定义了类之后，我们就可以方便地进行多项式处理了。例如我们可以建立两个多项

式对象 P(s)=x^3+4x^2-7 和 Q(s)=5x^4+3x^3-1.5x^2+7x+8，其相应的 MATLAB 语句为：

```
>> P=polynom([1,4,0,-7]), Q=polynom([5,3,-1.5,7,8])
P =
   x^3 + 4*x^2 - 7
Q =
   5*x^4 + 3*x^3 - 1.5*x^2 + 7*x + 8
```

然后调用下面函数就可以得出相应的计算结果：

```
>> P+Q
ans =5*x^4 + 4*x^3 + 2.5*x^2 + 7*x + 1
>> P-Q
ans =-5*x^4 - 2*x^3 + 5.5*x^2 - 7*x – 15
>> P*Q
ans =5*x^7 + 23*x^6 + 10.5*x^5 - 34*x^4 + 15*x^3 + 42.5*x^2 - 49*x – 56
>> X=P^3
X =x^9 + 12*x^8 + 48*x^7 + 43*x^6 - 168*x^5-336*x^4+147*x^3+588*x^2-343
```

由于前面的重载定义，下面的表达式也能得出期望的结果：

```
>> P+[1 2 3]
 ans =x^3 + 5*x^2 + 2*x - 4
```

使用 methods( ) 函数可以列出一个新类已经定义的方法函数名。

```
>> methods('polynom')
 Methods for class polynom:
 char      display  double   minus    mpower   mtimes   plus     polynom
```

2.4 流 程 控 制

MATLAB 语言也提供了丰富的流程控制语句，以完成具体的程序设计。MATLAB 的流程控制语句结构主要有 5 种。

2.4.1 for 循环结构

for 循环结构是流程控制语句中的基础，使用 for 循环能以指定的次数重复执行循环体内的语句。其调用方式如下：

```
for 循环控制变量=<循环次数设定>
循环体
end
```

【例 2-1】 一个简单的 for 循环示例。

```
for i=1:10;                    %i 依次取 1,2,…10
x(i)=i;                        %对每个 i 值，重复执行由该指令构成的循环体
end
```