普通高等教育机电类“十三五”规划教材

流体传动与控制系统计算机仿真

主　编　梁　全
副主编　徐　威　刘慧芳

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书全面系统地介绍了流体传动与控制系统计算机仿真的理论和方法，内容包括液压系统常见的仿真方法、液压流体力学基础、MATLAB 简介、古典控制理论基础、常用的数值计算方法及 MATLAB 数值计算函数、Simulink 仿真基础、液压 PLC 技术基础、常用液压元件及系统的建模方法。为了学以致用，在介绍基础理论的同时，还介绍了详细的建模操作步骤。

本书体系结构完整，是高等院校机械设计制造及其自动化专业教材，可作为高等院校机械类本科高年级学生及研究生的教学用书，也可作为从事液压控制系统技术研究与应用的工程技术人员的自学与参考书籍。

图书在版编目（CIP）数据

流体传动与控制系统计算机仿真 / 梁全主编. —北京：电子工业出版社，2018.11
普通高等教育机电类“十三五”规划教材
ISBN 978-7-121-34358-2

Ⅰ. ①流… Ⅱ. ①梁… Ⅲ. ①液压传动－控制系统－计算机仿真－高等学校－教材 Ⅳ. ①TH137

中国版本图书馆 CIP 数据核字（2018）第 115620 号

策划编辑：赵玉山
责任编辑：韩玉宏
印　　刷：
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×1 092　1/16　印张：12　字数：307.2 千字
版　　次：2018 年 11 月第 1 版
印　　次：2018 年 11 月第 1 次印刷
定　　价：38.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：zhaoys@phei.com.cn。

前　言

本书的写作目的是总结液压系统计算机仿真的关键技术，但在写作过程中，并没有仅仅局限在液压仿真方面，而是包含了机电控制等相关内容，所以最终本书命名为《流体传动与控制系统计算机仿真》。之所以命名为“传动与控制”，说明本书不仅包括“传动”系统的仿真，也包括“控制”系统的仿真。

“仿真”这一名词在计算机刚刚诞生时，是一个十分时髦的词汇，那时如果什么东西能和计算机搭上边，就都会显得高深莫测、新颖别致。随着时代的进步和发展，“仿真”被滥用了，渐渐地人们把“仿真”和“不实际”“不准确”“不现实”联系在一起，使仿真脱离了其本质。本书的写作目的中很重要的一条，是希望能够还原仿真的本质。

按“百度百科”中对仿真的定义，仿真是利用模型复现实际系统中发生的本质过程，并通过对系统模型的实验来研究存在的或设计中的系统，又称模拟。这句话基本说清了仿真的本质。笔者认为，仿真有两个直接的目的：一个是分析现有系统，另一个是辅助设计新系统。仿真使实际系统不存在时，人们就能在一定程度上对其进行了解、研究。试想，价格昂贵、搭建费时费力的传动系统，借助仿真就可以在一台小小的个人计算机上运行，在还没有搭建实际的系统时，就能够对将来要发生的事情做出规划和预测，那将在多大程度上节省金钱和时间呀！所以，同学们在学习本课程时，任何时候都应该把握住仿真的本质，不能主次颠倒、本末倒置，时刻要记住，仿真不是目的而是手段。

计算机仿真离不开软件，本课程采用的仿真软件是 MATLAB。之所以采用这个软件，是因为该软件能阐明仿真本质。该软件既能进行“功能仿真”，就是把液压、气动回路的功能在没有构建系统时，就能够比较清晰地展示给用户；又能进行“性能仿真”，就是深入系统内部，回答用户对系统性能探求的好奇心，是更高级的仿真。

另外，本书还介绍了在 MATLAB 环境下对 PLC 系统进行仿真的方法。

本书共 8 章。第 1 章介绍液压系统常见的仿真方法，第 2 章介绍液压流体力学基础，第 3 章是 MATLAB 简介，第 4 章介绍古典控制理论基础，第 5 章介绍常用的数值计算方法及 MATLAB 数值计算函数，第 6 章介绍 Simulink 仿真基础，第 7 章介绍液压 PLC 技术基础，第 8 章介绍常用液压元件及系统的建模方法。其中，第 3 章由沈阳工业大学徐威老师撰写，第 5 章由沈阳工业大学刘慧芳老师撰写，其余章节和最终统稿由沈阳工业大学梁全老师完成。

最后，还要回到仿真的本质上来，虽然笔者在写作过程中试图还原仿真的本质，但是由于水平有限，可能没有达到希望的目的，恳请读者在学习使用的过程中批评指正，使本书日趋完美。

作　者

目　　录

第 1 章　液压系统常见的仿真方法

1.1　动态系统的计算机仿真

1.1.1　系统与系统模型

1．系统

系统指具有某些特定功能、相互联系、相互作用的元素的集合。这里的系统是指广义上的系统，泛指自然界的一切现象与过程，举例来说，工程系统如控制系统、通信系统等，非工程系统如股市系统、交通系统、生物系统等。

2．系统模型

系统模型是对实际系统的一种抽象，是对系统本质（或系统的某些特性）的一种描述。系统模型具有与系统相似的特性。好的系统模型能够反映实际系统的主要特征和运动规律。

系统模型可以分为实体模型和数学模型。

实体模型又称物理效应模型，是根据系统之间的相似性而建立起来的物理模型，如建筑模型等。

数学模型包括原始系统数学模型和仿真系统数学模型。原始系统数学模型是对系统的原始数学描述。仿真系统数学模型是一种适合于在计算机上演算的模型，主要是指根据计算机的运算特点、仿真方式、计算方法、精度要求将原始系统数学模型转换为计算机程序。系统模型的分类见表 1-1。

表 1-1　系统模型的分类

<table>
<tr><th>静态系统模型</th><th colspan="3">动态系统模型</th></tr>
<tr><td rowspan="3">代数方程</td><td colspan="2">连续系统模型</td><td>离散系统模型</td></tr>
<tr><td>集中参数</td><td>分布参数</td><td rowspan="2">差分方程</td></tr>
<tr><td>微分方程</td><td>偏微分方程</td></tr>
</table>

1.1.2　仿真与计算机仿真

1．仿真的概念

仿真是以相似性原理、控制论、信息技术及相关领域的有关知识为基础，以计算机和各种专用物理设备为工具，借助系统模型对真实系统进行试验的一门综合性技术。

2．仿真的分类

实物仿真：又称物理仿真，是指研制某些实体模型，使之能够重现原系统的各种状态。早期的仿真大多属于这一类。其优点是直观、形象，至今仍然广泛应用；缺点是投资巨大、周期长、难以改变参数、灵活性差。

数学仿真：是用数学语言去描述一个系统，并编制程序在计算机上对实际系统进行研究的过程。其优点是灵活性高、便于改变系统结构和参数、效率高（可以在很短时间内完成实际系统很长时间的动态演变过程）、重复性好；缺点是对某些复杂系统可能很难用数学模型来表达，或者难以建立其精确模型，或者由于数学模型过于复杂而难以求解。

半实物仿真：又称数学物理仿真或混合仿真。为了提高仿真的可信度或针对一些难以建模的实体，在系统研究中往往把数学模型、物理模型和实体结合起来组成一个复杂的仿真系统，这种在仿真环节中存在实体的仿真称为半物理仿真或半实物仿真，如飞机半实物仿真。

3．计算机仿真

计算机仿真是在研究系统过程中根据相似性原理，利用计算机来逼真模拟研究系统。研究对象可以是实际的系统，也可以是设想中的系统。在没有计算机之前，仿真都是利用实物或其物理模型来进行研究的，即物理仿真。物理仿真的优点是直接、形象、可信，缺点是模型受限、易破坏、难以重用。计算机仿真可以用于研制产品或设计系统的全过程，包括方案论证、技术指标确定、设计分析、故障处理等各个阶段，如训练飞行员、宇航员的仿真工作台和仿真机舱。

1.2 计算机仿真的三要素和三项基本活动

计算机仿真的三个基本要素是系统、模型和计算机，联系着它们的三项基本活动是数学模型建立、仿真模型建立（又称二次建模）和仿真试验。

数学仿真采用数学模型，用数学语言对系统的特性进行描述，其工作过程如下。

（1）建立系统的数学模型。

（2）建立系统的仿真模型，即设计算法并转换为计算机程序，使系统的数学模型能为计算机所接受并能在计算机上运行。

（3）运行仿真模型，进行仿真试验，再根据仿真试验的结果进一步修正系统的数学模型和仿真模型。

1.3 常见的仿真方法

对液压元件和系统利用计算机进行仿真的研究和应用已有三十多年的历史。随着流体力学、现代控制理论、算法理论和可靠性理论等相关学科的发展，特别是计算机技术的突飞猛进，液压仿真技术也日渐成熟，成为液压系统设计人员的有力工具。

研究液压系统动态特性的方法，过去用得较多的是古典控制工程中的传递函数分析法，

近年来，随着现代控制理论及计算机应用的发展，给液压系统动态特性的研究开辟了新的途径。为了求得系统的瞬态响应，可以对系统的动态特性进行数字仿真，即写出系统在动态过程中的数学模型，然后在计算机中求出系统中各主要变量在过渡过程中的时域解，其仿真的具体方法是功率键合图法和节点容腔法。

下面将分别介绍液压系统常见的仿真方法。

1.3.1　传递函数分析法

这种方法主要适用于分析液压系统的稳定性，但只限于线性系统，对于非线性系统要进行线性化，而有些严重的非线性系统很难进行线性化。同时，这种方法主要适用于单输入、单输出，以及初始条件为零的情况。如果要用这种方法求系统的过渡过程（瞬态响应），那么只有在系统相当简单时才有可能。这些都是传递函数分析法的局限性。

1.3.2　功率键合图法

功率键合图是研究机械、液压、电气等系统模型的工具。它用于表示系统功率流程，即表示系统在各种因素作用下，在动态过程中，功率的流向、汇集、分配和能量转换等关系。键合图技术是建立在状态变量理论基础之上的一门研究系统动态特性的技术，它通过键合图直接得到描述系统的状态方程，列方程的方法与其他方法不同，但状态方程的形式完全一样。由一个完全增广定向的键合图模型就可以按一种非常规则的方式拟定系统的方程，有三个需要遵循的简单步骤，即选定输入能量状态变量及共能量变量、列出初步的系统方程组、将初步方程式归并为状态空间形式。真正意义上的自动建模应该是利用系统的功率键合图模型，由软件根据键合图变量之间的关系，推导出一组一阶微分方程组，即代表系统特征的状态方程。键合图理论的资料相对较少，理解应用相对比较困难。

1.3.3　节点容腔法

液压系统既可以看成单个元件，也可以看成是一个复杂的组件，其中各构件间的能量或信号传递通常是经过液压管道实现的。在计算机数字仿真中，液压系统可以描述成普通的微分方程组，并采用数值积分法进行求解。在求解压力、速度、位移等重要的状态变量来讨论动态特性时，可以采用节点容腔法进行仿真。

设$\sum Q$是进出容腔流量总和，则由连续性方程可以得出容腔压力基本方程为

$$p(t)=\frac{E_0}{\sum V}\int_0^t \sum Q\mathrm{d}t + p_0 \tag{1.1}$$

式中　$\sum V$ ——连接容腔节点所有元件的容积之和；

$\sum Q$ ——流过节点上所有元件的流量之和；

E_0 ——油液的体积弹性模量。

在液压系统中，流入或流出某节点的流量决定了其压力的大小，根据节点容腔法建模的原理，式（1.1）所示压力方程中的流量关系就对应于液压回路中的液压元件与管路的关系。根据这一原则，对液压系统中每个液压节点建立其流量-压力方程，这样液压系统就可以由一阶微分方程组来描述各节点容腔压力的动态特性。对此方程组联立求解，就可以对液压系统中各容腔的压力值进行动态特性仿真。

各种工作机械上的液压传动系统及其元件，迄今为止绝大多数都是按“克服阻力、保证速度”的静态指标来计算并设计的。在液压技术不断向前发展的今天，这种情况越来越难以适应高速、高压、大功率和高精度的要求。例如，机床在换向、启动等阶段及在负载突然变化时常常会出现振荡或颤抖，机床上的工作机构不能在外来扰动的作用下保持速度恒定的运动，有时还会产生持续的振荡等。为了查明这些现象的成因，提出解决办法，有必要对工作机械中的液压元件和系统进行动态特性的研究，以便了解它的主导因素和内在的作用规律。

研究液压元件或系统的动态特性必须使用自动控制理论。古典控制理论对分析、综合单变量的线性定常系统已发展得比较完善，可以简洁扼要地、形象地解释清楚许多问题。各种液压元件和系统都有各自的特点，在分析动态特性之前，必须建立该元件（或系统）的动态数学模型，然后再按自动控制理论的方法来进行分析。分析工作最后都汇聚到传递函数上，因为它就是古典控制理论中的动态数学模型。在分析中遇到非线性问题时，均用线性化的方法处理。

1.4 系统按数学模型分类

仿真其实就是用数学模型描述实际系统。因而，先要研究数学模型。从系统数学模型的观点出发，可把系统分成线性系统和非线性系统。

1.4.1 线性系统

尽管具体的物理系统可能是不相同的，但用微分方程来描述其动态特性时，对输出量、输入量及其各阶导数而言，都只是线性的组合，称这类微分方程为线性微分方程。凡是由线性微分方程来描述其动态特性的系统，称为线性系统。

线性系统微分方程的一般表达式为

$$a_n x_o^{(n)} + a_{n-1} x_o^{(n-1)} + \cdots + a_1 \dot{x}_o + a_0 x_o = b_m x_i^{(m)} + b_{m-1} x_i^{(m-1)} + \cdots + b_1 \dot{x}_i + b_0 x_i \qquad (1.2)$$

式中 x_o ——系统输出量；

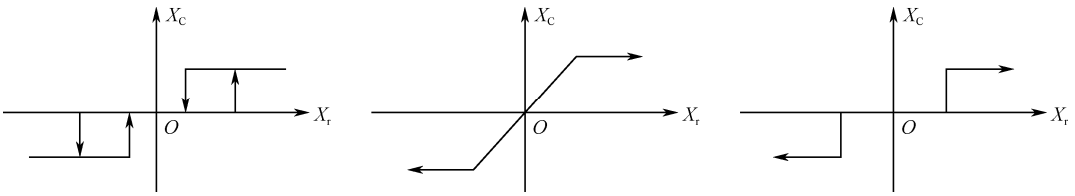
x_i ——系统输入量。

当 a_i 、 b_j （ $i = 0,1,\cdots,n$， $j = 0,1,\cdots,m$ ）为时间的变量时，则式（1.2）所描述的线性系统称为线性时变系统；当 a_i 、 b_j 为不随时间变化的常量时，则式（1.2）所描述的线性系统称为线性定常系统。线性系统可以运用叠加原理，当有几个输入量同时作用于系统时，可以逐个输入，求出对应的输出，然后把各个输出进行叠加，即为系统的总输出。

1.4.2 非线性系统

如果微分方程的系数与自变量有关，则称该微分方程为非线性微分方程。由非线性微分方程来描述其动态特性的系统，称为非线性系统。在自动控制系统中，即使只含有一个非线性元件，该系统便是非线性系统。

典型的非线性元件，就其 I/O 特性来看，可分为继电器特性、饱和特性和不灵敏区特性非线性元件等，如图 1-1 所示。



（a）继电器特性　　（b）饱和特性　　（c）不灵敏区特性

图 1-1　典型的非线性元件

非线性系统就其本质来说与线性系统是不同的。它不具备线性系统的叠加性和均匀性。因此，不能采用研究线性系统的方法去研究非线性系统。

应该指出，任何物理系统的特性精确地说都是非线性的。但在误差允许范围内，可以将某些非线性特性线性化，近似地用线性微分方程来描述，这样就可以按照线性系统来处理。

1.5　控制系统的线性化数学模型

通常所说的线性系统是有条件的，是在一定的工作范围之内保持着线性关系的。例如，系统中的阻尼器，在低速时可以看成是线性的，但在高速时，阻尼摩擦力则与速度的二次方成正比，系统就变成非线性的了。对于含有非线性关系的系统，非线性数学模型的建立和求解，特别是高阶非线性方程的求解，其过程是相当复杂的。

为了绕过非线性系统造成的数学上的困难，一般采用两种方法来处理：一是忽略那些次要的非线性因素，如死区、饱和及干摩擦等；二是当系统的信号或变量变化范围不大或非线性不太严重时，可以引入等效的线性系统来代替非线性系统，即应用线性化数学模型来代替非线性数学模型。

所谓线性化，就是指在一定条件下进行某种近似或缩小研究问题的范围，将非线性微分方程作为线性微分方程来处理。它的数学处理是将变量的非线性函数展开成泰勒（Taylor）级数，分解成这些变量在某工作状态的微增量的表达式，然后略去高于一阶微增量的无穷小，从而求得近似的线性函数，这时就可以用线性控制理论对系统进行分析了。

泰勒级数的一阶近似如下。

对于一元函数有

$$F(x)=F(x_0)+\left(\frac{\mathrm{d}F}{\mathrm{d}x}\right)_0\Delta x \tag{1.3}$$

对于二元函数有

$$F(x,y)=F(x_0,y_0)+\left(\frac{\partial F}{\partial x}\right)_0\Delta x+\left(\frac{\partial F}{\partial y}\right)_0\Delta y \tag{1.4}$$

在液压系统中，可采用这种方法建立滑阀节流口的线性化流量方程。滑阀节流口的流量公式为

$$Q = C_{\mathrm{d}} w x_{\mathrm{v}} \sqrt{\frac{2\Delta p}{\rho}} \tag{1.5}$$

式中 C_{d} ——流量系数；

w ——滑阀的面积梯度；

x_{v} ——阀芯的位移量；

Δp ——节流口的压降；

ρ ——油液的密度。

式（1.5）是一个二元非线性方程，应用泰勒级数进行线性化，有

$$Q = C_{\mathrm{d}} w x_{\mathrm{v0}} \sqrt{\frac{2\Delta p_0}{\rho}} + C_{\mathrm{d}} w \sqrt{\frac{2\Delta p_0}{\rho}} \Delta x_{\mathrm{v}} + C_{\mathrm{d}} w x_{\mathrm{v0}} \sqrt{\frac{2}{\rho}} \times \frac{1}{2\sqrt{\Delta p_0}} \Delta p \tag{1.6}$$

即

$$Q - Q_0 = C_{\mathrm{d}} w \sqrt{\frac{2\Delta p_0}{\rho}} \Delta x_{\mathrm{v}} + C_{\mathrm{d}} w x_{\mathrm{v0}} \sqrt{\frac{2}{\rho}} \times \frac{1}{2\sqrt{\Delta p_0}} \Delta p \tag{1.7}$$

所以有

$$\Delta Q = K_{\mathrm{q}} \Delta x_{\mathrm{v}} + K_{\mathrm{c}} \Delta p \tag{1.8}$$

式中 $K_{\mathrm{q}} = C_{\mathrm{d}} w \sqrt{\frac{2\Delta p_0}{\rho}}$ ——流量增益；

$K_{\mathrm{c}} = \frac{C_{\mathrm{d}} w x_{\mathrm{v0}}}{\sqrt{2\rho\Delta p_0}}$ ——流量-压力系数；

$Q_0 = C_{\mathrm{d}} w x_{\mathrm{v0}} \sqrt{\frac{2\Delta p_0}{\rho}}$ ——空载流量；

x_{v0} ——某个工作点的阀芯位移量（或阀开口量）；

Δp_0 ——某个工作点的压降。

由于 x_{v0} 和 Δp_0 分别是某个工作点的阀芯位移量（或阀开口量）和压降，这些值都是常数，因而 K_{q} 和 K_{c} 也都是常数。由公式 $\Delta Q = K_{\mathrm{q}} \Delta x_{\mathrm{v}} + K_{\mathrm{c}} \Delta p$ 可知，在某个工作点附近，滑阀节流口的流量随阀芯位移量的增加而成比例地增加，并随节流口压降的增加也成比例地增加。显然，它们之间为线性关系。

在处理线性化问题时，应注意以下几点。

（1）首先要确定系统处于平衡状态时各组成元件的静态工作点。这是因为在不同的静态工作点时，液压系统线性化方程的参数值有所不同。

（2）如果输入量工作在较大的范围内，所建立的线性化数学模型势必有较大误差，所以液压系统非线性模型线性化是有条件的。

（3）如果非线性特性是不连续的，则在不连续点附近不能得出收敛的泰勒级数，这时对系统不能进行线性化处理。

（4）线性化后的微分方程是以增量为基础的增量方程。

课后题

1．系统的定义是什么？
2．系统模型是什么？
3．仿真的基本概念是什么？
4．试列举一种生活中常见的设备，思考其是否能用仿真来替代。
5．试列举计算机仿真的三要素。

第 2 章　液压流体力学基础

事实上，任何一种所谓的计算机仿真方法，都是建立在用计算机求解数学问题的基础上的。无论对哪一门学科（如机械、电气、热力学、流体力学）进行仿真，都有赖于对数学公式的求解。因而，仿真工程师的任务就是用恰当的数学公式来描述所在领域的各种物理现象。

流体传动与控制技术的数学理论基础就是流体力学，本章将系统地回顾流体力学知识，这些知识是进行液压系统仿真建模最重要的理论依据。

2.1　液体的主要物理性质

2.1.1　密度和重度

单位容积的液体所具有的质量（重量）称为密度（重度），公式为

$$\rho = \frac{M}{V}$$
$$\gamma = G / V \qquad (2.1)$$
$$\gamma = \rho g$$

式中　ρ ——液体的密度（kg/m^3）；

M ——液体的质量（kg）；

V ——液体的体积（m^3）；

γ ——液体的重度（N/m^3）；

G——液体的重量，即液体受重力的大小（N）；

g——重力加速度（m/s^2）。

2.1.2　可压缩性

液体受压力作用时体积发生变化的性质称为可压缩性（对元件或系统进行动态分析时不可忽略）。单位压强变化所造成的液体体积的相对变化率，用体积压缩系数 β_e 来表示，即

$$\beta_e = -\frac{1}{\Delta p}\frac{\Delta V}{V_0} = -\frac{1}{V_0}\frac{\mathrm{d}V}{\mathrm{d}p} \qquad (2.2)$$

式中　β_e ——体积压缩系数（1/Pa）；

ΔV ——液体的体积变化量（m^3）；

V_0 ——液体的初始体积（m^3）；

Δp ——液体的压强变化量（Pa）。

由于压力增加时，液体的体积缩小，两者变化相反，为使 β_e 取正值，等号右端加一个负号。

体积压缩系数的倒数称为体积弹性模量，用 K 表示，即

$$K = 1 / \beta_e \qquad (2.3)$$

一般油液的体积弹性模量的中间值为 900MPa。

2.1.3　黏性

1．定义

液体在外力作用下流动时，分子之间的内聚力要阻止分子间的相对运动而产生的一种内摩擦力，液体的这一特性称为黏性。

2．液体内摩擦定理

实验测定表明，流动液体相邻液层间的内摩擦力 F_f 与液层间的接触面积 A、液层间的速度梯度 $\frac{dv}{dy}$ 成正比，即

$$F_f = \mu A \frac{dv}{dy} \tag{2.4}$$

式中　μ ——液体的动力黏度（Pa·s）；

A ——液层间的接触面积（m^2）；

$\frac{dv}{dy}$ ——速度梯度（1/s）。

单位面积上的内摩擦力定义为

$$\tau = \frac{F_f}{A} = \mu \frac{dv}{dy} \tag{2.5}$$

3．黏性的度量

黏性的大小用黏度来度量。黏度分三种：①动力黏度；②运动黏度；③相对黏度。

（1）动力黏度（μ）又称绝对黏度，是单位速度梯度下的单位面积上的内摩擦力，单位是 Pa·s 或 $N·s/m^2$（表征液体黏性的内摩擦程度）。过去的单位还有 P（泊）。规定 1Pa·s（帕·秒）=10P（泊）。1P（泊）=100cP（厘泊）。

$$\mu = \frac{F_f}{A\frac{dv}{dy}} = \frac{\tau}{\frac{dv}{dy}} \tag{2.6}$$

（2）运动黏度（ν）为动力黏度和密度的比值，即

$$\nu = \mu/\rho \tag{2.7}$$

运动黏度的法定单位为 m^2/s。过去的单位还有 St（斯）。$1m^2/s=10^4cm^2/s$（St，斯）$=10^6mm^2/s$（cSt，厘斯）。

2.2　液体静力学

2.2.1　液体静压力及其度量

1．定义

处于受力平衡状态的液体所受到的作用在内法线方向上的应力称为液体静压力。

2．液体静压力的度量

液体单位面积所受到垂直于该表面上的压力称为液体静压强，用 p 表示，法定单位为 Pa。1Pa=1N/m^2，1MPa=1×10^{6}Pa，一个大气压约为 0.1MPa。

2.2.2 帕斯卡原理

加在密闭液体上的压强，能够大小不变地由液体向各个方向传递。封闭容器中的平衡液体，其边界上任何一点的压强变化都将等效传递到液体内各点（施加于静止液体上的压强将等效传递到液体中所有各点）。

2.3 液体动力学

2.3.1 连续性方程

若液体不可压缩，则密度ρ为常数，在单位时间内流过管道各截面的液体质量一定是相等的，即

$$\rho A_1 v_1 = \rho A_2 v_2 = \text{const} \tag{2.8}$$

$$q = A_1 v_1 = A_2 v_2 = \text{const} \tag{2.9}$$

式中 q——流量。

也可以写成

$$q_1 = q_2 = \text{const} \tag{2.10}$$

在恒定流动中，通过管道各截面的不可压缩液体的流量是相等的（表现了质量守恒这一客观规律）。

2.3.2 伯努利方程（能量方程）

伯努利方程是以液体流动过程中的流动参数来表示能量守恒的一种数学表达式。根据能量守恒定律，有如下理想液体的伯努利方程。

$$p_1 + \rho g z_1 + \frac{1}{2}\rho v_1^2 = p_2 + \rho g z_2 + \frac{1}{2}\rho v_2^2 \tag{2.11}$$

式（2.11）的物理意义为：理想液体做恒定流动时具有压力能、位能和动能三种能量形式，在任一截面上这三种能量形式都可以相互转换，但三者之和为一定值，即能量守恒。

2.3.3 动量方程

对于做稳定流动的液体，若忽略液体的可压缩性，则液体的密度不变，有

$$\sum \boldsymbol{F} = \rho q(\boldsymbol{v}_2 - \boldsymbol{v}_1) \tag{2.12}$$

若考虑平均流速与实际流速之间存在的误差，应引入动量修正系数 β，故流动液体的动量方程为

$$\sum \boldsymbol{F} = \rho q \beta_2 \boldsymbol{v}_2 - \rho q \beta_1 \boldsymbol{v}_1 \tag{2.13}$$

式中 q——流量；

β_1、β_2——动量修正系数，层流为 1.33，紊流为 1，通常取 1。

动量方程为矢量方程，应用时可根据问题的具体要求向指定方向投影，列出该指定方向的动量方程。

2.4　阻力计算

2.4.1　管道中流体流动的两种状态

英国物理学家雷诺通过大量实验发现，液体在管道内流动时存在层流和紊流两种流动状态。流体质点平稳地沿管轴线方向运动，而无横向运动，流体就像分层流动一样，这种流动称为层流；流体质点不仅有纵向运动，而且有横向运动，处于杂乱无章的不规则运动状态，这种流动称为紊流。

流体做层流流动还是做紊流流动，与流速、管径及液体的黏性有关，不同的流动状态对损失的影响也不同。雷诺归纳出一个综合量——雷诺数 Re 来判断液体的流动状态。由相似理论得到雷诺数的公式为

$$Re=\frac{vD_{\mathrm{H}}}{\nu}=\frac{\rho vD_{\mathrm{H}}}{\mu} \tag{2.14}$$

式中　ν ——液体的运动黏度；

μ ——液体的动力黏度；

ρ ——液体的密度；

v ——液体运动的平均速度；

D_{H}——水力直径。

对圆形断面管，$D_{\mathrm{H}}=d$；对非圆形断面管，$D_{\mathrm{H}}=\dfrac{4A}{\chi}$。式中，$A$ 为过流断面面积；χ 为湿周，即过流断面上液体与固体壁面相接触的周界长度。

2.4.2　圆管层流

微小环形通流截面面积为

$$\mathrm{d}A=2\pi r\mathrm{d}r \tag{2.15}$$

所通过的流量为

$$\mathrm{d}q=u\mathrm{d}A=2\pi ur\mathrm{d}r=2\pi\frac{\Delta p}{4\mu l}(R^2-r^2)r\mathrm{d}r \tag{2.16}$$

积分得

$$q=\int_0^R 2\pi\frac{\Delta p}{4\mu l}(R^2-r^2)r\mathrm{d}r=\frac{\pi R^4}{8\mu l}\Delta p=\frac{\pi d^4}{128\mu l}\Delta p \tag{2.17}$$

平均流速的表达式为

$$v=\frac{q}{A}=\frac{d^2}{32\mu l}\Delta p \tag{2.18}$$

2.4.3 圆管紊流

液体做紊流流动时，其空间任一点处流体质点速度的大小和方向都是随时间变化的，本质上是非恒定流动。工程上处理紊流流动参数时，引入一个时均流速 $\bar{u}$ 的概念，从而把紊流当作恒定流动来看待。紊流时均流速为

$$u=\frac{1}{T}\int_0^T u\mathrm{d}t \tag{2.19}$$

2.5 孔口出流及缝隙流动

在液压传动中，常常碰到液体流经孔口的情况。例如，液压油流经滑阀、锥阀、阻尼孔、节流元件等都属于孔口出流问题。掌握孔口出流一般规律，对解决液压技术领域的具体问题具有非常重要的意义。

2.5.1 孔口出流

对孔前、孔后通道断面 1-1 和 2-2 列出伯努利方程，并设动能修正系数为 1，则有

$$\frac{p_1}{\rho g}+\frac{v_1^2}{2g}=\frac{p_2}{\rho g}+\frac{v_2^2}{2g}+\sum h_\xi \tag{2.20}$$

式中，$\sum h_\xi=h_{\xi 1}+h_{\xi 2}=(\xi+1)v_c^2/(2g)$，它包括两个部分，即流体流经截面突然缩小时的局部阻力系数 $h_{\xi 1}=\xi\frac{v_c^2}{2g}$ 和突然扩大时的局部阻力系数 $h_{\xi 2}=\left(1-\frac{A_c}{A_2}\right)^2\frac{v_c^2}{2g}$，因为 $A_c \ll A_2$，所以 $h_{\xi 2}=\frac{v_c^2}{2g}$。当 A_1=A_2 时，v_1=v_2，代入式（2.20），得

$$v_c=\frac{1}{\sqrt{\xi+1}}\sqrt{\frac{2}{\rho}(p_1-p_2)}=C_v\sqrt{\frac{2\Delta p}{\rho}} \tag{2.21}$$

式中 C_v——小孔速度系数。

流经小孔的流量为

$$q=A_c v_c=C_c A_0 v_c=C_c C_v A_0\sqrt{\frac{2\Delta p}{\rho}}=C_d A_0\sqrt{\frac{2\Delta p}{\rho}} \tag{2.22}$$

式中 A_0——小孔的截面积；

C_c——截面收缩系数，$C_c=A_c/A_0$；

C_d——流量系数，$C_d=C_c C_v$。

2.5.2 缝隙流动

1．压差流

压差流的流量为

$$q_l=\int_0^\delta u_l b\mathrm{d}z=\int_0^\delta\frac{\Delta p}{2\mu l}(\delta-z)z b\mathrm{d}z \tag{2.23}$$

得到

$$q_1 = \frac{b\delta^3}{12\mu l}\Delta p \tag{2.24}$$

2．剪切流

缝隙两端无压差，假设上平板以速度 v 沿 x 轴正向运动，液流速度近似呈线性规律分布，缝隙中液体在上平板带动下层层移动，称为剪切流。在缝隙 z 处流速为

$$u_2 = (v/\delta)z \tag{2.25}$$

则流量为

$$q_2 = \int_0^{\delta} u_2 b\mathrm{d}z = \frac{b\delta}{2}v \tag{2.26}$$

压差和剪切联合作用下的流动，这种流动沿缝隙高度方向上流速分布是由压差流和剪切流叠加而成的。总流量为

$$q = \frac{b\delta^3\Delta p}{12\mu l} \pm \frac{v}{2}b\delta \tag{2.27}$$

平板运动方向与压差流方向一致时，取“+”号；反之取“–”号。

3．平行缝隙流动

（1）壁面固定的平行缝隙中的流动如图 2-1 所示。缝隙两端压力差 $\Delta p = p_1 - p_2$，缝隙高度为 δ，宽度为 b，长度为 l，经理论推导可得

$$q = \frac{b\delta^3}{12\mu l}\Delta p \tag{2.28}$$

式中　μ ——液体的动力黏度。

（2）壁面相对移动的平行缝隙中的流动如图 2-2 所示。

$$q = \frac{b\delta^3}{12\mu l}\Delta p \pm \frac{v_0}{2}b\delta \tag{2.29}$$

式中　v_0 ——平行平板相对运动速度。

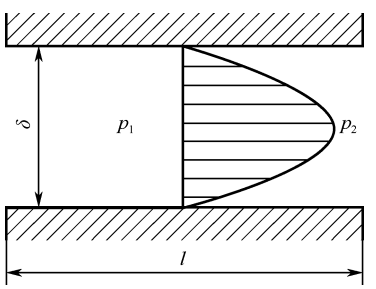


图 2-1　壁面固定的平行缝隙中的流动

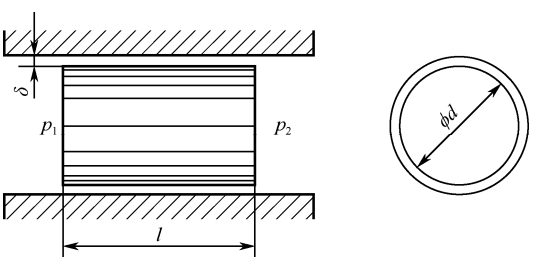


图 2-2　壁面相对移动的平行缝隙中的流动

4．环形缝隙流动

（1）流过同心圆环缝隙的流量为

$$q = \frac{\pi d\delta^3}{12\mu l}\Delta p \tag{2.30}$$

当环形缝隙的壁面有一侧以速度 v_0 运动时，有

$$q=\frac{\pi d\delta^3\Delta p}{12\mu l}\pm\frac{\pi d\delta}{2}v_0 \tag{2.31}$$

（2）流过偏心圆环缝隙的流量为

$$q=\frac{\pi d\delta^3}{12\mu l}\Delta p(1+1.5\varepsilon^2) \tag{2.32}$$

式中 ε ——相对偏心率，即两圆偏心距 e 与圆环同心时的缝隙 δ 的比值，$\varepsilon=e/\delta$ 。

课后题

1．液体密度的定义是什么？

2．某液压油在大气压下的体积是 50L，当压强升高后其体积减少到 49.9L，设液压油的体积弹性模量 K=7000×10^{5}Pa，求压强升高值。

3．液体黏性的大小用什么来度量？

4．液体动力学的三大方程式是什么？

5．圆管层流时，流量和压强差呈什么样的关系？缝隙流动时，流量和压强差呈什么样的关系？

6．薄壁孔的孔口流量公式是什么样的？式中各个符号的意义是什么？

第 3 章　MATLAB 简介

在本书中，所仿真的对象是液压系统，所采用的工具软件是 MATLAB，所以首先用一章来学习一下 MATLAB。

MATLAB 是 matrix&laboratory 两个词的组合，意为矩阵工厂（矩阵实验室）。它是由美国 MathWorks 公司发布的主要面对科学计算、可视化及交互式程序设计的高科技计算环境。它将数值分析、矩阵计算、科学数据可视化及非线性动态系统的建模和仿真等诸多强大功能集成在一个易于使用的视窗环境中，为科学研究、工程设计及必须进行有效数值计算的许多科学领域提供了一种全面的解决方案，并在很大程度上摆脱了传统非交互式程序设计语言（如 C、FORTRAN）的编辑模式，代表了当今国际科学计算软件的先进水平。

MATLAB 和 Mathematica、Maple 并称为三大数学软件。它在数学类科技应用软件中在数值计算方面首屈一指。MATLAB 的功能包括进行矩阵运算、绘制函数和数据、实现算法、创建用户界面、连接其他编程语言的程序等，主要应用于工程计算、控制设计、信号处理与通信、图像处理、信号检测、金融建模设计与分析等领域。

MATLAB 的基本数据单位是矩阵，它的指令表达式与数学、工程中常用的形式十分相似，故用 MATLAB 来解算问题要比用 C、FORTRAN 等语言完成相同的事情简捷得多，并且 MATLAB 也吸收了像 Maple 等软件的优点，使 MATLAB 成为一个强大的数学软件。在新的版本中也加入了对 C、FORTRAN、C++、Java 等编程语言的支持。可以直接调用，用户也可以将自己编写的实用程序导入 MATLAB 函数库中方便自己以后调用。此外，许多 MATLAB 爱好者都编写了一些经典的程序，用户可以直接进行下载就可以使用。

MATLAB 具有如下优势。

（1）高效的数值计算及符号运算功能，能使用户从繁杂的数学运算分析中解脱出来。

（2）具有完备的图形处理功能，实现计算结果和编程的可视化。

（3）友好的用户界面及接近数学表达式的自然化语言，使初学者易于学习和掌握。

（4）功能丰富的应用工具箱（如信号处理工具箱、通信工具箱等），为用户提供了大量方便实用的处理工具。

MATLAB 由一系列工具组成。这些工具方便用户使用 MATLAB 的函数和文件，其中许多工具采用的是图形用户界面，包括 MATLAB 桌面和命令窗口、历史命令窗口、编辑调试器、搜索路径和用于用户浏览帮助、工作空间、文件的浏览器。随着 MATLAB 的商业化及软件本身的不断升级，MATLAB 的用户界面也越来越精致，更加接近 Windows 的标准界面，人机交互性更强，操作更简单。而且新版本的 MATLAB 提供了完整的联机查询、帮助系统，极大地方便了用户的使用。简单的编程环境提供了比较完备的调试系统，程序不必经过编译就可以直接运行，而且能够及时地报告出现的错误及进行出错原因分析。

MATLAB 是一个高级的矩阵/阵列语言，它包含控制语句、函数、数据结构、输入/输出和面向对象编程特点。用户可以在命令窗口中将输入语句与执行命令同步，也可以先编写好一个较大的复杂的应用程序（M 文件）后再一起运行。新版本的 MATLAB 语言是基于最为流

行的 C++语言基础上的，因此语法特征与 C++语言极为相似，而且更加简单，更加符合科技人员对数学表达式的书写格式，使之更利于非计算机专业的科技人员使用。此外，这种语言可移植性好，可拓展性极强，这也是 MATLAB 能够深入到科学研究及工程计算各个领域的重要原因。

MATLAB 是一个包含大量计算算法的集合，其拥有六百多个工程中要用到的数学运算函数，可以方便地实现用户所需的各种计算功能。函数中所使用的算法都是科研和工程计算中的最新研究成果，而且经过了各种优化和容错处理。在通常情况下，可以用它来代替底层编程语言，如 C 和 C++。在计算要求相同的情况下，使用 MATLAB 的编程工作量会大大减少。MATLAB 的这些函数集包括从最简单最基本的函数到诸如矩阵、特征向量、快速傅里叶变换的复杂函数。函数所能解决的问题包括矩阵运算和线性方程组的求解、微分方程及偏微分方程组的求解、符号运算、傅里叶变换和数据的统计分析、工程中的优化问题、稀疏矩阵运算、复数的各种运算、三角函数和其他初等数学运算、多维数组操作及建模动态仿真等。

MATLAB 自产生之日起就具有方便的数据可视化功能，以将向量和矩阵用图形表现出来，并且可以对图形进行标注和打印。高层次的作图包括二维和三维的可视化、图像处理、动画和表达式作图，可用于科学计算和工程绘图。新版本的 MATLAB 对整个图形处理功能做了很大的改进和完善，使它不仅在一般数据可视化软件都具有的功能（如二维曲线和三维曲面的绘制和处理等）方面更加完善，而且对于一些其他软件所没有的功能（如图形的光照处理、色度处理及四维数据的表现等），MATLAB 同样表现了出色的处理能力。同时对一些特殊的可视化要求，如图形对话等，MATLAB 也有相应的功能函数，保证了用户不同层次的要求。另外，新版本的 MATLAB 还着重在图形用户界面（GUI）的制作上做了很大的改善，对这方面有特殊要求的用户也可以得到满足。

新版本的 MATLAB 可以利用 MATLAB 编译器及 C/C++数学库和图形库，将自己的 MATLAB 程序自动转换为独立于 MATLAB 运行的 C 和 C++代码。允许用户编写可以和 MATLAB 进行交互的 C 或 C++语言程序。另外，MATLAB 网页服务程序还允许在 Web 应用中使用自己的 MATLAB 数学和图形程序。MATLAB 的一个重要特色就是具有一套程序扩展系统和一组称为工具箱的特殊应用子程序。工具箱是 MATLAB 函数的子程序库，每一个工具箱都是为某一类学科专业和应用而定制的，主要包括信号处理、控制系统、神经网络、模糊逻辑、小波分析和系统仿真等方面的应用。

3.1 MATLAB 操作环境

3.1.1 启动和退出

以 Windows 操作系统为例，进入 Windows 后，选择“开始”→“程序”→“MATLAB”，便可以进入 MATLAB 主窗口。如果安装时选择在桌面上生成快捷方式，也可以单击快捷方式直接启动。MATLAB 的启动界面如图 3-1 所示。

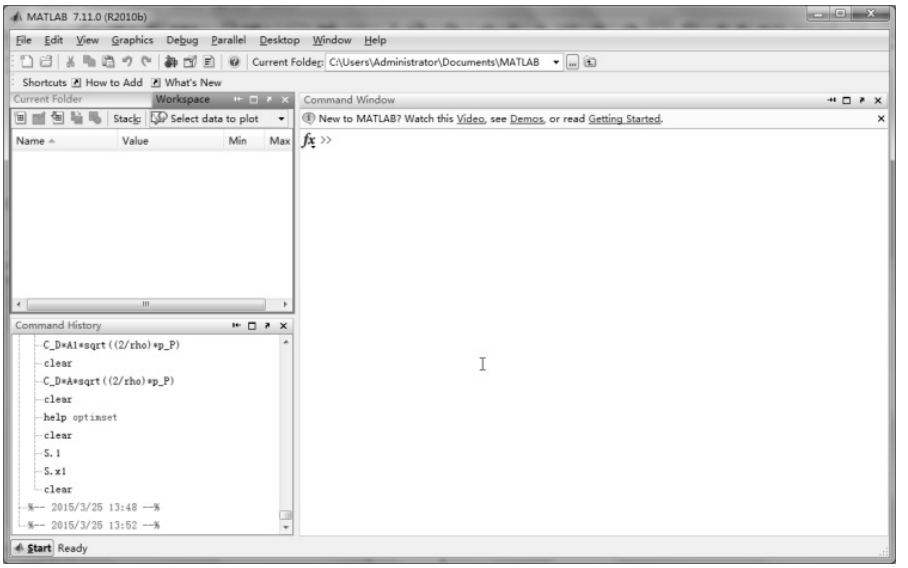


图 3-1　MATLAB 的启动界面

在启动 MATLAB、命令编辑区显示帮助信息后，将显示符号“|”。符号“|”表示 MATLAB 已准备好，正等待用户输入命令，这时就可以在提示符“|”后面输入命令，按下回车键后，MATLAB 就会解释执行所输入的命令，并在命令后面给出计算结果。如果在输入命令后再以分号结束，则不会显示结果。

举一个简单的例子，假设要计算一个半径为 2.5m 的圆的面积，在命令窗口（Command Window）中的操作如下。

```
>>area = pi*2.5^2
area =
      19.6350
```

当回车键敲下的一瞬间，结果被计算出来了，并存储到一个叫 area 的变量中（其实是一个 1×1 的数组），而且这个变量能进行进一步的计算。注意：pi 是 MATLAB 预先定义好的变量，表示π，所以 pi 不需要预先声明。

如果一个语句在一行内写不下，可能要另起一行接着写，在这种情况下需要在第一行末打上空格加英文状态下的 3 个点（...），再开始第二行的书写。

举例如下，下面这两个语句是等价的。

```
x1=1+1/2+1/3+1/4+1/5+1/6
```

和

```
x1=1+1/2+1/3+1/4 ...
+1/5+1/6
```

将一系列命令写入一个文件，在命令窗口中输入此文件的文件名，然后 MATLAB 就开始执行这个文件，而不是用直接在命令窗口中输入命令的方法，这样的文件叫作脚本文件（script file）。由于脚本文件的扩展名为“.m”，所以也叫它 M 文件。

3.1.2 菜单栏及其功能

在 MATLAB 界面的最上部是 MATLAB 的菜单栏，用户可以通过单击菜单栏来了解 MATLAB 的功能，这些都和普通的 Windows 系统一致，用户可以在不断的学习中逐渐摸索掌握，最开始可以忽略这一步骤。

3.1.3 命令窗口

MATLAB 的命令窗口是用户和 MATLAB 直接打交道最多的窗口，它具有两大主要功能。

（1）提供用户输入命令的操作平台，用户通过该窗口输入命令和数据。

（2）提供命令执行结果的显示平台，该窗口显示命令执行的结果。

计算机安装好 MATLAB 之后，双击 MATLAB 图标，就可以进入命令窗口，此时意味着系统处于准备接受命令的状态，可以在命令窗口中直接输入命令语句。

MATLAB 语句形式为：>>变量=表达式。

通过等号将表达式的值赋予变量。当按下回车键时，该语句被执行。语句执行之后，窗口自动显示出语句执行的结果。注意：本书中有的程序省略了“>>”指示符。

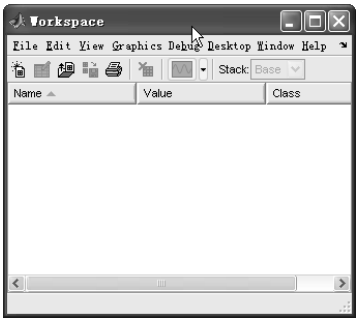


图 3-2 MATLAB 的工作空间

使用方向键和控制键可以编辑、修改已输入的命令，↑键回到上一行命令，↓键回到下一行命令。分号“;”的作用是指令执行结果将不显示在屏幕上，但变量将驻留在内存中。

3.1.4 工作空间

MATLAB 的工作空间如图 3-2 所示。

工作空间中的变量以变量名（Name）、数值（Value）和类型（Class）的形式显示出来。双击某个变量将进入矩阵编辑器（Array Editor），可以直接观察变量中的具体元素值，也可以直接修改这些元素。

3.1.5 历史命令窗口

历史命令窗口（History Command Window）用于记录用户在命令窗口中输入的命令，其顺序是按逆序排列的，即最早的命令排在最下面，最晚的命令排在最上面。这些命令会一直存在下去，直到被人为删除。双击这些命令可使其再次执行。在历史命令窗口中删除一个或多个命令，可以先选择，然后单击右键，这时就有一个弹出菜单出现，选择 Delete Section，任务就完成了。

3.1.6 编辑调试器

编辑调试器一般用于创建或修改已存在的 M 文件。当打开或修改一个 M 文件时，编辑调试器会自动被调用。创建一个 M 文件的方法如下。

（1）在菜单中选择“File/New/M-file”创建。

（2）单击图标。

打开一个已存在的 M 文件也有两种方法，具体如下。

（1）在菜单中选择“File/Open”打开。

（2）单击图标，然后选择 M 文件所在的路径。

编辑调试器是个重要的程序的文档编辑器，MATLAB 语言的一些特性会用不同的颜色表现出来。M 文件中的注释用绿色表示，变量和数字用黑色来表示，字符变量用红色表示，语言的关键字用蓝色表示。图 3-3 显示了一个包含 M 文件的简单的编辑调试窗口。这个文件用于计算半径已知的圆的面积并输出结果。

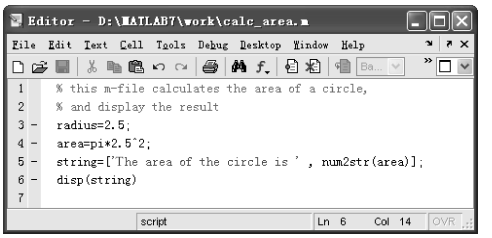


图 3-3　显示了一个包含 M 文件的简单的编辑调试窗口

该 M 文件如下。

```
% this m-file calculates the area of a circle,
% and display the result
radius=2.5;
area=pi*2.5^2;
string=['The area of the circle is ' , num2str(area)];
disp(string)
```

当 M 文件保存完后，在命令窗口中输入这个 M 文件的名字，它就可以被执行了。上面的 M 文件的输出结果为

```
>>calc_area
The area of the circle is 19.635 27
```

3.5.4 节将介绍编辑调试器在调试方面的应用。

3.1.7　图形窗口（Figure Window）

图形窗口主要用于显示 MATLAB 图形。它所显示的图形可以是数据的二维或三维坐标图、图片或用户图形接口（称为界面）。下面是一个简单的 M 文件，用于计算函数 sin(x)并打印出图像（见图 3-4）。

```
% this m-file calculates and plots the
% function sin(x) for 0<=x<=6
x=0:0.1:6;
y=sin(x);
plot(x,y)
```

如果此文件以 sin_x.m 为文件名保存，那么在命令窗口中输入此文件名即可执行文件了。

当 M 文件被编译后，MATLAB 将会打开一个图形窗口，并在窗口中打印出函数 sin(x)的图像。

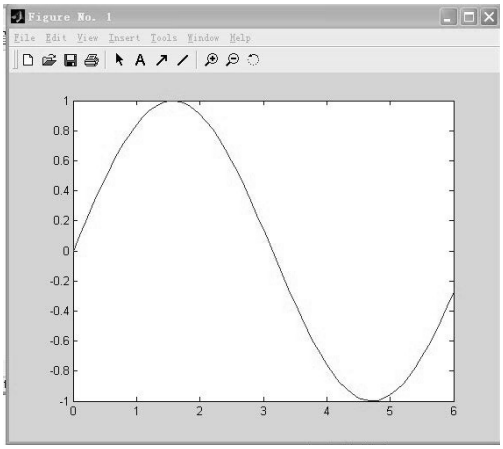


图 3-4　函数 sin(x)的图像

3.1.8　MATLAB 的帮助

有三种方法可以得到 MATLAB 的帮助。最好的方法是使用帮助窗口。单击 MATLAB 界面工具栏上的图标，也可以在命令窗口中输入 helpdesk 或 helpwin 来启动帮助窗口。此外，还可以通过浏览 MATLAB 参考证书或搜索特殊命令的细节得到帮助。帮助窗口如图 3-5 所示。

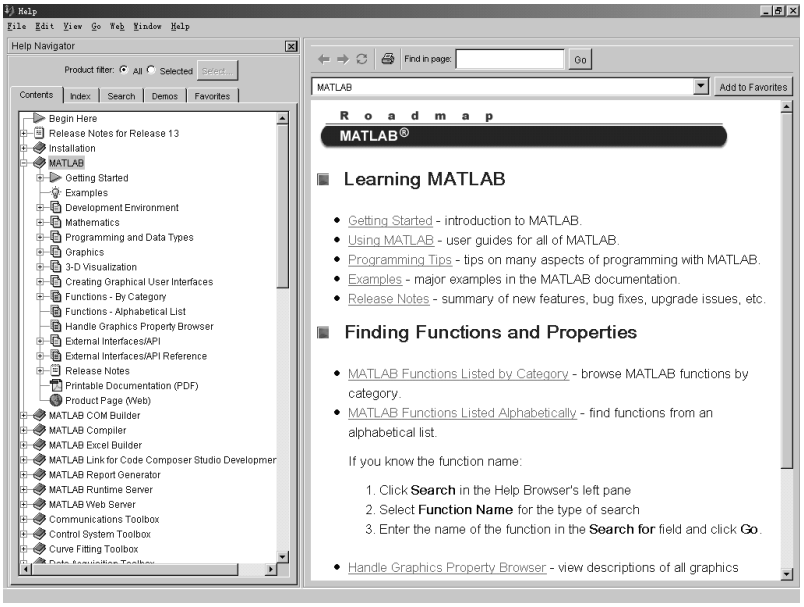


图 3-5　帮助窗口

另外，还有两种使用命令行得到帮助的方法。第一种方法是在 MATLAB 命令窗口中输入 help 或 help 和所需要的函数的名字。如果在命令窗口中只输入 help，那么将会显示一连串的函数。如果有一个专门的函数名或工具箱的名字包含在内，那么将会提供这个函数或工具箱。第二种方法是通过 lookfor 命令得到帮助。lookfor 命令与 help 命令不同，help 命令要求与函

数名精确匹配，而 lookfor 命令只要求与每个函数中的总结信息有匹配。lookfor 命令比 help 命令运行起来慢得多，但它提高了得到有用信息的机会。举个例子，假设想找到一个求矩阵的逆矩阵的函数，但是 MATLAB 中没有叫 inverse 的函数，这时 help 命令就不起作用了，只能用 lookfor 命令。

如果你是个 MATLAB 新手，一些示例可能有助于你理解它的功能。可以在命令窗口中输入 demo 或在启动平台中选择 demos 来运行 MATLAB 内建的示例。在任何时候你都可以用 clc 命令清空命令窗口中的内容，可以用 clf 命令清空当前图形窗口中的内容。在工作空间中的变量可用 clear 命令清除。正如我们看到的，工作空间中的变量在独立的命令和 M 文件间执行时，可能会出现第一个问题中的变量存留在工作区而影响到第二个问题解决的情况。为了避免这种情况的发生，在新的计算开始之前，应当用 clear 命令清空工作区。

另一个重要的命令是退出命令。如果一个 M 文件运行时间过长，里面可能含有无限循环而没有结束。在这种情况下，可在命令窗口中输入 control+c（简写为^c）。输入这个命令的方法是将光标放在命令窗口中，按住控制键 Ctrl，然后按下 C 键。当 MATLAB 检测到^c 时，它将中断正在运行的程序并回到命令行提示符处。感叹号（!）是另一个重要的特殊字符。它的特殊作用是给计算机操作系统发送一个命令，在感叹号后的字符会发送给计算机并执行，这与在计算机的命令行提示符后输入字符效果是一样的。这种特性使系统命令更容易植入 MATLAB 程序。

最后，能用 diary 命令记录下在 MATLAB 运行过程中每个线程所做的事。命令的格式如下。

```
diary filename
```

当这个命令被执行后，所有在命令窗口中的输入和输出将会被记录在 diary 文件中。这是一个非常重要的工具，当 MATLAB 发生错误而中断时，利用它可以重建重要的事件。diary off 命令中止写入 diary 文件，diary on 命令重新开始写入。

3.1.9　MATLAB 搜索路径

在 MATLAB 中，可用 MATLAB 搜索条寻找 M 文件。在文件系统中，MATLAB 的 M 文件是以目录形式被组织的。

如果用户在 MATLAB 提示符后输入一个名字，那么 MATLAB 的解释器将按以下顺序寻找这个名字。

（1）先检查这个名字是否是个变量名。如果它是一个变量名，MATLAB 将会显示出这个变量的值。

（2）然后检查它是否是内建函数或命令。如果是，则执行对应的函数或命令。

（3）检查是不是在当前目录下的一个 M 文件。如果是，则执行对应的函数或命令。

（4）检查是不是在 MATLAB 搜索路径的所有目录下的一个 M 文件。如果是，则执行对应的函数或命令。

如果首先检查到的是变量名，且这个变量名与 MATLAB 的某一个函数或命令同名，那么这个函数或命令将变得无法被访问。这是初学者易犯的错误之一。

如果有多个函数或命令重名，那么 MATLAB 将会执行在搜索路径中找到的第一个，其他

的将不会被执行。对于初学者，这也是一个常见的问题，往往使 M 文件的名字与 MATLAB 内建函数或命令重名，从而导致函数或命令不能被访问。不要创建和 MATLAB 内建函数或命令同名的 M 文件。

可以运用启动平台中的路径工具（path tool）随时检查和修改路径，或者在命令窗口中输入 editpath 命令。路径工具如图 3-6 所示。它允许使用者添加、删除路径和改变在目录中的顺序。

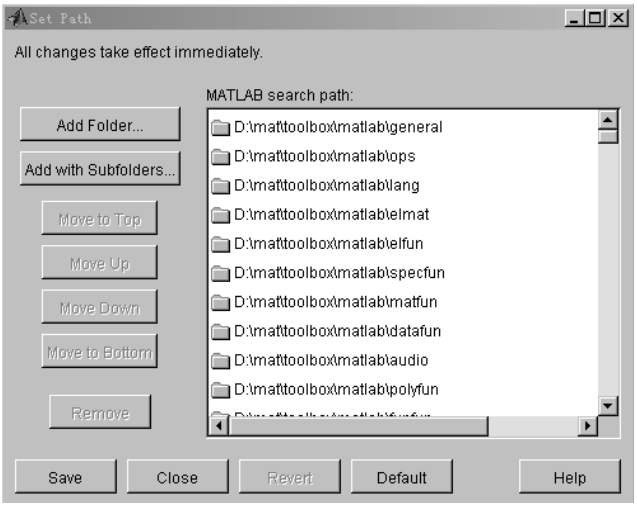


图 3-6　路径工具（path tool）

3.2　MATLAB 数值计算

MATLAB 是一门计算语言，它的运算指令和语法基于一系列基本的矩阵运算及它们的扩展运算，它支持的数值元素是复数，这也是 MATLAB 区别于其他高级语言的最大特点之一，它给许多领域的计算带来了极大的方便。

3.2.1　MATLAB 数据类型

MATLAB 包括四种基本的数据类型，即双精度数组、字符串数组、元胞数组、构架数组。数值之间可以相互转化，这为其计算功能开拓了广阔的空间。

MATLAB 程序的基本数据单元是数组。一个数组是以行和列组织起来的数据集合，并且拥有一个数组名。标量在 MATLAB 中也被当作数组来处理，即它被看作只有一行一列的数组。

数组可以定义为向量或矩阵。向量一般用来描述一维数组，而矩阵往往用来描述二维或多维数组。在本书中，当讨论一维数组时用向量表示，当讨论二维或多维向量时用矩阵表示。如果在特殊情况下同时遇到这两种数组，就把它们通称为“数组”。

数组的大小（size）由数组的行数和列数共同决定，注意行数在前。一个数组所包含的数据多少可由行数乘列数得到。

数组中的单个数据是可以被访问的，访问的方法是数组名后带一个括号，括号内是这个数据所对应的行标和列标。如果这个数组是一个行向量或列向量，则只需要一个下标。一个

MATLAB 变量是一段包含一个数组的内存区，并且拥有一个用户指定的变量名。通过适当的命令和它的变量名随时可以调用它和修改它。

MATLAB 的变量名必须以字母开头，后面可以跟字母、数字和下画线（_）。只有前 31 个字符是有效的，如果超过了 31 个字符，其余的字符将被忽略。如果声明两个变量，两个变量名只有第 32 个字符不同，那么 MATLAB 会将它们当作同一个变量对待。

当编写程序时，给变量起一个有意义的名字非常重要。有意义的名字可极大地提高程序的可读性和可维护性。像 day、month 和 year 这样的名字意义非常明确，即使第一次看到也能理解。尽管空格不能用在 MATLAB 变量名中，但是可以用下画线代替空格创造出有意义的变量名。例如，changerate 可以写成 change_rate。

在所写程序的开头列出一个数据字典（data dictionary）十分重要。数据字典列举了在本程序中用到的所有变量的定义，它的定义应包括本条目所要描述的内容和它在执行时所在的单元。当编写程序时，编写数据字典看似没有必要。但是设想一下，在过了一段时间后，你或其他人要对此程序进行修改，这时数据字典就显得十分有用了。

在 MATLAB 语言中是区分字母大小写的，也就是说，大写字母和小写字母代表的内容是不同的。所以，变量 NAME、Name、name 在 MATLAB 中是不同的。如果已用过的小写变量名与一个新建的大写变量名重名，这时使用时要特别小心。在一般情况下，一律用小写字母来表示。

两个最常见的变量类型是 char 型和 double 型。无论什么时候，将一个数值赋值于一个变量名，那么 MATLAB 将自动建立一个 double 型变量。例如，下面语句创建了一个以 var 为变量名的 double 型变量，包含一个 double 型的单个元素，存储了复数值（1+i）。

```
var=1+i
```

char 型的变量包括由 16 位数值构成的标量或数组，每一个 16 位数代表一个字符。这个类型的变量经常用于字符串操作，当将一个字符或字符串赋给一个变量名时，系统会自动建立一个 char 型变量。例如，下面的这个语句创建了一个 char 型变量 comment，并存储了一个字符串在其内。当这个语句执行后，系统将会建立一个 1×26 的字符串数组。

```
comment='this is a character string'
```

像 C 语言，变量类型和变量在使用之前必须强制声明。这种语言叫作强类型语言。相对地，像 MATLAB 这样的叫作弱类型语言。通过简单的赋值形式就可以创建变量，变量类型取决于创建时的类型。

3.2.2　MATLAB 变量的初始化

当变量初始化时，MATLAB 将会自动建立变量。有三种方法初始化 MATLAB 中的变量。

（1）用赋值语句初始化变量。

（2）用 input 函数从键盘输入初始化变量。

（3）从文件读取一个数据。

这里重点讨论第一种方法。

1．用赋值语句初始化变量

最简单的创建和初始化一个变量的方法是用赋值语句赋予变量一个或多个值。赋值语句的一般形式如下。

```
var = expression
```

其中，var 是变量名；expression 可以是一个标量、一个数组或常量、其他变量和数学运算符号的联合。这个表达式（expression）的值是通过一般的数学运算法则计算出来的，然后将产生的结果存储到变量 var 中。下面是一些用赋值语句初始化的变量。

```
var=40*i
var2=var/5
array=[1 2 3 4]
x=1,y=2
```

第一个例子创建了一个 double 类型的标量变量，存储了一个虚数 40i。第二个例子创建了一个变量 var2，把 var/5 的值存储于内。第三个例子创建了一个数组 array，并存储了一个 4 个元素的行向量。最后一个例子显示了多个赋值语句可写在同一行，中间用逗号或分号隔开。注意：如果在赋值语句执行时变量已经存在，那么这个变量原有的值将被覆盖。

正如第三个例子显示的，数据数组也可以初始化变量。可以用方括号、空格、逗号和分号建立数组。所有元素按行列排序，换句话说，每一行的值从左向右，顶部的行置于最前，底部的行置于最后。在一行内单个数值可用空格或逗号隔开，而行与行之间则用分号隔开，或者另起一行书写。表 3-1 中的表达式都是合法的，能用于建立一个变量。

表 3-1　初始化变量时的表达式举例

表　达　式	含　　义
[3.4]	这个表达式创建了 1×1 数组（一个标量），包含数值 3.4，这时方括号可以省略
[1.0 2.0 3.0]	这个表达式创建了 1×3 数组，为一维行向量[1 2 3]
[1.0;2.0;3.0]	这个表达式创建了一个 3×1 数组，为一维列向量 $\begin{bmatrix}1\\2\\3\end{bmatrix}$
[1,2,3;4,5,6]	这个表达式创建了一个 2×3 数组，为矩阵 $\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}$

注意：一个数组每一行元素的个数必须完全相同，每一列元素的个数也必须完全相同。

用于初始化数组的表达式可以包括代数符号或事先已经定义好的数组。例如，赋值语句

```
a=[0 1+7]
b=[a(2) 7 a]
```

定义了数组 a=[0 8]和数组 b=[8 7 0 8]。

当创建一个数组时，不是每一个元素都必须定义。如果要定义一个特殊的数组，或只有一个或几个元素没有定义，那么之前的那些元素将会自动创建，并初始化为 0。例如，如果数

组事先没有定义，语句 c(2,3)=5 将会创建一个矩阵 $c=\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 5\end{bmatrix}$。相似的情况是，指定一个值赋予一个存在的数组，但超过了这个数组的大小。例如，假设存在一个数组 d=[1 2]，下面这个语句

```
d(4)=4
```

将会创建数组 d=[1 2 0 4]。

每个赋值语句末的分号有特殊的目的，即无论在何时一个变量在赋值语句中被赋值，分号将会终止变量值的重复。如果句末没有分号，变量值将会自动显示在命令窗口中。如果句末有分号，这种重复将会消失。重复是一个用于检查工作极好的方法，但是它降低了运行速度。因此，在一般情况下总是禁止重复。尽管如此，重复计算的结果提供了一个强大的应急调试器。如果不能确定一个特定的赋值语句的结果是多少，这时可以去掉这个语句后的分号，当这个语句被编译时，结果会显示在命令窗口中。

2．用捷径表达式赋值

创建一个小数组用一一列举出元素的方法是比较容易的，但是当创建包括成千上万个元素的数组时怎么办？把每一个元素列举出来则不太现实。

MATLAB 提供一种专门的捷径标记法，这种方法用克隆运算符（colon operator）适用于上述情况。克隆运算符指定一系列的数值，它指定了这个系列数的第一个值、步长和最后一个值。它的一般顺序如下。

```
first:incr:last
```

其中，first 代表数组中的第一个值；incr 代表步长；last 代表这个数组中的最后一个值。如果步长为 1，那么步长可省略，而变成 first:last 格式。

例如，表达式 1:2:10 是创建一个 1×5 行向量[1 3 5 7 9]的简便方法。

```
>>x=1:2:10
x =
     1     3     5     7     9
```

捷径表达式可以联合转置运算符（'）来初始化行向量或更加复杂的矩阵。转置运算符可以在需要的情况下完成行和列的转换。例如，表达式

```
f = [1:4]'
```

先产生一个 4 个元素的行向量[1 2 3 4]，然后将其转换成 4 个元素的列向量 $f=\begin{bmatrix}1\\2\\3\\4\end{bmatrix}$。

3.2.3　多维数组

MATLAB 的数组可能是一维或多维的。一维数组可以形象地看作是一系列的数罗列出来，用一个下标就可以调用数组中的元素。

许多数据的类型需要多变量的函数。在这种情况下，就需要两个下标来调用这个数组中的元素：第一个下标选择行，第二个下标选择列。这样的数组叫作二维数组。二维数组中元素的个数取决于这个数组的行数和列数。

一个有 m 行和 n 列的二维数组包括 $m\times n$ 个元素，这些元素在计算机的内存中将会占有 $m\times n$ 个连续的内存空间。这些数组中的元素在内存中是如何排列的呢？MATLAB 以列主导顺序分配数组中的元素。也就是说，内存先分配第一列的元素，然后第二列、第三列……，以此类推，直到所有列都被分配完。

当数据重复在命令窗口中时，整数以整型形式显示，其他值将以默认格式显示。MATLAB 的默认格式是精确到小数点后 4 位。如果一个数太大或太小，那么将会以科学计数法的形式显示。改变默认输出格式要用到 format 命令。

3.2.4 标量运算和数组运算

在 MATLAB 中，赋值语句的一般形式如下。

```
variable_name = expression
```

赋值语句计算出等号右边表达式的值，然后赋值于等号左边的变量名。注意：这个等号并不是传统意义上的等号，它的意义是存储表达式的值到左边的变量。由于这个原因，等号在这里应叫作赋值号。

位于赋值号右边的表达式，可以包含标量、数组、括号和数学符号的任一个有效联合运算。两个标量间的数学运算符见表 3-2。

表 3-2 两个标量间的数学运算符

运算符名称	代 数 形 式	MATLAB 形式
加号	$A+B$	A+B
减号	$A-B$	A–B
乘号	$A\times B$	A*B
除号	$A\div B$	A/B
指数	A^B	A^B

当需要时，可以运用括号来控制运算顺序。括号内的表达式优先于括号外的表达式来计算。

MATLAB 对数组提供了两种不同类型的运算：一种是数组（array）运算，另一种是矩阵（matrix）运算。数组运算是一种用于元素对元素的运算。也就是说，这个运算是针对两个数组相对应的运算使用的。例如，$\boldsymbol{a}=\begin{bmatrix}1 & 3\\ 2 & 4\end{bmatrix}$，$\boldsymbol{b}=\begin{bmatrix}-1 & 3\\ -2 & 4\end{bmatrix}$，那么 $\boldsymbol{a}+\boldsymbol{b}=\begin{bmatrix}0 & 6\\ 0 & 5\end{bmatrix}$。注意，两个数组的行与列必须相同，否则，MATLAB 将产生错误。

数组运算可以用于数组与标量的运算。当一个数组和一个标量进行运算时，标量将会和数组中的每一个元素进行运算。例如，$\boldsymbol{a}=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$，则 $\boldsymbol{a}+4=\begin{bmatrix}5 & 6\\ 7 & 8\end{bmatrix}$。

相对地，矩阵运算则遵守线性代数的一般规则，如矩阵的乘法。在线性代数中，$\boldsymbol{c}=\boldsymbol{a}\times\boldsymbol{b}$ 的定义为

$$\boldsymbol{c}(i,j)=\sum_{k=1}^{n}\boldsymbol{a}(i,k)\boldsymbol{b}(k,j) \tag{3.1}$$

例如，$\boldsymbol{a}=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$，$\boldsymbol{b}=\begin{bmatrix}-1 & 3\\ -2 & 1\end{bmatrix}$，那么 $\boldsymbol{a}\times\boldsymbol{b}=\begin{bmatrix}-5 & 5\\ -11 & 13\end{bmatrix}$。注意：在矩阵相乘中，$\boldsymbol{a}$ 阵的列数必须等于 $\boldsymbol{b}$ 阵的行数。

MATLAB 用一个特殊的符号来区分矩阵运算和数组运算。在需要区分两者的不同时，把点置于符号前来指示这是一个数组运算（例如，.*）。表 3-3 给出了一些常见的数组运算和矩阵运算。

表 3-3　数组运算和矩阵运算

运　　算	MATLAB 形式	注　　释
数组加法	A+B	数组加法和矩阵加法相同
数组减法	A−B	数组减法和矩阵减法相同
数组乘法	A.*B	A 和 B 中的元素逐个对应相乘。两个数组之间必须有相同的形，或者其中一个是标量
矩阵乘法	A*B	A 和 B 的矩阵乘法。A 的列数必须和 B 的行数相同
数组右除法	A./B	A 和 B 中的元素逐个对应相除：A(i,j)/B(i,j)。两个数组之间必须有相同的形，或者其中一个是标量
数组左除法	A.\B	A 和 B 中的元素逐个对应相除：B(i,j)/A(i,j)。两个数组之间必须有相同的形，或者其中一个是标量
矩阵右除法	A/B	矩阵除法，等价于 A*inv(B)。inv(B)是 B 的逆阵
矩阵左除法	A\B	矩阵除法，等价于 inv(A)*B。inv(A)是 A 的逆阵
数组指数运算	A.^B	A 和 B 中的元素逐个进行 A(i,j)^B(i,j)和 A(i,j)/B(i,j)运算。两个数组之间必须有相同的形，或者其中一个是标量

初学者往往会混淆数组运算和矩阵运算。在一些情况下，两者相互替换会导致非法操作，MATLAB 将会报告产生了错误。在另一些情况下，两种运算都是合法的，那么这时 MATLAB 进行错误的运算，并产生错误的结果。当进行方阵运算时，极易产生这样的错误。两个方阵具有相同的大小，两者之间的数组运算和矩阵运算都是合法的，但产生的结果完全不同。在这种情况下，要万分小心。

3.3　MATLAB 符号运算

3.3.1　符号运算基础

MATLAB 提供了符号数学工具箱（Symbolic Math Toolbox），大大增强了 MATLAB 的功能。

1．符号运算简介

符号数学工具箱是操作和解决符号表达式的符号数学工具（函数）集合，有复合、简化、微分、积分及求解代数方程和微分方程的工具，另外还有一些用于线性代数的工具，用于求解逆、行列式、正则形式的精确结果。

1）符号表达式

符号表达式是代表数字、函数、算子和变量的 MATLAB 字符串或字符串数组。不要求变量有预先确定的值。符号方程式是含有等号的符号表达式。

2）符号变量和符号表达式

在 MATLAB 中，用 sym 或 syms 定义符号变量和符号表达式，定义多个符号变量之间用空格分开。例如，“sym a”定义了符号变量 a，“syms a b”定义了符号变量 a 和 b，“syms a b c x;f=sym('a*x^2+b*x+c')”创建表达式 $f=ax^2+bx+c$，“fcn=sym('f(x)')”创建函数 f(x)。

2．控制系统中常用的符号运算

下面对控制系统中常用的符号运算进行介绍，其他的符号运算读者可以参考 MATLAB 的帮助文档或其他关于符号数学工具箱的书籍，此处不再赘述。

控制系统中常用的符号运算有微积分、拉普拉斯变换和 Z 变换等积分变换。

diff 是求微分最常用的函数，其输入参数既可以是函数表达式，也可以是符号矩阵。常用的格式是：diff(f,x,n)。这表示 f 关于 x 求 n 阶导数。

int 是求积分最常用的函数，其输入参数可以是函数表达式。常用的格式是：int(f,r,x0,x1)。其中，f 为所要积分的表达式；r 为积分变量；若为定积分，则 x0、x1 分别为积分下限和积分上限。

【例 3-1】 已知表达式 $f=\sin(ax)$，分别对其中的 x 和 a 求导。

解：MATLAB 程序代码如下。

```
>>syms a x
>>f=sin(a*x)
>>dfx=diff(f,x)
>>dfa=diff(f,a)
```

程序的执行结果如下。

```
f=
sin(a*x)
dfx=
cos(a*x)*a
dfa=
cos(a*x)*x
```

【例 3-2】 已知表达式 $f=x\lg(1+x)$，求对 x 的积分和 x 在[0,1]上的积分值。

解： MATLAB 程序代码如下。

```
>>syms x
>>f=x*log(1+x)
```

```
>>int1=int(f,x)
>>int2=int(f,x,0,1)
```

程序的执行结果如下。

```
f=
x*log(1+x)
int1=
x/2-log(x+1)/2+x^2*(log(x+1)/2-1/4)
int2=
1/4
```

3.3.2　复数和复变函数运算

复数及在其基础上发展起来的复变函数这一重要的数学分支，解决了许多实变函数无法解决的问题，有着广泛的工程应用。

复变函数是控制工程的数学基础，常用来描述控制系统模型的传递函数就属于复变函数。MATLAB 支持在运算和函数中使用复数或复数矩阵，还支持复变函数运算。

在 MATLAB 中，可以采用符号数学工具箱进行拉普拉斯变换（也称拉氏变换）和拉普拉斯反变换（也称拉氏反变换），采用的函数是 laplace 和 ilaplace。使用前，用 syms 函数设置有关的符号变量。

拉氏变换函数 laplace 的格式为

```
L=laplace(F)
```

其中，F 是时域函数表达式，约定的自变量是 t，得到的拉氏函数是 L。

拉氏反变换函数 ilaplace 的格式为

```
F=ilaplace(L)
```

它将拉氏函数 L 变换为时域函数 F。

【例 3-3】 求函数 $f_1(t)=e^{at}$（a 为实数）、$f_2(t)=t-\sin t$ 的拉氏变换。

解：MATLAB 程序代码如下。

```
>>syms t s a
>>f1=exp(a*t);f2=t-sin(t);
>>L1=laplace(f1),L2=laplace(f2)
```

程序的执行结果如下。

```
L1=
1/(s-a)
L2=
1/s^2-1/(s^2+1)
```

【例 3-4】 求函数 $F_1(s)=\dfrac{1}{s(1+s^2)}$、$F_2(s)=\dfrac{s+3}{(s+1)(s+2)}$ 的拉氏反变换。

解：MATLAB 程序代码如下。

```
>>syms t s
>>F1=1/(s*(1+s^2));F2=(s+3)/((s+1)*(s+2));
>>f1=ilaplace(F1),f2=ilaplace(F2)
```

程序的执行结果如下。

```
f1=
1-cos(t)
f2=
2*exp(-t)-exp(-2*t)
```

3.4 MATLAB 常用绘图命令

在控制系统仿真中常常用到绘图，如绘制系统的响应曲线、根轨迹或频率响应曲线等。

MATLAB 提供了丰富的绘图功能，在命令窗口中输入 help graph2d 可得到所有画二维图形的命令，输入 help graph3d 可得到所有画三维图形的命令。

3.4.1 基本的绘图命令

plot(x1,y1,option1,x2,y2,option2,…)：x1、y1 给出的数据分别为 x、y 轴坐标值；option1 为选项参数，以逐点连折线的方式绘制一个二维图形；可以类似地绘制第二个、第三个二维图形。

选项参数 option 定义了图形曲线的颜色（用颜色英文单词的第一个字母表示，例如，r 表示红色，g 表示绿色，b 表示蓝色，w 表示白色，k 表示黑色，m 表示紫色，c 表示青色）、线型（如-.、*、-、+、o、x、square、.等）及标示符号，它由一对单引号括起来。

3.4.2 图形窗口处理命令

常用的图形窗口处理命令如下。

（1）打开不同的图形窗口命令 figure。figure(1),figure(2),…,figure(n)：它用来打开不同的图形窗口，以便绘制不同的图形。

（2）图形窗口拆分命令 subplot。subplot(m,n,p)：分割图形显示窗口，m 表示上下分割个数，n 表示左右分割个数，p 表示子图编号。

3.4.3 坐标轴相关命令

（1）在默认情况下，MATLAB 将自动选择图形的横、纵坐标的比例，当然也可以用 axis 命令控制。

axis([xmin xmax ymin ymax])：[xmin xmax ymin ymax]中分别给出 x 轴和 y 轴的最大值和最小值。

（2）在控制系统中，还会用到半对数坐标轴，MATLAB 中常用的对数坐标绘制命令为 semilogx。

semilogx：绘制以 x 轴为对数坐标（以 10 为底）、y 轴为线性坐标的半对数坐标图形。

3.4.4　文字标示命令

常用的文字标示命令如下。

（1）text(x,y,'字符串')：在图形的指定坐标位置(x,y)处标示单引号括起来的字符串。

（2）gtext('说明文字')：利用鼠标在图形的某一位置标示说明文字。执行完绘图命令后再执行 gtext('说明文字')命令，就可以在屏幕上得到一个光标，然后用鼠标选择放置说明文字的位置。

（3）title('字符串')：在所画图形的最上端显示说明该图形标题的字符串。

（4）xlabel('字符串')、ylabel('字符串')、zlabel('字符串')：设置 x、y、z 坐标轴的名称，输入特殊的文字需要用反斜杠（\）开头。

（5）legend('字符串 1', '字符串 2',…,'字符串 n')：在屏幕上打开一个小视窗，然后依据绘图命令的先后顺序，用对应的字符串区分图形上的线。

3.4.5　在图形上添加或删除栅格命令

在图形上添加或删除栅格命令如下。

（1）grid on：给当前坐标系加上栅格线。

（2）grid off：从当前坐标系中删去栅格线。

（3）grid 命令是一个交替转换命令，即执行一次转变一个状态（相当于 grid on、grid off）。

3.4.6　图形保持或覆盖命令

图形保持或覆盖命令如下。

（1）hold on：把当前图形保持在屏幕上不变，同时允许在这个坐标内绘制另外一个图形。

（2）hold off：使新图覆盖旧图。

（3）hold 命令是一个交替转换命令，即执行一次转变一个状态（相当于 hold on、hold off）。

【例 3-5】 绘图命令使用举例。绘制[0,4π]区间上的 x_1=10sin$t$ 和 x_2=5cos$t$ 曲线，并要求：（1）x_1 曲线的线型为点画线，颜色为红色，数据点标记为加号；（2）x_2 曲线为虚线，颜色为蓝色，数据点标记为星号；（3）标示坐标轴的显示范围和刻度线，添加栅格线；（4）标示坐标轴名称、标题、相应文本。

解：MATLAB 程序代码如下。

```
close all                                  %关闭打开了的所有图形窗口
clc                                        %清屏命令
clear                                      %清除工作空间中的所有变量
t=[0:pi/20:4*pi];
hold on
axis([0 4*pi -10 10])
plot(t,10*sin(t),'r+-.')
plot(t,5*cos(t),'b*:')
xlabel('时间 t'),ylabel('幅值 x')
title('简单绘图实例')
legend('x1=10sint:点画线','x2=5cost:虚线')
```

```
gtext('x1'),gtext('x2')
grid on
```

3.5 MATLAB 程序设计

3.5.1 MATLAB 编程步骤

MATLAB 程序类型包括三种：第一种是在命令窗口下执行的脚本 M 文件；第二种是可以存取的 M 文件，即程序文件：第三种是函数（function）文件。

为了帮助大家避免在编程过程中出现大量的错误，本节将向大家介绍正式的编程步骤，即自上而下的编程方法。接下来会向大家介绍一些普通的算法开发工具即伪代码。

假设你是在工厂工作的工程师，为了解决某个问题要编写一个程序。你将如何开始呢？当遇到一个新问题时，我们的心里会自然而然地产生这样的想法：马上坐在计算机前，开始编程，而不用浪费大量的时间思考所要解决的问题是什么。用这种不切实际的想法来编一些非常小的程序可能会成功。但在现实中，问题可能会非常大，程序员再用这种方法编程将会陷入困境。对一个大的程序来说，在编写代码之前要通盘地思考所要面临的问题和解决的方法。这里将向大家介绍正式的编程设计步骤，然后应用这个步骤来编写本书中所有的大的应用程序。对遇到的一些简单的例子来说，这个步骤好像有些画蛇添足，但是当要解决的问题变得越来越大时，这个步骤将会变得异常重要。

自上而下的编程方法是正规编程设计的基础。步骤如下。

1．清晰地陈述所要解决的问题

编写的程序大多数情况下要满足一些感觉上的需要，但这种需要不一定能够被人清晰地表达出来。例如，用户需要一个解线性方程组的表达式。像这样的要求就不够清楚，程序员就很难编出一个使用户满意的程序。程序员必须弄清楚要有多少问题需要解决，在这些方程式中有没有对称的形式使开发变得简单。程序设计者必须和使用者讨论所需的程序，他们必须要对完成的任务有一个精确细致的描述。对问题清晰的描述可以防止误解，并且能够帮助程序员合理地组织他的思想。上面的例子对问题合适的陈述应为：设计一个用于解决联立线性方程组的程序，这些方程中未知数的系数为实数，最多有 20 个未知数。

2．定义程序所需的输入量和程序所产生的输出量

指定输入量和输出量，只有这样新的程序才能适应全过程计划。在上面的例子中，方程式的系数可能有其预先存在的顺序，新程序必须能按照顺序读取它们。相似地，也需要产生出这个程序所要求的结果，即输出量，还要以一定的格式打印出来。

3．设计程序得以实现的算法

算法是指为某个问题找到答案一步接一步的程序。在这个阶段，自上而下的编程方法发挥了作用。编程设计者开始对这个问题进行逻辑划分，把它逐步分解为一个又一个子工作，这个过程叫作分解（decomposition）。如果一些子工作还是比较大，设计者还可以把它分解成更小的块。这个过程将会继续到问题被分解成许多简单且易理解的小块为止。

在问题被分解成小块之后，每一个小块要被进一步地求精，这个过程叫作逐步求精（stepwise refinement）。在这个过程中，设计者开始于对本小块代码总括性的描述，然后开始一步一步地定义所需的函数，越来越具体，直到它能够转化为 MATLAB 语句。逐步求精的过程中要用到的伪代码将会在后面为大家介绍。

在算法开发过程中，这个方法是非常有用的。如果设计者真正理解了解决问题的步骤，他将会对问题进行分解和逐步求精。

4．把算法转化为代码

如果分解和逐步求精的过程已经顺利完成，那么这一步将会异常简单。所有程序员都会将伪代码一句一句地转化为合适的 MATLAB 语句。

5．检测产生的 MATLAB 程序

这一步是真正的拦路虎。程序的每一部分将会被单独地检测，如果有可能的话，整个程序还要被检测一遍。在检测程序时，必须证明所有合法输入数据值都能够正常运行。用标准的输入值检测程序，看它是否产生了预期值。如果在一个程序中执行的算法包含不同的分支，那么必须检测每一个分支，以保证产生正确的答案。大程序在交付使用之前，必须经过一系列的检测。检测的第一步有时被称为单元检测（unit testing）。在单元检测过程中，程序的子程序将会被独立地检测以证明它的正确性。当单元检测结束之后，这个程序将进行一系列的组合，把独立的子程序联合产生出最后的程序。程序第一步的联合通常只包括很少的子程序。通过组合这些子程序，经常用于检查子程序或函数之间的联系。在一系列的组合过程中，越来越多的子程序被加了进来，直到整个程序完成。在每一次组合的过程中，每一个错误都会被发现并在进行下一次组合之前纠正过来。

在整个程序被组合之后，调试继续进行。程序的第一个版本通常被称为“alpha 版本”。程序员和其他有机会接近它的人可以想尽一切办法应用它，以发现其中的漏洞，然后改正。当许许多多大的错误从程序中去除，一个新的版本出现了，被称为“beta 版本”。beta 版本就要公开地发行给天天需要这个程序工作的人。这些用户使这个程序在不同的环境下、不同的输入条件下工作，就会发现许多的错误，并报告给程序员。当这些错误被更正后，这个程序就能够发行给公众使用了。因为本书中的程序都比较小，没有必要进行上述的大规模检测。但是要遵循基本的调试原则。

在大的编程项目中，花在编写程序上的时间是出奇地少。Frederick P Brooks 在他的 *The Mythical Man Month* 书中写道，对大的软件工程来说，1/3 的时间花在计划如何来做上，1/6 的时间花在编写程序上，近一半的时间用来调试程序。而我们能做的只有压缩调试用的时间。在计划阶段做好充分的准备和在编程过程中养成良好的编程习惯，这样会大大缩短调试所用的时间。好的编程习惯能减少出错的数量，也能使别人迅速地找出其中的错误。

3.5.2　伪代码的应用

作为编程步骤的一部分，描述出要执行的算法是非常必要的。算法的描述有一种标准形式，能让大家都能理解，这种描述将有助于从内容转化为 MATLAB 代码。用于描述算法的标准形式叫作构造（construct，有时也称 structure）。用这些结构描述出的算法称为结构化算法。

当在 MATLAB 中执行这个算法时，产生的程序叫作结构化程序。

可以用伪代码的形式建立算法的结构。伪代码是 MATLAB 和英语的混合体。和 MATLAB 一样，它是结构化的，一行表达一个明确的意思或代码的片段，但每一行的描述用的是英语或其他人类语言。因为修改简单灵活，所以伪代码在开发算法的过程中非常有用。因为伪代码是给编辑器或字处理器（通常用于编写 MATLAB 程序）的，而不需要其他的可视化功能。

3.5.3 关系运算符和逻辑运算符

选择结构的运算由一个表达式控制，这个表达式的结果只有 true(1)和 false(0)。有两种形式的运算符可以在 MATLAB 中运算得到 true/false，即关系运算符和逻辑运算符。

与 C 语言一样，MATLAB 没有布尔型/逻辑型数据类型。MATLAB 把 0 值作为结果 false，把所有的非 0 值作为结果 true。

1．关系运算符

关系运算符是指两个数值或字符操作数的运算符。关系运算将会根据两个操作数的关系产生结果 true 或 false。关系运算符见表 3-4。关系运算的基本形式为

$$a_1 \ op \ a_2$$

其中，a_1 和 a_2 是算术表达式、变量或字符串；op 代表关系运算符中的一个。如果两者的关系为真（true），那么这个运算将会返回 1；否则将会返回 0。

表 3-4 关系运算符

关系运算符	运　　算
==	等于
~=	不等于
>	大于
>=	大于或等于
<	小于
<=	小于或等于

等于运算符（==）：如果两个操作数相同，则返回 1；如果不同则返回 0。

不等于运算符（~=）：如果两个操作数不同，则返回 1；如果相同则返回 0。

用这两个运算符比较两个字符串是安全的，不会出现错误。但对两个数字数据的比较，将可能产生意想不到的错误。两个理论上相等的数不能有一丝一毫的差别，而在计算机计算的过程中存在修正误差，从而可能在判断相等与不相等的过程中产生错误，这种错误叫作 round off 错误。例如，考虑下面的两个数，两者均应等于 0。

```
a = 0;
b = sin(pi);
```

因为这两个数在理论上是相等的，所以关系式 a==b 应当返回 1。但在事实上，MATLAB 计算所产生的结果是

```
>> a = 0;
```

```
>> b = sin(pi);
>> a == b
ans =
     0
```

MATLAB 报告了 a 和 b 不同是由于修正误差造成的，在计算中，sin(pi)产生的结果是 1.2246×10^{-16}，而不是 0。两个理论上相等的值由于修正误差而引起了细微的差别。

可以检测两个数之间在一定范围内是不是近似相等，在这个精确范围内可能会存在修正误差。例如，检测

```
>> abs(a - b) < 1.0E-14
ans =
     1
```

将会产生正确的结果，尽管在 a 与 b 的计算中存在修正误差。

2．逻辑运算符

逻辑运算符（见表 3-5）是联系一个或两个逻辑操作数并能产生一个逻辑结果的运算符。有三个二元运算符，分别为逻辑与、逻辑或和逻辑异或运算符，还有一个一元运算符，即逻辑非运算符。二元逻辑运算的基本形式为

$$l_1 \text{ op } l_2$$

一元逻辑运算的基本形式为

$$\text{op } l_1$$

其中，l_1 和 l_2 代表表达式或变量；op 代表表 3-5 中的逻辑运算符。如果 l_1 和 l_2 的逻辑运算关系为 true，那么运算将会返回 1；否则将会产生 0。

表 3-5　逻辑运算符

逻辑运算符	运　　算
&	逻辑与
|	逻辑或
xor	逻辑异或
~	逻辑非

3．逻辑函数

MATLAB 中有大量的逻辑函数。在条件满足时，函数返回 1；在条件不满足时，函数返回 0。这些逻辑函数连同关系运算符和逻辑运算符一起实现程序的选择结构和循环结构。表 3-6 列出了一系列的逻辑函数。

表 3-6　MATLAB 逻辑函数

函　　数	用　　途
ischar(a)	a 是字符数组则返回 1，否则返回 0

续表

函　数	用　途
isempty(a)	a 是空数组则返回 1，否则返回 0
isinf(a)	a 是无穷大则返回 1，否则返回 0
isnan(a)	a 不是一个数则返回 1，否则返回 0
isnumeric(a)	a 是数值数组则返回 1，否则返回 0

3.5.4 MATLAB 程序流程控制

MATLAB 程序有顺序、分支、循环等程序结构及子程序结构。

1．顺序程序结构

顺序程序结构的程序从程序的首行开始，逐行顺序往下执行，直到程序最后一行，大多数简单的 MATLAB 程序都采用这种结构。它首先读取输入，然后运算得到所需结果，打印出结果，并退出。至于要多次重复运算程序的某些部分是没有办法的，也不能根据输入的值有选择地执行程序的某些部分。

2．分支程序结构

分支程序结构的程序要根据执行条件满足与否，确定执行方向。在 MATLAB 中，通过 if-else-end 结构、switch-case-otherwise 结构、try-catch 结构来实现。

1）if-else-end 结构

常见结构如下。

```
if control_expr_1
Statement 1
Statement 2
... %Block1
elseif control_expr_2
Statement 1
Statement 2
...%Block2
else
Statement 1
Statement 2
...%Block3
end
```

其中，控制表达式（control expression）控制该结构的运算。如果 control_expr_1 的值非 0，那么程序将会执行语句块 1（Block1），然后跳到 end 后面的第一个可执行语句继续执行。否则，程序将会检测 control_expr_2 的值。如果 control_expr_2 的值非 0，那么程序将会执行语句块 2（Block2），然后跳到 end 后面的第一个可执行语句继续执行。如果所有的控制表达式均为 0，那么程序将会执行 else 语句块。

在该结构中，可以有任意个 elseif 语句块，但 else 语句块最多有一个。只要上面每一个控

制表达式均为 0，那么下一个控制表达式将会被检测。一旦其中的一个表达式的值非 0，对应的语句块就要被执行，然后跳到 end 后面的第一个可执行语句继续执行。如果所有的控制表达式均为 0，那么程序将会执行 else 语句块。如果没有 else 语句块，程序将会执行 end 后面的语句，而不执行该结构中的部分。

MATLAB 通过 end 在 M 文件中的上下文来区分开它的用途。在大多数情况下，控制表达式均可以联合关系运算符和逻辑运算符。正如在本章前面学到的，当对应的条件为真时，关系运算和逻辑运算将会产生 1，否则产生 0。所以，当一个运算条件为真时，运算结果为非 0，则对应的语句块就会被执行。

2）swith-case-otherwise 结构

swith-case-otherwise 结构是另一种形式的选择结构。程序员可以根据一个单精度整型数、字符或逻辑表达式的值来选择执行特定的代码语句块。

常见结构如下。

```
switch switch_expr
case case_expr_1
Statement 1
Statement 2
... %Block 1
case case_expr_2
Statement 1
Statement 2
... %Block 2
...
otherwise
Statement 1
Statement 2
... %Block n
end
```

如果 switch_expr 的值与 case_expr_1 相符，那么第一个语句块将会被执行，然后程序将会跳到 end 后的第一个语句。如果 switch_expr 的值与 case_expr_2 相符，那么第二个语句块将会被执行，然后程序将会跳到 end 后的第一个语句。在这个结构中，用相同的方法来对待其他的情况。otherwise 语句块是可选的。如果它存在，则当 switch_expr 的值与其他所有的选项都不相符时，这个语句块将会被执行。如果它不存在，且 switch_expr 的值与其他所有的选项都不相符，那么这个结构中的任何一个语句块都不会被执行。这种情况下的结果可以看作没有选择结构，而直接执行 MATLAB 语言。

3）try-catch 结构

try-catch 结构是选择结构的一种特殊形式，用于捕捉错误。一般地，当一个 MATLAB 程序在运行时遇到了一个错误，这个程序就会终止执行。try-catch 结构修改了这个默认行为。

如果一个错误发生在这个结构的 try 语句块中，那么程序将会执行 catch 语句块，程序将不会中断。它将帮助程序员控制程序中的错误，而不用使程序中断。

常见结构如下。

```
try
Statement 1
Statement 2
... %Try Block
catch
Statement 1
Statement 2
...%Catch Block
end
```

当程序运行到 try-catch 结构时，在 try 语句块中的一些语句将会被执行。如果没有错误出现，那么 catch 语句块将会被跳过。如果一个错误发生在 try 语句块中，那么程序将终止执行 try 语句块，并立即执行 catch 语句块。

下面是一个包含 try-catch 结构的程序。它能创建一个数组，并询问用户显示数组中的哪一个元素。用户提供一个下标，那么这个程序将会显示对应的数组元素。try 语句块一般会在这个程序中执行，只有当 try 语句块执行出错时，catch 语句块才会被执行。

```
% Initialize array
a = [ 1 -3 2 5];
try
% Try to display an element
index = input('Enter subscript of element to display: ');
disp(['a(' int2str(index) ') = ' num2str(a(index))] )
catch
% If we get here an error occurred
disp( ['Illegal subscript: ' int2str(index)] )
end
```

这个程序的执行结果如下。

```
>> try_catch
Enter subscript of element to display: 3
a(3) = 2
>> try_catch
Enter subscript of element to display: 8
Illegal subscript: 8
```

4）调试程序的说明

含有选择结构和循环结构的程序出错的概率要比只含简单的顺序结构的程序出错的概率大得多。在完成了编程步骤之后，无论多大的一个程序，在第一次运行时都很难通过。假如创建了一个程序并调试它，只发现这个程序的输出是错误的，那么怎样找到程序错误并修改它呢?

一旦程序包含循环结构和选择结构，找到错误最好的方法是应用 MATLAB 支持的符号调试器（symbolic debugger）。这个调试器将会整合到 MATLAB 编辑器中。

应用这个调试器时，首先应该选择“File/Open”打开在 MATLAB 命令窗口中要调试的程序。当一个文件被打开，编辑器就加载了这个文件，代码根据语法的不同出现不同的颜色。这个文件中的注释显示为绿色，变量和数字显示为黑色，字符串显示为红色，语言的关键字显示为蓝色。图 3-7 显示的是含有文件 calc_roots.m 的编辑调试窗口。

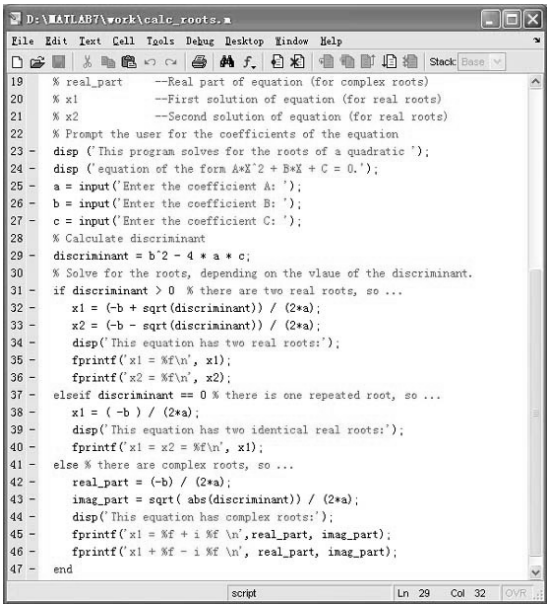


图 3-7　MATLAB 编辑调试窗口

当一个程序执行时，我们想知道什么事情发生了。为了达到此目的，可以用鼠标右击所关心的行并选择”set/clear breakpoint”选项。当一个断点被设置后，一个红色的点将会出现在行的左边，如图 3-8 所示。

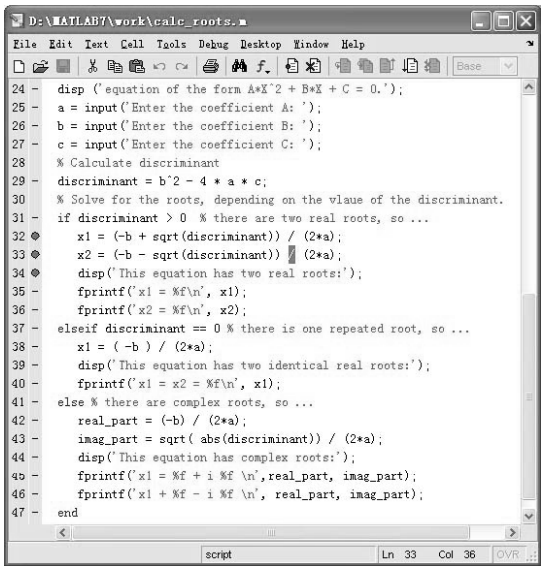


图 3-8　窗口断点已设置

一旦这些断点被设置，在命令窗口中输入 calc_roots 将会像往常一样执行这个程序，这个程序将会运行到第一个断点并在那里停止。在调试的过程中将会有一个绿色的箭头出现在当前行的左侧，如图 3-9 所示。

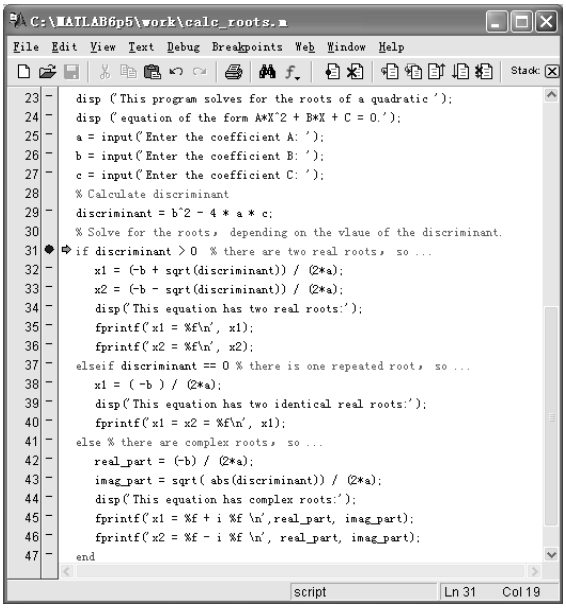


图 3-9 绿色的箭头出现在当前行的左侧

一旦到达某个断点，程序员可以通过在命令窗口中输入变量名的方法检查或修改在工作区内的任何一个变量。当程序员对程序的这一点感到满意时，可以通过重复按 F10 键一行一行地调试，也可以按 F5 键运行到下一个断点。它总是能检测程序中的每一个断点中的任何一个变量的值。

调试器的另处一个重要特性是可在 Breakpoints 菜单中找到。这个菜单包括两个选项：“stop if error”和“stop if warning”。如果程序中发生了一个错误，这个错误导致了计算机死机或产生了错误信息，那么程序员可以打开这些选项，并执行这个程序。这个程序将会运行到错误或警告的断点并停在那儿，它允许程序员检查任何一个变量的值，并帮助找出出错的原因。当一个错误被发现，程序员可用编辑器来更正这个 MALTAB 程序，并把更新的版本存到磁盘上，在调试结束之前，必须重复以上的动作。这个步骤将会重复下去直到这个程序没有错误出现。

3．循环程序结构

循环程序结构包括一个循环变量，循环变量从初始值开始计数，每循环一次就执行一次循环体内的语句，执行完后，循环变量以一定的规律变化，然后再执行循环体内的语句，直到循环变量大于循环变量的终止值为止。

常用的循环有 while 循环和 for 循环。while 循环和 for 循环的区别在于：while 循环结构中循环体的执行次数是不确定的，而 for 循环结构中循环体的执行次数是确定的。

1）for 循环结构

for 循环结构使用较为灵活，一般用于循环次数已经确定的情况，其结构如下。

```
for index = expr
Statement 1
... %Body
Statement n
end
```

其中，index 是循环变量（就是循环次数）；expr 是循环控制表达式。变量 index 读取的是数组 expr 的列，然后程序执行循环体（loopbody），所以 expr 有多少列，循环体就循环多少次。expr 经常用捷径表达式的方式，即 first:incr:last。

在 for 和 end 之前的语句被称为循环体。在 for 循环结构运转的过程中，它将被重复地执行。for 循环结构的工作过程如下。

（1）在 for 循环开始之时，MATLAB 产生了控制表达式。

（2）第一次进入循环，程序把循环控制表达式（数组）的第一列赋值于循环变量 index，然后执行循环体内的语句。

（3）在循环体内的语句被执行后，程序把循环控制表达式（数组）的下一列赋值于循环变量 index，程序将再一次执行循环体内的语句。

（4）只要在循环控制表达式（数组）中还有剩余的列，步骤（3）将会一遍一遍地重复执行。

下面要举例来说明 for 循环结构的工作过程。

第一，考虑下面的例子。

```
for ii = 1:10
Statement 1
...
Statement n
end
```

在这种情况下，控制表达式产生了一个 1×10 数组，所以语句 1 到 n 将会被重复执行 10 次。循环变量 ii 在第 1 次执行的时候为 1，第 2 次执行的时候为 2，以此类推。当最后一次执行时，循环变量为 10。在第 10 次执行循环体之后，再也没有新的列赋值给控制表达式，程序将会执行 end 后的第一个语句。注意：在循环体最后一次被执行后，循环变量将会一直为 10。

第二，考虑下面的例子。

```
for ii = 1:2:10
Statement 1
...
Statement n
end
```

在这种情况下，控制表达式产生了一个 1×5 数组，所以语句 1 到 n 将会被重复执行 5 次。循环变量 ii 在第 1 次执行时为 1，第 2 次执行时为 3，以此类推，最后一次执行时为 9。在第 5 次执行循环体之后，再也没有新的列赋值给控制表达式，程序将会执行 end 后的第一个语句。注意：在循环体最后一次被执行后，循环变量将会一直为 9。

第三，考虑下面的例子。

```
for ii = [5 9 7]
Statement 1
...
Statement n
end
```

在这里，控制表达式是一个直接写出的 1×3 数组，所以语句 1 到 n 将会被重复执行 3 次。循环变量 ii 在第 1 次执行时为 5，第 2 次执行时为 9，第 3 次执行时为 7。循环变量在循环结束之后一直为 7。

第四，考虑下面的例子。

```
for ii = [1 2 3; 4 5 6]
Statement 1
...
Statement n
end
```

在这里，控制表达式是一个直接写出的 2×3 数组，所以语句 1 到 n 将会被重复执行 3 次。循环变量 ii 在第 1 次执行时为列向量$\begin{bmatrix}1\\4\end{bmatrix}$，第 2 次执行时为$\begin{bmatrix}2\\5\end{bmatrix}$，第 3 次执行时为$\begin{bmatrix}3\\6\end{bmatrix}$。这个例子说明循环变量可以为向量。

for 循环结构的伪代码如下。

```
for index = expression
Statement 1
...
Statement n
end
```

2）while 循环结构

while 循环结构一般用于事先不能确定循环次数的情况，其结构如下。

```
while 表达式
    循环体
end
```

while 循环结构的伪代码如下。

```
while expr
...
...
...
end
```

若表达式为真，则执行循环体的内容，执行后再判断表达式是否为真；若不为真，则跳出循环体，向下继续执行。在 while 循环结构中，可用 break 语句退出循环。

3）break 语句和 continue 语句

有两个附加语句可以控制 while 循环和 for 循环：break 语句和 continue 语句。

（1）如果 break 语句在循环体中执行，那么循环体的执行终止，然后执行循环后的第一个可执行性语句。用在 for 循环中的 break 语句的例子如下。

```
%test_break.m
for ii = 1:5
if ii == 3
break
end
fprintf('ii = %d \n', ii)
end
disp('End of loop!')
```

程序的执行结果如下。

```
>> test_break
ii = 1
ii = 2
End of loop!
```

注意：break 语句在 ii 为 3 时执行，然后执行 disp 语句，而不执行 fprintf 语句。

（2）continue 语句只终止本次循环，然后返回循环的顶部，在 for 循环中的循环变量将会更新到下一个值，循环将会继续进行。下面是在 for 循环中的 continue 语句的例子。

```
%test_continue.m
for ii = 1:5
if ii == 3
continue
end
fprintf('ii = %d \n', ii)
end
disp('End of loop!')
```

程序的执行结果如下。

```
>> test_continue
ii = 1
ii = 2
ii = 4
ii = 5
End of loop!
```

注意：continue 语句在 ii 为 3 时执行，然后程序返回循环的顶部，而不执行 fprintf 语句。

break 语句和 continue 语句可用在 while 循环和 for 循环中。

3.5.5 自定义函数

1．函数简介

到目前为止，我们看到的所有的 M 文件都是脚本文件。脚本文件只是用于存储 MATLAB 语句。当一个脚本文件被执行时，与直接在命令窗口中输入 MATLAB 语句所产生的结果是一样的。脚本文件分享命令窗口中的工作区，所以所有的在脚本文件运行之前定义的变量都可以在脚本文件中运行，所有在脚本文件中创建的变量在脚本文件运行之后仍然存在于工作区。一个脚本文件没有输入参数，也不返回结果，但是所有脚本文件都可以通过存于工作区的数据进行交互。

相对地，MATLAB 函数是一种特殊形式的 M 文件，它运行于独立的工作区。它通过输入参数列表接受输入数据，它通过输出参数列表返回结果。MATLAB 函数的基本形式如下。

```
function [outarg1, outarg2, ...] = fname(inarg1, inarg2, ...)
%H1 comment line
%Other comment lines
...
(Executable code)
...
(return)
```

function 语句标志着这个函数的开始，它指定了函数的名称和输入参数列表、输出参数列表。输入参数列表显示在函数名后面的圆括号中，输出参数列表显示在等号左边的中括号中（如果只有一个输出参数，中括号可以省略）。

输入参数列表是名字的列表，这些名字代表从调用者到函数的值。这些名字被称为形参。当函数被调用时，它们只是从调用者得来实际变量的占位符而已。相似地，输出参数列表也由形参组成，当函数结束运行时，这些形参是返回到调用者的值的占位符。

在一个表达式中，调用一个函数需要用到实参列表。在命令窗口中直接（或在脚本文件中、另一个函数中）输入函数的名字就可以调用这个函数了。当调用一个函数时，第一个实参的值用在第一个形参的位置，而且其余的形参和实参都一一对应。

函数的执行从函数的顶部开始，结束于 return 语句或函数的终点。因为函数执行到结尾就会结束，所以 return 语句在大部分的程序中没有必要使用。输出参数列表中每一个项目都必须出现在 function 语句中的左边。当函数返回时，存储于输出参数列表的值就会返回给调用者，用于下一步的运算。

在一个函数中的初始注释行有特定的目的。该初始注释行被称为 H1 注释行，它应当是对本函数功能的总结。这一行的重要性在于，通过 lookfor 命令它能被搜索到并显示出来。从 H1 注释行到第一个空行或第一个可执行性语句可以通过 help 命令或帮助窗口搜索到。它们应包含如何使用这个函数的简单总结。

例如，编写一个函数文件 cirarea.m 计算圆的面积。

```
%M-file function,cirarea.m
%Calculate the area of a circle with radius r
```

```
%r can be a scalar or an array
function c = cirarea(r)
c = pi*r^2;
```

应用该函数计算圆的面积。

```
>>r=1:3;
>>ar = cirarea(r) %呼叫 cirarea 函数，以数组 r 为输入变量
ar=
    3.1416 12.5664 28.2743
>>disp(ar)                          %使用 disp 可以将变量值直接列出
    3.1416 12.5664 28.2743
```

2．MATLAB 中自定义函数的几种方法

MATLAB 中自定义函数的几种方法如下。

1）自定义函数文件+调用函数文件

该方法需要单独编写一个自定义函数的 M 文件。

首先，编写自定义函数文件 mylfg.m，该文件描述了要被调用的函数。

```
%自定义函数文件：mylfg.m
function y=mylfg(x) %注意：函数名（mylfg）必须与文件名（mylfg.m）一致
y=x^(1/3);
```

其次，编写 M 文件 myfile.m，该文件是调用函数文件。

```
%调用函数文件：myfile.m
clear
clc
for t=1:10
y=mylfg(t);
fprintf('M^(1/3)=%6.4f\n',t,y)
end
```

注意：这种方法要求自定义函数必须单独写一个 M 文件，不能与调用函数文件写在同一个 M 文件中。

2）自定义函数文件+子函数

该方法需要编写定义一个具有多个子函数的 M 文件。

```
%自定义函数文件：funtry2.m
function []=funtry2()
for t=1:10
y=lfg2(t);
fprintf('M^(1/3)=%6.4f\n',t,y)
end
function y=lfg2(x)
y= x^(1/3);
```

注意：自定义函数文件 funtry2.m 中可以定义多个子函数。子函数 lfg2 只能被主函数和主函数中的其他子函数调用。

3）使用 inline 命令

该方法无需 M 文件，直接定义。

```
%inline 命令用来定义一个内联函数
f=inline('函数表达式', '变量 1', '变量 2',…);
```

调用方式为

```
y=f(数值列表) %注意：代入的数值列表顺序应与 inline 定义的变量名顺序一致
```

举例如下。

```
>>f=inline('x^2+y','x','y');
>>z=f(2,3)
z=
    7
```

注意：这种函数定义方法是将它作为一个内部函数调用。优点是，它是基于 MATLAB 的数值运算内核的，所以运算速度较快，程序效率更高；缺点是，该方法只能对数值进行代入，不支持符号代入，且对定义后的函数不能进行求导等符号运算。

4）使用函数句柄操作符@

使用 MATLAB 函数句柄操作符@，可以定义指向 MATLAB 内置函数和用户自定义函数的函数句柄。函数句柄也可以像函数一样地使用。例如：

```
>>x=-pi:0.1:pi;
>> fh=@(x,y)sin(x)+cos(y)
fh =
    @(x,y)sin(x)+cos(y)
>>fh(pi/2,0)
ans =
     2
```

使用函数句柄操作符有以下两种方式。

```
handle = @functionname
handle = @(arglist)anonymous_function
```

第一种方式是创建一个函数句柄。函数句柄是 MATLAB 的一种数据类型，通过函数句柄，用户可以间接地调用函数。用户可以将函数句柄作为另一个函数的参数。

第二种方式用来创建匿名函数，并返回该匿名函数的句柄。括号右边的函数体是单个的 MATLAB 语句或 MATLAB 命令。arglist 是一个用逗号“,”分隔的输入变量列表。

5）syms+subs

该方法无需 M 文件，直接定义。用 syms 定义一个符号表达式，用 subs 调用。例如：

```
>>syms f x %定义符号
>>f=1/(1+x^2); %定义符号表达式
```

```
>>subs(f, 'x', 代替 x 的数值或符号)
```

注意：对于在 syms 中已经定义过的符号变量，在 subs 中进行替换时，单引号可以省略。但是，如果在 syms 后又被重新定义为其他类型，则必须加单引号，否则不可替换。

这种函数定义方法的优点是，可以用符号进行替换。例如：

```
>>syms f x
>>f=1/(1+x^2);
>>subs(f, 'x', 'y^2')
ans=
1/(1+(y^2)^2)
```

注意：该方法的缺点是，由于使用符号运算内核，运算速度会大大降低。

6）字符串+subs

该方法无需 M 文件，直接定义。该方法直接定义一个字符串，用 subs 命令调用。例如：

```
>>f='1/(1+x^2)';                    %定义字符串
>>z=subs(f, 'x',2)
>>g=subs(f, 'x', 'y^2')
```

注意：优点是，占用内存最少，定义格式方面自由；缺点是，无法对字符进行符号转化。当所要替换的符号在调用前都已经有了数值定义，则可以直接调用字符串。例如：

```
>>f='x^2*y';
>>x=2;y=3;
>>subs(f)
ans=
     12
```

课后题

1．MATLAB 是哪几个英文单词的缩写？

2．在 MATLAB 环境中，创建一个 M 文件的方法有哪些？

3．在 MATLAB 环境中，编写一个两个矩阵求和的程序。

4．利用 MATLAB 的符号运算功能，求函数 $f=t^2+\cos(t)$的二阶导数。

5．利用 MATLAB 软件，求函数 $f(t)=t^3$ 的拉氏变换。

6．利用 MATLAB 软件，绘制函数 $f=\sin(x)+\cos(x)$的曲线图。

7．编写 MATLAB 程序，计算表达式“0 & 1 | 0 xor 1”的值。

8．用 MATLAB 编写程序，求 e 的值。已知：$e\approx 1+1/1!+1/2!+1/3!+\cdots+1/n!$。（1）用 for 循环，计算前 50 项；（2）用 while 循环，要求直至最后一项的值小于 0.00001。

9．在 MATLAB 中，自定义函数有几种方法？用这些 MATLAB 自定义函数的方法，实现题 8 中求 e 的值的函数程序。

第 4 章　古典控制理论基础

由第 1 章的学习知道，古典控制理论是建立系统模型和仿真实际系统的重要工具，也是本书所采用的主要理论依据。事实上，古典控制理论的核心是采用拉普拉斯数学变换方法，将求解困难的微分方程，转换成求解容易的代数方程，然后再将得到的代数方程用拉普拉斯反变换的方法求得原微分方程的解。可见，拉普拉斯变换方法是古典控制理论的核心。因此，下面用一章的内容将古典控制理论复习一下。

4.1　复数与复变函数

拉普拉斯变换方法是将时域中的方程转换成复域（复数域）内的方程。因此，下面先复习一下复数与复变函数。

4.1.1　复数的定义

复数的定义为

$$s = \sigma + \mathrm{j}\omega, \quad \sigma, \omega \in \mathbf{R} \tag{4.1}$$

式中　j——虚数单位，$\mathrm{j}^2 = -1$；

σ——实部，$\sigma = \mathrm{Re}[s]$；

ω——虚部，$\omega = \mathrm{Im}[s]$。

如果两个复数相等，则其实部、虚部应分别相等。

4.1.2　复数的表示方法

复数有多种表示方法，式（4.1）是其中的一种。另外的表示方法如下所示。

1．向量表示法（实部、虚部对应向量横、纵坐标）

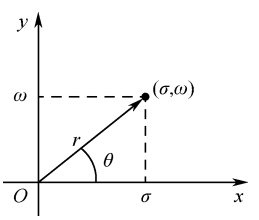


图 4-1　复数的向量表示法

复数的向量表示法如图 4-1 所示。图中，r 称为模，其定义为 $r = |s| = \sqrt{\sigma^2 + \omega^2}$，向量的长度称为复数 s 的模；θ 称为辐角，辐角表达式为 $\theta = \arg(s) = \arctan\dfrac{\omega}{\sigma}$，向量与实轴的夹角称为复数 s 的辐角，其取值范围为 $-\pi \leqslant \theta \leqslant \pi$。

2．三角函数表示法

复数的三角函数表示法为

$$s = r\cos\theta + \mathrm{j}r\sin\theta = r(\cos\theta + \mathrm{j}\sin\theta) \tag{4.2}$$

3．指数表示法

根据欧拉公式

$$e^{j\theta}=\cos\theta+j\sin\theta \tag{4.3}$$

则复数可以表示为

$$s=re^{j\theta} \tag{4.4}$$

【例 4-1】 已知复数 $s_1=2+j3$，$s_2=-2+j3$，写成指数表示形式。

解： $s_1=\sqrt{13}e^{j\arctan\frac{3}{2}}$，$s_2=\sqrt{13}e^{j\left(\pi-\arctan\frac{3}{2}\right)}$

4.1.3　复数的四则运算

已知两个复数为

$$s_1=\sigma_1+j\omega_1=r_1(\cos\theta_1+j\sin\theta_1),\quad s_2=\sigma_2+j\omega_2=r_2(\cos\theta_2+j\sin\theta_2) \tag{4.5}$$

这两个复数的加法为

$$s_1+s_2=\sigma_1+j\omega_1+\sigma_2+j\omega_2=(\sigma_1+\sigma_2)+j(\omega_1+\omega_2) \tag{4.6}$$

这两个复数的减法为

$$s_1-s_2=\sigma_1+j\omega_1-(\sigma_2+j\omega_2)=(\sigma_1-\sigma_2)+j(\omega_1-\omega_2) \tag{4.7}$$

这两个复数的乘法为

$$\begin{aligned}s_1s_2&=r_1r_2[(\cos\theta_1\cos\theta_2-\sin\theta_1\sin\theta_2)+j(\sin\theta_1\cos\theta_2+\cos\theta_1\sin\theta_2)]\\&=r_1r_2[\cos(\theta_1+\theta_2)+j\sin(\theta_1+\theta_2)]\end{aligned} \tag{4.8}$$

即

$$\begin{aligned}|s_1s_2|&=|s_1||s_2|\\\arg(s_1s_2)&=\arg(s_1)+\arg(s_2)\end{aligned} \tag{4.9}$$

同理，可得这两个复数的除法为

$$\begin{aligned}|s_1/s_2|&=|s_1|/|s_2|\\\arg(s_1/s_2)&=\arg(s_1)-\arg(s_2)\end{aligned} \tag{4.10}$$

4.1.4　复变函数

1．定义

以复数为自变量的函数称为复变函数。下列函数均为复变函数：$G(s)=\dfrac{2s+1}{5s^2+6s+1}$，$G(s)=\dfrac{s-2}{s+3}$，$G(s)=5s+6$，$G(s)=e^{-Ts}$。其中，$s=\sigma+j\omega$。

2．极点、零点

极点和零点是复变函数重要的定义。使复变函数分子为零的复数，称为该复变函数的零点；使复变函数分母为零的复数，称为该复变函数的极点。写成公式的形式为

$$G(s)=\frac{k(s+z)}{(s+p_1)(s+p_2)} \tag{4.11}$$

式中　$s_1=-p_1$ 和 $s_2=-p_2$ ——极点；

$s=-z$ ——零点。

4.2 拉普拉斯变换及拉普拉斯反变换

4.2.1 拉普拉斯变换的定义

设函数 $f(t)$，在$[0,+\infty)$有定义，且积分$\int_0^{+\infty} f(t)\mathrm{e}^{-st}\mathrm{d}t$（$s$ 为复数）在 s 的某一域内收敛，则由此积分所确定的函数 $F(s)=\int_0^{+\infty} f(t)\mathrm{e}^{-st}\mathrm{d}t$，称为函数 $f(t)$ 的拉普拉斯变换，记为 $F(s)=L[f(t)]$。$f(t)$ 称为原函数，$F(s)$ 为象函数。拉普拉斯变换也称拉氏变换。

【例 4-2】 求常数 A 的拉氏变换。

解：$f(t)=A$，$L[f(t)]=\int_0^{+\infty} A\mathrm{e}^{-st}\mathrm{d}t=A\int_0^{+\infty}\mathrm{e}^{-st}\mathrm{d}t=A\left(-\frac{1}{s}\right)\mathrm{e}^{-st}\Big|_0^{+\infty}=\frac{A}{s}$

【例 4-3】 求 $f(t)=t$ 的拉氏变换。

解：

$$L[f(t)]=\int_0^{+\infty} t\mathrm{e}^{-st}\mathrm{d}t=-\frac{1}{s}\int_0^{+\infty} t\mathrm{d}(\mathrm{e}^{-st})$$

$$=-\frac{1}{s}t\mathrm{e}^{-st}\Big|_0^{+\infty}+\frac{1}{s}\int_0^{+\infty}\mathrm{e}^{-st}\mathrm{d}t$$

$$=\frac{1}{s}\left(-\frac{1}{s}\right)\mathrm{e}^{-st}\Big|_0^{+\infty}=\frac{1}{s^2}$$

4.2.2 典型时间函数的拉氏变换

先来学习几种典型时间函数的拉氏变换，它们是进行复杂时间函数拉氏变换的基础。

1．单位阶跃函数

单位阶跃函数的定义为

$$x_{\mathrm{i}}(t)=u(t)=\begin{cases}1, & t\geqslant 0\\ 0, & t<0\end{cases} \tag{4.12}$$

其图像如图 4-2 所示。

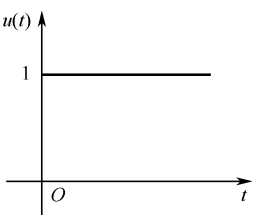


图 4-2 单位阶跃函数的图像

其拉氏变换为

$$L[u(t)]=\int_0^{+\infty}\mathrm{e}^{-st}\mathrm{d}t=-\frac{1}{s}\mathrm{e}^{-st}\Big|_0^{+\infty}=\frac{1}{s} \tag{4.13}$$

说明：当幅值不为 1 时，称为阶跃函数。

2．单位脉冲函数

单位脉冲函数的定义为

$$x_{\mathrm{i}}(t)=\delta(t)=\begin{cases}+\infty, & t=0\\ 0, & t\neq 0\end{cases} \tag{4.14}$$

其图像如图 4-3 所示。

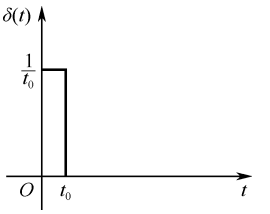


图 4-3　单位脉冲函数的图像

因为 $\int_{-\infty}^{+\infty}\delta(t)\mathrm{d}t=1$，$\int_{-\infty}^{+\infty}\delta(t)f(t)\mathrm{d}t=f(0)$，所以有

$$L[\delta(t)]=\int_{0}^{+\infty}\delta(t)\mathrm{e}^{-st}\mathrm{d}t=\mathrm{e}^{-st}\Big|_{t=0}=1 \tag{4.15}$$

从单位脉冲函数的定义可以看出当 $t_0\to 0$ 时，时间与强度的乘积为 1，所以称为“单位”脉冲函数。

3．单位速度（单位斜坡）函数

单位斜坡函数的定义为

$$x_{\mathrm{i}}(t)=r(t)=\begin{cases}t, & t\geqslant 0\\ 0, & t<0\end{cases} \tag{4.16}$$

其图像如图 4-4 所示。

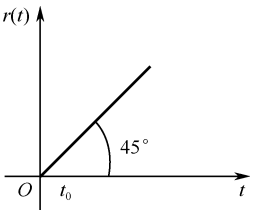


图 4-4　单位斜坡函数的图像

其拉氏变换为

$$L[r(t)]=\int_{0}^{+\infty}t\mathrm{e}^{-st}\mathrm{d}t=-\frac{1}{s}\int_{0}^{+\infty}t\mathrm{d}(\mathrm{e}^{-st})=\frac{1}{s^2} \tag{4.17}$$

说明：当直线的斜率不等于 1 时，称为斜坡函数。

4．指数函数

指数函数的定义为

$$x_i(t)=o(t)=e^{-at} \quad （a 为常数）\tag{4.18}$$

其拉氏变换为

$$\begin{aligned}L[o(t)]&=\int_0^{+\infty}e^{-at}e^{-st}dt=\int_0^{+\infty}e^{-(a+s)t}dt\\&=-\frac{1}{a+s}e^{-(a+s)t}\Big|_0^{+\infty}=\frac{1}{s+a}\end{aligned}\tag{4.19}$$

5．单位正弦函数

单位正弦函数的定义为

$$x_i(t)=b(t)=\begin{cases}\sin\omega t, & t\geqslant 0\\ 0, & t<0\end{cases}\tag{4.20}$$

其拉氏变换为

$$L[b(t)]=\int_0^{+\infty}\sin\omega t e^{-st}dt=\frac{\omega}{s^2+\omega^2}\tag{4.21}$$

同理，可求得单位余弦函数的拉氏变换为

$$L[\cos\omega t]=\frac{s}{s^2+\omega^2}\tag{4.22}$$

值得说明的是，通常求时间函数的拉氏变换并不一定用定义求解，也可以从拉氏变换表中查到。常用函数的拉氏变换表见表 4-1。

表 4-1　常用函数的拉氏变换表

$f(t)$	$F(s)$	$f(t)$	$F(s)$
$\delta(t)$（单位脉冲函数）	1	t（单位斜坡函数）	$1/s^2$
$u(t)$（单位阶跃函数）	$1/s$	t^n	$n!/s^{n+1}$
e^{-at}	$1/(s+a)$	$t^n e^{-at}$	$n!/(s+a)^{n+1}$
$\cos\omega t$	$s/(s^2+\omega^2)$	$\sin\omega t$	$\omega/(s^2+\omega^2)$
$e^{-at}\cos\omega t$	$\dfrac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\sin\omega t$	$\dfrac{\omega}{(s+a)^2+\omega^2}$
$\dfrac{1}{b-a}(e^{-at}-e^{-bt})$	$\dfrac{1}{(s+a)(s+b)}$	$\dfrac{1}{b-a}(be^{-bt}-ae^{-at})$	$\dfrac{s}{(s+a)(s+b)}$
$\sin(\omega t+\phi)$	$\dfrac{\omega\cos\phi+s\sin\phi}{s^2+\omega^2}$	$\dfrac{\omega_n}{\sqrt{1-\xi^2}}e^{-\xi\omega_n t}\sin\omega_n\sqrt{1-\xi^2}t$	$\dfrac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2}$ $(0<\xi<1)$

4.2.3　拉氏变换的性质

拉氏变换的性质对于分析控制系统的特性是有用的，下面不加证明地给出拉氏变换的一些主要性质。

1．线性性质

已知 α、β 为常数，设 $L[f_1(t)]=F_1(s)$，$L[f_2(t)]=F_2(s)$，则

$$L[\alpha f_1(t)+\beta f_2(t)]=\alpha F_1(s)+\beta F_2(s)\tag{4.23}$$

2．微分性质

设 $L[f(t)]=F(s)$，则 $L\left[\dfrac{\mathrm{d}f(t)}{\mathrm{d}t}\right]=sF(s)-f(0)$。

进一步有

$$L\left[\frac{\mathrm{d}^n f(t)}{\mathrm{d}t^n}\right]=s^nF(s)-s^{n-1}f(0)-\cdots-sf^{(n-2)}(0)-f^{(n-1)}(0) \tag{4.24}$$

若 $f(0)=f'(0)=\cdots=f^{n-1}(0)=0$（当全部初始值为 0 时），则

$$L\left[\frac{\mathrm{d}^n f(t)}{\mathrm{d}t^n}\right]=s^nF(s) \tag{4.25}$$

3．积分性质

设 $L[f(t)]=F(s)$，则 $L\left[\int f(t)\mathrm{d}t\right]=\dfrac{F(s)}{s}+\dfrac{1}{s}f^{-1}(0)$。

进一步有

$$L\left[\underbrace{\int\cdots\int}_{n} f(t)(\mathrm{d}t)^n\right]=\frac{1}{s^n}F(s)+\frac{1}{s^n}f^{-1}(0)+\cdots+\frac{1}{s}f^{-n}(0) \tag{4.26}$$

当初始条件为零时，则

$$L\left[\underbrace{\int\cdots\int}_{n} f(t)(\mathrm{d}t)^n\right]=\frac{F(s)}{s^n} \tag{4.27}$$

4．终值定理

设 $L[f(t)]=F(s)$，且 $\lim\limits_{t\to\infty}f(t)$ 存在且唯一，则

$$\lim_{t\to\infty}f(t)=\lim_{s\to 0}sF(s) \tag{4.28}$$

5．延迟定理

设 $L[f(t)]=F(s)$，且 $t<0$ 时，$f(t)=0$，如果 $f(t)$ 沿时间轴延迟一个恒值 a，用 $f(t-a)$ 表示，则

$$L[f(t-a)]=\mathrm{e}^{-as}F(s) \tag{4.29}$$

【例 4-4】 已知时间函数 $r(t)=1+2t+0.5t^2$，试用查表法求该函数的拉氏变换。

解： $R(s)=L[r(t)]=L[1+2t+0.5t^2]=\dfrac{1}{s}+\dfrac{2}{s^2}+\dfrac{1}{s^3}$

4.2.4　拉普拉斯反变换

描述控制系统的动态数学模型可以是微分方程、差分方程、传递函数和状态方程，也可以用信号流图或模拟图符号表示。分析和研究控制系统的动态特性，就是分析和研究系统数学模型的特征。对微分方程求解，可得到系统输出随时间变化的规律。当微分方程的阶次较高时，微分方程的求解就变得十分困难。因此，常采用拉氏变换方法将微分方程转换成代数方程，在复数域内对获得的代数方程求解后，再通过拉普拉斯反变换（也称拉氏反变换）得到时域内的微分方程的解。拉氏变换与拉氏反变换的关系如图 4-5 所示。这在时域分析中用途很大。

时域函数经过拉普拉斯变换后得到拉普拉斯函数，拉普拉斯反变换定义为

$$f(t)=\frac{1}{2\pi \mathrm{j}}\int_{\sigma-\mathrm{j}\omega}^{\sigma+\mathrm{j}\omega}F(s)\mathrm{e}^{st}\mathrm{d}s \tag{4.30}$$

用符号表示为 $f(t)=L^{-1}[F(s)]$。直接对 $F(s)$ 积分来计算 $f(t)$ 是十分复杂的，因此由 $F(s)$ 求 $f(t)$ 常用部分分式法。首先将 $F(s)$ 分解成一些简单有理分式之和，然后由常用函数的拉氏变换表（见表 4-1）查各有理分式的反变换函数，即得到所求原函数 $f(t)$。

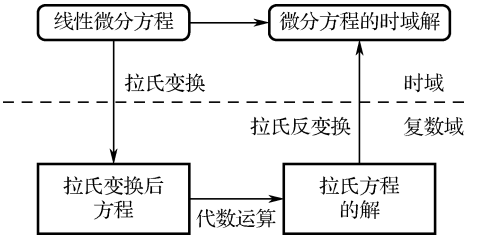


图 4-5　拉氏变换与拉氏反变换的关系

4.2.5　MATLAB 中的拉氏变换

在 MATLAB 中，可以采用符号数学工具箱进行拉氏变换和拉氏反变换，采用的函数是 laplace 和 ilaplace。使用前，用 syms 函数设置有关的符号变量。

要得到在输入信号 $u(t)$ 下过程输出的时域解 $y(t)$，可计算 $u(t)$ 的拉氏变换 $U(s)$，并根据过程的拉氏变换 $G(s)$ 计算出过程的输出 $Y(s)=G(s)U(s)$，再对 $Y(s)$ 进行拉氏反变换，从而得到系统的输出 $y(t)$。

MATLAB 的符号数学工具箱中有拉氏变换和拉氏反变换的运算函数，以及有关定义变量、表达式简化等的函数，下面进行简单介绍。

1．符号变量设置函数的格式：syms arg1 arg2...

使用该函数设置符号运算中的变量 arg1、arg2 等，也可以用：arg1=sym('arg1'); arg2= sym('arg2'); …

当需说明变量的数据类型时，可用格式：syms arg1 arg2 ... real。这表示 arg1、arg2 等变量是实型数据。数据类型可以是实型（real）、正型（positive）、非实型（unreal）等。非实型常用于清除原设置的数据类型。arg1 等变量的第一个字符必须以字母开始，并由字母和数字组成变量名，变量间用空格分隔。

2．拉氏变换函数的格式：L=laplace(F)

F 是时域函数表达式，约定的自变量是 t，得到的拉氏函数是 L。

3．拉氏反变换函数的格式：F=ilaplace(L)

使用该函数把拉氏函数 L 变换为时域函数 F。

4．简化函数的格式：simple(S)

使用该函数将符号表达式 S 化简。用于转换和简化的函数还有 simplify、expand、collect、factor 等，可用 help 命令求助。

5．显示打印函数的格式：pretty(S)

使用该函数将允许结果的表达式较好地显示和打印。用于显示和打印的函数还有 ccode、fortran 和 latex 等，可用 help 求助命令了解其使用方法。

下面看几个例子。

【例 4-5】 计算时域函数 $f(t)=\frac{1}{13}(2\mathrm{e}^{-3t}+3\sin 2t-2\cos 2t)$ 的拉氏变换。

解：MATLAB 程序代码如下。

```
syms t s
y=laplace(1/13*(2*exp(-3*t)+3*sin(2*t)-2*cos(2*t)));
b=simple(y)
```

程序中的第 1 行定义运算的符号，第 2 行进行拉氏变换，之后简化。简化后显示的结果如下。

```
b=
2/(s+3)/(s^2+4)
```

在表达式中，也可以包含其他变量。

【例 4-6】 计算时域函数 $f(t)=\mathrm{e}^{at}$ 的拉氏变换。

解：MATLAB 程序代码如下。

```
syms t s a
y=laplace(exp(a*t))
```

程序的执行结果如下。

```
y=
1/(s-a)
```

【例 4-7】 计算拉氏函数 $F(s)=\frac{2}{(s+3)(s^2+4)}$ 的拉氏反变换。

解：MATLAB 程序代码如下。

```
syms t s
y=ilaplace(2/(s+3)/(s^2+4));
simple(y)
```

【例 4-8】 计算拉氏函数 $F(s)=\frac{(s+a)}{(s+b)(s+c)}$ 的拉氏反变换。

解：MATLAB 程序代码如下。

```
syms t s a b c
y=ilaplace((s+a)/(s+b)/(s+c));
simple(y)
```

4.3　动态过程的传递函数描述

传递函数是在拉氏变换的基础上，以系统本身的参数所描述的线性定常系统输入量和输

出量的关系式，它表达了系统内在的固有特性，而与输入量和驱动函数无关。它可以是有量纲的，也可以是无量纲的，视系统的输入量、输出量而定，它包含着联系输入量与输出量的量纲。它通常不能表明系统的物理特性和物理结构，许多物理性质不同的系统却有着相同的传递函数，正如一些不同的物理现象可以用相同的微分方程描述一样。

线性定常系统的传递函数定义为：在零初始条件下，输出量（响应函数）的拉氏变换与输入量（驱动函数）的拉氏变换之比。现在来考虑一个单变量的线性定常系统，它的运动方程是一个 n 阶的常系数线性微分方程：

$$y^{(n)}+a_{n-1}y^{(n-1)}+\cdots+a_1\dot{y}+a_0y=b_mu^{(m)}+b_{m-1}u^{(m-1)}+\cdots+b_1\dot{u}+b_0u \tag{4.31}$$

式中 u ——输入函数；

y ——输出量，且 $n\geqslant m$ 。

在零初始条件下，系统的传递函数为

$$G(s)=\frac{Y(s)}{U(s)}=\frac{b_ms^m+b_{m-1}s^{m-1}+\cdots+b_1s+b_0}{s^n+a_{n-1}s^{n-1}+\cdots+a_1s+a_0} \tag{4.32}$$

利用传递函数的概念，可用以 s 为变量的代数方程表示系统的动态特性。如果传递函数分母中 s 的最高次数为 n，则称该系统为 n 阶系统。

对线性定常系统，式（4.32）中 s 的系数均为常数，且 $a_0\neq 0$，这时系统在 MATLAB 中可以方便地由分子和分母系数构成的两个向量唯一地确定出来，这两个向量分别用 num 和 den 表示，即

$$\text{num}=[b_m,b_{m-1},\cdots,b_0],\quad \text{den}=[1,a_{n-1},\cdots,a_0]$$

注意：它们都是按 s 的降幂进行排列的。

在 MATLAB 中，函数 tf 用来建立传递函数模型，函数 zpk 用来建立零极点传递函数模型。与建立传递函数模型相关的函数有 tf、tfdata、zpk、zpkdata 和 pzmap，其调用格式如下。

（1）G=tf(num,den)：num 是分子多项式系数行向量，den 是分母多项式系数行向量。

（2）[tt,ff]=tfdata(G)：提取 tf 对象模型中的分子、分母多项式。

（3）GG=zpk(G)：将传递函数 tf 对象转换成零极点模型。

（4）[z,p,k]=zpkdata(G)：提取零极点模型中的零极点及增益项。

（5）pzmap(G)：绘制零极点。

【例 4-9】 某一微分方程描述系统的传递函数，其微分方程描述如下。

$$\frac{\mathrm{d}^3y}{\mathrm{d}t^3}+11\frac{\mathrm{d}^2y}{\mathrm{d}t^2}+11\frac{\mathrm{d}y}{\mathrm{d}t}+10y=\frac{\mathrm{d}^2u}{\mathrm{d}t^2}+4\frac{\mathrm{d}u}{\mathrm{d}t}+8u$$

试使用 MATLAB 建立其模型。

解：对已知条件两边进行拉氏变换，得

$$(s^3+11s^2+11s+10)Y(s)=(s^2+4s+8)U(s)$$

由上式求出系统的传递函数为

$$G(s)=\frac{Y(s)}{U(s)}=\frac{s^2+4s+8}{s^3+11s^2+11s+10}$$

根据上式建立模型的 MATLAB 程序代码如下。

```
num=[1 4 8];
den=[1 11 11 10];
sys=tf(num,den)
```

4.4　系统模型的连接

一般来说，一个系统是由许多环节或子系统按一定方式连接起来组合而成的，它们之间的连接方式有串联、并联、反馈、附加等。MATLAB 提供了模型连接函数。

4.4.1　模型串联

函数 series 用于两个 SISO 系统模型串联，如图 4-6 所示，其调用格式为

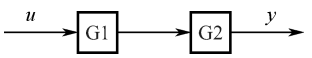


图 4-6　模型串联

```
G=series(G1,G2)
```

该函数的执行结果等价于模型算术运算式 $G=\mathrm{G1}\times\mathrm{G2}$ 。

4.4.2　模型并联

函数 parallel 用于两个 SISO 系统模型并联，如图 4-7 所示，其调用格式为

```
G=parallel(G1,G2)
```

该函数的执行结果等价于模型算术运算式 $G=\mathrm{G1}+\mathrm{G2}$ 。

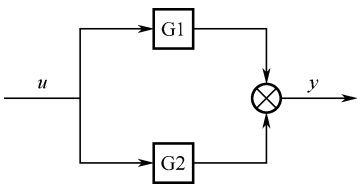


图 4-7　模型并联

4.4.3　反馈连接

函数 feedback 用于 SISO 模型的反馈连接，如图 4-8 所示，其调用格式为

```
G=feedback(G1,H1,sign)
```

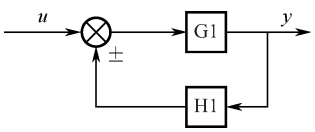


图 4-8　反馈连接

当采用负反馈时，sign=−1 且可省略；当采用正反馈时，sign=+1。

【例 4-10】 已知系统前向通道传递函数为 $G(s)=\dfrac{2s^2+5s+1}{s^2+2s+3}$，负反馈通道传递函数为 $H(s)=\dfrac{s(s+2)}{s+10}$，求系统整体的传递函数。

解：MATLAB 程序代码如下。

```
G=tf([ 2 5 1],[1,2,3]);
H=zpk([0 -2],  -10,1);
Gcloop=feedback(G,H)
```

4.5　MATLAB/Simulink 在时域分析中的应用

一个动态系统的性能常用典型输入作用下的响应来描述。响应是指零初始值条件下，某种典型的输入函数作用下对象的响应。液压控制系统常用的输入函数为单位阶跃函数和单位

脉冲函数。在 MATLAB 的控制系统工具箱中提供了这两种输入下系统响应的函数。常用的输入响应函数有单位阶跃响应函数 step、单位脉冲响应函数 impulse 和时域分析函数。

4.5.1 时域分析中 MATLAB 函数的应用

1．step 函数的用法

（1）step(G)：用于绘制系统 G 的单位阶跃响应曲线，曲线的点数和时间的长度自动确定，系统 G 可以是传递函数模型、零极点模型。

（2）y=step(num,den,t)：num 和 den 分别为系统传递函数描述中的分子和分母多项式系数向量；t 为选定的仿真时间向量，一般可由 t=0:step:end 等步长地产生；该函数返回值 y 为系统在仿真中所得输出组成的矩阵。

（3）[y,x,t]=step(num,den)：时间向量 t 由系统的模型特性自动生成，状态变量 x 返回为空矩阵，不绘制单位阶跃响应曲线。

2．impulse 函数的用法

求取单位脉冲响应的 impulse 函数调用方法与 step 函数基本一致。

（1）impulse(G)

（2）y=impulse(num,den,t)

（3）[y,x,t]=impulse(num,den)

【例 4-11】 绘制一阶惯性环节的单位阶跃响应曲线。

解：MATLAB 程序代码如下。

```
z=[];
for T=5:5:30
    p=[-1/T];k=1/T;G=zpk(z,p,k);
    step(G,100),hold on
end
grid
```

程序中的第 1 行设置零点（空矩阵）；第 2～5 行用于计算时间常数 T 为 5～30 时的系统传递函数 G，用 step 函数计算时间为 0～100 时的输出响应，并绘制响应曲线。

【例 4-12】 绘制二阶环节的单位脉冲响应曲线。

解：MATLAB 程序代码如下。

```
wn=0.15;w=wn^2;num=w;
for zeta=[0:0.2:0.8,1:0.5:2.0]
    den=[1 2*zeta*wn w];
    G=tf(num, den);
    y=impulse(G,100);
    plot(y),hold on
end
grid
```

【例 4-13】 已知系统的闭环传递函数为 $G(s)=\dfrac{1}{s^2+0.4s+1}$，试绘制所述系统的单位阶跃响应曲线和单位脉冲响应曲线。

解：MATLAB 程序代码如下。

```
num=1;
den=[1 0.4 1];
sys=tf(num,den);
subplot(121)
step(sys)
grid on
subplot(122)
impulse(sys)
grid on
```

程序的执行结果如图 4-9 所示。

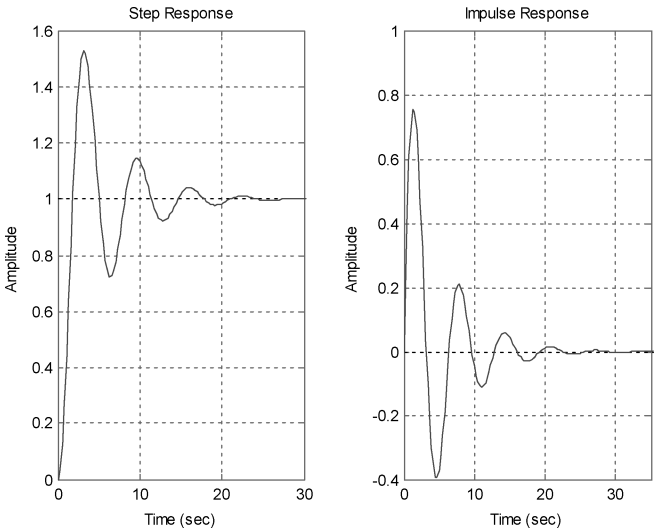


图 4-9　单位阶跃响应曲线与单位脉冲响应曲线

【例 4-14】 已知一阶惯性系统的传递函数为 $G(s)=\dfrac{1}{1+Ts}$，单位阶跃函数的拉氏变换为 $X_{\mathrm{i}}(s)=\dfrac{1}{s}$，求一阶惯性系统在单位阶跃信号作用下的输出。

解：系统输出的拉氏变换为 $X_{\mathrm{o}}(s)=X_{\mathrm{i}}(s)G(s)=\dfrac{1}{s}\dfrac{1}{1+Ts}$，则系统的输出为其拉氏反变换。

MATLAB 程序代码如下。

```
syms F s t T
F=1/(s*(T*s+1));
xo=ilaplace(F,s,t)
```

程序的执行结果如下。

```
xo =
1-1/exp(t/T)
```

【例 4-15】 已知某单位反馈系统，其开环传递函数为 $G(s)=\dfrac{k}{s(Ts+1)}$，其中 $T=1$，试绘制 k 分别为 0.1、0.2、0.5、0.8、1.0、2.4 时的单位阶跃响应曲线。

解：MATLAB 程序代码如下。

```
T=1;
k=[0.1 0.2 0.5 0.8 1.0 2.4];
t=linspace(0,20,200);
num=1;
den=conv([1 0],[T 1]);
for j=1:6
    s1=tf(num*k(j),den);
    sys=feedback(s1,1);
    y(:,j)=step(sys,t);
end
plot(t,y(:,1:6),'k')
grid on
```

程序的执行结果如图 4-10 所示。

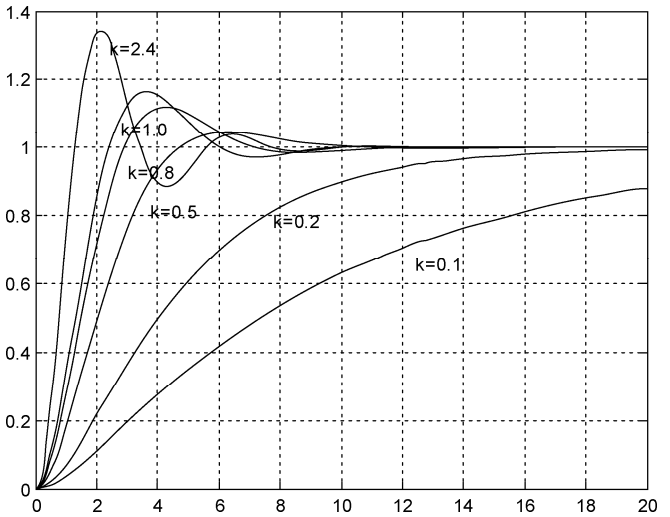


图 4-10 输出结果

4.5.2 时域响应性能指标求取

时域响应分析的是系统对输入和扰动在时域内的瞬态行为。系统特征（如上升时间、调节时间、超调量和稳态误差）均能从时域响应上反映出来。

利用 MATLAB，不仅可以方便、快捷地计算系统的时域响应，绘制响应曲线，而且还能直接在响应曲线图上求取时域响应性能指标，或者通过简单的编程来求取。

1．游动鼠标法求取性能指标

在求取时域响应的程序执行完毕后，用鼠标左键单击时域响应曲线上任意一点，系统会自动跳出一个小方框，小方框中显示了这一点的横坐标（时间）和纵坐标（幅值）。按住鼠标左键在曲线上移动，可找到曲线上幅值最大的一点，即曲线最大峰值，此时小方框中显示的时间就是此二阶系统的峰值时间，根据观测到的稳态值和峰值可计算出系统的超调量。系统的上升时间和调节时间可以此类推。

需要注意的是，由于显示精度和鼠标动作误差的原因，求取的性能指标可能与实际值有误差，但这对分析问题是没有影响的。另外，游动鼠标法不适合用于用 plot 函数画出的图形，也就是说它只能在用非 plot 函数输出的曲线上进行求取。

2．编程法求取性能指标

调用单位阶跃响应函数 step，可以获得系统的单位阶跃响应，当采用[y,t]=step(G)的调用格式时，将返回响应值 y 及相应的时间 t，通过对 y 和 t 进行计算，可以得到时域响应性能指标。

1）求取峰值时间

峰值时间可由以下 MATLAB 程序代码求取。

```
[Y,k]=max(y);                 %求出 y 的峰值及相应的时间
timetopeak=t(k);              %获得峰值时间
```

应用取最大值函数 max 求出 y 的峰值及相应的时间，并存于变量 Y 和 k 中；然后在变量 t 中取出峰值时间，并将它赋给变量 timetopeak，它就是峰值时间。

2）求取超调量

超调量可由以下 MATLAB 程序代码求取。

```
C=dcgain(G);                  %求取系统的终值
[Y,k]=max(y);                 %求出 y 的峰值及相应的时间
percentovershoot=100*(Y-C)/C; %计算超调量
```

dcgain 函数用于求取系统的终值，将终值赋给变量 C，然后依据超调量的定义，由 Y 和 C 计算出百分比超调量。

3）求取上升时间

上升时间可利用 MATLAB 中的循环控制语句 while 编制 M 文件来求取，程序代码如下。

```
C=dcgain(G);                  %求取系统的终值
n=1;
while y(n)<C                  %通过循环，求取输出第 1 次达到终值时的时间
    n=n+1;
end
risetime=t(n) ;               %获得上升时间
```

在阶跃输入条件下，y 值由零逐渐增大，当以上循环满足 y=C 时，退出循环，此时对应的时间即为上升时间。

对于输出无超调的系统响应，上升时间定义为输出从稳态值的 10%上升到 90%所需的时

间，这时求取上升时间的 MATLAB 程序代码如下。

```
C=dcgain(G);                    %求取系统的终值
n=1;
while y(n)<0.1*C                %通过循环，求取输出第 1 次达到终值的 10%时的时间
    n=n+1;
end
m=1;
while y(n)<0.9*C                %通过循环，求取输出第 1 次达到终值的 90%时的时间
    m=m+1;
end
risetime=t(m)-t(n);             %上述两个时间相减，即为上升时间
```

4）求取调节时间

求取调节时间的 MATLAB 程序代码如下。

```
C=dcgain(G);                    %求取系统的终值
i = length(t);                  %求取仿真时间 t 序列的长度
while (y(i)>0.98*C)&(y(i)<1.02*C)
    i=i-1;
end
settlingtime=t(i);              %获得调节时间
```

【例 4-16】 已知二阶系统的传递函数为 $G(s)=\dfrac{3}{(s+1-3\mathrm{i})(s+1+3\mathrm{i})}$，试分别用游动鼠标法和编程法求取系统的性能指标。

解： 首先采用游动鼠标法来求取性能指标，在 MATLAB 命令窗口中输入以下程序代码。

```
G=zpk([],[-1+3*i,  -1-3*i],3);  %建立零极点模型
step(G)                         %绘制单位阶跃响应曲线
```

执行程序后，输出单位阶跃响应曲线，利用游动鼠标法，可大致求出系统的性能指标，如图 4-11 所示。

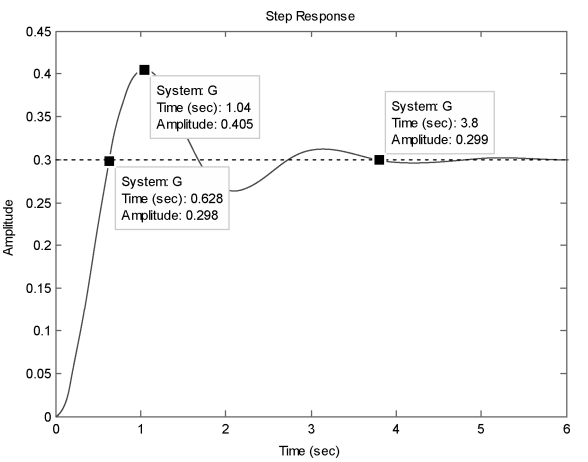


图 4-11　单位阶跃响应曲线

从图 4-11 中可以看出，峰值时间约为 1.04s，上升时间约为 0.628s，超调量为(0.405−0.298)/0.298≈35.9%，调节时间约为 3.8s。

下面采用编程法来求取性能指标，MATLAB 程序代码如下。

```
G=zpk([ ],[-1+3*i,-1-3*i],3);                    %建立零极点模型
%求取峰值时间和它对应的超调量
C=dcgain(G)
[y,t]=step(G);                                   %求取单位阶跃响应
plot(t,y)
grid
[Y,k]=max(y);
timetopeak=t(k)                                  %取得峰值时间
percentovershoot=100*(Y-C)/C                     %计算超调量
%求取上升时间
n=1;
while y(n)<C
    n=n+1;
end
risetime=t(n)
%求取调节时间
i=length(t);
while(y(i)>0.98*C)&(y(i)<1.02*C)
    i=i-1;
end
settlingtime=t(i)
```

程序的执行结果如下。

```
C=
    0.3000
timetopeak=
    1.0492
percentovershoot=
    35.0913
risetime=
    0.6820
settlingtime=
    3.5147
```

对比游动鼠标法的结果可知，峰值时间、上升时间、超调量相差不大，而调节时间的差距较大，那是因为游动鼠标时，找到稳态输出范围时有所误差，当然这不会影响到问题的分析。

4.6　系统误差分析与计算

“精确”是对控制系统提出的一个重要性能要求。对实际系统来说，输出量常常不能绝对

精确地达到所期望的数值，期望的数值与实际输出的差就是所谓的误差。当存在随机干扰作用时，可能带来随机误差；当元件的性能不完善、变质，或者存在诸如干摩擦、间隙、死区等非线性时，也可能带来误差。

4.6.1 系统的误差与偏差

系统的误差是以系统的输出端为基准来定义的，设 $x_{or}(t)$ 是控制系统所期望的输出，$x_o(t)$ 是其实际输出，则误差 $e(t)$ 定义为

$$e(t)=x_{or}(t)-x_o(t) \tag{4.33}$$

其拉氏变换记为 $E_1(s)$（为避免与偏差 $E(s)$ 混淆，用下标 1 区别），故有

$$E_1(s)=X_{or}(s)-X_o(s) \tag{4.34}$$

系统的偏差则是以系统的输入端为基准来定义的，即

$$\varepsilon(t)=x_i(t)-b(t) \tag{4.35}$$

其拉氏变换为

$$E(s)=X_i(s)-B(s)=X_i(s)-H(s)X_o(s) \tag{4.36}$$

式中 $H(s)$ ——反馈回路的传递函数。

现求偏差 $E(s)$ 与误差 $E_1(s)$ 之间的关系。

一个闭环控制系统之所以能对输出 $X_o(s)$ 起自动控制作用，就在于运用偏差 $E(s)$ 进行控制。当 $X_o(s)\neq X_{or}(s)$ 时，由于 $E(s)\neq 0$， $E(s)$ 就起控制作用；反之，当 $X_o(s)=X_{or}(s)$ 时，应有 $E(s)=0$，而使 $E(s)$ 不再对 $X_o(s)$ 进行调节。因此，当 $X_o(s)=X_{or}(s)$ 时，有

$$E(s)=X_i(s)-B(s)=X_i(s)-H(s)X_o(s)=X_i(s)-H(s)X_{or}(s)=0 \tag{4.37}$$

故

$$X_i(s)=H(s)X_{or}(s) \tag{4.38}$$

或

$$X_{or}(s)=\frac{X_i(s)}{H(s)} \tag{4.39}$$

将式（4.39）代入式（4.34），有

$$E_1(s)=\frac{X_i(s)}{H(s)}-X_o(s) \tag{4.40}$$

对比式（4.40）和式（4.36），可见

$$E_1(s)=\frac{E(s)}{H(s)} \tag{4.41}$$

4.6.2 系统的稳态误差与稳态偏差

系统的稳态误差是指系统进入稳态后的误差，因此不讨论动态过程中的情况。只有稳定的系统存在稳态误差。

稳态误差的定义为

$$e_{ss}=\lim_{x\to\infty}e(t) \tag{4.42}$$

为了计算稳态误差，可首先求出系统的误差信号的拉氏变换 $E_1(s)$，再用终值定理求解，即

$$e_{ss}=\lim_{x\to\infty}e(t)=\lim_{s\to 0}sE_1(s) \tag{4.43}$$

同理，系统的稳态偏差为

$$\varepsilon_{ss}=\lim_{x\to\infty}\varepsilon(t)=\lim_{s\to 0}sE(s) \tag{4.44}$$

4.6.3　误差的一般计算

为了在一般情况下分析、计算系统的误差 $e(t)$，设输入 $X_i(s)$ 与干扰 $N(s)$ 同时作用于系统，如图 4-12 所示。

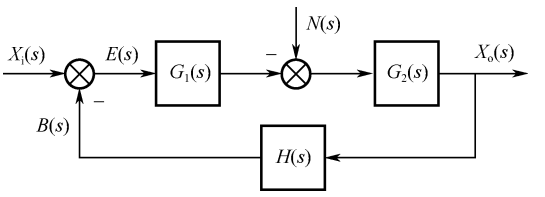


图 4-12　考虑干扰的反馈控制系统

现可求得在如图 4-12 所示情况下的 $X_o(s)$，即线性系统的输出。它是 $X_i(s)$ 引起的输出与干扰 $N(s)$ 引起的输出的叠加，即

$$\begin{aligned}X_o(s)&=\frac{G_1(s)G_2(s)}{1+G_1(s)G_2(s)H(s)}X_i(s)-\frac{G_2(s)}{1+G_1(s)G_2(s)H(s)}N(s)\\&=G_{Xi}(s)X_i(s)-G_N(s)N(s)\end{aligned} \tag{4.45}$$

式中　$G_{Xi}(s)$ ——输入与输出之间的传递函数；

$G_N(s)$ ——干扰与输出之间的传递函数。

将式（4.45）、式（4.39）代入式（4.34），有

$$\begin{aligned}E_1(s)&=X_{or}(s)-X_o(s)=\frac{X_i(s)}{H(s)}-G_{Xi}(s)X_i(s)+G_N(s)N(s)\\&=\left[\frac{1}{H(s)}-G_{Xi}(s)\right]X_i(s)+G_N(s)N(s)=\phi_{Xi}(s)X_i(s)+\phi_N(s)N(s)\end{aligned} \tag{4.46}$$

式中　$\phi_{Xi}(s)$ 和 $\phi_N(s)$ ——总称为误差传递函数，反映了系统的结构参数对误差的影响。

【例 4-17】 已知单位反馈系统如图 4-12 所示，$G_1(s)=\dfrac{5}{0.2s+1}$，$G_2(s)=\dfrac{2}{s(s+1)}$，$H(s)=1$，其中输入信号 $x_i(t)=t$，扰动信号 $n(t)=1(t)$，试分析系统的稳态偏差。

分析：这是一个单位反馈系统，输入信号 $x_i(t)=t$，即 $X_i(s)=\dfrac{1}{s^2}$，干扰信号 $n(t)=1(t)$，即 $N(s)=\dfrac{1}{s}$。

由式（4.44）可知，输入信号作用下的稳态偏差为

$$\varepsilon_{ss1}=\lim_{s\to 0}s\frac{1}{1+G_1(s)G_2(s)}\cdot\frac{1}{s^2}$$

同样可知干扰信号作用下的稳态偏差为

$$\varepsilon_{ss2}=\lim_{s\to 0}\left[s\frac{1}{s}\frac{-G_2(s)}{1+G_1(s)G_2(s)}\right]$$

总稳态偏差为 $\varepsilon_{ss}=\varepsilon_{ss1}+\varepsilon_{ss2}$。

解：计算系统的总稳态偏差的 MATLAB 程序代码如下。

```
G1=tf(5,[0.2 1]);
G2=tf(2,[1 1 0]);
H=1;
sys1=feedback(1,G1*G2,-1);
xi=tf(1,[1 0 0]);
si=tf([1 0],1);
sys=si*sys1*xi;
sysr=minreal(sys);
 [num,den]=tfdata(sysr,'v');
ess1=polyval(num,0)/polyval(den,0);
sys2=feedback(-G2,-G1*H,1);
n=tf(1,[1 0]);
si=tf([1 0],1);
sys=si*sys2*n;
sysr=minreal(sys);
[num,den]=tfdata(sysr,'v');
ess2=polyval(num,0)/polyval(den,0);
ess=ess1+ess2
```

程序的执行结果如下。

```
ess =
    0.3000
```

4.7 控制系统的频率特性

4.7.1 频率响应

线性系统对谐波输入信号的稳态响应，称为频率响应。

其中，谐波信号为 $x_i(t)=X_i\sin\omega t$，拉氏变换为

$$X_i(s)=\frac{X_i\omega}{s^2+\omega^2} \tag{4.47}$$

设线性系统的传递函数为

$$G(s)=\frac{X_o(s)}{X_i(s)}=\frac{b_m s^m+\cdots+b_1 s+b_0}{a_n s^n+a_{n-1}s^{n-1}+\cdots+a_1 s+a_0} \tag{4.48}$$

则系统输出为

$$X_o(s)=G(s)X_i(s)=G(s)\frac{X_i\omega}{s^2+\omega^2} \tag{4.49}$$

若系统无重极点，有

$$X_o(s)=\sum_{i=1}^{n}\frac{A_i}{s-s_i}+\left(\frac{B}{s-\mathrm{j}\omega}+\frac{B^*}{s+\mathrm{j}\omega}\right) \tag{4.50}$$

求拉氏反变换，得

$$x_{\mathrm{o}}(t)=\sum_{i=1}^{n}A_i\mathrm{e}^{s_i t}+(B\mathrm{e}^{\mathrm{j}\omega t}+B^{*}\mathrm{e}^{-\mathrm{j}\omega t}) \tag{4.51}$$

对稳定的系统而言，特征根 s_i 具有负实部，当 $t\to\infty$ 时，系统的稳态响应为

$$x_{\mathrm{o}}(t)=B\mathrm{e}^{\mathrm{j}\omega t}+B^{*}\mathrm{e}^{-\mathrm{j}\omega t} \tag{4.52}$$

式中，有

$$\begin{aligned}B&=G(s)\frac{X_{\mathrm{i}}\omega}{(s-\mathrm{j}\omega)(s+\mathrm{j}\omega)}(s-\mathrm{j}\omega)\Big|_{s=\mathrm{j}\omega}=G(\mathrm{j}\omega)\frac{X_{\mathrm{i}}}{2\mathrm{j}}=|G(\mathrm{j}\omega)|\,\mathrm{e}^{\mathrm{j}\angle G(\mathrm{j}\omega)}\frac{X_{\mathrm{i}}}{2\mathrm{j}}\\ B^{*}&=G(-\mathrm{j}\omega)\frac{X_{\mathrm{i}}}{-2\mathrm{j}}=\left|G(\mathrm{j}\omega)\right|\mathrm{e}^{-\mathrm{j}\angle G(\mathrm{j}\omega)}\frac{X_{\mathrm{i}}}{-2\mathrm{j}}\end{aligned} \tag{4.53}$$

将 B、B^{*} 代入式（4.52），得系统的稳态响应为

$$\begin{aligned}x_{\mathrm{os}}(t)&=\lim_{t\to\infty}x_{\mathrm{o}}(t)=|G(\mathrm{j}\omega)|\,X_{\mathrm{i}}\frac{\mathrm{e}^{\mathrm{j}[\omega t+\angle G(\mathrm{j}\omega)]}-\mathrm{e}^{-\mathrm{j}[\omega t+\angle G(\mathrm{j}\omega)]}}{2\mathrm{j}}\\ &=|G(\mathrm{j}\omega)|\,X_{\mathrm{i}}\sin[\omega t+\angle G(\mathrm{j}\omega)]\end{aligned} \tag{4.54}$$

称 $x_{\mathrm{os}}(t)=|G(\mathrm{j}\omega)|\,X_{\mathrm{i}}\sin[\omega t+\angle G(\mathrm{j}\omega)]=X_{\mathrm{o}}(\omega)\sin[\omega t+\varphi(\omega)]$ 为系统的频率响应。

定义：输出与输入的幅值比 $A(\omega)=X_{\mathrm{o}}(\omega)/X_{\mathrm{i}}=|G(\mathrm{j}\omega)|$ 为幅频特性，输出与输入的相位差 $\varphi(\omega)=\angle G(\mathrm{j}\omega)$ 为相频特性。称 $G(\mathrm{j}\omega)=|G(\mathrm{j}\omega)|\,\mathrm{e}^{\mathrm{j}\angle G(\mathrm{j}\omega)}$ 为系统的频率特性，且 $\mathrm{Re}[G(\mathrm{j}\omega)]$ 为实频特性，$\mathrm{Im}[G(\mathrm{j}\omega)]$ 为虚频特性。

4.7.2　频率特性的求法

简单说，频率特性的求法就是将控制系统传递函数中的 s 用 $\mathrm{j}\omega$ 取代。

4.7.3　频率特性的伯德图

因为系统的频率特性 $G(\mathrm{j}\omega)$、幅频特性和相频特性都是 ω 的函数，人们想用图形来研究它们与 ω 的关系。这个图形就是伯德图。

伯德图用两张图来描述系统的频率响应。其中，一张图描述系统输出与输入的幅值比与频率的关系，称为幅频图；另一张图描述系统输出与输入的相位差与频率的关系，称为相频图。通常，伯德图的频率坐标采用对数坐标，因此，伯德图又被称为对数坐标图，相应的图是对数幅频图和对数相频图。

在对数幅频图中，不同教材中幅值比的表示也有所不同。例如，有的直接用输出与输入的幅值比 $|G(\mathrm{j}\omega)|$，有的用该比值的对数值 $\lg|G(\mathrm{j}\omega)|$ 表示，有的用比值的分贝数 $20\lg|G(\mathrm{j}\omega)|$ 表示。MATLAB 中的伯德图函数 bode 采用分贝数表示幅值比。

分贝（dB）这个单位源于电信技术，用于表示信号功率的衰减程度。设信号功率从 N_1 衰减到 N_2，若 $\lg\dfrac{N_1}{N_2}=1$，则称 N_1 比 N_2 高 1 贝尔（B），即 N_1 为 N_2 的 10 倍。贝尔这个单位太大，分贝是其 1/10。若 $10\lg\dfrac{N_1}{N_2}=1$，即 $\lg\dfrac{N_1}{N_2}=0.1$，则称 N_1 比 N_2 高 1dB，即 N_1 约为 N_2 的 1.2589 倍。

后来，贝尔和分贝推广到其他领域，并将分贝的意义引申为：若 $10\lg\frac{p_2^2}{p_1^2}=1$，即 $20\lg\frac{p_2}{p_1}=1$，则称 p_2 比 p_1 高 1dB。式中，p_2 和 p_1 可代表电压、电流、速度等物理量，其平方与功率成正比。

将分贝的意义引入到控制工程中，幅频特性即稳态输出信号与输入信号的幅值比，故将分贝定义为 $\mathrm{dB}=20\lg|G(\mathrm{j}\omega)|$。

4.8 利用 MATLAB 绘制控制系统的伯德图

在 MATLAB 中，可以直接使用 bode 函数绘制伯德图。下面介绍有关的函数格式。

（1）bode(G)：用于绘制用传递函数 G 描述系统的伯德图。

（2）bode(G,w)：对所需的频率值 w 和已知的系统传递函数 G 绘制伯德图。注意：频率 w 所使用的单位是 rad/sec。

（3）bode(G,{wmin,wmax})：对选定的频率范围[wmin,wmax]和已知的系统传递函数 G 绘制伯德图。注意：输入的频率采用元胞数组。

（4）[MAG,PHASE,W]=bode(G)或[MAG,PHASE]=bode(G,w)：输出数据包括系统频率响应的幅值 MAG 和相位 PHASE。

【例 4-18】 绘制一阶惯性环节的伯德图。

解：MATLAB 程序代码如下。

```
Ga=tf([1],[2 1]);
bode(Ga,'k')
```

程序中的第 1 行给出一阶惯性环节的传递函数 Ga，第 2 行用 bode 函数绘制伯德图。程序绘制的频率响应曲线由两张图组成。曲线的颜色在 bode 函数中设置，'k'表示采用黑色（black）。

【例 4-19】 绘制二阶环节的伯德图。

解：MATLAB 程序代码如下。

```
wn=1;
w=[0,logspace(-2,2,200)];
for zeta=[0.1 0.5 1 2]
    G=tf(1,[wn^-2 2*zeta/wn 1]);
    bode(G,w)
    hold on
end
```

本例采用 bode 函数绘制伯德图，zeta（ξ）的取值范围也有所不同。从伯德图中可以看到，当频率 ω 接近 ω_n 时，产生谐振，阻尼比 zeta 的大小确定谐振峰值的大小，阻尼比越小，谐振峰值越大。频率较小时，相位角为 0°。频率响应曲线的低频渐近线是 0dB 的水平线。高频渐近线是斜率为−40dB/十倍频程的直线。高频渐近线与低频渐近线相交点的频率称为转角频率，转角频率处的相位角恒为−90°；频率趋于∞时，相位角为−180°。相位角曲线是以相位角等于−90°时的点进行点对称的。

【例 4-20】 已知二阶环节的传递函数为 $G(s)=\dfrac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2}$，其中 $\omega_n=0.7$，试绘制 ξ 分别为 0.1、0.4、1.0、1.6、2.0 时的伯德图。

解：MATLAB 程序代码如下。

```
w=[0,logspace(-2,2,200)];                    %w 为 10^-2～10^2 对数等间距分布的 200 个数
wn=0.7;                                      %自然振荡角频率
tou=[0.1,0.4,1.0,1.6,2.0];                   %阻尼比的不同取值
for j=1:5
    sys=tf([wn*wn], [1, 2*tou(j)*wn, wn*wn]);   %不同阻尼比下的系统传递函数
    bode(sys,w)                              %绘制伯德图
    hold on
end
%放置 tou 取不同值的文字注释
gtext('tou=0.1'),gtext('tou=0.4'),gtext('tou=1.0'),gtext('tou=1.6'),gtext('tou=2.0')
```

输出的伯德图如图 4-13 所示。

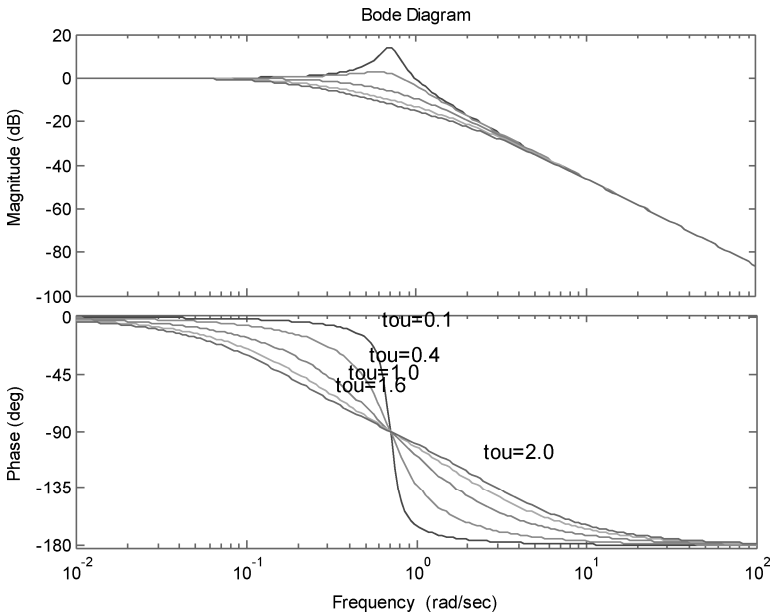


图 4-13　二阶环节的伯德图

【例 4-21】 已知二阶系统的传递函数为 $G(s)=\dfrac{3.6}{s^2+3s+5}$，试计算此系统的谐振峰值和谐振频率。

解：计算谐振峰值和谐振频率的 MATLAB 程序代码如下。

```
function [Mr, Pr, Wr]=mr(G)
[mag, pha, w] = bode(G);        %得到系统伯德图相应的幅值 mag、相位角 pha 与角频率 w 矢量
magn(1,:)=mag(1,:); phase(1,:)=pha(1,:);
[M,i]=max(magn);
```

```
Mr=20*log10(M)              %求得谐振峰值
Pr=phase(1,i)
Wr=w(i,1)                   %求得谐振频率
end
```

主函数的 MATLAB 程序代码如下。

```
num=[3.6];den=[1,3,5];G=tf(num,den);        %建立传递函数
[Mr, Pr, Wr]=mr(G)                          %求取谐振峰值和谐振频率
```

程序的执行结果如下。

```
Mr =
    -2.8098
Pr =
    -24.6446
Wr =
    0.6915
```

由执行结果可知，系统的谐振峰值 $M_r = -2.8098\text{dB}$，谐振频率 $\omega_r = 0.6915\text{rad/s}$。

此外，还可以从 MATLAB 绘制的频率响应曲线上直接得到谐振峰值和谐振频率。步骤是先生成频率响应曲线，然后在频率响应图内部空白处用鼠标右键单击，弹出菜单，选择“Peak Response”菜单项，将在频率响应图上出现一个圆点，该点就是系统的谐振频率。

绘制伯德图的 MATLAB 程序代码如下。

```
num=[3.6];den=[1,3,5];G=tf(num,den);        %建立传递函数
bode(G)                                     %绘制伯德图
```

程序输出如图 4-14 所示的伯德图（图中的弹出菜单非伯德图内容，作用见下）。

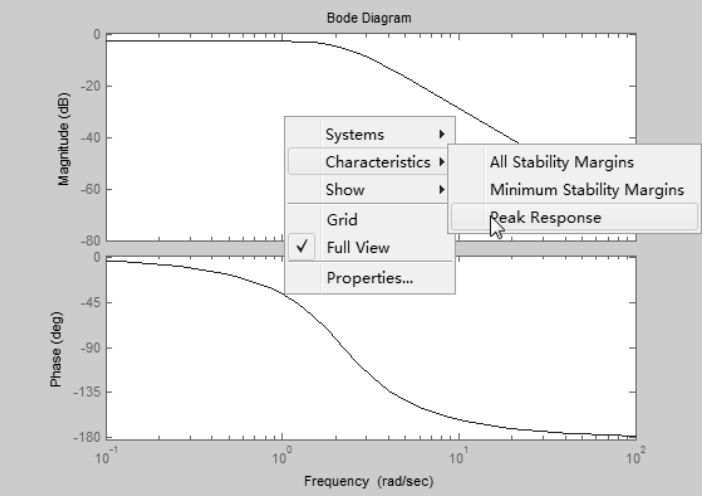


图 4-14　系统伯德图

单击鼠标右键，弹出菜单，选择“Peak Response”菜单项，将光标移至圆点处，输出如图 4-15 所示的图形。从图中可以看出，系统的谐振峰值 $M_r = -2.81\text{dB}$，谐振频率 $\omega_r = 0.692\text{rad/s}$，与前面的计算结果是一致的。

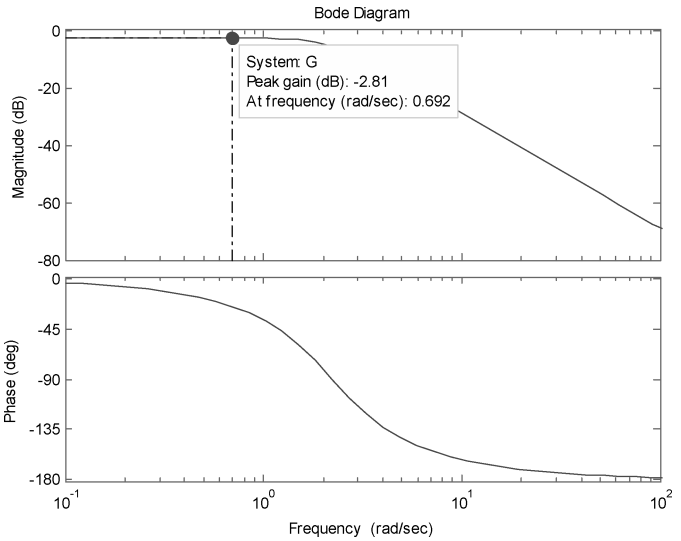


图 4-15　系统伯德图（显示谐振峰值和谐振频率）

4.9　Routh 判据

实际系统通常是高阶的，无法求出所有解，考虑通过特征方程的系数和特征根的关系来判断特征根是否全部具有负的实部。

线性系统的特征方程为：$a_n s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = 0$ 。

判断系统的稳定性，通常使用 Routh（劳斯）判据。其充要条件如下。

① 充分条件：$a_i > 0$ 。

② 必要条件：列劳斯数列

$$\begin{array}{c|ccccc} s^n & a_n & a_{n-2} & a_{n-4} & a_{n-6} & \cdots \\ s^{n-1} & a_{n-1} & a_{n-3} & a_{n-5} & a_{n-7} & \cdots \\ s^{n-2} & b_1 & b_2 & b_3 & b_4 & \cdots \\ s^{n-3} & c_1 & c_2 & c_3 & c_4 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ s^0 & & & & & \end{array}$$

式中

$$b_1 = -\frac{1}{a_{n-1}}\begin{vmatrix} a_n & a_{n-2} \\ a_{n-1} & a_{n-3} \end{vmatrix}, \qquad c_1 = -\frac{1}{b_1}\begin{vmatrix} a_{n-1} & a_{n-3} \\ b_1 & b_2 \end{vmatrix}$$

$$b_2 = -\frac{1}{a_{n-1}}\begin{vmatrix} a_n & a_{n-4} \\ a_{n-1} & a_{n-5} \end{vmatrix}, \qquad c_2 = -\frac{1}{b_1}\begin{vmatrix} a_{n-1} & a_{n-5} \\ b_1 & b_3 \end{vmatrix}$$

$$b_3 = -\frac{1}{a_{n-1}}\begin{vmatrix} a_n & a_{n-6} \\ a_{n-1} & a_{n-7} \end{vmatrix}, \qquad c_3 = -\frac{1}{b_1}\begin{vmatrix} a_{n-1} & a_{n-7} \\ b_1 & b_4 \end{vmatrix}$$

若劳斯数列首列均为正（不变号），则系统稳定。

4.10 频率法的稳定性分析

4.10.1 稳定判据

利用开环对数频率特性曲线判断闭环系统稳定的判据称为对数频率特性稳定性判据，在介绍对数频率特性稳定性判据之前先了解一下有关穿越的概念。

正负穿越的概念：在伯德图上，在开环对数幅频特性曲线的大于零的区间内，沿 ω 增加的方向，对数相频特性曲线自下向上穿过-180° 线为正穿越；对数相频特性曲线自上向下穿过-180° 线为负穿越。

闭环系统稳定的充要条件是：在开环伯德图对数幅频特性大于零的所有频段内，开环对数相频特性曲线对-180° 线的正穿越数与负穿越数之差 N 等于开环传递函数的右极点数 P 的一半，即 $N=P/2$。

4.10.2 稳定裕度

稳定判据可以判断系统是否稳定，但不能知道稳定程度如何，一个实际的控制系统，不仅要求稳定，而且还必须具有一定的稳定储备，即相对稳定性的概念。所谓相对稳定性，是指稳定系统的稳定状态距离不稳定（或临界稳定）状态的程度。

4.10.3 相位裕度

当 ω 为剪切频率 ω_c（$\omega_c>0$）时，相频特性距-180° 线的相位差 γ 称为相位裕度。

对于稳定系统，γ 必在伯德图横轴以上，这时称为正相位裕度；对于不稳定系统，γ 必在伯德图横轴之下，这时称为负相位裕度。

4.10.4 幅值裕度

当 ω 为相位交界频率 ω_g（$\omega_g>0$）时，开环幅频特性的倒数，称为系统的幅值裕度，即

$$K_g=\frac{1}{|G(j\omega_g)H(j\omega_g)|}$$

在伯德图上，幅值裕度改以分贝表示为

$$20\lg K_g=20\lg\frac{1}{|G(j\omega_g)H(j\omega_g)|}=-20\lg|G(j\omega_g)H(j\omega_g)|\rightarrow K_g\ (\mathrm{dB})$$

此时，对于稳定系统，K_g（dB）必在 0dB 线以下，K_g（dB）>0，此时称为正幅值裕度；对于不稳定系统，K_g（dB）必在 0dB 线以上，K_g（dB）<0，此时称为负幅值裕度。

综上所述，对开环为 $P=0$ 的系统的闭环系统来说，$G(j\omega)H(j\omega)$ 具有正幅值裕度与正相位裕度时，其闭环系统是稳定的；$G(j\omega)H(j\omega)$ 具有负幅值裕度或负相位裕度时，其闭环系统是不稳定的。

【例 4-22】 已知控制系统的开环传递函数为

$$G(s)H(s)=\frac{K}{s(s+1)(s+5)}$$

试分别求取 K 为 10 和 100 时的相位裕度 γ 和幅值裕度 K_g（dB），并判断闭环系统的稳定性。

解：在 MATLAB 环境中编制程序，其源代码如下。

```
den=conv([1 5],[1 1 0]);
K=[10 100];
%
[Gm1 Pm1 Wg1 Wc1]=margin(K(1),den);
subplot(211)
margin(K(1),den)
grid
%
[Gm2 Pm2 Wg2 Wc2]=margin(K(2),den);
subplot(212)
margin(K(2),den)
grid
[K(1) 20*log10(Gm1) Pm1 Wg1 Wc1;K(2) 20*log10(Gm2) Pm2 Wg2 Wc2]
```

执行上述程序得到如图 4-16 所示的曲线图，计算结果见表 4-2。

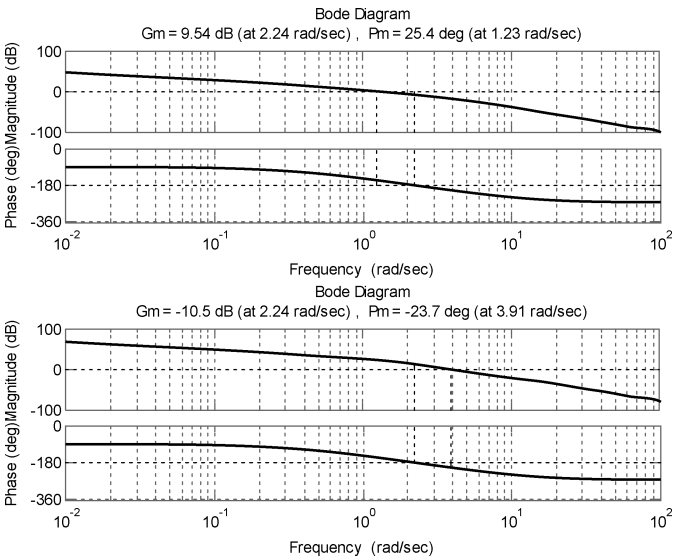


图 4-16　不同增益下的伯德图

表 4-2　计算结果

K	K_g（dB）	γ	ω_g	ω_c
10	9.5424	25.3898	2.2361	1.2271
100	−10.4576	−23.6504	2.2361	3.9073

根据表 4-2 的计算结果：K=10，开环系统是稳定的；K=100，开环系统是不稳定的。

课后题

1．已知系统的动力学方程如下，试写出其传递函数并用 MATLAB 来描述。

（1）$\frac{\mathrm{d}y^3(t)}{\mathrm{d}t^3}+15\frac{\mathrm{d}y^2(t)}{\mathrm{d}t^2}+50\frac{\mathrm{d}y(t)}{\mathrm{d}t}+500y(t)=\frac{\mathrm{d}r^2(t)}{\mathrm{d}t^2}+2\frac{\mathrm{d}r(t)}{\mathrm{d}t}$；

（2）$5\frac{\mathrm{d}y^2(t)}{\mathrm{d}t^2}+25\frac{\mathrm{d}y}{\mathrm{d}t}=0.5\frac{\mathrm{d}r(t)}{\mathrm{d}t}$。

2．若某线性定常系统在单位阶跃输入作用下，其输出为 $y(t)=1-\mathrm{e}^{-3t}+2\mathrm{e}^{-t}$，试求系统的传递函数。

3．求一阶惯性系统在斜坡信号作用下系统的输出。

4．假设系统由两个模块 $G_1(s)$ 和 $G_2(s)$ 连接而成，已知

$$G_1(s)=\frac{s+1}{s^2+3s+4},G_2(s)=\frac{s^2+3s+5}{s^4+4s^3+3s^2+2s+1}$$

试求：（1）将两个传递函数模型进行串联，然后写出整个系统的传递函数模型；

（2）求串联系统的阶跃响应。

5．什么是时间响应？

6．时间响应由哪两部分组成？各部分的定义是什么？

7．系统的单位脉冲响应函数如下，试求这些系统的传递函数。

（1）$x_o(t)=0.0125\mathrm{e}^{-1.25t}$；

（2）$x_o(t)=5t-10\sin\left(4t+\frac{\pi}{4}\right)$；

（3）$x_o(t)=0.1(1-\mathrm{e}^{-t/3})$。

8．使用 MATLAB 表示下列系统，并绘制它们的单位阶跃响应和单位脉冲响应曲线。

（1）$G(s)=\frac{0.8s+4}{s^2+1.6s+4}$；

（2）$G(s)=\frac{s^3+7s^2+24s+24}{s^4+10s^3+35s^2+50s+24}$。

9．图 4-17 为某数控机床系统的位置随动系统方框图，试求：

（1）阻尼比 ξ 和无阻尼固有频率 ω_n；

（2）该系统的 M_p、t_p、t_s 和 N。

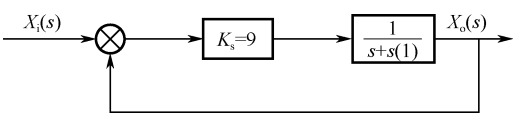


图 4-17　位置随动系统方框图

10．某系统结构优化设计后，得到最优闭环传递函数为

$$G_b(s)=\frac{1500}{s^4+185.8s^3+1088.4s^2+1800s+1500}$$

试用 MATLAB 编程的方法和计算机辅助图解法证明系统的动态性能指标为 $M_p\leqslant 5\%$，

$t_p \leqslant 3.25s$，$t_s \leqslant 3.5s$。

11．什么是频率特性？

12．试用 MATLAB 编制程序和利用 MATLAB 函数，分别绘制具有下列传递函数的各系统的伯德图。

（1）$G(s)=\dfrac{1}{1+0.01s}$；

（2）$G(s)=\dfrac{1}{1-0.01s}$；

（3）$G(s)=\dfrac{1}{s(1+0.5s)(1+2s)}$；

（4）$G(s)=\dfrac{7.5(0.2s+1)(s+1)}{s(s^2+16s+100)}$；

（5）$G(s)=\dfrac{20(s+5)(s+40)}{s(s+0.1)(s+20)^2}$。

13．已知单位反馈系统的开环传递函数为

$$G_k(s)=\frac{10}{s(0.05s+1)(0.1s+1)}$$

试用 MATLAB 编制程序计算系统的 M_r 和 ω_r。

14．系统稳定性的定义是什么？

15．一个系统稳定的充分和必要条件是什么？

16．系统特征方程如下。

$$s^4+Ks^3+s^2+s+1=0$$

应用 Routh 判据，确定系统稳定时 K 值的范围。

17．设系统如图 4-18 所示，试判别该系统的稳定性，并求出其稳定裕度，其中，$K_1=0.5$。

（1）$G(s)=\dfrac{2}{s+1}$；

（2）$G(s)=\dfrac{2}{s}$。

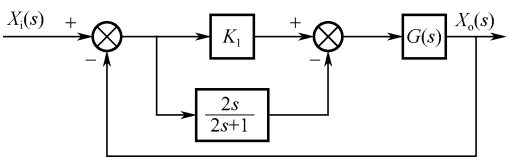


图 4-18　控制系统方框图

第 5 章　常用的数值计算方法及 MATLAB 数值计算函数

5.1　非线性方程（组）的数值解法

在科学和工程技术计算中的许多问题常归结为求解方程，即

$$f(x)=0 \tag{5.1}$$

当函数 $f(x)$ 是一次多项式时，称式（5.1）为线性方程；若式（5.1）中包含三角函数、指数函数等超越函数（如 $\sin x$ 、 e^x 、 $\ln x$ 等）时，则称式（5.1）为超越方程。它与 n（$n\geqslant 2$）次代数方程一起统称非线性方程。

如果有 x^* 使得 $f(x^*)=0$，则称 x^* 为式（5.1）的根，或称为 $f(x)$ 的零点。设有正整数 m 使得

$$f(x)=(x-x^*)^m g(x)$$

且 $g(x^*)\neq 0$，则当 $m\geqslant 2$ 时，称 x^* 为式（5.1）的 m 重根；当 m=1 时，称 x^* 为式（5.1）的单根。

关于 n 次代数方程

$$a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x+a_n=0 \tag{5.2}$$

当 $n\leqslant 4$ 时，有求根公式；当 n>4 时，方程就没有现成的求根公式了。但是由数学基本定理知，n 次代数方程一定有 n 个根，包括复数，当然重根应按照重数计算根的个数。对于超越方程不仅没有一般的求根公式，而且若只依据方程本身，那么连有没有根，有几个根也难以判断。然而，在实际应用问题中，不一定要求得到根的解析式，只要求获得具有一定精确程度的近似值（即数值解）就可以了。求解非线性方程的数值解一般可分如下 3 步进行。

（1）判断根的存在性，即方程有没有根，如果有根，有几个根。

（2）确定根的近似位置，即求出根所在区间或确定根的初始近似值。

（3）根的精确化，即按某种方法将初始近似值逐步精确化，直到满足所要求的精确度。

在 MATLAB 中，求解方程 $f(x)=q(x)$ 的根可以用下面的命令。

```
>>x=solve('方程 f(x)=q(x)'，'待求符号变量 x')
```

求方程组 $f_i(x_1,\cdots,x_n)=q_i(x_1,\cdots,x_n)\ (i=1,2,\cdots,n)$ 的根可以用 MATLAB 命令：

```
>>E1=sym('方程 f1(x1,...,xn)=q1(x1,...,xn)');
…
En=sym('方程 fn(x1,...,xn)=qn(x1,...,xn)');
[x1,x2,...,xn]=solve(E1,E2,...,En,x1,...,xn);
```

【例 5-1】 求超越方程组 $\begin{cases} x^x - 4 = 0 \\ 2xy + x = 1 \end{cases}$。

解：编写 MATLAB 文件，代码如下。

```
syms x y
[x,y]=solve('x^x-4=0','2*x*y+x=1');
x1=double(x),y1=double(y)
```

执行结果为如下。

```
x1 =
     2
y1 =
   -0.2500
```

5.2　微分方程的数值计算

微分方程在科技、工程、经济管理、生态、环境、人口、交通等各个领域中常用于建立数学模型，它是研究函数变化规律的有利工具，如在研究弹性物体的振动，电阻、电容、电感电路的瞬变，热量在介质中的传播，抛射体的轨迹，污染物浓度的变化，人口增长的预测，种群数量的演变，交通流量的控制等过程中，作为研究对象的函数，要和函数的导数一起，用一个符合其内在规律的方程，即微分方程来描述。

建立微分方程只是解决问题的第一步，通常需要求出方程的解来说明实际现象，并加以检验。如果能够得到解析解固然便于分析和应用，但是我们知道，只有线性常系数微分方程，并且自由项是某些特殊类型的函数时，才可以肯定得到这样的解，而绝大多数变系数方程、非线性方程都很难得到解析解。所以，对用微分方程解决实际问题来说，数值解法就是一个十分重要的手段。

综上所述，求解常微分方程，可以采用符号解（解析解）的方法，也可以采用数值解的方法。对于以上两种方法，MATLAB 都提供了支持。

5.3　常微分方程（组）的解析解法

在 MATLAB 系统中提供了常微分方程（组）解析解（symbolic solution of ordinary differential equations）的函数 dsolve。

在调用此函数前，必须首先将给定的常微分方程（组）中的一阶导数用 D 表示，例如，d*y*/d*x* 写成 Dy，*n* 阶导数用 Dn 表示，例如，d^6y/dx^6 表示为 D6y。由此，常微分方程 $y'' + 4y' = y^2 + x^2$ 写成 D2y+4Dy=y^2+x^2。下面具体介绍函数 dsolve 解常微分方程（组）的方法。

5.3.1　用 MATLAB 求常微分方程（组）的通解

用 MATLAB 函数 dsolve 求常微分方程

$$F(x, y, y', y'', \cdots, y^{(n)}) = 0 \tag{5.3}$$

的通解的主要调用格式如下。

调用格式一：S=dsolve('eqn','var')

输入量：eqn 是常微分方程（5.3）改用符号方程表示的常微分方程[即 F(x, y, Dy, D2y,⋯, Dny) = 0]；var 表示自变量，默认的自变量为 t。

输出量：S 是常微分方程（5.3）的通解。

求常微分方程组

$$\begin{cases} F_1(x, y_1, y_1', y_1'', \cdots, y_1^{(n)}) = 0 \\ F_2(x, y_2, y_2', y_2'', \cdots, y_2^{(n)}) = 0 \\ \cdots \\ F_n(x, y_n, y_n', y_n'', \cdots, y_n^{(n)}) = 0 \end{cases} \tag{5.4}$$

的通解的主要调用格式如下。

调用格式二：S=dsolve('eqn1','eqn2',...,'eqn*m*','var')

输入量：eqn1,eqn2,⋯,eqn*m* 分别是常微分方程组（5.4）中用符号方程表示的 *m* 个常微分方程。默认的自变量为 t，也可以将自变量 t 变为其他的符号变量 var。

输出量：S 是常微分方程组（5.4）的通解。

5.3.2 用 MATLAB 求常微分方程（组）的特解

如果给定常微分方程（5.3）的初始条件

$$y(x_0) = a_0, y'(x_0) = a_1, \cdots, y^{(n)}(x_0) = a_n \tag{5.5}$$

则求方程（5.3）的特解的主要调用格式如下。

调用格式三：S=dsolve('eqn','condition1',...,'condition*n*','var')

输入量：eqn 是常微分方程(5.3)改用符号方程表示的常微分方程；condition1,⋯,condition*n* 是初始条件式（5.5）；var 表示自变量，默认的自变量为 t。

输出量：S 是常微分方程（5.3）的特解。

同理，如果给定常微分方程组（5.4）的初始条件，则求方程组（5.4）的特解的主要调用格式如下。

调用格式四：S=dsolve('eqn1','eqn2',...,'eqn*m*','condition1','condition2',...,'var')

输入量：eqn1,eqn2,⋯,eqn*m* 分别是常微分方程组（5.4）中用符号方程表示的 *m* 个常微分方程；condition1,condition2,⋯是常微分方程组的初始条件；默认的自变量为 t，也可以将自变量 t 变为其他的符号变量 var。

输出量：S 是常微分方程组（5.4）的特解。

【例 5-2】 求下列微分方程组的特解。

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = y \\ \dfrac{\mathrm{d}y}{\mathrm{d}t} = -x \end{cases} \tag{5.6}$$
$$x(0) = 10, y(0) = 2$$

解：输入 MATLAB 程序代码如下。

```
z=dsolve('Dx=y','Dy=-x','x(0)=10','y(0)=2');
z.x
4/exp(t)+6*exp(t)
z.y
6*exp(t)-4/exp(t)
```

5.4　常微分方程（组）的数值求解

5.4.1　数值解决基本原理

在 5.3 节的讨论中我们看到，利用符号运算可以寻求常微分方程（组）的初等函数公式解，或称符号解（包含通解和特解）。但是存在这种解的常微分方程（组）的类型很少，往往只局限于线性常系数常微分方程（组）或少数低阶的常微分方程（组）以及少数线性变系数常微分方程（组）。对于更加广泛的、非线性的一般常微分方程（组），通常不存在有初等函数公式解。由于实际问题求解的需要，经常采用数值解法求常微分方程（组）的数值解。常见的数值解法有欧拉（Euler）法、亚当斯（Adams）方法、龙格−库塔（Runge-Kutta）方法等。因为龙格−库塔方法的精度较高，计算量适中，常被人们采用。

解微分方程通常需要附加某种定解条件。微分方程和定解条件组成定解问题。定解条件通常有两种方法给出：一种是给出积分曲线在初始时刻的性态，这类条件称为初始条件，对应的定解问题称为初值问题；另一种是给出积分曲线在首末两端的性态，这类条件称为边界条件，对应的定解问题称为边值问题。

关于一阶常微分方程初值问题有下面的定理。

对于常微分方程初值问题有

$$\frac{\mathrm{d}y}{\mathrm{d}x}=f(x,y),y(x_0)=y_0 \tag{5.7}$$

如果 $x_0\in[a,b]$，函数 $f(x)$ 对 x 连续，且对 y 满足利普希茨（Lipschitz）条件，即存在常数 $L>0$，使

$$|f(x,y_1)-f(x,y_2)|\leqslant L|y_1-y_2| \tag{5.8}$$

对所有 $x\in[a,b]$ 及任意实数 y_1 和 y_2 都成立，则初值问题式（5.3）和式（5.3）在闭区间 $[a,b]$ 上有唯一解。

如果微分方程（组）难以求解或根本无解析解，则用微分方程（组）的数值解法。所谓数值解法就是求初值问题式（5.7）的解 $y=y(x)$ 在一系列离散点

$$x_0<x_1<x_2<\cdots<x_n<\cdots$$

处 $y(x_n)$ 的近似值 $y_n(n=1,2,\cdots)$，通常取等步长 h，则 $x_n=x_0+nh(n=1,2,\cdots)$。

建立数值解法，首先将微分方程离散化，一般经常采用数值微分法、数值积分法和泰勒（Taylor）逼近法等，下面简单介绍这些方法的基本思想。

1．数值微分法的基本思想

如果 $f(x,y)$ 中的 x 取小区间 $[x_n,x_{n+1}]$ 的左端点 x_n，用向前差商 $\dfrac{y(x_{n+1})-y(x_n)}{h}$ 代替 $y'(x_n)$

代入式（5.7）中的导数，则式（5.7）中的微分方程化为

$$\frac{y(x_{n+1})-y(x_n)}{h}\approx f[x_n,y(x_n)]\,(n=0,1,2,\cdots)$$

化简得

$$y(x_{n+1})\approx y(x_n)+hf[x_n,y(x_n)]\,(n=0,1,2,\cdots)$$

如果用 $y(x_n)$ 的近似值 y_n 代入上式的右端，所得的结果 y_{n+1} 作为 $y(x_{n+1})$ 的近似值，则有

$$y_{n+1}=y_n+hf(x_n,y_n)\,(n=0,1,2,\cdots) \tag{5.9}$$

这样，求常微分方程初值问题式（5.8）的数值解，可以通过求解

$$\begin{cases}y_{n+1}=y_n+hf(x_n,y_n)\,(n=0,1,2,\cdots)\\ y_0=y(x_0)\end{cases} \tag{5.10}$$

得到。将初值 (x_0,y_0) 代入式（5.9），得到 $y(x_1)$ 的近似值，即

$$y_1=y_0+hf(x_0,y_0)$$

再将 (x_1,y_1) 代入式（5.9），得到 $y(x_2)$ 的近似值为

$$y_2=y_1+hf(x_1,y_1)$$

继续下去，可以逐次求出 $y_1,y_2,\cdots$。式（5.9）称为向前欧拉公式。式（5.10）是个离散化的问题，也称为差分方程初值问题。式（5.7）中方程的右端，已知函数 $f(x,y)$ 中的 x 在小区间 $[x_n,x_{n+1}]$ 上取不同的点，或用不同的差商近似导数公式，则将得到不同的公式。

2．数值积分法的基本思想

将初值问题式（5.7）的解 $y(x)$ 表示成积分形式，然后用数值积分方法离散化。例如，将式（5.7）中的方程改写成 $\mathrm{d}y=f(x,y)\mathrm{d}x$，等式两边在小区间 $[x_n,x_{n+1}]$ 上积分得

$$y(x_{n+1})-y(x_n)=\int_{x_n}^{x_{n+1}}f(x,y(x))\mathrm{d}x\,(n=0,1,2,\cdots) \tag{5.11}$$

如果用 y_n 和 y_{n+1} 作为 $y(x_n)$ 和 $y(x_{n+1})$ 的近似值分别代入式（5.11）的左端，而右端的积分用左端点矩形公式

$$\int_{x_n}^{x_{n+1}}f[x,y(x)]\mathrm{d}x\approx hf(x_n,y_n)\,(n=0,1,2,\cdots)$$

近似替代，则有

$$y_{n+1}-y_n=hf(x_n,y_n)\,(n=0,1,2,\cdots)$$

移项后也得到向前欧拉公式（5.9），再代入初值问题式（5.7）中，得式（5.10）。用不同的数值积分公式，将得到不同的计算微分方程初值问题的数值公式。

3．泰勒（Taylor）多项式逼近法基本思想

如果将函数 $y(x)$ 在小区间 $[x_n,x_{n+1}]$ 的左端点 x_n 进行泰勒展开，取一阶泰勒多项式逼近，则得

$$y(x_{n+1})=y(x_n)+hy'(x_n)+\frac{1}{2}h^2y''(\xi_n)\approx y(x_n)+hf(x_n,y(x_n))\,(n=0,1,2,\cdots)$$

如果用 y_n 和 y_{n+1} 作为 $y(x_n)$ 和 $y(x_{n+1})$ 的近似值分别代入上式，也得到向前欧拉公式（5.9），代入初值问题式（5.7）中，得式（5.10）。$y(x)$ 在不同的点展开泰勒公式，将得到不同的公式。

以上三种方法都是将微分方程离散化的常用方法，取不同的点或多项式的次数，又可以

得到不同类型的微分方程的数值计算公式。其中泰勒多项式逼近法不仅可以推出数值计算公式，而且可以得到截断误差公式。

从另一种角度上说，常微分方程初值问题的数值解法分为两大类：单步法和多步法。单步法是在计算 y_{n+1} 时，只用到了前一步的值 y_n 、 x_n 和 x_{n+1}。因此，有了初值就可以往下计算。例如，龙格-库塔方法。多步法是在计算 y_{n+1} 时，除了用到前一步的值 x_{n+1} 、 x_n 和 y_n 以外，还要用到前面 k 步的值 x_{n-j} 和 y_{n-j} $(j=1,2,\cdots,k)$。例如，亚当斯方法。

5.4.2　解微分方程（组）的欧拉法

一般微分方程的数值解法很大一类是关于微分方程初值问题的数值解法，这类问题需要用一阶显示的微分方程（组）来描述，即

$$\dot{x}(t)=f(t,x(t)) \tag{5.12}$$

其中， $x^{\mathrm{T}}(t)=[x_1(t),x_2(t),\cdots,x_n(t)]$ 称为状态向量，方程左侧是状态变量的一阶导数向量。右侧的函数 $f^{\mathrm{T}}(\cdot)=\{f_1(\cdot),f_2(\cdot),\cdots,f_n(\cdot)\}$ 可以是任意非线性函数。所谓初值问题是指，若已知初始状态 $x_0=[x_1(0),\cdots,x_n(0)]^{\mathrm{T}}$ ，用数值求解方法求出在某个时间区间 $t\in[0,t_{\mathrm{f}}]$ 内各个时刻状态变量 $x(t)$ 的数值，这里 t_{f} 又称为终止时间。

对于多元非线性常微分方程初值问题来说，Euler 算法是最直观的一类求解算法。虽然该算法比较简单，但对理解其他复杂的微分方程算法是很有帮助的，故这里将以 Eular 算法为例介绍微分方程初值问题的数值算法。

假设已知在 t_0 时刻系统状态向量的值为 $x_0(t)$ ，若选择一个很小的计算步长 h，则可以将微分方程左侧的导数近似为

5.4.3　四阶定步长龙格-库塔算法

四阶定步长的龙格-库塔算法是传统数值分析课程和系统仿真课程中经常介绍的算法，被认为是求解微分方程的一种有效的方法，该算法结构很简单，可以先定义如下 4 个附加向量。

$$\begin{cases} k_1=hf(t_k,x_k) \\ k_2=hf\left(t_k+\dfrac{h}{2},x_k+\dfrac{k_1}{2}\right) \\ k_3=hf\left(t_k+\dfrac{h}{2},x_k+\dfrac{k_2}{2}\right) \\ k_4=hf(t_k+h,x_k+k3) \end{cases} \tag{5.13}$$

式中　h——计算步长。

在实际应用中该步长是一个常数，这样由四阶龙格-库塔算法可以求解出下一个步长的状态变量值为

$$x_{k+1}=x_k+\frac{1}{6}(k_1+2k_2+2k_3+k_4) \tag{5.14}$$

这样，用迭代的方法由给定的初值问题逐步求出在所选择的时间段 $t\in[t_0,t_h]$ 内各个时刻 t_0+h 、 t_0+2h 等处的原问题数值解。

5.4.4 常微分方程（组）的刚性和非刚性

根据常微分方程（组）的描述问题的性质，可将常微分方程划分为刚性方程和非刚性方程。这里对刚性方程只给直观的描述，更详细的内容请参见有关书籍。

自然界各种现象的发生的每一个过程都是极其复杂的，往往包含许多子过程及它们之间的相互作用，其中有些过程表现为快的变化，而有些则相对慢些，且变化的速度可以相差大的数量级，相应地描述这些子过程的常微分方程（组）的解中也将包含快变化和慢变化的分量，如果在一个过程中的快变量过程与慢变量过程的变化速度相差非常大，在数学上称这种过程具有“刚性”，而描述这种过程的常微分方程（组）称为刚性方程（组），也称为病态方程（组）或坏条件方程（组），否则称为非刚性方程（组）。对于一阶常系数线性微分方程组 $\dot{\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{f}$，如果它的雅可比矩阵 $\boldsymbol{A}_{n\times n}$ 的特征值相差十分悬殊，则称此微分方程组是刚性的。刚性问题在控制系统工程、电子网络、生物学、物理学及化学动力学过程中经常遇到。这是由于这种性质，使得传统的常微分方程（组）数值解法遇到极大的困难。在 MATLAB 系统中，提供了数值解刚性和非刚性常微分方程的程序。根据微分方程组刚性的不同情况（特殊值相差悬殊的情况），应该采用不同的求解函数，才能得到较好的效果。

5.4.5 解常微分方程（组）初值问题的 MATLAB 库函数

在 MATLAB 系统中，有 7 个专门用于解常微分方程初值问题的库函数，具体的库函数名、解决问题的类型、精度及其适用范围见表 5-1。

表 5-1 解常微分方程初值问题的库函数

Odeij	问 题 类 型	精 度	适 用 对 象
ode45	非刚性	中等	多数情况下可优先适用，但不能用来解决刚性问题
ode23	非刚性	低到高	不能解刚性问题，当误差容限要求严格时，效果较 ode45 好
ode113	非刚性	低到高	不能解刚性问题，当误差容限要求严格时，效果较 ode 好
ode23t	适度刚性	低	可用来解刚性问题，但要求无数值衰减
ode15s	刚性	低到高	可用来解刚性问题，当采用 ode45 失败或效果很差时，可考虑适用
ode23s	刚性	低	可用来解刚性问题，当误差容限较宽时，效果较 ode15s 好
ode23tb	刚性	低	可用来解刚性问题，当误差容限较宽时，效果较 ode15s 好

在表 5-1 中，ode23、ode45、ode23s 等库函数主要采用自适应龙格-库塔方法，其中 ode23 系列为采用二阶、三阶龙格-库塔方法，ode45 系列为采用四阶、五阶龙格-库塔方法。一般来说，ode45 比 ode23 的积分段少、运算速度更快一些。表 5-1 中每个库函数的详细说明，请在 MATLAB 工作窗口输入命令“help 函数名”进行查询。

表 5-1 中每个库函数的一般调用格式有如下几种。

调用格式一：[T,Y]=odeij(@ODEFUN,TSPAN,Y0)

也可以书写为：[T,Y]=odeij('ODEFUN',TSPAN,Y0)，下同。

输入量：①ODEFUN 是由待解得微分方程（组）$y' = f(t,y)$ 中函数 $f(t,y)$ 写成 M 函数文件名；Y0 是初始条件；用于解 n 个未知函数的方程组时，Y0 和 Y 均为 n 维向量，M 文件中

的待解方程组应以 Y 的分量形式写成。

② TSPAN=[T0 TFINAL]是积分区间，其中 T0、TFINAL 分别为自变量 t 的初值和终值；一般情况下输入 TSPAN=[T0 TFINAL];如果要去输入在指定点 t=T0,T1,…,TFINAL 处的微分方程的数值解 Y=Y0,Y1,…,YFINAL，则输入 TSPAN=[T0 T1 ... TFINAL]；等步长时用 t=T0:k:TFINAL，输出 Y 是在[T0 TFINAL]上的等分点 t=T0+nk(n=0,1,2,…,m)处的数值解。

③ t 的步长是程序根据误差限自动选定的；

④ odeij 表示表 5-中每个库函数名。

输出量：T 和 Y 是自变量 t 和 $y' = f(t, y)$ 在自变量 t 的值（列向量）T 处的数值解（列向量）。

调用格式二：[T,Y]=odeij(@ODEFUN,TSPAN,Y0,OPTIONS)

① 输入量 ODEFUN、TSPAN、Y0、odeij 和输出 Y 和 T 同上。

② OPTIONS 是用来设置一些可选的参数值，缺省时相当调用格式一。一般地被用于设定误差限，即数量的相当误差限‘RelTol’（默认值为 1e-3）和向量的各分量的绝对误差限‘AbsTol’（各分量的默认值为 1e-6）。OPTIONS 是用函数 ODESET 创建的，详细内容请查看“help ODESET”。

调用格式三：[T,Y]=odeij(@ODEFUN,TSPAN,Y0,OPTIONS,P1,P2,...)

① 输入量 ODEFUN、TSPAN、Y0、odeij、OPTIONS 和输出 Y 和 T 同上。

② P1,P2,…的作用是将附加参数 P1,P2,…传递给 ODE 文件（如 ODEFUN(T,Y,P1,P2,...)）和在 OPTIONS 中的所有函数。如果没有 OPTIONS 参数设置，则用 OPTIONS=[ ]代替，以便正确传递参数。如果 TSPAN 或 Y0 没有被指定，则 odeij 函数调用 ODE 文件，令[TSPAN,Y0,OPTIONS]= ODEFUN([],[],'init')来获得在 odeij 函数调用时没有提供的相关参数值，甚至调用参数列表中末尾的没有的参数，如格式 odeij(@ODEFUN)。

调用格式四：[T,Y,TE,YE,IE]=odeij(@ODEFUN,TSPAN,Y0,OPTIONS,P1,P2,...)

此调用格式在 OPTIONS 中设置了一个函数 EVENTS，使其调用格式具有‘Events’性质，在求如上的解得同时，还找(T,Y)的哪些函数值是零。在设定函数 EVENTS 时，才输出 TE、YE、IE。其中 TE 是事件发生时的自变量 t 构成的列向量，YE 的每一行是对应 TE 的数值解，向量 IE 指出哪一个事件发生。

求解一阶显示微分方程组的关键是用 MATLAB 语言编写一个函数，描述需要求解的微分方程组。该函数的入口应为

```
function xd=funname(t,x)
function xd=funname(t,x,flag,p1,p2,…)
```

其中，t 是时间变量或自变量，即使需要求解的微分方程不是时变的，也需要给出 t 占位，否则在变量传递过程中将出现问题。变量 x 为状态向量，返回的 xd 为状态向量的导数。flag 一般用来控制求解过程，可以用其给初值状态赋值。

在微分方程求解中有时需要对求解算法及控制条件进行进一步设置，这可以通过求解过程中的 options 变量进行修改。初始 options 变量可以通过 odeset()函数获取，该变量是一个结构体变量。

修改该变量有两种方式：其一是用 odeset()函数设置；其二是直接修改 options 的成员变量。例如，若想将相对误差设置成较小的 10^{-7}，则需要给出

```
options=odeset('RelTol',1e-7);
```

或者更直观地写成

```
options=odeset;options.RelTol=1e-7;
```

在实际求解过程中经常需要定义一些附加参数，这些参数由 p1,p2,…,pm 表示，在编写方程函数时也应该一一对应地写出，在这种调用格式下还应该使用 flag 变量占位。

【例 5-3】 在区间[0,10]上求解微分方程 $\frac{\mathrm{d}y}{\mathrm{d}t}=2t$ 在初始值 $y(0)=0$ 的数值解。

解：① 先编写一个 M-Function，并以 Fun.m 为文件名存盘。

```
%常微分方程
function f = fun(t,y)
    f = 2*t;
end
```

② 再编写一个 M-Script，调用 ode45 对微分方程进行数值求解。

```
%数值法求解微分方程
[t,y] = ode45(@Fun,[0,10],0);
plot(t,y)
```

从上面的微分方程求解实例看，如果能建立一个描述微分方程的 MATLAB 函数，则调用 ode45()将立即得出该微分方程的数值解。可以看出，编写一个 MATLAB 函数来描述微分方程是常微分方程数值求解的关键。下面将介绍能描述微分方程的另一种方法—匿名函数定义法。

如微分方程组

$$\begin{cases} \dot{x}_1(t)=-x_2(t)-x_3(t) \\ \dot{x}_2(t)=x_1(t)+0.2x_2(t) \\ \dot{x}_3(t)=0.2+[x_1(t)-5.7]x_3(t) \end{cases} \tag{5.15}$$

则该函数文件可以这样书写：

```
function dx=rossler(t,x)                    %虽然不显示时间，还应该写成占位
dx=[-x(2)-x(3); %对应方程第一行，直接将参数代入
    x(1)+0.2*x(2);
    0.2+(x(1)-5.7)*x(3)];
```

如果上述微分方程，采用匿名定义的方法，可以写成

```
rossler=@(t,x)[-x(2)-x(3);x(1)+0.2*x(2);0.2+(x(1)-5.7)*x(3)]
```

其中，rossler 为生成的函数名，可以用 ode45()这类微分方程求解函数直接调用。这里自变量仍为 t 和 x。

另外，在基于 MATLAB 语言的微分方程求解中，还可以引入附加参数。引入附加参数的目的是，若微分方程的某些参数可以选择不同的值，对不同值求解时考虑附加参数可以避免每次修改模型文件。同样还是上例，可以这样编写 MATLAB 函数文件，即

```
rossler=@(t,x,a,b,c)[-x(2)-x(3);x(1)+a*x(2);b+(x(1)-c)*x(3)]
```

这样求解的微分方程语句可以类似修改成

```
t_final=100;
x0=[0;0;0];
a=0.3;b=2;c=3;
opt=odeset;
opt.RelTol=1e-8;
[t,x]=ode45(rossler,[0,t_final],x0,opt,a,b,c);
plot(t,x),figure
plot3(x(:,1),x(:,2),x(:,3))
```

5.5　高阶常微分方程（组）的数值解法

ode23()和 ode45()不能直接求解高阶常微分方程，但可以利用变量变换（change of variables）法将高阶常微分方程改写成一阶微分方程组的形式，然后再利用 ode23()或 ode45()求解。

例如 n 阶 ODE 方程为

$$y^n = g(x, y, y', y'', \cdots, y^{n-1})$$

先定义 n 个新的变量来取代上式中的 $y, y', y'', \cdots, y^{n-1}$，即

$$u_1(x) = y^{n-1}, u_2(x) = y^{n-2}, \cdots, u_{n-1}(x) = y', u_n(x) = y$$

将上述的新变量代入原常微分方程，连同这些新变量构成联立的一阶 ODE 方程组，即

$$\begin{cases} u_1'(x) = y^n = g(x, y, y', y'', \cdots, y^{n-1}) \\ u_2' = u_1(x) \\ \cdots \\ u_n'(x) = u_{n-1}(x) \end{cases}$$

【例 5-4】 二阶常微分方程 $y'' = y'(1 - y^2) - y$ 的初始条件为 $y(0) = 0$，$y'(0) = 0.25$，用数值方法求解该方程。

分析： 定义 $u_1(x) = y', u_2(x) = y$，代入原常微分方程，构成一阶常微分方程组为

$$\begin{cases} u_1' = y'' = u_1(1 - u_2^2) - u_2 \\ u_2' = u_1 \end{cases}$$

解： 下面是求解二阶常微分方程组的程序。

```
%m-function,eqns.m
function du = eqns(x, u)
du = [u(1).*(1-u(2).^2)-u(2);u(1)];                    %用矩阵表示方程组
```

然后编写求解的 M-Script 文件

```
[x,y] = ode23('eqns',[0,20],[0 0.25]);
subplot(2,1,1)
plot(x,y(:,1)),grid
title('1st derivative of y'),xlabel('x')
subplot(2,1,2)
```

```
plot(x,y(:,2)),grid
title('y'),xlabel('x')
```

5.6 微分代数方程求解

所谓微分代数方程（Differential Algebraic Equations，DAE），是指在微分方程中，某些变量满足某些代数方程的约束。假设微分方程的更一般形式可以写成

$$M(t,x)x' = f(t,x) \tag{5.16}$$

前面所介绍的 ODE 数值解法主要针对能够转换为一阶常微分方程组的类型，故 DAE 就无法使用前面介绍的常微分方程解法直接求解，必须借助 DAE 的特殊解法。

其实对于我们使用 MATLAB 求解 DAE 时，却没有太大的改变，只需增加一个 Mass 参数即可。描述 $f(t,x)$ 的方法和普通微分方程完全一致。

值得注意的是，ode15i 无法设置 Mass 属性，换句话说除了 ode15i 外其他 ode 计算器都可以求解 DAE 问题。

使用时需要注意下面的问题。

（1）当 $M(t,x)$ 非奇异时，我们可以将微分方程等效的转换为 $x' = inv(M)f(t,x)$，此时就是一个普通的 ODE（常微分方程组，当然我们可以将它当成 DAE 处理），对任意一个给定的初值条件都有唯一的解。

（2）当 $M(t,x)$ 奇异时，我们叫它为 DAE（微分代数方程组）。DAE 问题只有在同时提供状态变量初值 x0 和状态变量一阶导数初值 py0，且满足 M(t0,y0)*py0=f(t0,y0)时才有唯一解，假如不满足上面的方程，DAE 解算器会将提供的 y0 和 py0 作为猜测初始值，并重新计算与提供初值最近的封闭初值。

（3）质量矩阵可是一个常数矩阵（稀疏矩阵），也可以是一个自定义函数的输出。但是 ode23s 只能求解 Mass 是常数的 DAE。

（4）对于 Mass 奇异的 DAE 问题，特别是设置 MassSingular 为 yes 时，只能 ode15s 和 ode23t 解算器。

（5）对于 DAE 我们还有几个参数需要介绍。

① Mass：质量矩阵，这个是 DAE 的关键。

② MStateDependence：质量矩阵 $M(t,y)$是否是 y 的函数，可以选择 none|{weak}|strong，none 表示 M 与 y 无关，weak 和 strong 都表示与 y 相关。

③ MvPattern：注意这个必须是稀疏矩阵，$S(i,j)$=1 表示 $M(t,y)$的第 i 行中任意元素都与第 j 个状态变量 yi 有关，否则为 0。

④ MassSingular：设置 Mass 矩阵是否奇异，当设置为 yes 时，只能使用 ode15s 和 ode23t。

⑤ InitialSlope：状态变量的一阶导数初值 py0，和 y0 具有相同的 size，当使用 ode15s 和 ode23t 时，该属性默认为 0。

假设一个质量弹簧系统，为了使问题简化，可以列写成如下微分方程：

$$\begin{cases} \dfrac{\mathrm{d}^2x}{\mathrm{d}t^2} = \dfrac{F - F_2}{m} \\ F_2 = kx \end{cases} \tag{5.17}$$

参考 5.5 节，首先将式（5.17）进行降阶处理。设

$$u_1(t)=x',u_2(t)=x,u_3(t)=F_2 \tag{5.18}$$

则式（5.17）可化成

$$\begin{cases} u_1'(t)=[F-u_3(t)]/\mathrm{m} \\ u_2'(t)=u_1(t) \\ u_3'(t)=ku_2(t)-u_3(t) \end{cases} \tag{5.19}$$

则用矩阵形式表示的 DAE 为

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} u_1'(t) \\ u_2'(t) \\ u_3'(t) \end{bmatrix}=\begin{bmatrix} [F-u_3(t)]/m \\ u_1(t) \\ ku_2(t)-u_3(t) \end{bmatrix} \tag{5.20}$$

编写 MATLAB 代码如下。

```
F = 10;                                          %外力
k = 10;                                          %弹簧刚度
m = 1;                                           %重物质量
odefun=@(t,x)[(F-x(3))/m;
              x(1);
              k*x(2)-x(3)];                      %微分方程组
M=[1 0 0;0 1 0;0 0 0];                           %质量矩阵
options = odeset('mass',M);                      %对于 DAE 问题，mass 属性必须设置
x0=[0 0 0];                                      %初值
[t,x]=ode15s(odefun,[0 10],x0,options);
figure('numbertitle','off','name','DAE demo')
plot(t,x)
legend('v','x','F_f')
```

执行程序，得到输出结果如图 5-1 所示。

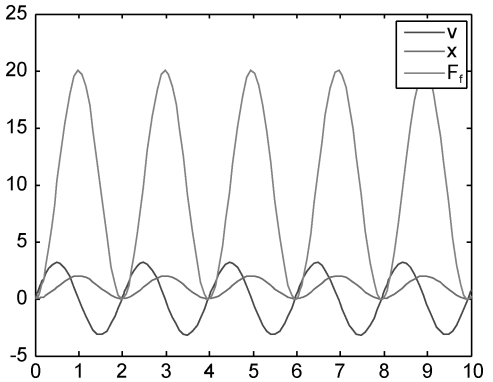


图 5-1　质量弹簧系统仿真曲线输出

其实，有些简单 DAE 是可以转换为 ODE 的，由于代数方程只是状态变量的一个约束，假如约束充分的话，我们就可以使用消参的思想，将约束反代回 ODE 中，这样就将那个约束方程消去了，最后只剩下那些微分方程了。

在本例中，式（5.17）可以修改成

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2}=\frac{F-kx}{m} \tag{5.21}$$

对上式进行降阶处理

$$u_1(t)=x',u_2(t)=x$$

则式（5.21）可化成

$$\begin{cases}u_1'(t)=[F-ku_2(t)]/m\\ u_2'(t)=u_1(t)\end{cases} \tag{5.22}$$

重新编写 MATLAB 程序如下。

```
F = 10;                                  %外力
k = 10;                                  %弹簧刚度
m = 1;                                   %重物质量
odefun=@(t,x)[(F-k*x(2))/m;
                x(1)];                   %微分方程组
x0=[0 0];                                %初值
[t,x]=ode45(odefun,[0 10],x0);
figure('numbertitle','off','name','DAE demo')
plot(t,x)
legend('v','x')
```

仿真图形如图 5-2 所示。

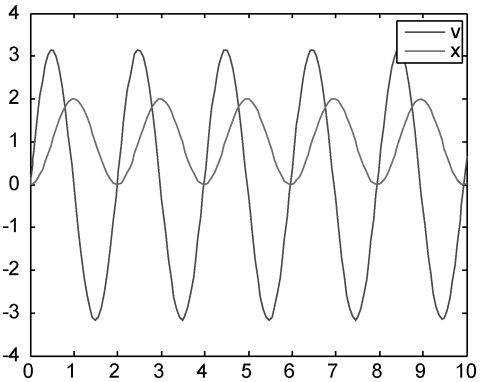


图 5-2　改进版的质量弹簧系统仿真图形

课后题

1．编写 MATLAB 程序，求解如下非线性方程组。

$$\begin{cases}\sin x+\cos x=0.5\\ xy=1\end{cases}$$

2．在 5.6 节的例子中，增加一个刚度为 100N/m 的弹簧，试采用微分方程的数值解法可以得出一个质量、弹簧和阻尼组成的系统的模型，来进行仿真。

第 6 章　Simulink 仿真基础

6.1　Simulink 简介

6.1.1　简介

Simulink 是 MATLAB 最重要的组件之一，它提供了一个动态系统建模、仿真和综合分析的集成环境。在该环境中，无须大量书写程序，只需要通过简单直观的鼠标操作，就可构造出复杂的系统。Simulink 具有适应面广、结构和流程清晰及仿真精细、贴近实际、效率高、灵活等优点，并基于以上优点 Simulink 已被广泛应用于控制理论和数字信号处理的复杂仿真和设计。同时有大量的第三方软件和硬件可应用于 Simulink。

6.1.2　功能

Simulink 是 MATLAB 中的一种可视化仿真工具，是一种基于 MATLAB 的框图设计环境，是实现动态系统建模、仿真和分析的一个软件包，被广泛应用于线性系统、非线性系统、数字控制及数字信号处理的建模和仿真中。Simulink 可以用连续采样时间、离散采样时间或两种混合的采样时间进行建模。它也支持多速率系统，也就是系统中的不同部分具有不同的采样速率。为了创建动态系统模型，Simulink 提供了一个建立模型方块图的图形用户接口（GUI），这个创建过程只需单击和拖动鼠标操作就能完成，它提供了一种快捷、直接明了的使用方式，而且用户可以立即看到系统的仿真结果。

Simulink 是用于动态系统和嵌入式系统的多领域仿真和基于模型的设计工具。对各种时变系统，包括通信、控制、信号处理、视频处理和图像处理系统，Simulink 提供了交互式图形化环境和可定制模块库来对其进行设计、仿真、执行和测试。

构架在 Simulink 基础之上的其他产品扩展了 Simulink 多领域建模功能，也提供了用于设计、执行、验证和确认任务的相应工具。Simulink 与 MATLAB 紧密集成，可以直接访问 MATLAB 大量的工具来进行算法研发、仿真的分析和可视化、批处理脚本的创建、建模环境的定制以及信号参数和测试数据的定义。

6.1.3　特点

Simulink 具有以下特点。

（1）有丰富的可扩充的预定义模块库。

（2）用交互式的图形编辑器来组合和管理直观的模块图。

（3）以设计功能的层次性来分割模型，实现对复杂设计的管理。

（4）通过 Model Explorer 导航、创建、配置、搜索模型中的任意信号、参数、属性，生成模型代码。

（5）提供 API 用于与其他仿真程序的连接或与手写代码集成。

（6）使用 Embedded MATLAB™ 模块在 Simulink 和嵌入式系统执行中调用 MATLAB 算法。

（7）使用定步长或变步长运行仿真，根据仿真模式（Normal、Accelerator、Rapid Accelerator）来决定以解释性的方式运行或以编译 C 语言代码的形式来运行模型。

（8）使用图形化的调试器和剖析器来检查仿真结果，诊断设计的性能和异常行为。

（9）可访问 MATLAB 从而对结果进行分析与可视化，定制建模环境，定义信号参数和测试数据。

（10）模型分析和诊断工具来保证模型的一致性，确定模型中的错误。

6.1.4 启动

1．在 MATLAB 命令窗口中输入 Simulink

结果是在桌面上出现一个称为 Simulink Library Browser 的窗口，在这个窗口中列出了按功能分类的各种模块的名称。

当然用户也可以通过 MATLAB 主窗口的快捷按钮来打开 Simulink Library Browser 窗口。

2．在 MATLAB 命令窗口中输入 Simulink3

结果是在桌面上出现一个用图标形式显示的 Library :Simulink3 的 Simulink 模块库窗口。

两种模块库窗口界面只是不同的显示形式，用户可以根据各人喜好进行选用，一般来说第二种窗口直观、形象，易于初学者使用，但使用时会打开太多的子窗口。

6.1.5 Simulink 建模仿真

一个典型的 Simulink 模型由以下三种类型的模块构成。

（1）信号源模块。信号源为系统的输入，它包括常数信号源、函数信号发生器（如正弦波和阶跃函数等）以及用户自己在 MATLAB 中创建的自定义信号。

（2）被模拟的系统模块。系统模块作为仿真的中心模块，它是 Simulink 仿真建模所要解决的主要部分。

（3）输出显示模块。系统的输出由显示模块接收。输出显示的形式包括图形显示、示波器显示和输出到文件或 MATLAB 工作空间三种，输出模块主要在 Sinks 库中。

6.1.6 Simulink 建模仿真的基本过程

启动 Simulink 后，便可在 Simulink 中进行建模仿真了。Simulink 建模仿真的基本过程如下。

（1）打开一个空白的 Simulink 模型窗口。

（2）进入 Simulink 模块库浏览界面，将相应模块库中所需的模块拖拉到编辑窗口里。

（3）按照给定的框图修改编辑窗口中模块的参数。要对模块进行参数设置时，可双击响应的模块，这时将打开此模块的参数设置对话框，进行参数设置。

（4）将各模块按给定的框图连接起来，搭建所需要的系统模型。

（5）用菜单选择命令或在命令窗口输入命令进行仿真分析，在仿真的同时，可以观察仿

真结果，如果发现有不正确的地方，可以停止仿真，对参数进行修改。

（6）如果对结果满意，可以将模型保存。

6.2　Simulink 的建模方法

通过本章的学习，读者应该能够利用 Simulink 建立一个简单的模型。

6.2.1　打开模型

Simulink 中主要有以下两种方法来打开模型。

（1）在 MATLAB 中选择 File|Open 命令，然后按照 Windows 系统的常规操作进行。

（2）直接在 MATLAB 窗口中输入模型名，然后 MATLAB 就会自动在工作目录中搜索。例如，在当前工作目录中存在模型文件为 vdp.mdl，只需在命令窗口中输入模型名：

```
>>vdp
```

6.2.2　模块操作

1．调整模块大小

当选中模块后，在模块的四周会出现控制点，将鼠标移动到控制点附近，按住左键，拖动鼠标，就可以调整模块的大小。

2．模块旋转

主要的旋转模块的方法是：单击选中要旋转的模块，选择右键菜单，在菜单中选择 Format|Rotate block 命令。其中，Rotate 是顺时针旋转 90°，Flip block 是旋转 80°。

3．模块复制和删除

单击要复制的模型，按 Ctrl+C 组合键复制，再按 Ctrl+V 组合键粘贴。

删除方法是：选中要删除的模型，按下 Delete 键。

4．模块的其他操作

选择多个模块（按 Shift 键）；修改标签（单击标签项）。

6.2.3　模块的连线操作

模型中的模块必须用连线联系起来才能够变成一个有机整体。

绘制连线的操作步骤如下。

（1）新建模型窗口。向窗口中添加相应的模块。

（2）将鼠标指针移动到模块的输出端，鼠标指针呈十字形，然后按住鼠标左键，拖动到所要连接的模块的输入端，松开鼠标左键，如果连线变成黑色，则说明连接成功；如果连线为红色，则可以再调整连线的终点位置，直到连接成功。

6.2.4 Simulink 模型的基本结构

一个典型的 Simulink 模型包括如下三种类型的元素（见图 6-1）。

（1）信号源模块。

（2）被模拟的系统模块。

（3）输出显示模块。

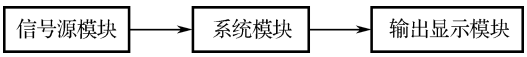


图 6-1 Simulink 模型元素关联图

信号源为系统的输入，包括常数信号源、函数信号发生器（如正弦波和阶跃函数波等）和用户自己在 MATLAB 中创建的自定义信号。

系统模块作为中心模块是 Simulink 仿真建模所要解决的主要部分。

系统的输出由显示模块接收。输出显示的形式包括图形显示、示波器显示和输出到文件或 MATLAB 工作空间中三种。输出模块主要在 Sinks 库中。

6.3 Simulink 运行仿真

运行 Simulink 模型的仿真模式主要有两种：一种是窗口模式；另一种是命令模式。本节将主要讲解窗口模式。

利用 Simulink 窗口进行仿真，主要有以下几个操作。

（1）设置仿真参数。

（2）运行仿真。

（3）终止仿真。

（4）暂停仿真。

（5）仿真诊断。

1．仿真参数的设置

初学者暂时可以忽略该项，随着学习的深入，再回头进行研究。

2．运行仿真

设置好 Simulink 模型运行环境之后，就可以进行仿真了。选择 Simulation|Start 命令进行仿真，或者使用 Ctrl+T 组合键，或单击工具栏上的向右的黑色箭头。

3．终止仿真

选择 Simulation|Stop 命令，或使用 Ctrl+T 组合键，或单击工具栏上的黑色方框按钮。

4．暂停仿真

选择 Simulation|Pause 命令。

5．其他

另外，还可以通过工具栏上的文本框输入终止仿真时间。

6.4　Simulink 模块库

单击工具栏上的按钮，就可以启动 Simulink。

我们先来介绍一下 Simulink 的模块库。

熟悉 Simulink 模块库所包含的内容是建立模型的基础，只有熟练地掌握了模块库才能让用户快速地建立模型，或者以最少的模块来建立模型。

6.4.1　模块库简介

启动 Simulink 后，将打开 Simulink 模型库浏览器窗口，如图 6-2 所示。

图 6-2　Simulink 模型库浏览器窗口

单击工具栏上的新建按钮，可以创建一个新的 Simulink 模型文件，这和一般的 Windows 软件操作方法类似。

6.4.2　常用模块组

Simulink 的常用模块组如图 6-3 所示。该分类集成了常用的模块，在其他组中也都包括。

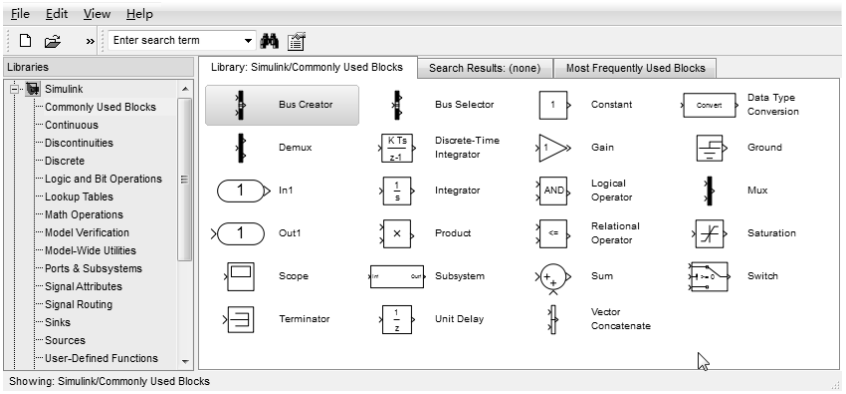


图 6-3 Simulink 的常用模块组

6.4.3 连续模块组

常用的连续模块组包括积分模块、微分模块、状态空间模块、传递函数模块、零极点模块等。

6.4.4 离散模块组

包括常用的离散模块。

6.4.5 非连续模块组

包括常用的非连续模块。

6.4.6 逻辑运算模块组

包括常用的逻辑运算模块。

6.4.7 函数与表格模块组

包括一维查表模块、二维查表模块等。

6.4.8 数学运算模块组

常用的包括求和模块、增益模块、点乘模块、数学函数模块等。

6.4.9 端口与子系统模块

包括空白子系统模块等。

6.4.10 信号通道模块组

包括常用的混路器（Mux）和分路器（Demux）等。

6.4.11 信号接受模块组

包括输出到工作空间模块（Out1）、终结模块（Terminator）、示波器模块（Scope）、显示

数据模块（Display）等。

6.4.12　信号源模块组

包括输入端口模块（In1）、常数模块（Constant）、信号发生器模块（Signal Generator）、信号构造模块（Signal Builder）、斜坡信号模块（Ramp）、正弦信号模块（Sine Wave）、阶跃信号模块（Step）、随机信号模块（Random Number）、时钟模块（Digital Clock）等。

6.4.13　用户自定义模块组

其中比较重要的是 MATLAB 函数模块（MATLAB Fcn）

6.5　子系统及其封装技术

本节要求读者掌握 Simulink 子系统的建模方法和子系统的封装技术。

前面的章节介绍了使用 Simulink 进行建模的基本方法，使用这些方法基本可以创建任何物理系统的模型。然而随着系统越来越复杂，用这些基本操作创建的 Simulink 模型变得越来越庞大而难以读懂。本节将介绍一系列 Simulink 的特殊处理技术，使得模型变得更加简洁、易懂、易用。

绝大多数程序设计语言都有使用子程序的功能，如 MATLAB 的子程序——M 文件。随着模型变得越来越大、越来越复杂，人们很难轻易读懂它们。在这种情况下，子系统通过把大的模型分割成几个小的模型以使得整个模型更加简洁、可读性更高，MATLAB 中引进了子系统技术。

6.5.1　创建 Simulink 子系统的两种方法

（1）对已经存在的模型的某些部分或全部使用菜单命令【Edit→Create Subsystem】将执行压缩转换，使之成为子系统。

（2）使用“Subsystems”模块库中的“Subsystems”模块直接创建子系统。

6.5.2　Simulink 子系统的两种作用

（1）系统模型更加简洁和可读性更高。

（2）子系统可以反复调用，节省建模时间。

6.5.3　例子

下面以一个例子说明压缩子系统的使用方法。有如下系统

$$\ddot{x}+0.4\dot{x}+0.8x=\sin t$$

方程可以转化为

$$\ddot{x}=-0.4\dot{x}-0.8x+\sin t$$

根据方程，创建如图 6-4 所示的 Simulink 仿真模型。

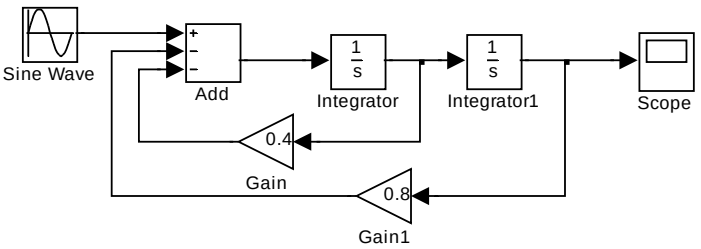


图 6-4　子系统封装前的仿真模型

按下述步骤进行子系统封装。

（1）使用范围框将要压缩成子系统的部分选中，包括模型框和信号线（注意：只能使用范围框，而不能使用 Shift 键逐个选定）。

（2）在模块窗口选项中选择【Edit→Create Subsystem】，Simulink 将用一个子系统模块代替被选中的模块组（或者使用右键菜单，从中选择 Create Subsystem 命令，如图 6-5 所示）。

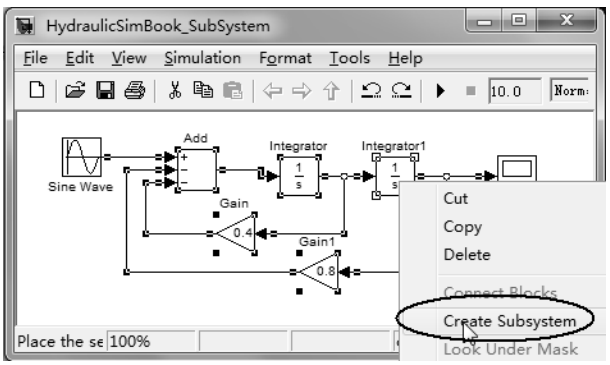


图 6-5　创建子系统

（3）进行模型美观调整，最终效果如图 6-6 所示。

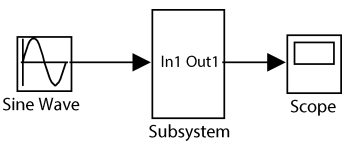


图 6-6　修饰之后的系统

子系统创建后，双击该子系统，会出现一个显示子系统内容的新窗口，如图 6-7 所示。在新窗口中，除了原始的模块外，Simulink 自动添加了输入模块和输出模块，分别代表子系统的输入端口和输出端口。

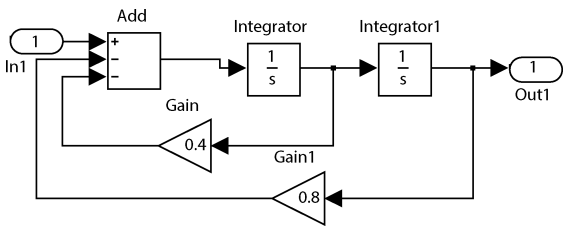


图 6-7　子系统内容

两点说明如下。

（1）子系统窗口无须保存，只需保存主程序窗口即可。保存主程序窗口后，子系统窗口将自动保存。

（2）菜单命令【Edit→Create Subsystem】没有相反的操作命令，也就是说，一旦一组模块压缩成了子系统，就没有可以直接还原的处理方法（undo 除外）。因此，一个理想的处理方法是在压缩子系统之前，先将模型进行保存，作为备份。

6.6　Simulink 模型仿真

6.6.1　仿真的基本过程

基本操作过程如下。

（1）根据系统具体情况，建立数学仿真模型。

（2）打开一个空白的模型编辑窗口。

（3）拖放模块建立模型。

（4）设置模块参数。

（5）对模块进行连线。

（6）设置仿真模型的系统参数。

（7）运行仿真。

（8）查看仿真结果。

（9）保存文件退出。

6.6.2　对单自由度系统进行仿真

本节将通过实例介绍如何建立一个动力学系统，在此过程中，读者会看到如何通过命令流和鼠标来一步一步地建造模型。

1．建立系统的数学模型

建立动力学方程如下。

$$m\frac{\mathrm{d}^2y}{\mathrm{d}t^2}+c\frac{\mathrm{d}y}{\mathrm{d}t}+ky=F \tag{6.1}$$

式中　m ——值为 1；

c ——值为 4；

k ——刚度，k=3；

F ——外激励，$F=2\sin(2t+\pi/3)$ 。

带入参数，整理成

$$\frac{\mathrm{d}^2y}{\mathrm{d}t^2}=2\sin(2t+\pi/3)-4\frac{\mathrm{d}y}{\mathrm{d}t}-3y \tag{6.2}$$

2．创建文件

打开一个空白模型编辑窗口，保存文件。

3．创建相应的模块

通过拖拉的方式，在模型编辑窗口中建立图 6-8 所示模块。

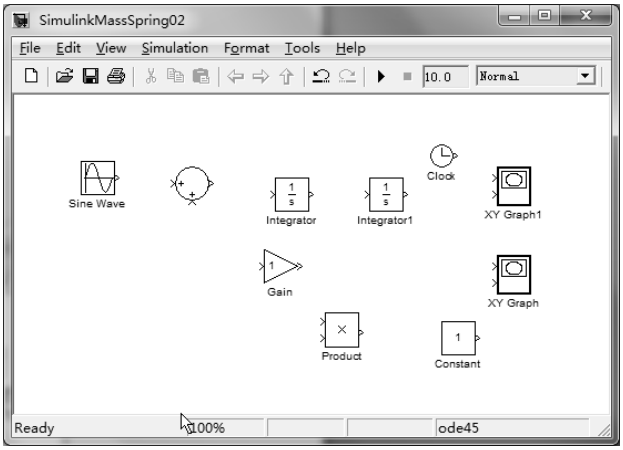


图 6-8　模型草图

4．对模块进行参数设置

设置 Sine Wave 模块。双击模块弹出参数设置对话框，如图 6-9 所示。参数 Amplitude 为 2，参数 Bias 为 0，参数 Frequency 为 2，参数 Phase 为π/3，然后单击 OK 按钮。

对 Sum 模块进行设置，双击模块弹出设置参数的对话框，如图 6-10 所示。其中参数 Icon shape 选择 rectangle 选项，List of signs 为 “+-”，然后单击 OK 按钮。

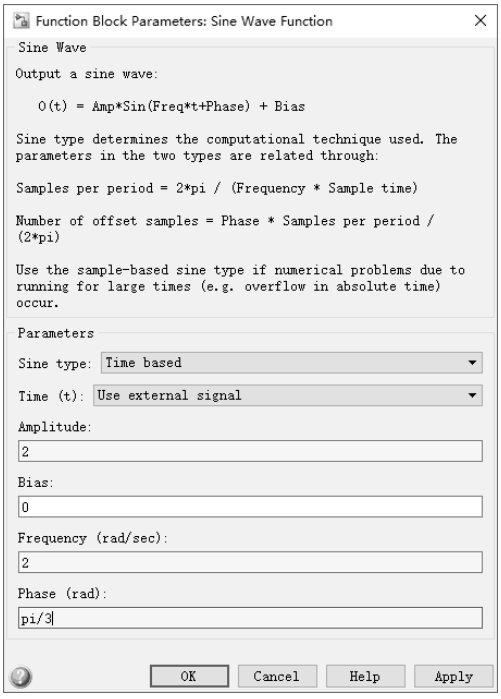


图 6-9　Sine Wave 参数设置对话框

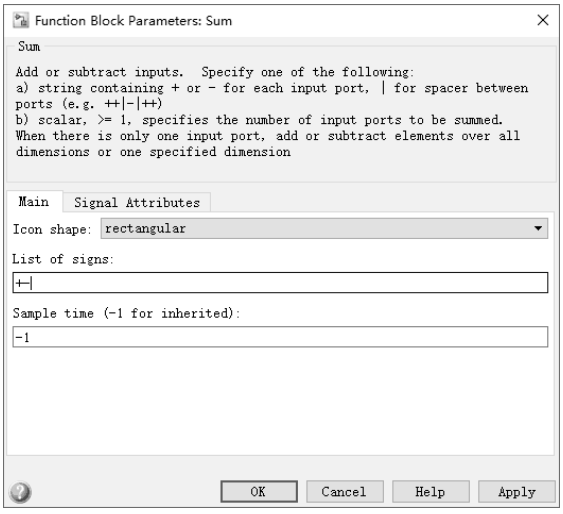


图 6-10　Sum 参数设置对话框

设置 Constant 模块，双击模块弹出参数对话框，设置 Constant value 参数为 3，单击 OK 按钮。右击模块，在弹出的右键菜单中单击 Format|Flip Block 命令。

设置 Gain 模块，双击模块弹出设置参数的对话框，设置 Gain 为 4，单击 OK 按钮。右击模块，在弹出的右键菜单中选择 Format|Flip Block 命令。

设置 Product 模块，右击模块，在弹出的右键菜单中选择 Format|Flip Block 命令。

设置 Integrator 和 Integrator1 模块，双击 Integrator 模块，弹出参数设置对话框。在 Integrator 模块的参数设置对话框中，设置 Initial condition 参数为 0.5，即初始速度为 0.5，单击 OK 按钮。对 Integrator1 模块进行相同操作，即初始位移为 0.5。

设置 XY Graph 模块，修改其中范围参数：x-min 为−0.7，x-max 为 0.7，y-min 为−0.5，y-max 为 0.5。

设置 XY Graph1 模块，修改其中范围参数：x-min 为 0，x-max 为 10，y-min 为−2，y-max 为 2。

5．连接模块

连接模块如图 6-11 所示。

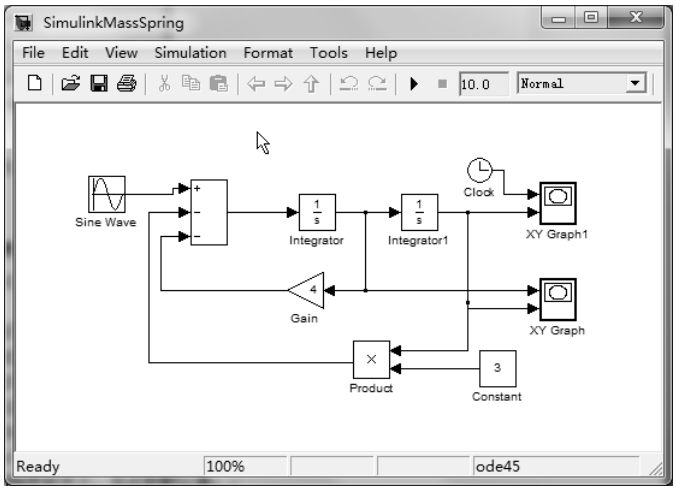


图 6-11　连接后的模型

6．对模型系统参数进行设置

选择 Simulation|Configuration Parameters 命令，弹出 Configuration Parameters 对话框。

Start time 为仿真开始时间，在此取默认值 0。

Stop time 为仿真结束时间，在此修改为 20s。

Type 接受默认选项。

Solver 为计算方法，此处选默认的 ode45 选项。

Max step size 和 Min step size 分别设定为 0.05 和 0.01。

Relative tolerance 和 Absolute tolerance 分别取默认值。

Initial step size 为初始步长大小，在此取默认值。

Zero crossing control 取默认值。

7．运行仿真

按下 Ctrl+T 组合键，或单击工具栏上的向右按钮，运行仿真。

8．查看运行结果

XY Graph 会自动弹出运行结果，横坐标是位移，纵坐标是速度，如图 6-12 所示。XY Graph1 也会自动显示时间和位移的曲线，如图 6-13 所示。

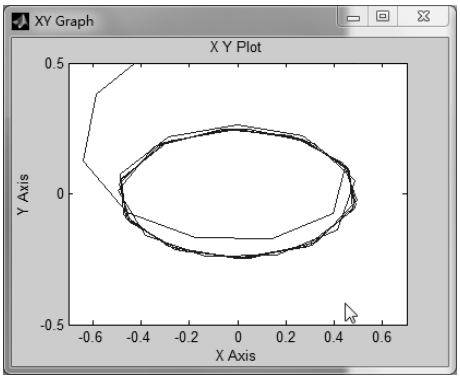


图 6-12　速度和位移之间的关系曲线

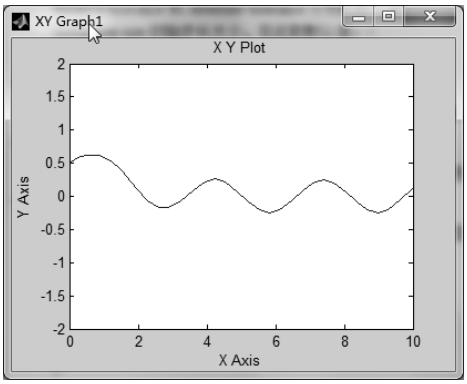


图 6-13　时间和位移之间的关系曲线

9．保存文件

至此我们已经完成了一个简单模型的建立、模块的设置、模型设置和仿真以及最后的结果显示，其实复杂的模型也都具有类似的过程。

6.7　仿真过程中代数环的消除方法

控制系统的仿真需要计算机按照一定的时序执行相应的计算步骤，再将最终结果以图形或数组等方式提供给用户。对于存在反馈回路的控制系统，输入和输出存在计算时序，当输入直接取决于输出，同时输出也直接取决于输入时，仿真模型中会出现代数环。

代数环（Algebraic Loop）的出现会严重降低系统的仿真速度，甚至降低仿真精度或得到错误的仿真结果。目前，Simulink 对于代数环问题没有提出令人满意的解决方案。为了保证系统仿真的速度和精度，有必要掌握代数环的消除方法。

目前代数环的消除方法有以下两种方案：①利用变换法消除代数环，即对系统中有代数环的环节进行变换，从而消除输入和输出的“直接”联系；②利用拆解法消除代数环，即采用“非直通、同功能”模块代替系统中的“直通”模块（直接传输环节），从而切断仿真系统中的代数环。第一种方案，只适用于简单的控制系统，当系统方程非常复杂或者输入、输出只能以隐函数的方式表达出来时，系统方程的变换无法实现；第二种方案，采用在反馈通道加入延迟环节的传统方法，可以消除大多数场合下系统仿真中的代数环。

6.7.1　代数环产生的原因

在数字计算机仿真中，当输入信号直接取决于输出信号，同时输出信号也直接取决于输入信号时，由于数字计算的时序性，而出现由于没有输入而无法计算输出，没有输出也无法得到输入的“死锁环”，称之为“代数环”。图 6-14 所示为一个简单的代数环。

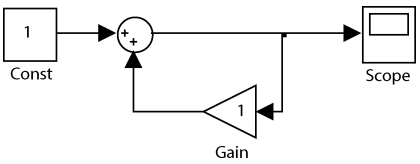


图 6-14　简单代数环

如果此时运行该仿真回路，Simulink 会弹出如图 6-15 所示对话框，在其说明中指明该回路包含一个 Simulink 无法求解的代数环（Simulink cannot solve the algebraic loop）。

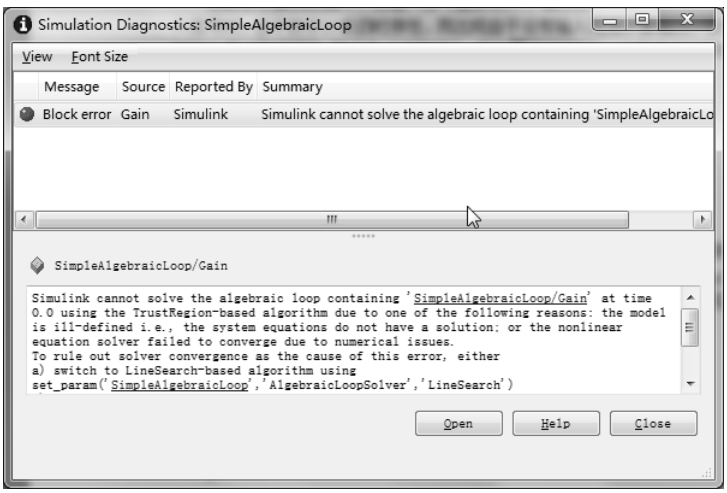


图 6-15　无法求解代数环警告

事实上，代数环问题本质上是方程（组）求解的问题。例如，如图 6-16 所示的代数环。在仿真模型窗口中单击菜单【Simulation→Configuration parameters...】。

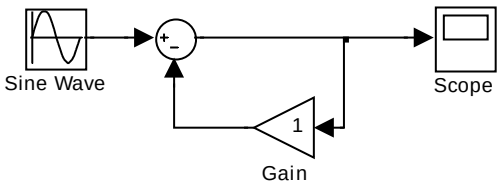


图 6-16　可以化简的代数环问题

在弹出的“Configuration Parameters”对话框中，选择左侧的“Diagnostics”树形控件，在求解器（Solver）的“Algebraic loop”中选择“error”，然后运行仿真。此时，Simulink 会弹出对话框，告知用户该仿真回路中包含一个代数环，并将代数环部分用红色标识出来，以提醒用户，如图 6-17 所示。

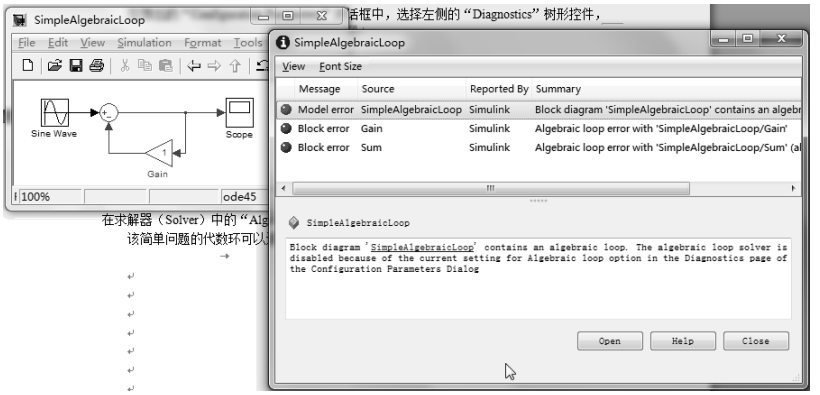


图 6-17　代数环错误警告

事实上，Simulink 是可以求解一部分代数环的，如果在上面的例子中，我们没有在“Algebraic loop”中选择“error”，而是选择“warning”或“none”，Simulink 就能够求解出该代数环的运行结果。事实上，这个简单问题的代数环可以通过数学变换的方法消除。由图 6-16 可得

$$u - y = y \tag{6.3}$$

即代数环是一种输入和输出有关的方程，这就是导致死锁的原因。要解决死锁，只要把式（6.3）右边的输出移到左边，得出输出只和输入有关的方程即可。所以，解决代数环其实就是解方程问题。对式（6.3）做变换得

$$y = \frac{1}{2}u \tag{6.4}$$

这时，输出只和输入有关，消除了代数环。根据式（6.4）可得到相应的 Simulink 模型如图 6-18 所示。可以看出，回路中已经没有了反馈，自然就不会产生代数环问题。

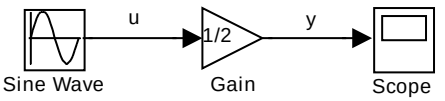


图 6-18　变换后的模型

6.7.2　产生代数环的条件

代数环是一种特殊的反馈回路，它的特殊之处就在于除了输入直接决定于输出外，输出还直接决定于输入。在 Simulink 中，这种“直接”是由直通模块导致的。在一个闭合路径中，只有各模块都是直通（无延时）模块时才会形成代数环，这就是代数环产生的充分必要条件。可知，并非所有的反馈回路都是代数环。

Simulink 的模块库中提供的很多模块都是直通模块，如数学函数模块、加减法模块、乘积运算模块、积分器模块等。

Simulink 仿真模型中，产生代数环的条件如下。

（1）前馈通道中含有信号的“直通”模块，如比例环节或含有初始值输出的积分器。

（2）系统中大部分模型表现为非线性。

（3）前馈通道的传递函数的分子分母同阶。

（4）用状态空间描述时，输出方程中输出矩阵 D=0。

6.7.3　代数环的消除

1．Simulink 对代数环的解决

在 Simulink 中，参数对话框中的 Diagnositic 选项可以设置代数环显示出警告信息，以便能让系统诊断所设计的模型中是否有代数环。

为了解决存在代数环情况下的仿真问题，Simulink 提供了计算代数环的方法，即方程求根的牛顿法。牛顿法也称牛顿-拉弗逊法或切线法，是一种基于一阶泰勒级数展开的逐次迭代逼近法。随着代数环变长，其中的模块功能越复杂，精度要求越高，则迭代计算量就越大，仿真速度就越慢。另外，当不满足收敛条件时，Simulink 对代数环就会得出错误的解。

2．利用变换法消除代数环

对一些简单的代数环，可采用数学变换法求解。用变换法消除代数环的方法在前文已讲过，这里不再赘述。要说明的是，当模型变得很复杂时，这种方法几乎是不能实现的。

3．在直通模块前插入存储器模块

这种方法是在直通模块前插入存储器模块切断代数环，如图 6-19 所示。其中存储器模块在 Simulink 的“Discrete”库中。

这种方法是在反馈通道上增加了存储器环节，使得输入只与上一步的输出有直接联系，这样就消除了代数环。由于该存储环节在系统中相当于一个纯延时环节，会影响系统的仿真精度，特别是当系统稳定裕度不大时，由于延时环节传递函数众多极点的影响，可能会导致系统不稳定。

4．在反馈通道中加入高频传递环节

在反馈通道上增加一个延迟环节，可以切断输入、输出的直接联系，如图 6-20 所示。当该环节截止频率足够高时，它对系统精度几乎没有影响，然而此环节增加的计算量会影响系统的仿真速度。

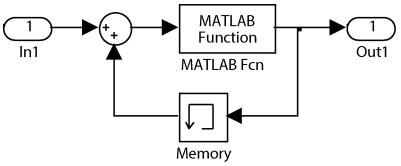


图 6-19　插入存储器模块

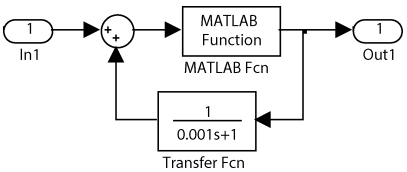


图 6-20　加入高频传递环节

除以上几种方法外，Simulink 还提供了一些专门手段来拆解代数环，如代数约束模块（algebraic constraint）、积分模块（以及离散积分模块）的状态输出端等。这些手段可以解决一些特定的代数环问题。例如，Simulink 专门为积分模块设计了一个状态端口（state port），其输入与输出端口完全相同，仅在内部计算的时序上有细微区别，而无论是从积分模块的初始值输入端口还是从复位输入端口到状态端口都是非直通的。因此，当有代数环问题时，可以从积分模块的状态端口引出信号，这样代数环就会被消除。

课后题

1．创建一算术运算的仿真文件，并分别对加、减、乘、除运算建立用户自定义模块。

2．对正弦信号 $y=\sin x$ 和斜坡信号 $y=2x$ 进行仿真，并将其图形显示在同一窗口中。

3．建立传递函数 $G(s)=2(s+3)/(s^3+4s^2+3s+7)$的仿真模型，并调试运行。

4．已知某单位负反馈系统的开环传递函数为

$$G(s)=\frac{3}{s^2+3s+5}\times\frac{7}{s+3}+\frac{2}{3s+1}$$

试建立其闭环传递函数的仿真模型并进行仿真。

第 7 章　液压 PLC 技术基础

绝大部分流体传动与控制系统都是用 PLC 实现自动控制的，因此，对流体传动控制系统的仿真，离不开对 PLC 的仿真，本章将介绍在 MATLAB 环境下如何进行 PLC 控制的仿真。

7.1　定义

国际电工委员会（IEC）于 1987 年颁布了可编程序控制器标准草案第三稿。在草案中对可编程序控制器定义如下：“可编程序控制器是一种数字运算操作的电子系统，专为工业环境应用而设计。它采用可编程序的存储器，用来在其内部存储执行逻辑运算、顺序控制、定时、计数和算术运算等操作的指令，并通过数字式和模拟式的输入和输出，控制各种类型的机械或生产过程。可编程序控制器及其有关外围设备，都应按易于与工业系统联成一个整体，易于扩充其功能的原则设计”。

7.2　适用领域

根据工厂自动化和 FMS（柔性制造系统）的发展，PLC 从中小型控制板向多功能、高速度的大型系统设备发展。

PLC 的适用领域见表 7-1。

表 7-1　PLC 的适用领域

领　域	控 制 对 象
食品工厂	综合传输控制、流水线自动控制
钢铁制造工厂	货物控制、原料运输控制
纺织、化学工厂	原料输送控制、织物染色生产线控制
汽车厂	传输生产线控制、自动装配生产线控制、喷漆生产线控制
机械制造厂	工业机器人的控制、机床刀具控制、抽水泵控制
水利、排水系统	水的净化控制、污水排放控制、传输线的控制
后勤	自动入库控制、传输线的控制
厂房设备	压缩机的控制
环境	垃圾焚化自动控制、防污染设备的控制

7.3　组成

PLC 的基本组成元素包括：输入模块（input modules）或称输入端口（input points），中央处理单元（Central Processing Unit，CPU），输出模块（output modules）或称输出端口（output

points）及编程设备。输入模块或端口的类型取决于所使用的输入设备的类型。一些输入端口或模块属于数字输入，也称离散输入，其值或者为 On 或者为 Off。另一些模块或端口属于模拟信号。这些模拟信号代表在某种设备或工艺过程在一定范围内变化的电压或电流值。PLC 输入电路的功能是将多种开关或传感器的输入信号转换成 CPU 能够处理的逻辑信号。

CPU 通过执行内部存储的程序，计算输入信号、输出信号和其他变量的状态，然后发出信号更新输出变量的状态。

输出模块将 CPU 的控制信号转换成数字量或模拟量，以便控制多种输出设备。

输入设备用来输入或改变 PLC 的程序或改变存储的值。一旦将程序输入，程序本身及其相关联的变量就存储在 CPU 中。

除了这些基本的元素，PLC 系统还包括与其相配合的操作人机接口设备，以简化对机械或工艺过程的监控。

在下面的例子中，按钮与 PLC 的输入相连，用于启动或停止通过电动机启动器与 PLC 输出相连的电动机。在本例中，没有编程设备和人机接口。

本章介绍西门子 PLC 的基本概念和编程方法。西门子 PLC 包括在 SIMATIC S7 产品族中，它们是 S7-200、S7-300 和 S7-400。其中，S7-200 就是所谓的微型 PLC，因为其尺寸小巧。S7-200 采用集成设计方法，这意味着该 PLC 的 CPU、电源、输入/输出接口（I/O）都包括在一个集成的结构中。该 PLC 可以单独使用，也可以与其他 PLC 相连接构成更加复杂的系统。

西门子 S7-200 PLC 的 CPU 有五种类型：CPU 221、CPU 222、CPU224、CPU 224XP 和 CPU 226，每种类型有两种电源。其类型表见表 7-2。

表 7-2 西门子 S7-200 PLC 的 CPU 类型

模 块 描 述	电 源	输 入 类 型	输 出 类 型
221 DC/DC/DC	DC 20.4～28.8V	6DC	4DC
221 AC/DC/Relay	AC 85～264V，47～63Hz	6DC	4Relay
222 DC/DC/DC	DC 20.4～28.8V	8DC	6DC
222 AC/DC/Relay	AC 85～264V，47～64Hz	8DC	6Relay
224 DC/DC/DC	DC 20.4～28.8V	14DC	10DC
224 AC/DC/Relay	AC 85～264V，47～64Hz	14DC	10Relay
224XP DC/DC/DC	DC 20.4～28.8V	14DC，2 Analog	10DC，1Analog
224XP AC/DC/Relay	AC 85～25.4V，47～64Hz	14DC，2 Analog	10Relay，1Analog
226 DC/DC/DC	DC 20.4～28.8V	24DC	16DC
226 AC/DC/Relay	AC 85～25.4V，47～64Hz	24DC	16Relay

表 7-2 描述了 PLC 的 CPU 类型、电源、输入和输出的类型，如 CPU222 DC/DC/DC 中的 3 个 DC 按顺序分别表示电源类型为直流（DC）、输入端口类型为直流（DC）和输出端口类型为直流（DC）。AC 表示交流，Relay 表示继电器类型。

7.3.1 硬件

PLC 由像人类大脑的中央处理器、存储器、与外设相连的输入/输出设备、电源以及输入

程序的外设组成，如图 7-1 所示。

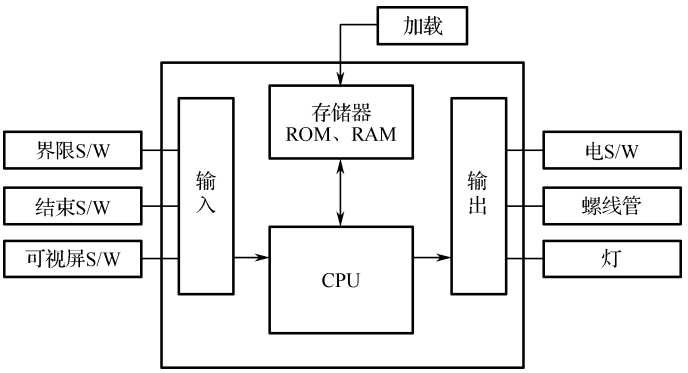


图 7-1　PLC 的硬件结构

1．PLC 的 CPU

CPU 是 PLC 的“大脑”，它解释存储的程序，并按程序执行操作。在这个过程中，所有信息都用二进制数处理。

2．存储器

1）存储器的种类

（1）ROM（只读存储器）。ROM 只用于读操作，不能更改存储器内容。存储器里的信息不会丢失，即使是电源断电也不会有影响。

（2）RAM（随机存储器）。存储器里的信息可以随时读/写，主要用于暂时保存重要的内容，但是当电源断电时，存储器里的内容会消失。

2）存储器的使用

PLC 存储器分为用户程序存储区、数据存储区和系统存储区三部分。

用户程序存储区是保存程序的地方，程序是根据系统说明编写控制。程序在写入前后都可以进行修改，因此先存放在 RAM，程序完成后存入 RM。

数据存储区是存放像输入/输出传输、辅助存储器、计时器、连接条件、设定值和当前值等这种数据的地方。由于信息会不断变化，因此使用了 RAM。

系统存储区存储了系统程序，这是由 PLC 制造厂商提供的程序，也是决定 PLC 功能的重要程序。制造厂商直接把它写到了 ROM 中。

3．输入/输出

PLC 的输入输出用于直接连接外设。尽管 PLC TTL 电路内部直流电压为 DC +5V，由于输入/输出使用的是不同电压，PLC 内部接口和输入/输出间应建立一个平衡点，以保证系统稳定。

下面是 PLC 输入/输出的需求。

（1）电压标准必须与外部设备吻合。

（2）外设的噪声禁止被传输到 CPU（使用光耦合器）。

（3）与外设相连的接口必须平缓。

（4）输入/输出接触情况明显。输入（LED 显示）检测连接情况或引导面板控制外设，输出激发外设或者显示情况，见表 7-3。

表 7-3 PLC 的输入/输出接口

I/O	类 别	附 属 位 置	外 设 名 称
输入	操作输入	控制板和操作板	按钮开关 制动开关 波动开关
	传感器输入	机械设备	限位开关 光敏开关 接近开关
输出	显示警告输出	控制板和操作板	指示灯 蜂鸣
	激励输出	机械设备	电磁阀 电离合器 电闸 开关

7.3.2 软件

1．PLC 程序与继电器序列的不同

PLC 是像集成电路这样的电子元器件组合体，但它没有继电器序列中的物理触点和线圈。由于触点的连接是靠程序控制的，因此，PLC 的操作是不可见的。

继电器序列是通过线圈的激励来触发触点，而 PLC 是靠不停地读取存储器里的程序来运作。

由于 PLC 控制依赖于程序，因此程序的功能必须符合要求。

2．IEC-1131

Automation Studio 中采用了 IEC-1131 PLC 国际标准，这样有利于方便地使用系统，也方便了不同语言的 PLC 使用者之间的交流和统一。

IEC 是制定电子技术国际标准的国际性组织，始建于 1906 年，它实现了电子器材和设备领域的国际协作。

（IEC-1131）标准包括以下五部分。

（1）基本功能和术语的定义。

（2）设备必需功能和测试条件。

（3）程序语言。

（4）用户指南。

（5）网络交流。

最新介绍的 IEC 语言重要特点如下。

（1）支持各种数据类型。

（2）上游或者下游的设计和结构编程由程序集成元件导入支持，如功能、功能块和程序。

（3）用户编写的程序可以编入库，在其他的项目中再次使用。

（4）支持其他语言，用户可以选择适合自己的语言。

（5）IEC 语言由 IEC 制定标准，由两个图形语言、两个文本语言和 SFC 组成。

① 图形语言。

a．LD（梯形图）：表示继电器逻辑的图形语言。

b．FBD（功能块）：用功能块的连接表达语言。

② 文本语言。

a．IL（指令表）：汇编语言。

b．ST（结构文本）：像 Pascal 这样的高级语言。

③ SFC（顺序功能表）。

7.4　S7-200 的寻址与基本指令

S7-200 CPU 的基本功能就是监视现场的输入信号，根据用户的控制逻辑进行控制运算，输出控制信号去控制现场设备的运行。

在 S7-200 系统中，控制逻辑由用户编程实现，CPU 按照循环扫描的方式，完成包括执行用户程序在内的各项任务。

S7-200 CPU 周而复始地执行一系列任务。任务执行一个循环称为一个扫描周期。在一个扫描周期内，CPU 执行如下操作。

（1）读输入：S7-200 CPU 读取物理输入点上的状态并复制到输入过程映像寄存器中。

（2）执行用户控制逻辑：从头到尾地执行用户程序，一般情况下，用户程序从输入映像寄存器获得外部控制和状态信号，把运算结果写到输出映像寄存器中，或存入到不同的数据保存区中。

（3）写输出：复制输出过程映像寄存器中的数据状态到物理输出点。

7.4.1　S7-200 的寻址

S7-200 CPU 将信息存储在不同的存储单元，每个单元都有唯一的地址。S7-200 CPU 使用数据地址访问所有的数据，称为寻址。输入/输出点、中间运算数据等各种数据类型具有各自的地址定义方式。S7-200 的大部分指令都需要指定数据地址。

1．数据长度

S7-200 寻址时，可以使用不同的数据长度。不同的数据长度表示的数值范围不同，S7-200 指令也分别需要不同的数据长度，见表 7-4。

表 7-4　数据长度

数 据 长 度	字节/B	字/W	双字/D
无符号整数	0～255（十进制） 0～FF（十六进制）	0～65535（十进制） 0～FFFF（十六进制）	0～4294967295（十进制） 0～FFFF FFFF（十六进制）

续表

数 据 长 度	字节/B	字/W	双字/D
符号整数	−128～+127（十进制） 80～7F（十六进制）	−32768～32768（十进制） 8000～7FFF（十六进制）	−2147483648～+2147483647（十进制） 8000 0000～7FFF FFFF（十六进制）
实数（单精度） 32 位浮点数			+1.175495E-38～+3.402823E+38（正数） −1.175495E-38～−3.402823E+38（负数） （十进制）

在 S7-200 PLC 中，可以按位、字节、字和双字对存储单元寻址。

寻址时，数据地址以代表存储区类型的字母开始，随后是表示数据长度的标记，然后是存储单元编号；对于二进制位寻址，还需要在一个小数点分隔符后指定位编号。

2．位寻址举例

I3.4：其中“I”代表存储器标识符；“3”代表字节地址，字节 3（第 4 个字节）；“.”代表字节地址与位号之间的分隔符；“4”代表字节的位或位号。（注：I 表示存储器是输入过程映像区。）

VB100：其中“V”表示存储区域标识符，“B”表示访问一个字节，“100”表示起始字节地址。

VW100：“W”表示访问一个字，其余意义同前。

VD100：“D”表示访问一个双字，其余意义同前。

7.4.2 各数据存储区寻址

1．输入过程映像寄存器：I

在每次扫描周期的开始，CPU 对物理输入进行采样，并将采样值写入输入过程映像寄存器中。可以按位、字节、字或双字来存取输入过程映像寄存器中的数据。

位：I[字节地址].[位地址]，I0.1。

字节、字或双字：I[长度][起始字节地址]，IB4，IW1，ID0。

2．输出过程映像寄存器：Q

在每次扫描周期的结尾，CPU 将输出过程映像寄存器中的数值复制到物理输出点上。可以按位、字节、字或双字来存取输出过程映像寄存器中的数据。

位：Q[字节地址].[位地址]，Q1.1。

字节、字或双字：Q[长度][起始字节地址]，QB5，QW1，QD0。

3．变量存储区：V

用户可以用 V 存储器存储程序执行过程中控制逻辑操作的中间结果，也可以用它来保存与工序或任务相关的其他数据。可以按位、字节、字或双字来存取 V 存储器中的数据。

位：V[字节地址].[位地址]，V10.2。

字节、字或双字：V[长度][起始字节地址]，VB100，VW200，VD300。

4．位存储区：M

可以用位存储区作为控制继电器来存储中间操作状态和控制信息。可以按位、字节、字

或双字来存取位存储区中的数据。

位：M[字节地址].[位地址]，M26.7。

字节、字或双字：M[长度][起始字节地址]，MB0，MW13，MD20。

5．定时器存储区：T

S7-200 CPU 中，定时器可用于时间累计。定时器寻址有以下两种形式。

（1）当前值：16 位有符号整数，存储定时器所累计的时间。

（2）定时器位：按照当前值和预置值得比较结果置位或者复位。

两种寻址使用同样的格式，用定时器地址（T+定时器号，如 T33）来存取这两种形式的定时器数据。究竟使用哪种形式取决于所使用的指令。

6．计数器存储区：C

在 S7-200 CPU 中，计数器可以用于累计其输入端脉冲电平由低到高的次数，计数器有以下两种寻址方式。

（1）当前值：16 位有符号整数，存储累计值。

（2）计数器位：按照当前值和预置值得比较结果来置位或者复位。

可以用计数器地址（C+计数器号，如 C0）来存取这两种形式的计数器数据。究竟使用哪种形式取决于所使用的指令。

7．模拟量输入：AI

S7-200 将模拟量值（如温度或电压）转换成 1 个字长（16 位）的数据。可以用区域标识符（AI）、数据长度（W）及字节的起始地址来存取这些值。因为模拟值输入为 1 个字长，且从偶数位字节（如 0、2、4）开始，所以必须用偶数字节地址（如 AIW0、AIW2、AIW4）来存取这些值。模拟量输入值为只读数据。模拟量转换的实际精度是 12 位。

格式：AIW[起始字节地址]，AIW4。

8．模拟量输出：AQ

S7-200 把 1 个字长（16 位）数字值按比例转换为电流或电压。可以用区域标识符（AQ）、数据长度（W）及字节的起始地址来改变这些值。因为模拟量为一个字长，且从偶数字节（如 0、2、4）开始，所以必须用偶数字节地址（如 AQW0、AQW2、AQW4）来改变这些值。模拟量输出值为只写数据。模拟量转换的实际精度是 12 位。

格式：AQW[起始字节地址]，AQW4。

9．常数

表 7-5 显示了西门子 S7-200 PLC 常用的数据类型。

表 7-5　西门子 S7-200 PLC 常用数据类型

数　制	格　式	举　例
十进制	[十进制]	20047
十六进制	16#[十六进制值]	16#4E4F
二进制	2#[二进制数]	2#1010_0101_1010_0101

续表

数　制	格　式	举　例
ASCII	'[ASCII 码文本]'	'Text goes between single quotes'
实数	ANSI/IEEE 754-1985	+1.175495E-38（正数）−1.175495E-38（负数）

7.4.3 基本指令

1．位逻辑指令

位逻辑指令的基础是触点和线圈。触点是对二进制位的状态测试，测试的结果用于进行位逻辑运算；线圈用来改变二进制位的状态，其状态根据它前面的逻辑运算结果而定。

一个二进制位，既可以在程序中作为触点，也可以作为线圈。线圈可以作为触点在程序中被多次引用；如果同一地址的线圈在不止一个程序段中出现，则其状态以最后一次运算的结果为准。

位逻辑运算的基本关系是“与”和“或”。

1）标准触点

常开触点指令（LD、A 和 O）与常闭触点指令（LDN、AN 和 ON）从存储器或者过程映像寄存器中得到参考值。

当位等于 1 时，常开触点闭合（接通）；当位等于 0 时，常闭触点闭合（断开）。

标准触点的梯形图如图 7-2 所示。

2）立即触点

立即触点不依靠 S7-200 扫描周期进行更新，它会立即更新。常开立即触点指令（LDI、AI 和 OI）和常闭立即触点指令（LDNI、ANI 和 ONI）在指令执行时得到物理输入值，但过程映像寄存器并不刷新。

立即触点的梯形图如图 7-3 所示。

图 7-2　常开和常闭标准触点的梯形图

图 7-3　常开和常闭立即触点的梯形图

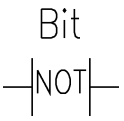


图 7-4　取反指令的触点

3）取反指令

取反指令（NOT）改变功率流输入的状态（也就是说，它将栈顶值由 0 变为 1，由 1 变为 0）。

取反指令的梯形图如图 7-4 所示。

2．传送指令

数据传送指令在不改变原值得情况下，将 IN（输入端）的值传送到 OUT（输出端）。

3．比较指令

比较指令用来比较两个数值，结果反映了比较表达式是否成立。

7.4.4　定时器

S7-200 CPU 提供了 256 个定时器。定时器分为以下三种类型。

（1）TON（接通延时定时器）：输入端通电后，定时器延时接通。

（2）TONR（有记忆接通延时定时器）：输入端通电时定时器计时，断开时计时停止；除非复位端接通，计时值累计。

（3）TOF（断开延时计时器）：输入端通电时输出端接通，输入端断开时定时器延时关断。

定时器对时间间隔计数，时间间隔又称分辨率（或时基）。S7-200 CPU 提供三种定时器分辨率，即 1ms、10ms、100ms。

选择不同的定时器号就决定了定时器的类型和分辨率。建议在一个项目中，一个定时器号只使用一次。

定时器规格见表 7-6。

表 7-6　定时器规格

定时器类型	分辨率/ms	最长定时值/s	定 时 器 号
TONR	1	32.767	T0、T64
	10	327.67	T1～T4、T65～T68
	100	3276.7	T5～T31、T69～T95
TON、TOF	1	32.767	T32、T96
	10	327.67	T33～T36、T97～T100
	100	3276.7	T37～T63、T101～T255

最长定时时间=时基×最大定时计数值。

定时器使用一个字长的有符号整数对时基计数，最大值为 32767。

定时器指令的梯形图如图 7-5 所示。图中，IN 代表输入端，PT 代表预置值，TON 代表定时器类型，T33 代表定时器号。

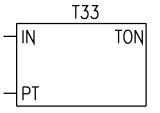


图 7-5　定时器指令的梯形图

定时器指令操作数见表 7-7。

表 7-7　定时器指令操作数

输入/输出	数 据 类 型	操　作　数
T××	WORD	常数（T0～T255），指定定时器号
IN	BOOL	I、Q、V、M、能流，启动定时器
PT	INT	IW、QW、VW、MW、常数，规定预置值

定时器工作规律见表 7-8。

表 7-8　定时器工作规律

类　　型	当前值≥预设值	使能输入（IN）的状态	上电周期/首次扫描
TON	定时器位 ON，从当前连续计数到 32767	ON：从当前值对时间间隔计数，定时器工作 OFF：定时器位关闭，当前值=0	定时器位 OFF，当前值=0

续表

类　型	当前值≥预设值	使能输入（IN）的状态	上电周期/首次扫描
TONR	定时器位 ON，从当前连续计数到 32767	ON：从当前值对时间间隔计数，定时器工作 OFF：定时器和当前值保持最后状态	定时器位 OFF，当前值可以保持
TOF	定时器位 OFF 当前值=预设值时停止计数	ON：定时器位接通，当前值=0 OFF：在接通至断开转换后定时器开始计数	定时器位 OFF 当前值=0

接通延时定时器实例梯形图如图 7-6 所示。

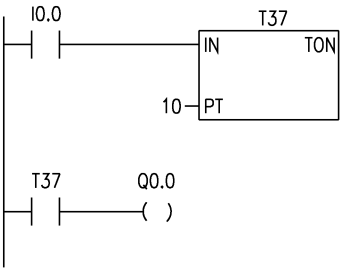


图 7-6　接通延时定时器实例梯形图

定时器 T37 时基为 100ms，预置值设定为 10，实际延时时间为 1s。这个程序的时序图如图 7-7 所示。

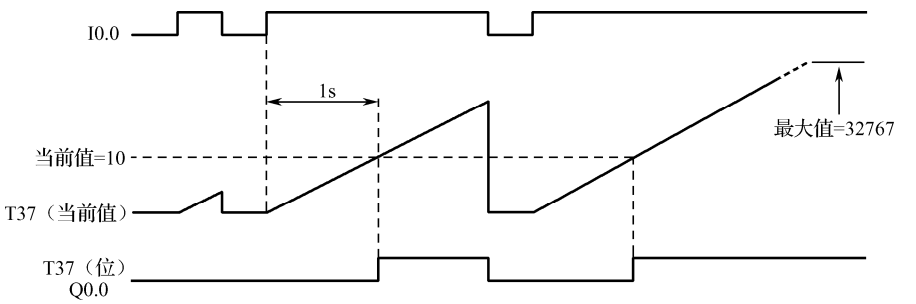


图 7-7　时序图

关断延时定时器实例梯形图如图 7-8 所示，时序图如图 7-9 所示。

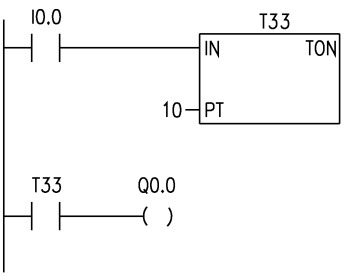


图 7-8　关断延时定时器实例梯形图

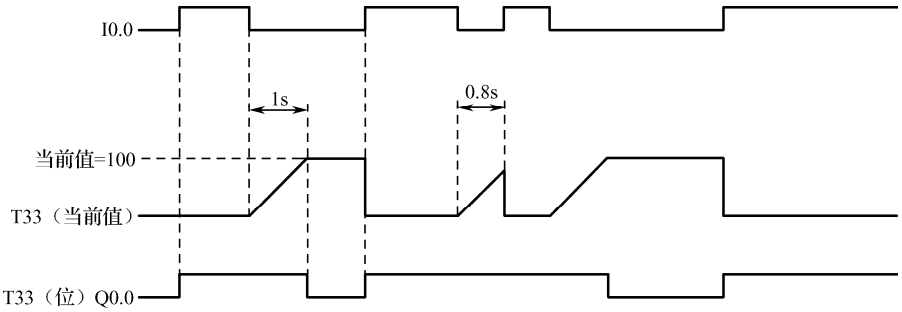


图 7-9　关断延时定时器实例时序图

7.4.5　计数器

S7-200 CPU 提供了 256 个计数器。计数器分为以下三种类型。

（1）CTU：增计数器。

（2）CTD：减计数器。

（3）CTUD：增/减计数器。

计数器指令的梯形图如图 7-10 所示。图中，CU 为增计数信号输入端，CD 为减计数信号输入端，R 为复位输入，LD 为装载预置值，PV 为预置值。

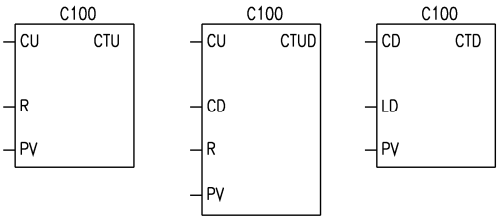


图 7-10　计数器指令的梯形图

计数器指令操作数见表 7-9。

表 7-9　计数器指令操作数

输入/输出	数 据 类 型	操　作　数
C××	WORD	常数（C0～C255），指定计数器信号
CU、CD、LD、R	BOOL	I、Q、V、M、能流
PV	INT	IW、QW、VW、MW、常数，规定预置值

计数器按表 7-10 所列的规律工作。

表 7-10　计数器指令的工作规律

类　型	操　作	计 数 器 位	上电周期/首次扫描
CTU	CU 增加当前值，当前值持续增加直至 32767	当前值≥预设值时，计数器位接通	计数器位关断
CTUD	CU 增加当前值 CD 使当前值减少	当前值≥预设值时，计数器位接通	计数器位关断
CTD	CD 使当前值减少直至当前值为 0	当前值≥预设值时，当前值=0	计数器位关断

减计数器实例梯形图如图 7-11 所示，时序图如图 7-12 所示。

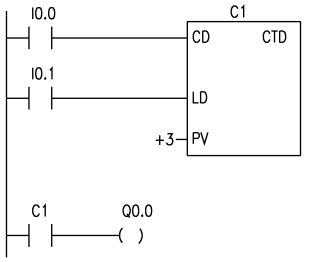


图 7-11　减计数器实例梯形图

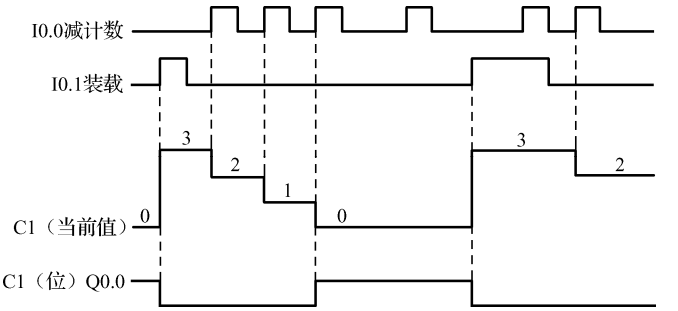


图 7-12　减计数器实例时序图

增/减计数器实例梯形图如图 7-13 所示，时序图如图 7-14 所示。

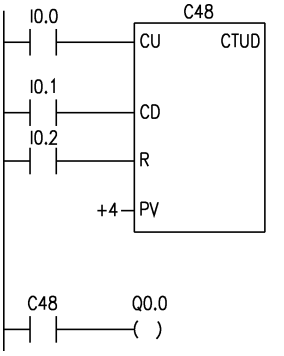


图 7-13　增/减计数器实例梯形图

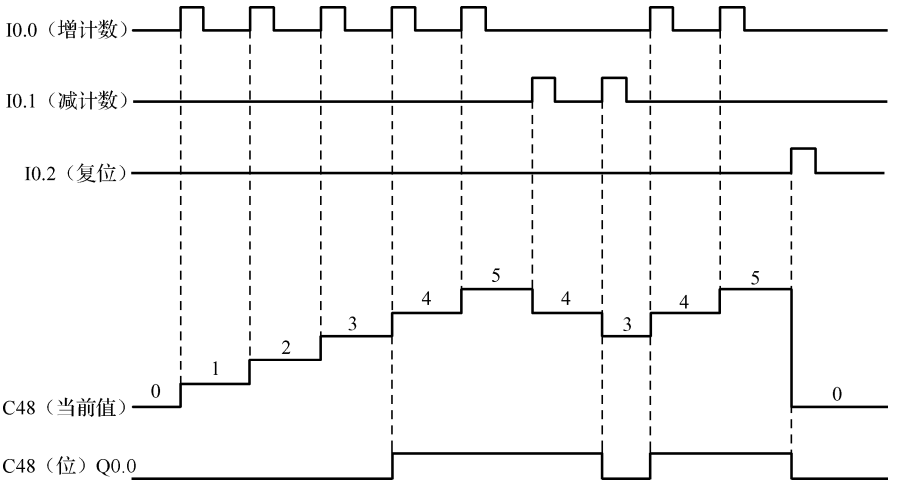


图 7-14　增/减计数器实例时序图

7.5　用 MATLAB 仿真 PLC

7.5.1　仿真方法概述

7.4 节我们介绍了西门子 S7-200 PLC 的主要指令。如果我们针对指定系统设计了 PLC 程序，要想验证程序的正确性，就必须拥有 PLC 的硬件实物，才能够运行程序，检验其正确性。但是，在实际的工程项目中，由于种种原因，往往不能将实际的程序在系统上运行，并且有时输入、输出信号也不能够获得，这时要想对 PLC 程序的正确性进行验证就比较困难。

由于上述原因，很多时候我们希望能够在编写实际 PLC 程序前，先在某仿真系统中对要编写的 PLC 程序进行仿真，检验程序的正确性，而 MATLAB 的 Simulink 环境为这一要求提供了良好的仿真平台。

为了说明用 MATLAB 仿真 PLC 的基本原理，我们用图形来进行说明。

在 MATLAB/Simulink 中工业 PLC 控制过程用一个“Industrial Process Simulation Block”（工业过程仿真模块）来模拟。该模块的输出是检测器和传感器（Sensors and Detectors），这些信号将作为“PLC 控制程序”（PLC Control Program）MATLAB/Simulink 函数模块。该模块将模拟 PLC 的操作，它的输出将对应 PLC 的输出触点，并连接到“工业过程仿真模型”中执行器的输入端。

在图 7-15 中，“PLC 控制程序”是所提供的仿真过程的关键，该模块将模拟 PLC 的循环扫描工作过程。该函数块是一个 MATLAB 的 M 文件。为了自动建立这个模块，必须完成下面的过程。

（1）首先要建立 PLC 所控制的过程的仿真环节，该环节如图 7-15 所示，是在 MATLAB/Simulink 环境下开发的。

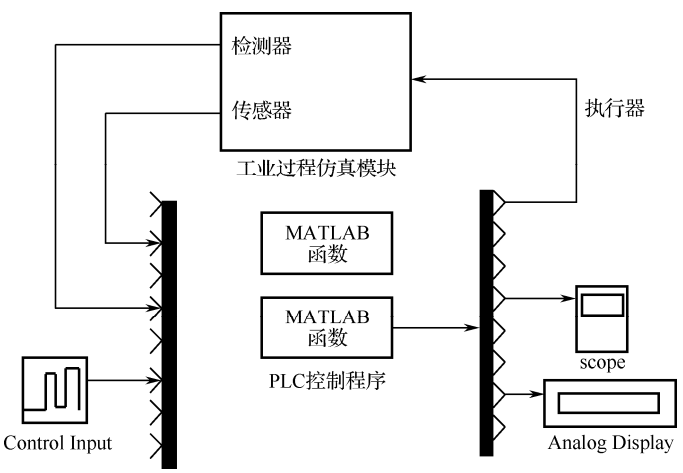


图 7-15　PLC 在 MATLAB 中的仿真原理

（2）建立 PLC 控制过程的控制语句。

（3）指定控制该过程的 PLC 型号。

（4）利用 MATLAB 的函数语句，实现对应的 PLC 控制程序。

（5）用 MATLAB 语言完成 PLC 控制程序的书写。

（6）以文本文件的方式保存 PLC 控制程序。

（7）运行开发的 PLC-MATLAB/Simulink 程序包，将 PLC 程序转换成 MATLAB 语言的 M 文件。

（8）用工业过程仿真模块仿真运行 PLC 控制过程。

（9）进行必要的修改，最终生成真正的工业控制过程程序。

对于所开发的 PLC 翻译程序包来说，在自动翻译 PLC 控制程序为 MATLAB/Simulink 语言之前，还需要下面一些信息。

（1）PLC 的类型。

（2）PLC 的输入和输出触点数。

（3）要翻译的 PLC 控制程序文件。

1．PLC 的类型

PLC 的类型是至关重要的，其类型决定了它的指令系统。虽然不同类型的 PLC 都是基于布尔逻辑关系进行设计的，但是每种 PLC 的生产商都开发了自己的编程语言。因此翻译包的开发必须在熟知 PLC 生产商的指令系统的情况下才能进行。

2．PLC 的输入/输出点数

PLC 的输入/输出点数清楚地定义了 MATLAB 函数 PLC 控制程序（PLC Control Program）的参数个数。该函数负责在 MATLAB/Simulink 环境中执行 PLC 控制程序，该函数以 MATLAB 的 M 文件的形式存在。

在该 M 文件中，di0～di*n* 代表了 PLC 的数字输入，ai0～ai*m* 代表了 PLC 的模拟输入，do0～do*p* 代表了 PLC 的数字输出，ao0～ao*q* 代表了 PLC 的模拟输出，*n*+1、*m*+1、*p*+1 和 *q*+1 分别代表了 PLC 的数字输入触点、模拟输入触点、数字输出触点和模拟输出触点的个数。

从上面的定义可以看出，*n*+1+*m*+1 决定了图 7-15 中输入 Mux 块的维数；类似地，*p*+1+*q*+1 定义了输出 Demux 块的维数。

该 MATLAB 的 M 文件的大致结构如下。

```
function [output] = PLC_Control_Program(di0,...,din,ai0,...,aim)
…
（用 MATLAB/Simulink 语言描述的 PLC 控制程序）
…
output=[do0,...,dop,ao0,...,aoq]
```

3．PLC 控制文件的翻译

PLC 控制程序的语言可以使用各种编程语言来进行编写。PLC 的运行模式和语法规则在 IEC1131-3 标准中做了定义。每个 PLC 生产商都提供了自己独有的 PLC 编程语言，这些编程语言可以被划分成以下五种类型。

（1）语句表：此种类型的编程语言类似汇编语言被认为是低等级的编程语言。

（2）梯形图：从历史上的电气回路延续下来的编程方法，其程序语言更接近电气回路，但该类程序语言不适合编写复杂的、模块化的程序。

（3）顺序功能图：方块图，非常类似图形编程语言，它将 PLC 的内部结构功能转换成步进方式进行编程。

（4）功能块图：这是另一种图形化编程语言，每个功能块都能处理几个 PLC 信号。

（5）结构文本：从 Pascal 语言转化而来，这是一种高等级的语言，能够进行复杂的模块化编程。

典型的 PLC 控制程序是以众所周知的被称为梯形图的图形化编程语言为代表的。但是，几乎所有的 PLC 程序开发包都允许使用文本为导向的编程语言。所有的编程环境也都支持梯形图程序和文本程序之间的转化。本书介绍的程序转化方法是以文本为对象的 PLC 控制程序的转化。典型的西门子 PLC 语句表程序如下所示。

```
    //
    // Program title comments
    //
NETWORK 1
    LD      I0.0
    A       I0.1
    LD      I0.2
    A       I0.3
    OLD
    =       Q0.0
    //
NETWORK 2
    LD      I0.4
    LD      I0.5
    CTU     C5,+6
    //
END
```

PLC 控制程序的翻译软件包可以由 Visual Basic 等高级开发语言开发，翻译软件包能够自动将以文本方式书写的 PLC 控制程序转化成对应的 MATLAB 的 M 文件。知道了 PLC 的输入/输出触点的个数，自动转换程序将在该 M 文件中建立处理函数的正确输入、输出参数。PLC 控制程序本身的转换将依赖依据 PLC 指令设定的转换规则。本书限于篇幅，没有给出自动翻译软件包的编写方法，而是讲解了如何手动将 PLC 指令的梯形图翻译成 M 文件的方法。

完整的 PLC 指令可以被区分成如下几种类型：布尔、比较、输出、定时器、计数器、数学、递增/递减、移位、程序控制等。

下面将依据西门子 PLC 的语句格式，给出指令转换规则。

布尔操作指令将使用标准 MATLAB 布尔函数翻译成 MATLAB/Simulink 语言，见表 7-11。在该表格中，I x.y 代表数字输入，Q x.y 代表数字输出，x 和 y 分别是对应的数字输入、输出的字节和位。进一步地，do_g 是 MATLAB 变量，代表数字输出 g；di_h 是 MATLAB 变量，代表数字输入 h。在 MATLAB 环境中，布尔状态“TRUE”代表“1”，布尔状态“FALSE”代表“0”。布尔状态的组合也应用上述规则进行转换。表 7-11 中的最后一列是布尔指令组合

的示例。

表 7-11 布尔指令的仿真代码

布 尔 指 令	PLC 指令	MATLAB/Simulink 翻译
AND	LD I a.b A I c.d = Q e.f	do_g = di_h & di_i
OR	LD I a.b O I c.d = Q e.f	do_g = di_h | di_i
NOT	LDN I a.b = Q c.d	do_g = ~di_i
布尔指令的组合	LD I 0.0 A I 0.1 LD I 0.2 OLD =Q 0.0	do_1 = (di_1 & di_2) | di_3

值得注意的是，我们这里的逻辑控制，是用 MATLAB 的仿真环境仿真 PLC 的扫描过程。在 PLC 实际的扫描工作过程中，有寄存器可以记忆触点的状态，而在 Simulink 仿真环境中，没有这个记忆环节，只能采用全局变量的方式来保存逻辑状态，使其拥有记忆功能。

其方法是在 function 里声明全局变量，如下所示。

```
global x;
```

声明全局变量后，可以在 PLC 的第一个扫描周期将这个（些）全局变量进行初始化。在之后的仿真运行中，就不再初始化了，而是根据逻辑控制程序，来修改全局变量的值。所以还要配上定时器时间来使用。

为了说明 PLC 的 MATLAB 仿真方法，下面以一个最简单的自锁环节为例，来说明仿真过程。

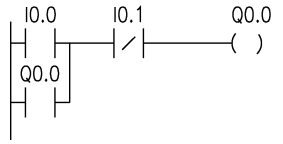


图 7-16 自锁环节的梯形图

自锁环节的梯形图如图 7-16 所示。这在机电传动控制书籍中均有介绍。

要在 Simulink 环境中仿真该 PLC 梯形图，需要进行如下步骤的操作。

（1）启动 MATLAB，进入 Simulink 环境。

（2）新建一个 Simulink 文件。

（3）在 Simulink 环境左侧的树形控件中选“User-Defined Function”，选择“MATLAB Fcn”图标到 Simulink 工作空间中。

（4）再拖动“Commonly Used Blocks”下的“Mux”和“Demux”到 Simulink 工作空间中，如图 7-17 所示。

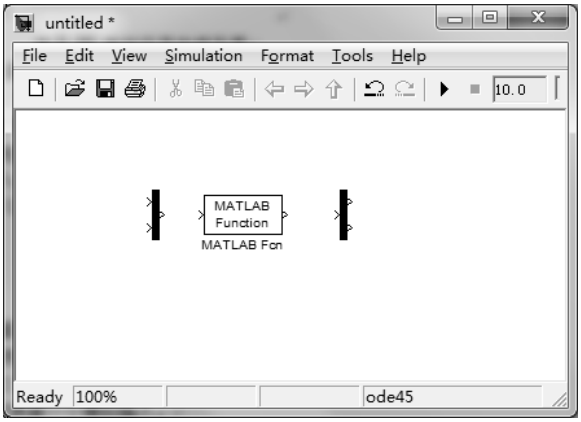


图 7-17　PLC 建模草图

（5）在 MATLAB 中新建一个 M 文件，输入如下代码，保存文件，并起名为 cpu220。

```
001 function output = S7200(u1,u2,clock)
002 global do0 do1
003 if clock == 0
004 do0 = 0; do1 = 0;
005 end
006
007 do0 = (u1 | do0) & ~u2; %梯形图逻辑功能实现
008
009 output = [do0 do1];
```

从程序代码可以看到，在 001 行定义了 PLC 有 3 个输入，但只有前两个是 PLC 真正的输入触点，而第 3 个输入变量用来记录仿真时间。002 行声明了全局变量。003 行至 005 行，在第一个扫描周期中，对全局变量进行了初始化，以后的仿真中，均不用初始化了。007 行是自锁逻辑的 m 文件翻译代码，实现了逻辑控制功能。

（6）回到 Simulink 工作空间中，双击“MATLAB Fcn”图标，在弹出的对话框中进行如图 7-18 所示的修改。

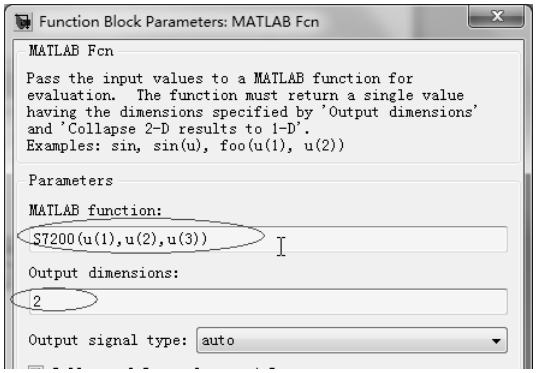


图 7-18　MATLAB Function 的修改

从图中可以观察到，我们设置 S7-200 的输入参数为 3 个，分别为 u(1)、u(2)和 u(3)；设

置 S7-200 的输出参数为 2 维。这正好印证了上一步（第 5 步）的代码，第一行中，定义输入参数为 3 个，分别为 u1、u2 和 clock；第 5 步中程序的最后一行，输出为 2 维，即“[do0 do1]”。

（7）双击“Mux”图标，在弹出的对话框中，将“Number of inputs”修改为 3，表明 PLC 接收 3 个通道的输入。

（8）在 Simulink 的树形控件中选“Source”，拖动 2 个“Signal Builder”到工作空间中，再拖动一个“Digital Clock”到工作空间中，再在树形控件中选“Sink”，拖动一个 Scope 图标到工作空间中，将它们用连线连接起来，如图 7-19 所示。

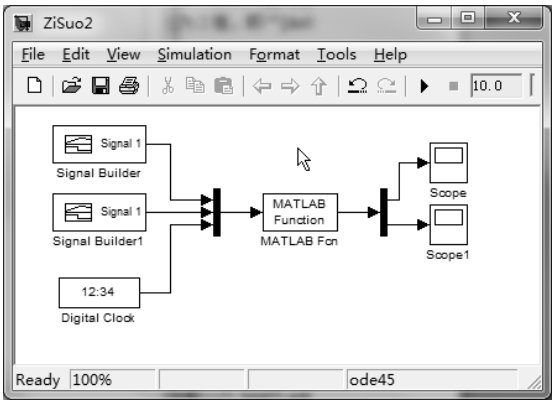


图 7-19 PLC 仿真系统最终原理图

从图 7-19 大致可以看出，“MATLAB Function”模拟了 PLC 本体，与其相连接的程序决定了 PLC 的功能。左侧的信号发生器模拟了现场的工业环境，其个数对应了 PLC 的输入触点的个数。右侧的“Scope”用来监视 PLC 的输出，其个数也等于 PLC 的输出触点的个数。

通过这个几乎是最简单的回路，完整地模拟了工业 PLC 的运行环境，完全能够验证 PLC 梯形图逻辑关系的正确性。

（9）双击“Single Builder”图标，将信号的波形修改成如图 7-20 所示。同样双击“Single Builder1”图标，将信号的波形修改成如图 7-21 所示。从图 7-20 可以看出，与 I0.0 相连接的控制信号在第 2s 到第 3s 间发出，在自锁的配合下，系统的输出触点 Q0.0 应该在第 2s 开始接通。从图 7-21 中可以看出，与 I1.1 相连接的信号，在第 7s 到第 8s 接通，所以系统的输出应该在第 7s 关断。

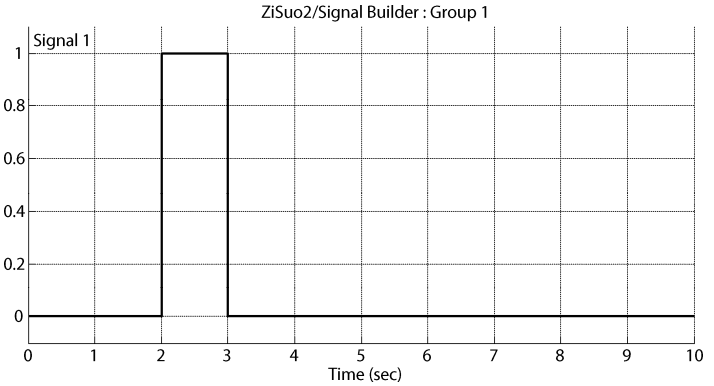


图 7-20 Single Builder 的波形

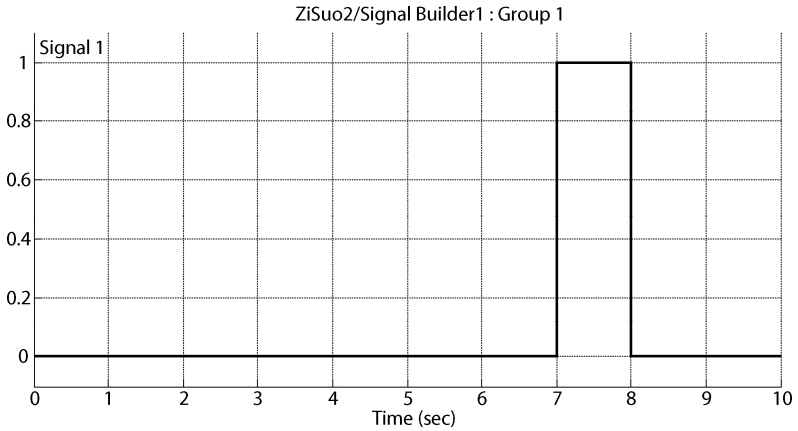


图 7-21　Single Builder1 的波形

（10）修改完成后，单击 Simulink 工具栏上的按钮，运行仿真，可以得到仿真结果。从图 7-22 中可以观察到，系统在第 2s 到第 7s 维持输出，正好与设计的输入信号的控制思想是一致的。

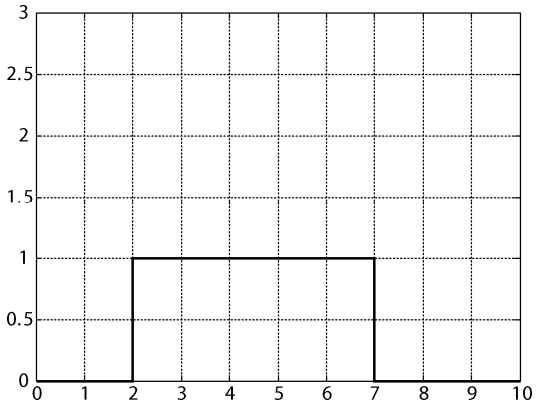


图 7-22　PLC 仿真结果

7.5.2　数学指令的转换

PLC 的数学指令通常都是布尔使能的。这一特点暗含了应该使用 MATLAB 的 if 函数来实现这一功能。作为示例，考虑表 7-12 中的 PLC 整形加法指令。AIW0 和 AQW0 分别代表 PLC 的模拟输入和 PLC 的模拟输出。变量 ao_1 是 MATLAB 变量，代表第一个模拟输出，ai_1 是 MATLAB 变量代表第一个模拟输入。

另一方面，PLC 控制程序经常使用内部标志变量（位或内部寄存器的状态）代表状态或存储模拟值。无论何时，当解释程序发现一个 PLC 内存数据，就自动地分配一个 MATLAB 变量给它。这些变量通常根据它们是数字的还是模拟的，标示为 m 或者 v。PLC 的乘法指令通常包括对一个附加内存地址的使用，见表 7-12。在该表格中，也使用了 MOVW 指令，把一个 16 位的字移动到另一个变量里）。注意，PLC 内部变量 VD 代表 32 位字。

表 7-12 非布尔指令的仿真代码

指 令	PLC 指令	MATLAB/Simulink 译码
INCREMENT	LD I 0.0 +I AIW0, AOW0	if di_1 ao_1 = ai_1 + ao_1 end

MULTIPLY	LD I 0.0
	MOVW +6, VD4
	MUL +9, VD4
		if di_1
		v_4 = +6
		v_4 = +9*v_4
		end

7.5.3 计数器和定时器指令的转换

PLC 计数器指令（CTUD-向上计数或向下计数）需要几个布尔输入，即一个向上计数，一个向下计数和一个重置计数器。因为计数仅发生在布尔操作的上升沿，MATLAB/Simulink 必须记住布尔输入之前的状态。表 7-13 显示了一个计数器的例子。在该例子中，di_1_prev 是 MATLAB 变量，代表了变量 di_1 之前的状态。所谓之前的状态指 PLC 当前扫描周期之前的状态。在本例中，当计数器数到 4 时，计数器的布尔状态改变为 true。在 MATLAB 环境中，c_10 代表了计数器 10 的布尔状态，c_10_value 代表了该计数器的计数值。

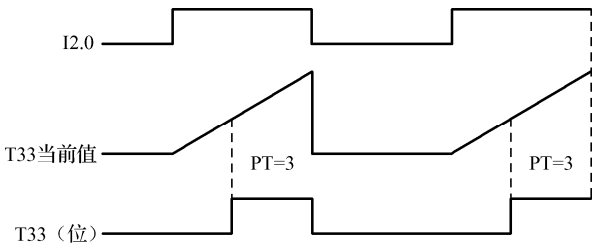


图 7-23 PLC 定时器时序图

PLC 的定时器指令，例如接通延时定时器，只需要一个数字输入来计数。如图 7-23 所示，对于延时接通定时器（本例中为 T33）来说，无论何时数字输入（本例为 I2.0）为使能，并且延时时间达到设定值（本例为 3s）后，T33 才开始工作。

定时器的代码也列写在表 7-13 中。无论何时定时器启动（对应的数字输入 di_20 为 true，且此时定时器还没有启动），预设时间（t33_start）被加到当前时钟值（clock_in）以建立时钟停止时间（t33_end_time）。当达到停止时间，定时器的逻辑值（t1_bin）改变为 true。当其对应的布尔输入改变为 FALSE 时，定时器重新启动。

表 7-13 计数器、定时器 PLC 仿真代码

指 令	PLC 指令	MATLAB/Simulink 译码
计数器	LD I 0.0 // 增计数 LD I0.1 // 减计数 LD I0.2 // 重置计数器 CTUD C10, +4	if di_1 & ~di_1_prev %增计数 c_10_value = c_10_value+1 end if di_2 & ~di_2_prev %减计数 c_10_value = c_10_value-1 end if di_3 %复位计数器 c_10_value = 0 c_10 = 0 end if c_10_value >= +4 %重置计数器 c_10 = 1 end

续表

指　令	PLC 指令	MATLAB/Simulink 译码
定时器	LD　I2.0 TON　T33,3	%开启定时器 if di_20 & t33_start == 0 　　t33_start = 3 　　t33_end_time = clock_in + 3 end %定时器打开 if t33_start & clock_in >= t33_end_time 　　t33_bin = 1 end %定时器关闭 if ~di_20 　　t33_bin = 0 　　t33_start = 0 end

7.6　应用实例一

第一个实例我们仿真一个纯布尔操作的工程。假设我们操作一台自动切割机，如图 7-24 所示。在图 7-24 中，也绘制了用于控制机器运动的行程开关（由 Si 代表，i=1,2,⋯,5）。在原位时，切割机压下行程开关 S5、S3。当按下启动按钮后，切割机向右移动。在切割圆木之前（碰到行程开关 S1 时），锯片开始运转。当切割完成后（碰到行程开关 S2），锯片关闭，同时锯片必须升起，当升起到顶端时（碰到行程开关 S4），停止向上运动，锯片开始向左运动，碰到行程开关 S5，锯片开始向下运动，直到碰到行程开关 S3，表明回到原点，此时锯片在原位停止。

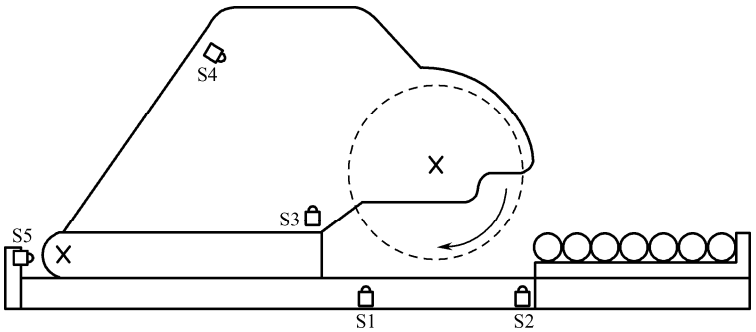


图 7-24　切割机

表 7-14 列出了 PLC 输入和输出触点的分配。每一行描述了信号的功能，其对应的 PLC 地址，和对应的 MATLAB 地址。为了完成本自动过程，使用了 6 个数字输入和 5 个数字输出。

我们选择标准的带有 8 个数字输入和 6 个数字输出的西门子 S7-200 PLC 执行自动控制功能。

表 7-14 PLC 地址单元的分配

信 号 描 述	PLC I/O 地址	对 应 元 件	MATLAB 变量分配
启动	I 0.0	按钮	di0
电锯启动	I 0.1	行程开关 S1	di1
电锯关断	I 0.2	行程开关 S2	di2
机器下限	I 0.3	行程开关 S3	di3
机器上限	I 0.4	行程开关 S4	di4
机器在起始位置	I 0.5	行程开关 S5	di5
向左运行	Q 0.0	电动机 M1 顺时针旋转	do0
向右运行	Q 0.1	电动机 M1 逆时针旋转	do1
电锯旋转	Q 0.2	电动机 M2 旋转	do2
机器上升	Q 0.3	电动机 M3 顺时针旋转	do3
机器下降	Q 0.4	电动机 M3 逆时针旋转	do4

绘制出 PLC 的硬件接线图有助于我们编写自动控制的梯形图程序，根据表 7-14 PLC 地址单元的分配，可以得到 PLC 的硬件接线图如图 7-25 所示。

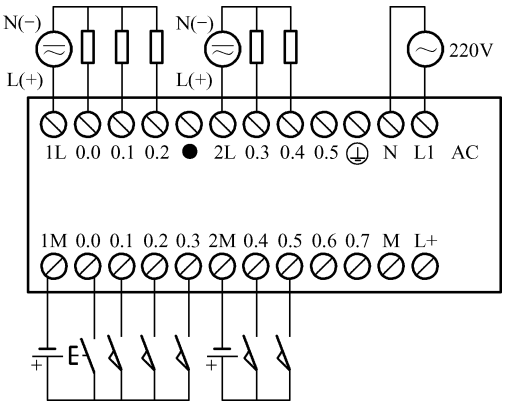


图 7-25 原木切割机 PLC 硬件接线图

根据控制逻辑要求，编写自动控制的梯形图如图 7-26 所示。梯形图的编写主要采用了自锁环节和按步递进的编程技巧，具体的设计方法读者可以查找机电传动的相关书籍，本书此处不再赘述。

MATLAB 的 M 文件的代码如下。

```
001 function output = cpu222( di0,di1,di2,di3,di4,di5,clock_in)
002 %输出初始化
003 global do0 do1 do2 do3 do4
004 if clock_in == 0
005 do0 = 0; do1 = 0; do2 = 0; do3 = 0; do4 = 0;
006 end
```

```
007 %梯形图译码
008 do1 = (di0 & di5 & di3) | do1;
009 do2 = (di1 | do2) & ~di2;
010 do3 = (di2 | do3) & ~di4;
011 do0 = (di4 | do0) & ~di5;
012 do4 = (di5 | do4)& ~di3;
013
014 output = [do0 do1 do2 do3 do4];
```

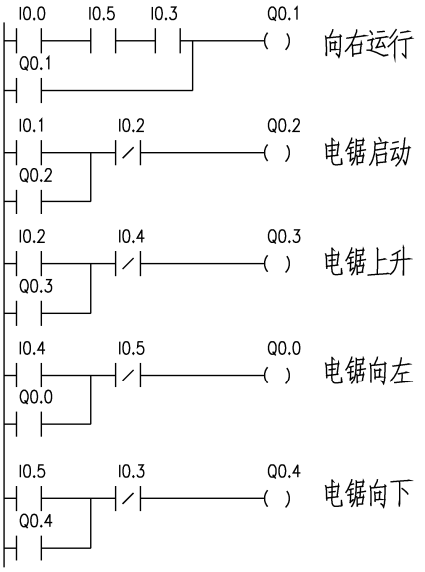


图 7-26　切割机的自动控制梯形图

下面对程序的代码进行说明。

程序的第 001 行进行了函数的声明。尤其值得注意的是输入和输出。输入变量的个数根据题目中的输入触点的个数设置，但应该在最后增加一个“clock_in”，作为时钟脉冲的输入，这个时钟脉冲很重要，它的功能一方面可以用来实现定时器，另一方面可以在第一个时钟中期对系统的全局变量进行初始化。

程序中的 003 行至 006 行定义了全局变量，并在第一个时钟周期中将这些全局变量进行了初始化。在剩余的仿真周期中，这些全局变量将根据程序中的逻辑关系的计算结果进行赋值。

程序中的 008 行至 012 行，将图 7-26 中梯形图的逻辑关系翻译成了 MATLAB 语言，完成了梯形图的译码。

程序中的 014 行，将输出触点的状态（全局变量的值）整合在一个矩阵中完成输出，这是实现 PLC 控制多触点输出的一个方法。

7.7　应用实例二

第二个例子演示了定时器和计数器的使用方法。

考虑一个接受随机注水的水箱，该水箱的水位要求保持在 4～5m 的高度，如图 7-27 所示。为了实现上述目标，使用一个电气控制的阀门来控制水位，该阀门的打开或关闭由 PLC 的输出触点控制。PLC 根据两个水位检测器（安装在水箱的 4m 和 5m 高度处）提供的信息控制阀的开度的大小。在本例中，同时记录阀打开的次数。当阀的打开次数达到 8 次时，将发出两种类型的警报：指示灯信号和蜂鸣器警报。当计数器达到 8 次时，蜂鸣器应该被触发 5s。

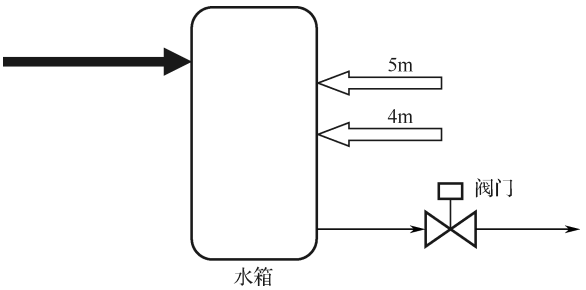


图 7-27　水箱水位控制原理图

7.7.1　水箱的物理建模

要想完整地对本 PLC 控制系统进行仿真，还需要构建水箱注水的数学模型，该数学模型应该能够模拟水箱注水和排水的物理过程，该物理过程的描述，可以利用累加的原理来实现。水箱模拟程序的代码如下。

```
001 %水箱模拟程序
002 %u1: 随机注入的水的体积
003 %u2: 水箱排水的体积
004 %u3: 仿真的时刻
005 function output_h = TankMode(u1,u2,u3)
006 %初始容积
007 global V0 h_prev
008 if u3 == 0   %如果是初始时刻，水箱内的水和排放的水都为 0
009         V0 = 0;
010         h_prev = 0;
011 end
012 %水箱截面积
013 S = 2;           %2m^2
014 %当前水箱中水的体积
015 V0 = V0 + u1 - u2;
016 %当前水箱中水位的高度
017 h = V0 / S;
018 %水位传感器的实现
019 if h > 5       %水位高于 5m，输出信号 1
020         output_h = 1;
021         h_prev = output_h;
022 elseif h < 4       %水位低于 4m，输出信号 0
```

```
023        output_h = 0;
024        h_prev = output_h;
025 else                    %如果处于 4m 到 5m 之间，输出信号等于上一个扫描周期
026        output_h = h_prev;
027 end
```

在 MATLAB 中新建 M 文件，输入上述代码（行号除外）。保存文件名为“TankMode.m”。

下面对程序代码进行说明。

程序代码的 001 行至 005 行，定义了水箱物理模型的输入变量，分别为 u1、u2 和 u3。其中，u1 代表随机注入水箱中水的体积；u2 代表水箱排水的体积；u3 代表仿真时刻。

程序中的第 007 行到第 011 行是在第一个仿真周期初始化各物理量，主要是将初始容积 V0 清零，将初始信号 h_prev 清零。

程序的第 012、013 行是设定水箱的横截面积，下面用来计算水位的高度。在每个仿真周期，水箱中水的实际体积为“V0 = V0 + u1 −u2”，其中，u1 代表注入水的体积，u2 代表排放水的体积，每个仿真周期水箱中的水量等于上一个仿真周期的水量加上本仿真周期注入的水量再减去排出的水量。程序第 017 行计算当前水平（用总水量除以水箱的横截面积）。

程序 018 行至 027 行是为了模拟水位传感器的逻辑信号输出值。根据条件判断语句，读者不难看出，当水位高度大于 5m 时，水位传感器输出的信号为 1（output_h = 1）；当水位高度小于 4m 时，水位传感器输出的信号为 0（output_h = 0）；否则如果水位处于 4～5m，水位传感器输出的信号保持上一个仿真周期的输出信号（output_h = h_prev）。

为了降低仿真的难度，我们可以先建立水箱物理模型（而不建立 PLC 控制模型），来验证我们水箱物理模型代码的正确性。

（1）新建一个 Simulink 仿真文件。

（2）在 Simulink 库中寻找下列图标，拖动到工作区域中，如图 7-28 所示。其中的“Uniform Random Number”用来产生随机数，以模拟注入的水量；“Constant”用来模拟排出的水量；“Digital Clock”用来模拟仿真时间；“MATLAB Fcn”用来设定仿真水箱的函数代码；“Scope”用来观察输出。

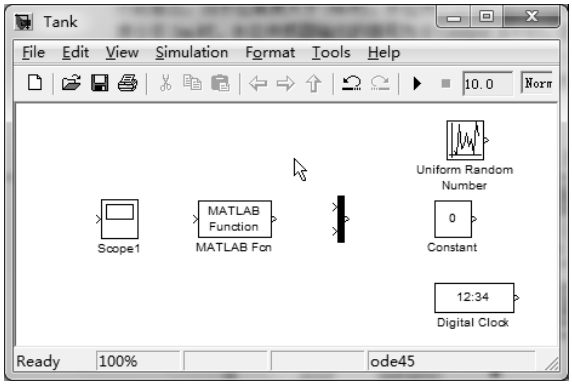


图 7-28　仿真水箱容积所需要的物理模型

（3）选中适当的元件，单击鼠标右键，选择菜单“Format”下的“Flip Block”，或直接按动键盘上的“Ctrl+I”键，可以翻转对应的图标，整理成如图 7-29 所示。

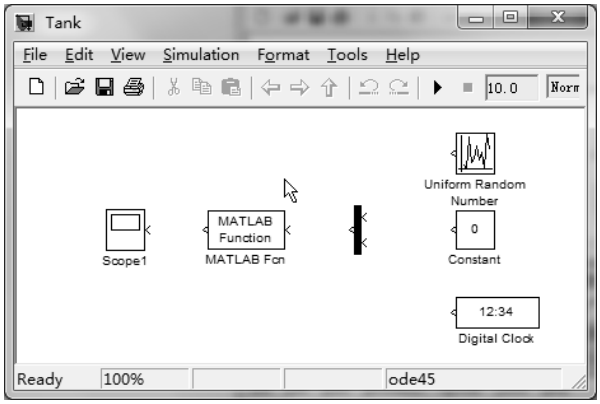


图 7-29　整理后的图标状态

（4）从水箱的物理模型代码我们可以看出，我们的水箱模型有 3 个输入，所以我们要将“Mux”模型的输入修改成 3 个。单击“Mux”图标，将“Number of inputs”修改为 3。

（5）常量图标“Constant”先修改为 0。然后连接图形中各个图标，如图 7-30 所示。

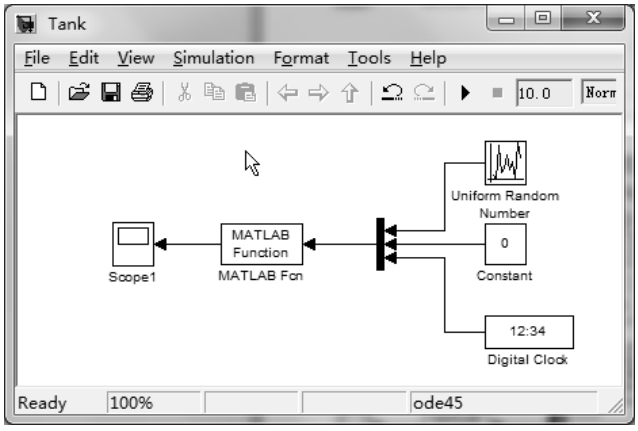


图 7-30　连接后的模型

（6）双击图标“MATLAB Fcn”，修改其选项。将“MATLAB function”修改为“TankMode(u(1), u(2), u(3))”，将“Output dimensions”修改为“1”，如图 7-31 所示。

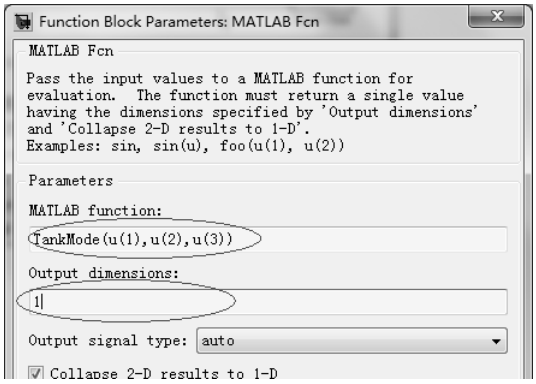


图 7-31　修改“MATLAB Fcn”参数

（7）双击图标“Uniform Random Number”图标，修改其“Minimum”选项为“0”，这样保证注入水箱中的水量最小值为 0，既都是大于 0 的数，这符合实际的物理原理。

（8）修改完成后，运行工具栏上的开始仿真按钮，然后双击“Scope”图标，就可以看到仿真结果曲线，如图 7-32 所示。

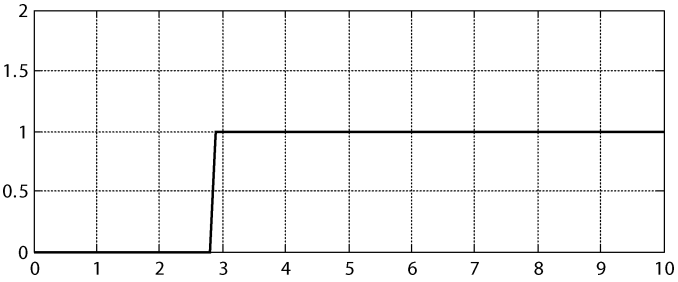


图 7-32　水箱模型仿真结果

（9）如果我们修改“Constant”的值，将其修改为“0.2”，相当于加入了一个排水常数，再运行仿真，可以得到图 7-33 所示图形。

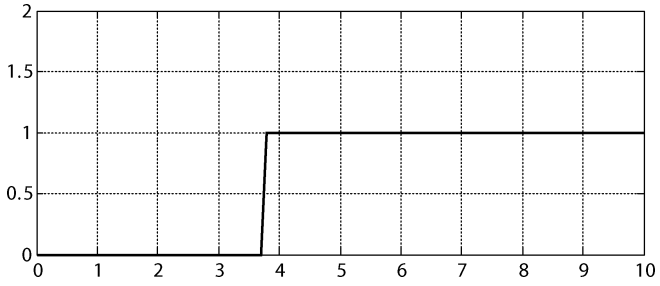


图 7-33　加入排水常数后得仿真结果

对比图 7-32 和图 7-33，可以观察到后者比前者用了更多的时间（前者接近 3s，后者接近 4s）达到水位 5m 的状态，接近实际的效果。

7.7.2　PLC 的物理建模

7.7.1 节只仿真了水箱的物理模型，要想仿真对水箱水位的控制，还需要将 PLC 的逻辑控制功能仿真出来，本节将讲解 PLC 控制的仿真方法。

本工程采用的 PLC 的型号为西门子 S7-200，该 PLC 带有 8 个数字输入、4 个模拟输入、6 个数字输出和 4 个模拟输出。根据这些输入和输出的类型，定义了 MATLAB 子系统“PLC Control Program”的输入和输出参数。该子系统的输入是西门子 S7-200 的输入和时钟，子系统的输出参数是 S7-200 的输出。表 7-15 定义了 PLC 的触点定义和在 MATLAB 环境中的对应项。

表 7-15　水箱水位控制 PLC 触点分配

信 号 描 述	PLC 信号	对 应 元 件	MATLAB/Simulink 变量
最高水位	I 0.0	开关	di0
最低水位	I 0.1	开关	di1

续表

信 号 描 述	PLC 信号	对 应 元 件	MATLAB/Simulink 变量
外部计数器复位	I 0.2	按钮	di3
阀	Q 0.0	电磁阀	do0
灯光报警信号	Q 0.1	记录 8 次阀操作	do1
蜂鸣器信号警报	Q 0.2	等亮后延时 1 秒动作	do2
计数器的值	QW10	计数器的值	ao1
阀操作计数器	C1	内部计数器（CTU）	c1
计数器警报定时器	T1	内部定时器（延时吸合）	t1

根据表 7-15，可以绘制 PLC 的硬件接线图如图 7-34 所示。

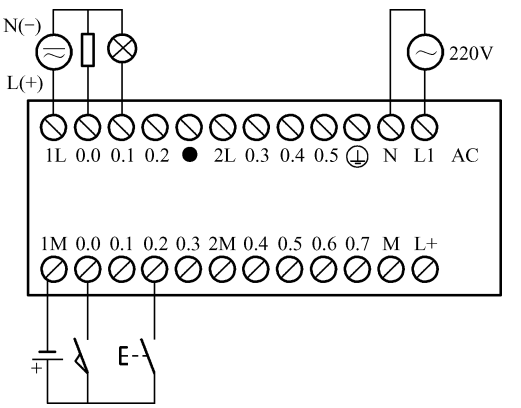


图 7-34　水位控制 PLC 硬件接线图

PLC 的控制梯形图如图 7-35 所示。

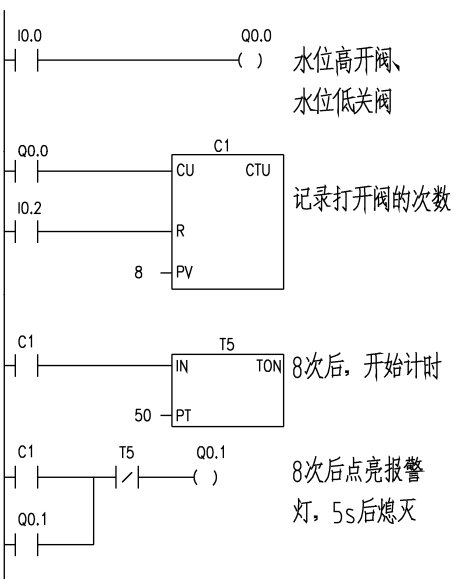


图 7-35　水位控制 PLC 梯形图

根据图 3-35 所示的梯形图程序，手工翻译的 MATLAB 程序代码如下。

```
%水箱水位控制 PLC 程序
function output = TankPLC(di0, di1, di2, clock_in)
global c_1 c_1_value t5_start t_5 t5_end_time
global do0 do1 do2 do0_prev do1_prev
%如果定时器为 0，说明为初始化
if clock_in == 0
    c_1 = 0;
    c_1_value = 0;
    do0 = 0;
    do1 = 0;
    do2 = 0;
    t_5 = 0;
    t5_start = 0;
    t5_end_time = 0;
    do0_prev = 0;
    do1_prev = 0;
end

%梯形图译码
do0 = di0;

%计数器
if (do0 == 1) & ~do0_prev
    c_1_value = c_1_value + 1;
end
if di2
    c_1_value = 0;
    c_1 = 0;
end
if do0 ==1 & c_1_value >=8    %达到指定计数值
    c_1 = 1;
end

%定时器
if c_1 & t5_start == 0
    t5_start =1;
    t5_end_time = clock_in + 5;
end
%定时器开
if t5_start & (clock_in >= t5_end_time)
    t_5 = 1;
end
```

```
043 %梯形图译码
044 do1 = (c_1 | do1) & ~t_5;
045 %保存上一个扫描周期输出触点的状态
046 do0_prev = do0;
047 do1_prev = do1;
048
049 do0 = double(do0);
050 do1 = double(do1);
051
052 output = [do0 do1]
```

程序代码的 001～018 行定义了输入变量、全局变量和全局变量的初始化。019～020 行定义了排水阀的控制语句。022 到 032 行为计数器的 M 文件指令。034～042 行为计数器 M 文件指令。034～044 行还是梯形图 M 文件指令。045～047 行是为了保存上一个扫描周期输出触点的状态。049～050 行是为了将输出进行强制类型转换（如果不转换成 double 类型，Simulink 仿真报错）。

在 7.6 节中，我们曾经建立过一个只拥有逻辑控制功能的 PLC 程序，本例中也与其相似，我们也要建立一个类似的 PLC 仿真模型。所不同的只是输入触点的个数和输出触点的个数，以及 PLC 逻辑控制功能的程序。在 7.7.1 节中，我们还建立了水箱物理模型，在本节中，我们需要将水箱模型和 PLC 控制模型结合起来。通过对其原理进行分析，可以建立如图 7-36 所示的仿真草图。

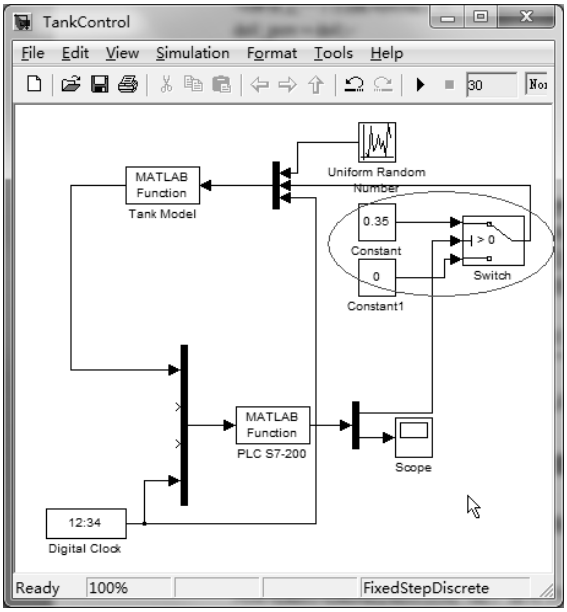


图 7-36　仿真回路草图

值得说明的是，图 7-36 中画圈的部分，这部分采用“Switch”图标模型排水控制阀，当“Switch”图标的第二个输入大于 0 时，“Switch”的输出为 0.35，表明排水每个周期为 0.35m^3，如果输入小于 0（在本例中为 0），则关闭排水阀，即排水量每个周期为 0m^3。

图 7-36 中其余元件的使用方法我们在前面已经进行了介绍，此处不再赘述。

在 Simulink 工件栏上将仿真时间修改为 30s，如图 7-37 所示。

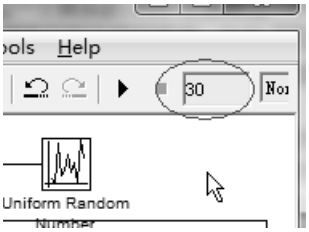


图 7-37　修改仿真时间

其余值得注意的问题如图 7-36 所示，名称为“Tank Model”的“MATLAB Function”的函数应该设置为“TankMode(u(1), u(2), u(3))”。同样，名称为“PLC S7-200”的“MATLAB Function”的函数应该设置为“TankPLC(u(1), u(2), u(3), u(4))”，这两个函数的输出维数也要设置正确，以上参数的设置方法如图 7-18 所示，此处不再赘述。

运行系统，可以得到仿真结果。双击图 7-36 中的 Scope 图标，可以观察到系统在大约 29s 处产生一个输出（见图 7-38），此处模拟点亮报警灯。

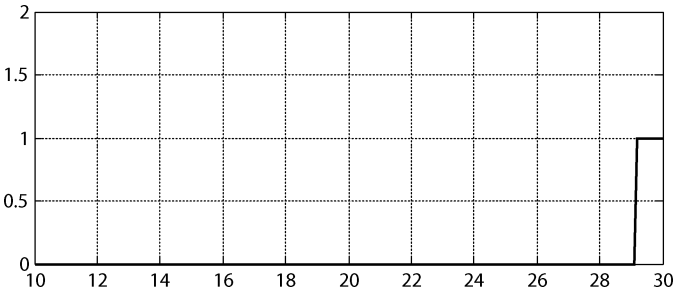


图 7-38　系统报警灯引脚的输出

为了对系统进行全方位的了解，可以在自己关心的位置处加上 Scope 图标，观察系统的输出，如图 7-39 所示。

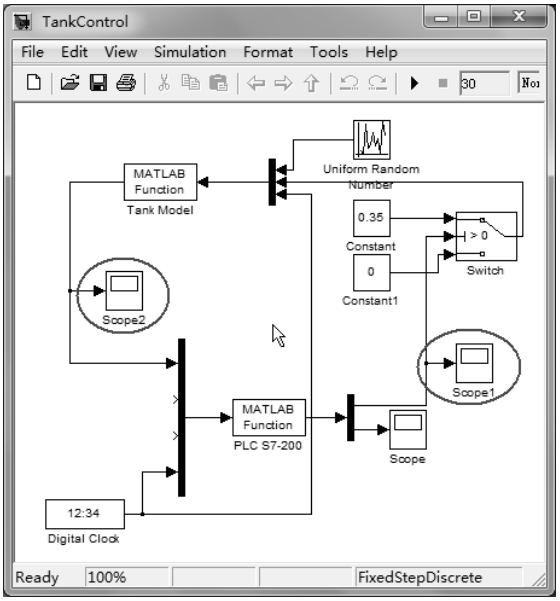


图 7-39　添加了多个 Scope 图标

以 Scope1 为例，双击该图标可以观察到本水位控制系统电磁阀的接通和关断次数，系统在大约第 29s 处，接通关断次数达到了 8 次，与此同时，报警灯应该点亮，如图 7-38 所示。

图 7-38 和图 7-40 印证了系统 PLC 控制逻辑的正确性。

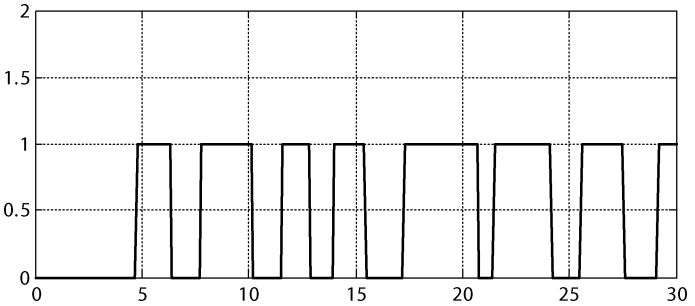


图 7-40　电磁阀的接通和关断信号波形

课后题

1．试编写程序，实现对电动机单向运行的启动/停止控制。要求：有两个输入控制器件，即启动按钮 SB1 和停止按钮 SB2；有 3 个输出器件，即控制电动机启动的接触器线圈 KM、绿色指示灯 HL1 和红色指示灯 HL2。编程元件的地址分配如下。

I0.0：SB1；I0.1：SB2；Q0.0：KM；Q0.1：HL1；Q0.2：HL2

编写上述程序，并在 MATLAB 的 Simulink 环境下对上述程序进行仿真。

2．有 3 台皮带运输机，分别由电动机 M1、M2、M3 驱动。要求：按启动按钮 SB1 后，启动顺序为 M1、M2、M3，间隔时间为 5s；按停止按钮 SB2 后，停车时的顺序为 M3、M2、M1，间隔时间为 3s。3 台电动机 M1、M2、M3 分别通过接触器 KM1、KM2、KM3 接通三相交流电源，用 PLC 控制接触器的线圈。

试编写实现上述控制功能的 PLC 程序，可自由分配地址，并在 MATLAB 的 Simulink 环境下对上述程序进行仿真。

3．有一个密码锁，它有 4 个按键，分别为 SB1～SB4，其控制要求如下。

（1）SB1 为启动键，按下 SB1 键，才可进行开锁作业。

（2）SB2 为复位键，按下 SB2 后，可重新进行开锁作业。如果按错键，则必须进行复位操作，所有计数器都被复位。

（3）SB3、SB4 为可按压键。开锁条件为：SB3 设定按压次数为 3 次，SB4 设定按压次数为 5 次，如果按上述规定按压，则 5s 后，密码锁自动打开。

（4）除启动键外，不考虑按键的顺序。

试编写 PLC 程序，可自由分配地址，实现上述控制功能，并在 MATLAB 的 Simulink 环境下对上述程序进行仿真验证。

第 8 章　常用液压元件及系统的建模方法

本书的主要目的是介绍液压系统的仿真方法，但是单就液压系统的仿真来说，这涉及仿真的基本技术和仿真的基本原理。本书前面的章节主要介绍了仿真的基本技术（MATLAB 软件的操作方法和技巧）和基本原理（节点容腔法、控制工程基础等），本章将主要介绍上述基本技术和基本原理如何应用于液压元件和液压系统的仿真中。

8.1　节点容腔法建模举例

液压系统由多个液压元件和管路组成，液压元件通常具有多个油口并与管路相连，通过管路相连的多个元件之间构成液压容腔。把液压管路的交汇点定义为节点容腔，对每个节点建立流量平衡方程，以表达节点压力和进出该节点流量之和的关系，从而得到一组方程，即为液压系统的集中参数数学模型。此方法称为节点容腔建模法。

8.1.1　孔口流量公式的仿真方法

本节我们将对液压系统中的节流孔口进行仿真。节流孔口的数学模型是孔口流量公式，该公式是液压系统中最基本、同时也是最重要的模型。

需要说明的是，在本节中，我们对孔口流量公式进行了适当的简化，主要是为了阐明液压系统中的节流孔口在 MATLAB 的 Simulink 环境下应该怎样进行仿真。

通常，利用 MATLAB 环境来对已知数学模型系统的仿真建模方法有两种：一种是利用 Simulink 环境中提供的现成的图标模型库，将数学公式用 Simulink 模型库中的模型来描述，利用图形来建立仿真模型，这有时会用到子系统的封装技巧；另一种方法是用 Simulink 的“MATLAB Function”模型用编程方法建模。其中前一种方法适合较简单的模型，后一种方法适合较复杂的模型。下面我们将对两种方法分别进行介绍。

1．利用图标模型来仿真节流孔

首先给出节流孔口的数学公式，利用流体力学中的质量守恒定律——伯努利方程，可以推导出节流孔口流量公式为

$$q = C_{\mathrm{d}} A \sqrt{\frac{2}{\rho} \Delta p} \tag{8.1}$$

式中　C_{d} ——流量系数，一般取值为 0.7（无单位）；

A ——节流孔口的面积（m^2）；

ρ ——液压油的密度，一般可取 850（$\mathrm{kg/m}^3$）；

Δp ——节流孔口的前后压差（Pa）。

在进行上式的计算时，一定要注意公式中每个变量的单位，当全部采用上式中的规定单位时，计算出了流量的单位就是 m^3/s。

另外还要注意到，要想在计算机中建立孔口流量公式的仿真模型，还要对孔口流量公式做一定的修改。修改后的公式如下。

$$Q = \mathrm{sign}(\Delta p) C_{\mathrm{d}} W x \sqrt{\frac{2}{\rho} |\Delta p|} \tag{8.2}$$

修改部分主要体现在压力差的处理上：①为了保证平方根下的数值总为正，需要将平方根下的压力差添加一个绝对值符号；②另外还添加了一个 sign 符号，以保证当压差为正时，流量为正，即流量是流入的。当压差为负时，流量为负，即流量是流出的。

要利用 MATLAB 的 Simulink 模型搭建孔口流量公式的仿真模型，其操作步骤如下。

（1）启动 MATLAB，在工具栏上找到 Simulink 图标，单击启动，将弹出“Simulink Library Browser”窗口。

（2）在“Simulink Library Browser”窗口中，单击新建按钮，建立一个空白仿真文件，如图 8-1 所示。

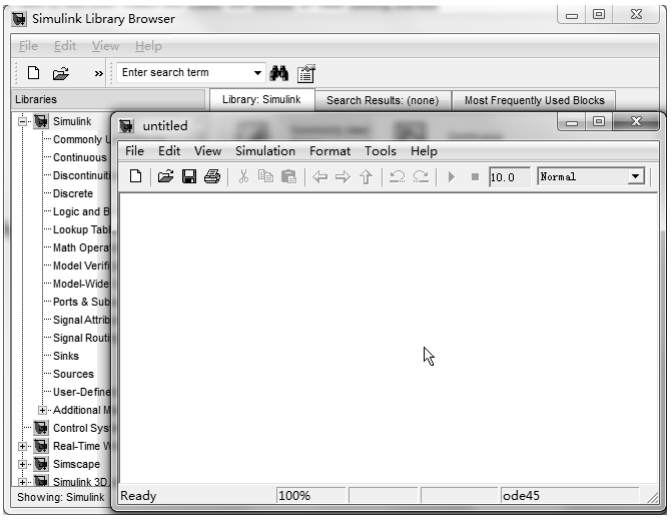


图 8-1 在 Simulink 中建立一个空白仿真文件

在真正建立孔口流量仿真模型之前，我们先来分析一下式（8.1）。在该式中，已知固定不变的量是 C_{d}，说明它是一个常数，这可以用 Simulink 的“Constant”（常量）来模拟。另外，密度也是一个不变量，也可以用“Constant”来模拟。而节流孔的大小，根据我们的仿真要求，应该是一个可以改变的变量，这样我们就能仿真不同的开口量下的流量，这也可以用“Constant”来模拟，只不过每次都要人为地修改这个变量。Δp 是节流孔口的入口和出口的压力差，我们可以设置两个输入变量，一个作为入口，一个作为出口，再设置一个输出变量，作为流量计算的结果。但实际上，节流孔的入口和出口都有流量流过，只不过入口的流量是流入的，而出口的流量是流出的。所以我们这个仿真系统有两个输入参数，也有两个输出参数。

（3）建立仿真模型，首先拖动 5 个常量图标（“Constant”）到空白 Simulink 文件中，分别用这 5 个“Constant”图标模拟孔口流量公式中的 C_{d}、A、ρ、p_1 和 p_2（$\Delta p = p_1 - p_2$），可以修改图标的注释以利记忆，修改后的结果如图 8-2 所示。

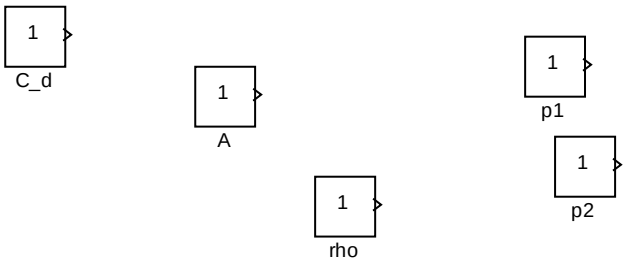


图 8-2　仿真常量

（4）下面的工作就是将这些常量进行有机的结合（主要是利用数学运算图标），来模拟孔口流量的数学公式。观察孔口流量公式，各个变量之间的关系主要是相乘、取倒数、减法，还有开平方根的关系，这都可以用 Simulink 中的“Math Operations”库中的仿真图标来完成，主要采用的是乘法（Product）、减法（Add）、平方根和函数（倒数）仿真图标（Math Function）。将这两个图标拖入 Simulink 仿真文件，如图 8-3 所示。

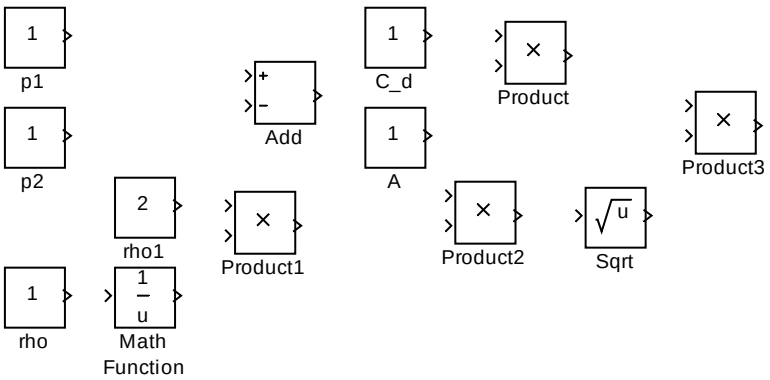


图 8-3　加入数学函数的仿真模型

按照孔口流量公式，将图 8-3 中的图标进行连接，如图 8-4 所示。

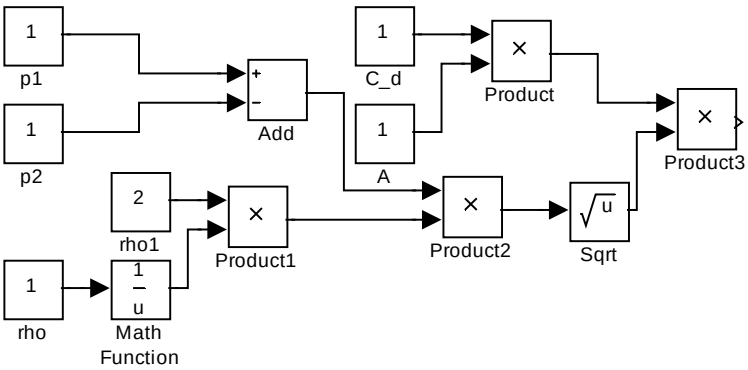


图 8-4　孔口流量公式的仿真模型

值得说明的是，为了实现减法、平方根、相减和乘法的关系，需要对数学运算图标进行设置。其中倒数计算，可以通过单击“Math Function”图标，在弹出的对话框中，选择“Function”下拉列表框中的“reciprocal”选项实现，如图 8-5 所示。

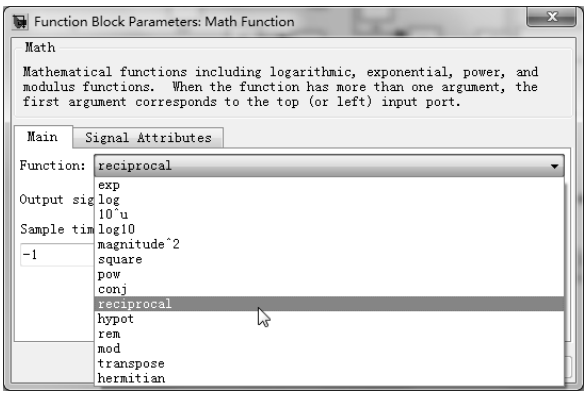


图 8-5 设置倒数运算

减法的实现需要修改加法运算图标为减法，双击加法“Add”图标，将“List of signs”文本框中的字符为“+-”，如图 8-6 所示。

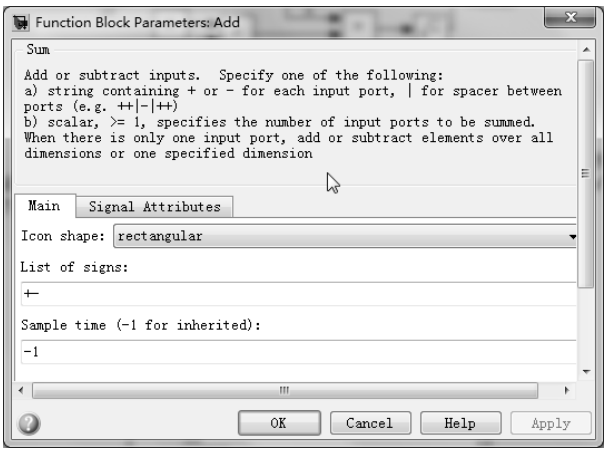


图 8-6 实现减法运算

其余图标直接相连接就可以。

（5）然后是为系统中的元件赋值。根据实际情况，$C_d = 0.7$，面积 $A = 1\times10^{-6}\,\mathrm{m}^2$，密度 $\rho = 850\,\mathrm{kg/m}^3$，$p_1 = 5\times10^6\,\mathrm{Pa}$，$p_2 = 0\mathrm{Pa}$。赋值后的图形如图 8-7 所示。

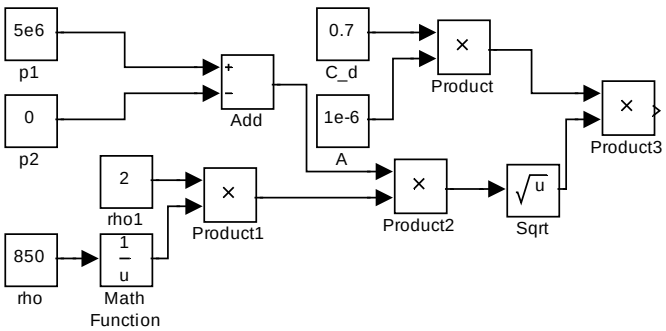


图 8-7 添加参数后的仿真回路

（6）添加观察元件，这里我们选“Sinks”库中的“Display”，添加后的结果如图 8-8 所示。

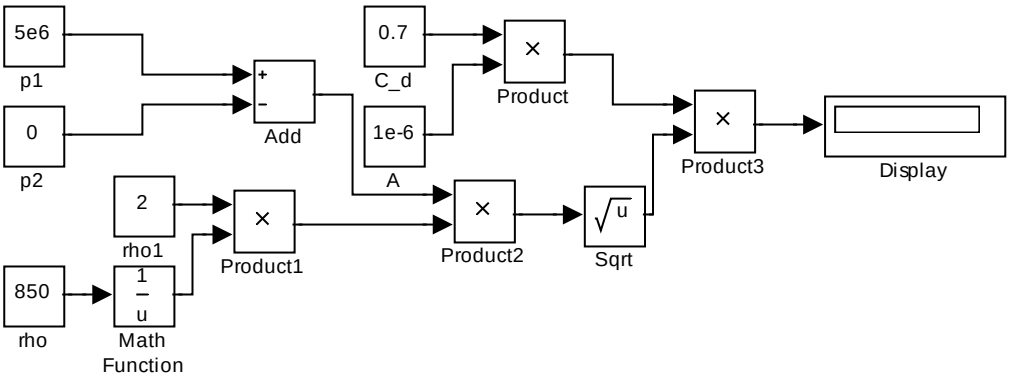


图 8-8　添加了观察元件的仿真回路

完成了上述工作，可以单击工具栏上的黑色箭头，开始仿真了。仿真过程很快就会完成，结果如图 8-9 所示。

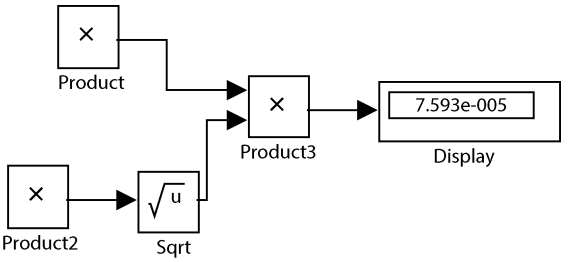


图 8-9　仿真结果

从图 8-9 中可以观察到，在液力差为 5×10^{6}Pa、孔口截面积为 1×10^{-6}m^2 的情况下，通过孔口的流量为 75.93×10^{-6}m/s。

2．仿真模型的封装

对于简单的系统来说，可以直接建立系统的模型，并分析模块之间的相互关系以及模块的输入输出关系。但是对于一个复杂系统来说，或者一个大系统中存在多个相对独立的子系统时，Simulink 中将会包含非常多的模块，使得各个模块之间的相互关系显得非常复杂，不利于分析。子系统正是针对以上原因而设计的，可以将联系比较紧密的模块或者归属于一个子系统的模块进行封装，这样就能够对大系统模型一目了然。

子系统可以理解为一种“容器”，我们可以将一组相关的模块封装到这个子系统模块当中，并且等效于原系统模块群的功能。

下面来介绍两种建立子系统的方法。

设已有系统如图 8-8 所示。用鼠标选择或用框选的方法（用 Shift 键和鼠标左键配合）选中要封装在子系统中的元件。具体选择哪些元件，需要用户进行事先思考。在本例中我们以孔口流量公式为仿真对象，我们设想为孔口通入不同的压力，用仿真的方法计算通过孔口的流量。所以我们希望这个子系统对外界的输入接口是入口和出口的压力，输出接口是计算得到的流量。这样我们就将除输入、输出压力和流量之外的模型封装进子系统。

如图 8-10 所示，选中要封装进子模型的元件后，将鼠标放到某个模型上，单击鼠标右键，在弹出的菜单中选择“Create Subsystem”，则系统将执行封装子系统命令，封装后的结果如图 8-11 所示。

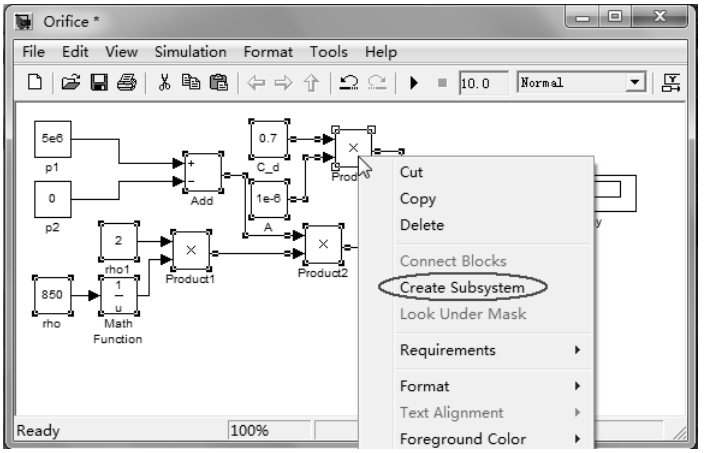


图 8-10　封装子系统菜单

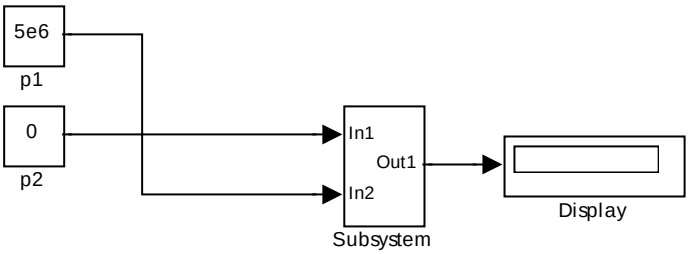


图 8-11　孔口流量公式封装后的效果图

可以修改封装后子模型的名字，以方便管理和记忆。从图 8-11 中可见，此时系统模型相当简单，方便了管理和使用。

3．利用编程方法来仿真节流孔

前面我们讲解了利用模型图标的方式建立孔口流量公式的仿真模型。当面对的模型较复杂时，采用这种方法建立的仿真图形将非常复杂和庞大，并且采用这种方式，对用数学公式描述的模型建模也不够方便。所以，本节采用另外一种方法——编程代码的方式，介绍孔口流量公式的仿真方法。

这种方法主要采用了 Simulink 仿真库“User-Defined Functions”中的“MATLAB Fcn”仿真模块，该模块可以与一个指定的 M 文件函数相关联，这样就可以用一个 M 文件来描述孔口流量数学公式。函数的输入参数为孔口的入口压力和出口压力，函数的返回值为计算得到的出口流量。具体操作步骤如下。

（1）新建一个空白的 Simulink 文件。

（2）在 Simulink 库中，选择“User-Defined Functions”分支，拖动一个“MATLAB Fcn”图标到工作空间中。

（3）双击该图标，弹出如图 8-12 所示界面。其中对我们有用的是第一、二个文本框。仔细阅读对话框上的提示文字，可知其中第一个文本框设定了与“MATLAB Fcn”相关联的 M 文件函数名和输入参数列表，如果有多个参数，用“u(1),u(2)...”这样的形式表示。第二个文本框指定了输出参数的个数（维数）。

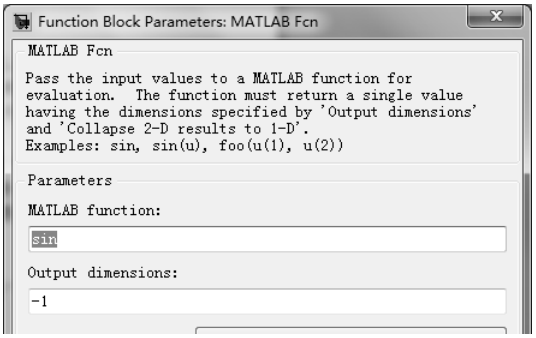


图 8-12 “MATLAB Fcn”模块参数设置界面

此时仍然需要用户思考如何设计孔口流量公式的仿真模型。观察孔口流量公式（8.1），可知其输入参数为 2 个，则“MATLAB function”文本框中应该填入“OrificeFcn(u(1),u(2))”。我们设计孔口流量公式仿真模型输出的结果为入口和出口的流量，其中出口的流量为正值，而入口的流量为负值，则“Output dimensions”文本框中应输入“2”。

（4）拖动“Commonly Used Blocks”中的“Demux”和“Mux”图标到工作空间中，如图 8-13 所示。

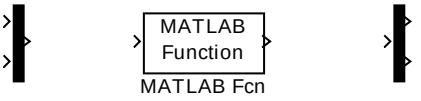


图 8-13　添加了“Demux”和“Mux”图标后的仿真模型

很容易能够理解到，这两个图标添加的目的是为了实现“MATLAB Fcn”仿真模块的多输入、多输出功能（本例是 2 输入、2 输出）。

（5）下面就是根据公式描述，编写程序。在 MATLAB 的主界面中单击新建按钮，新建一个 M 文件，输入如下程序代码。

```
function output = OrificeFcn(p1, p2)
    C_d = 0.7;          %孔口流量系数
    A = 1e-6;           %m^2，孔口面积
    detP = p1 - p2;
    rho = 850;          %kg/m^3，液体密度
    q = C_d * A * sqrt((2/rho)*abs(detP));
    output = [-q q];
end
```

从程序中的注释直接就能够读懂程序代码，对比图 8-7，可见程序代码描述的算法更加清晰、直观。

（6）最后为程序添加合适的输入参数，添加适当的输出观测器，并设置必要的参数，如图 8-14 所示。

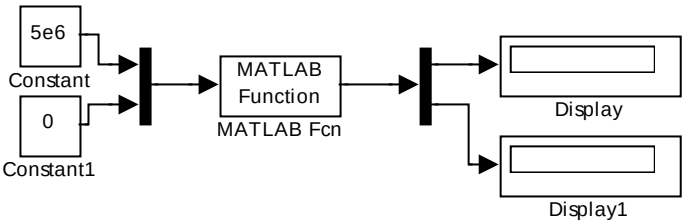


图 8-14　采用“MATLAB Fcn”模块的孔口流量公式仿真回路

此时可以运行仿真，仿真结果如图 8-15 所示。

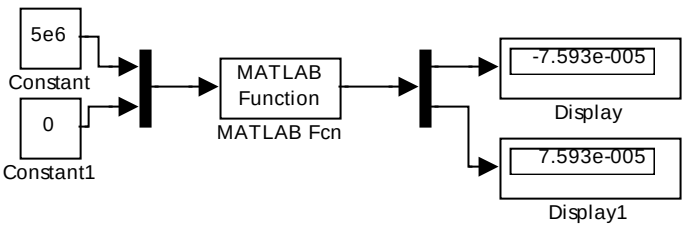


图 8-15　孔口流量公式仿真运行结果

可见图 8-9 和图 8-15 的仿真结果是一致的，这也证明了仿真的正确性。

8.1.2　液压缸节点容腔法建模

双活塞杆液压缸的工作原理模型如图 8-16 所示。要对其进行建模，可以采用节点容腔的方法进行。根据液压缸的结构原理，可以将液压缸拆分成如图 8-17 所示的几个组成部分。

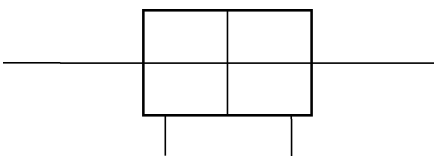


图 8-16　双活塞杆液压缸的工作原理模型

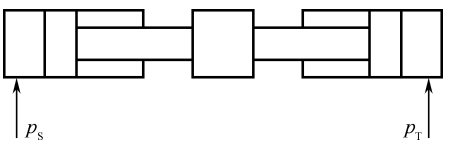


图 8-17　双活塞杆液压缸功率守恒模型

可以将液压缸模型进一步地拆分成液压缸 A 腔、液压缸 B 腔和质量负载，如图 8-18 所示。

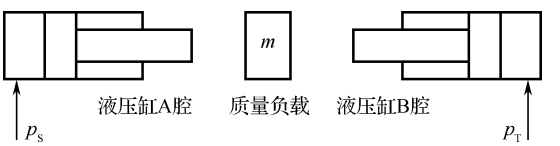


图 8-18　按功率守恒模型所拆分成的元件

这时，就可以对每一个部分分别进行建模，然后再组合成一个整体，就可以仿真液压缸了。下面我们来对每一个部分分别建模。

1．质量负载

根据牛顿第二定律，有

$$F = ma = m\dot{v} \tag{8.3}$$

则可以得到

$$v = \frac{1}{m}\int_{t_0}^{t} F\mathrm{d}t \tag{8.4}$$

图 8.19 中箭头的方向代表了输入参数和输出参数。事实上，对本例来说，也可以将速度 v 设定为输入参数。但是，这将导致对速度计算微分而求得输出力，这是我们应当尽力避免的。

2．液压缸 A 腔

根据帕斯卡原理，我们有

$$F = pA \tag{8.5}$$

根据流量连续性方程，我们有

$$q = Av \tag{8.6}$$

按照前述规则，图 8-20 中箭头指向外的为输出变量；箭头指向内的为输入变量。

液压缸 B 腔的建模方法也是类似的，这里省略。

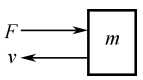


图 8-19　质量负载的功率守恒原理

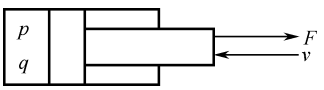


图 8-20　液压缸 A 腔的功率守恒原理

3．节点容腔

根据体积压缩系数的定义有

$$\beta_e = -\frac{1}{V_0}\frac{\Delta V}{\mathrm{d}p} = -\frac{1}{V_0}\frac{A\mathrm{d}y}{\mathrm{d}p} = -\frac{1}{V_0}\frac{q(t)}{\mathrm{d}p} \tag{8.7}$$

用体积弹性模量来表示，则有

$$p = -\int_{t_0}^{t}\frac{Kq(t)}{V_0}\mathrm{d}t + p_0 \tag{8.8}$$

根据上式，由输入流量就可以计算出输出压力了。

从上面的公式可以看出，要对节点容腔进行建模，在 MATLAB 的 Simulink 中，是可以定义积分的初值的，就是在积分元件的 initial condition 属性框中写入初值。

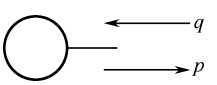


图 8-21　节点容腔

有了上述子模型，就可以利用节点容腔法建立液压缸的仿真模型了。其具体建立方法参考 8.1.1 节。

8.2　常用液压元件建模

液压元件组成了液压系统，因此要对液压系统进行仿真，首先应该建立典型液压元件的仿真模型，然后就能够对系统进行仿真了。本节将介绍典型液压元件的建模方法。

8.2.1 液压泵的建模

液压泵是一种能量转换装置，它把驱动电动机的机械能转换成输到系统中的油液的压力能，供液压系统使用。在液压系统的仿真中，必然要用到液压源（液压泵），因此有必要对其进行建模。

1．理想泵的模型

理想泵的模型是不考虑泵的容积效率和机械效率的，也就是没有泄漏和摩擦。在实际使用过程中，理想泵有两种形式：一种是恒流泵，另一种是恒压泵。前者对应泵的流量恒定，后者对应泵的压力恒定。

1）恒流泵

事实上，只需用 Simulink 中的一个常数模块就可以代表恒流泵的流量，这个常数模块即代表了这个泵。

2）恒压泵

恒压泵类似于恒流泵，也只需用 Simulink 中的一个常数模块来代表就可以了。

理想泵的 Simulink 模型本节省略。

2．实际泵的模型

在工程上，液压泵在能量转换过程中存在容积损失，容积损失是因为内泄露、气穴和油液在高压下的压缩（主要是内泄漏）而造成的流量上的损失。对于液压泵来说，输出压力增大时，泵的实际输出流量 q 会减小。

泵内机件间泄漏的油液流态可以看作为层流，可以认为流量损失 q_l 和泵的输出压力 p 成正比，则

$$q_l = k_l p \tag{8.9}$$

式中 k_l ——流量损失系数（泄漏系数）。

设泵的转速为 n，排量为 V_p，则泵的实际输出流量为

$$q = V_p n - k_l p \tag{8.10}$$

由式(8.10)可见，泵的输出流量是转速、负载压力和排量 3 个输入参数的函数。在 Simulink 中建立泵的仿真模型如图 8-22 所示。

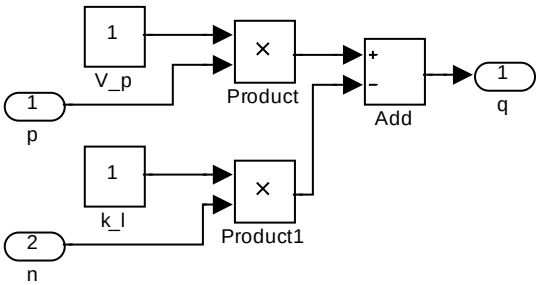


图 8-22 实际泵的仿真模型

3．流量源和节流孔之间的连接

从孔口流量仿真模型［式（8.1）］可见，节流孔口进口、出口的压力（差）是孔口模型的输入参数，而液压泵也以出口压力作为输入，如果直接将两者进行连接，将面临压力没有输入的问题。解决的方法就是加入节点容腔。

8.2.2　节点容腔的仿真模型

节点容腔是仿真中经常用到的模型，可用来解决流量到压力的转换问题。

节点容腔的数学公式参考式（2.2），根据该公式，可以用图形建模法建立仿真草图，如图 8-23 所示。

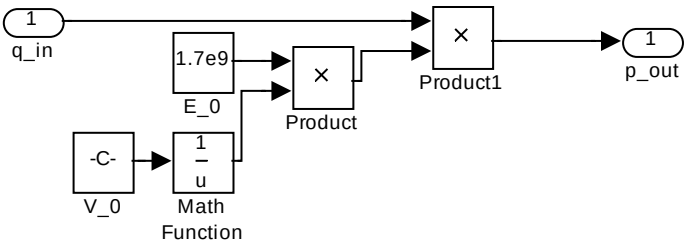


图 8-23　节点容腔的数学模型

如图中所示，“q_in”代表输入的流量，“E_0”代表液压油液的弹性模量，“V_0”代表节点容腔的体积。

在建模的过程中，要有这样的概念，根据帕斯卡原理，连通容器中的压力是处处相等的，所以如果不考虑管道的动态效应，只要管道是连通的，压力就是处处相等的。

流量的数值是可正可负的，流量为正，表明所考虑的对象是流进了流量；流量为负，表明所考虑的对象流出了流量。而流量的正负是可以求和叠加的。

8.2.3　液压缸的仿真模型

液压缸是机液耦合系统模型中最重要的元件。其中单级液压缸结构简单，应用范围广泛。由于液压缸的使用条件多种多样，其构型也各不相同，所以对其建模与仿真也没有统一的形式和方法。本节给出的仿真模型只是一种参考形式，还可能需要根据实际情况进行修改。

依据前几节介绍的仿真方法，要想对特定的液压元件进行建模，只需要列写出其相关的数学公式。液压缸相关的数学公式如下。

$$\begin{aligned} &p_1A_1 - p_2A_2 = ma \\ &q_1 = vA_1 \\ &q_2 = vA_2 \\ &\dot{v} = a \end{aligned} \tag{8.11}$$

如式（8.11）所示，第 1 个公式是液压缸的受力平衡方程式，其中 p_1、p_2 是液压缸两个油口通入的压力；第 2、3 个公式用来计算液压缸输出的流量；第 4 个公式表示速度的微分为加速度。由此可以按图形法建立液压缸的仿真模型草图，如图 8-24 所示。

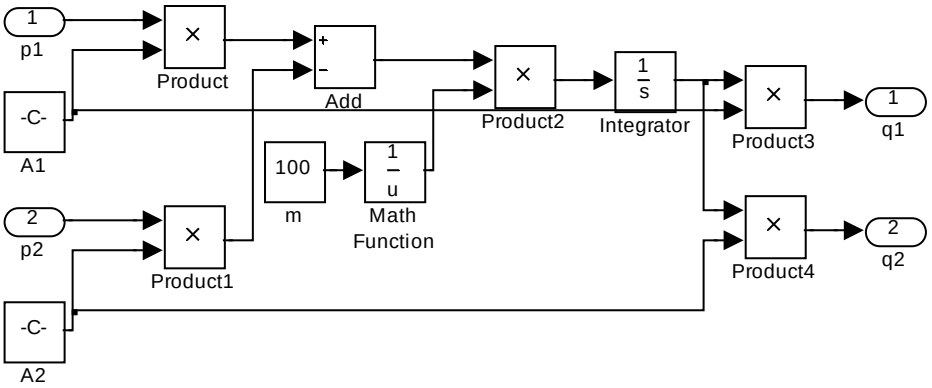


图 8-24 液压缸仿真模型

值得一提的是，在仿真过程中，经常需要反复使用某个参数，比如说重力加速度、油液的密度、油液的黏度、液压缸的面积等，这些参数要在模型中共享使用。当然可以建立这些参数的多个相同值的副本，但这样管理起来较困难，同时模型也显得比较凌乱。事实上，Simulink 中已经提供了两种方法可以实现系统仿真模型参数的共享。方法一是在一个 M 文件中声明 global 全局变量，并赋值。然后在 Simulink 的 M Function 中再次用 global 声明这些变量；方法二是通过菜单“File→Model Properties→Callback→InitFcn”，使用 M 语言定义变量即可。

8.2.4 管道方块图与传递函数

所谓管道的动态特性，是指管道一端液体的流量、压力发生变化时，另一端流量、压力也发生相应变化的情况。实践证明，管道中某处液体的流动参数变化时，管内液流原来的稳定状态便遭到破坏，只有经历一段瞬变过程后，才能达到新的稳定。这个瞬变过程历时虽短，但对整个液压系统来说影响却很大。如系统中常出现的冲击、噪声和振动等往往都与管道振动的谐振频率有关。因此，在设计液压传动系统时，应考虑液压管道的频率特性，它是建立低振级液压系统的必要条件。

分析管道动态特性的方法有分散参数法和集中参数法两种。管道中油的质量、摩擦力及可压缩性等因素是沿着管道分布的。因此对于长管道、高频扰动，必须采用分散参数法才符合实际情况。反之，对于管道较短，输入变量频率较低（压力波主波长比管道长度大得多）的情况，管道内的油柱完全可以看成是一个集中质量，管内摩擦力和油液可压缩性都可并在一起考虑，这时就可用集中参数法进行分析。

在这里，我们仅介绍集中参数分析法。关于分布参数分析法，请参考林国重的《液压传动与控制》。

集中参数分析法的管道模拟图如图 8-25 所示。观察 I-II 断面间长为 l_{T} 的一段短管。设 $t=0$ 时，管道中液流参数与断面 II 上的参数 p_2 、Q_2 相同，设 dt 时间 I 断面上的参数突然变成 p_1、Q_1，则压力波传到 II 断面处，整段管道中的液流参数变为 $p_2+\mathrm{d}p_2$ 、$Q_2+\mathrm{d}Q_2$，以前一段时间内，出口处截面 II 参数仍保持 p_2 、Q_2 不变。如油液的流动状态为层流，则可列出短管油柱的受力平衡方程和液流连续性方程如下。

$$\rho A_{\mathrm{T}} l_{\mathrm{T}} \frac{\mathrm{d}(Q_2 / A_{\mathrm{T}})}{\mathrm{d}t} = (p_1 - p_2)A_{\mathrm{T}} - \frac{128\mu l_{\mathrm{T}}}{\pi d_{\mathrm{T}}^4} Q_2 A_{\mathrm{T}} \tag{8.12}$$

$$Q_1 = Q_2 + \frac{A_{\mathrm{T}} l_{\mathrm{T}}}{K} \frac{\mathrm{d}p_2}{\mathrm{d}t} \tag{8.13}$$

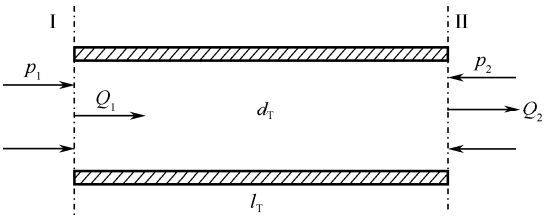


图 8-25　管道的集中参数模型

式中　A_{T}、d_{T} ——管道的截面积和直径；

l_{T} ——管道的长度；

K ——油液等效体积弹性模量；

μ、ρ ——油液的黏度和密度。

令

$$\begin{aligned} R &= 128\mu l_{\mathrm{T}} / \pi d_{\mathrm{T}}^4 \\ L &= \rho l_{\mathrm{T}} / A_{\mathrm{T}} \\ c &= A_{\mathrm{T}} l_{\mathrm{T}} / K \end{aligned} \tag{8.14}$$

则式（8.12）、式（8.13）可写成

$$p_1 = p_2 + RQ_2 + L(\mathrm{d}Q_2 / \mathrm{d}t) \tag{8.15}$$

$$Q_1 = Q_2 + c(\mathrm{d}p_2 / \mathrm{d}t) \tag{8.16}$$

式（8.15）中各项都表示压差，式（8.16）中各项都表示流量。如用压差表示式（8.15）中右边的第二项、第三项，用流量表示式（8.16）右边的第二项，即

$$\begin{aligned} \Delta p &= RQ \\ \Delta p &= L(\mathrm{d}Q / \mathrm{d}t) \\ Q &= c(\mathrm{d}p / \mathrm{d}t) \end{aligned} \tag{8.17}$$

显然上面的压差与流量表达式类似于电学中的电压与电流表达式，因此可采用液阻、液感和液容等参量。

液阻：流体流经管道、缝隙、孔口产生压力损失，压力损失与流量之间的关系用液阻 R 表示，即

$$\Delta p = RQ^a \tag{8.18}$$

式中　a ——值为 1～2，它取决于液阻的性质、流体的流动状态和具体的结构参数。

当流体流经圆管并为层流时，有

$$R = \frac{128\mu l}{\pi d^4} \qquad a = 1 \tag{8.19}$$

当流体流经圆管并为紊流时，有

$$R = \lambda\rho(l / d)(Q^2 / 2A^2) \qquad a = 2 \tag{8.20}$$

同理可推导出缝隙、孔口及其他情况下液阻 R 和 a 的值。

液容：在静力学中通常不考虑油液的可压缩性，而在动力学中必须考虑，即当压力变化时，油液的体积也发生变化，压力变化率与体积变化率（流量）的关系，可用液容 C 表示，即

$$\begin{aligned} K &= -\frac{\mathrm{d}p}{\mathrm{d}V}V \\ -\frac{\mathrm{d}V}{\mathrm{d}t} &= Q = \frac{V}{K}\frac{\mathrm{d}p}{\mathrm{d}t} = C\frac{\mathrm{d}p}{\mathrm{d}t} \end{aligned} \tag{8.21}$$

式中，负号表示压力增加时，油液体积减小。液容 C 值大表示压力变化时油液的容积变化率（流量）大。

液感：当导管内液体为非稳定流（流量随时间发生变化），液体流动具有加速度，说明液体压力发生了变化，液体压力增量与流量变化率之比称为液感 L，即

$$L = \frac{\Delta p}{\mathrm{d}Q/\mathrm{d}t} \tag{8.22}$$

根据力学第二定律，可求得导管液感的具体数值为

$$\begin{aligned} &\frac{F}{A} = \Delta p = \frac{ma}{A_{\mathrm{T}}} = \frac{\rho A_{\mathrm{T}} l_{\mathrm{T}}}{A_{\mathrm{T}}}\frac{\mathrm{d}v}{\mathrm{d}t} = \frac{\rho l_{\mathrm{T}}}{A_{\mathrm{T}}}\frac{\mathrm{d}Q}{\mathrm{d}t} = L\frac{\mathrm{d}Q}{\mathrm{d}t} \\ &L = \rho l_{\mathrm{T}} / A_{\mathrm{T}} \end{aligned} \tag{8.23}$$

由式（8.23）可知，管道越长，液感越大，说明在动态过程中，由于惯性而产生的压力降越大。

1．管道方块图

将式（8.15）与式（8.16）取增量并进行拉氏变换得

$$\begin{aligned} P_1(s) &= A_{\mathrm{T}}(s)P_2(s) + B_{\mathrm{T}}(s)Q_2(s) \\ Q_1(s) &= C_{\mathrm{T}}(s)P_2(s) + D_{\mathrm{T}}(s)Q_2(s) \end{aligned} \tag{8.24}$$

式中

$$\begin{aligned} &A_{\mathrm{T}}(s) = 1, B_{\mathrm{T}}(s) = R + Ls \\ &C_{\mathrm{T}}(s) = Cs, D_{\mathrm{T}}(s) = 1 \end{aligned} \tag{8.25}$$

将式（8.24）写成矩阵形式为

$$\begin{bmatrix} P_1(s) \\ Q_1(s) \end{bmatrix} = G_{\mathrm{T}}(s)\begin{bmatrix} P_2(s) \\ Q_2(s) \end{bmatrix} \tag{8.26}$$

式中 $G_{\mathrm{T}}(s)$——管道传递矩阵，$G_{\mathrm{T}}(s) = \begin{bmatrix} A_{\mathrm{T}}(s) & B_{\mathrm{T}}(s) \\ C_{\mathrm{T}}(s) & D_{\mathrm{T}}(s) \end{bmatrix}$，矩阵中的各元素称为传递矩阵元素。

根据式（8.24）绘制管道方块图如图 8-26 所示。

2．管道传递函数

根据研究和计算任务的不同，由图 8-26、式（8.24）可以求出不同形式的传递函数。计算和研究液压系统的动态特性可以按照两种扰动形式进行：作用在机床工作机构上的外力及由于泵的供油脉动所引起的流量周期性变化。在外载荷扰动下进行研究时，需要进油管和回油管道的传递函数。对于液压系统的进油管道，其输入扰动是出口处流量的变化 $Q_2(s)$，而输出量是同一点处压力的变化 $P_2(s)$。因此，压力管道的传递函数为

$$W_1(s) = \frac{P_2(s)}{Q_2(s)} = \frac{B_{\mathrm{T}}(s) - D_{\mathrm{T}}(s)P_1(s)/Q_1(s)}{-A_{\mathrm{T}}(s) + C_{\mathrm{T}}(s)P_1(s)/Q_1(s)} \tag{8.27}$$

式中　$P_1(s)/Q_1(s)$ ——管道进口处元件的传递函数。

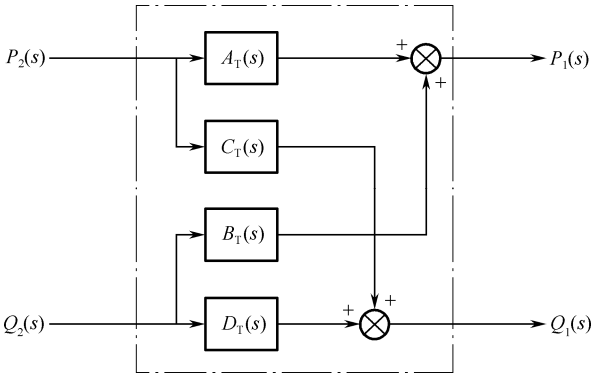


图 8-26　管道方块图

对于回油管道，其输入扰动为回油管道起点处（即液压缸回油腔出口处）流量变化 $Q_1(s)$，而输出量为同一点处压力变化 $P_1(s)$，其传递函数为

$$W_2(s)=\frac{P_1(s)}{Q_1(s)}=\frac{A_T(s)P_2(s)/Q_2(s)+B_T(s)}{C_T(s)P_2(s)/Q_2(s)+D_T(s)} \tag{8.28}$$

式中　$P_2(s)/Q_2(s)$ ——回油管道出口处元件的传递函数。

建立了上述传递函数模型后，就可以应用前面章节介绍的知识，用 MATLAB 或 Simulink 来仿真上述管道系统。事实上，如果不做特殊说明，本节下面介绍的液压元件的建模方法，都是基于传递函数模型来开展的。

8.2.5　限压式变量泵的动态特性

在 8.2.1 节，我们已经建立了液压泵的模型，但是读者们请注意，我们在那一节所建立的模型是液压泵的静态模型，而本节我们建立的是液压泵的动态模型。

动态模型仿真的是液压泵的动态特性，所谓液压泵的动态特性，指的是泵输出流量的瞬时变化引起输出压力的瞬时变化。实践和理论分析证明，如果忽略泵的固有流量脉动的影响，对于定量泵来说，它的动态特性主要取决于泵的内外泄漏及泵压油腔油液的可压缩性。对于变量泵来说，动态特性在很大程度上还取决于具体变量机构的调节特性。

下面我们以限压式变量叶片泵为例，建模仿真其动态特性。

限压式变量叶片泵的工作原理简图如图 8-27 所示。限压式变量叶片泵的工作压力大于其拐点压力时，压力的任何变化都将通过定子偏心距的改变影响输出流量。但是，由于惯性和阻尼的存在，定子不能对压力变化及时作出响应（偏心距不能立即改变），因此泵内会出现一个瞬时的压力急剧变化，要经历一段时间，工作压力才会重新稳定下来。

变量叶片泵的连续性方程为

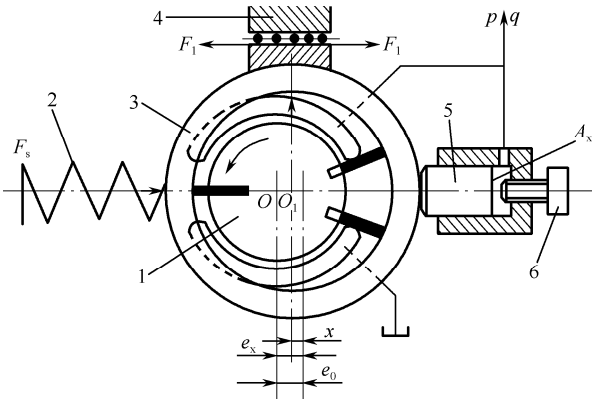
$$q_t=q+k_1p+\frac{V}{K}\frac{\mathrm{d}p}{\mathrm{d}t}+A_x\frac{\mathrm{d}x}{\mathrm{d}t} \tag{8.29}$$

式中　k_1 ——泵的泄漏系数；

V ——泵的压油腔容积；

K ——油液的体积弹性模量；

A_x ——为反馈柱塞面积。



1—转子；2—弹簧；3—定子；4—滑块滚针支撑；5—反馈柱塞；6—流量调节螺钉

图 8-27 限压式变量叶片泵的工作原理简图

由图 8-27 知，泵的理论流量为

$$q_t = k_q e = k_q(e_{max} - e_0 - x) \tag{8.30}$$

式中 k_q ——泵的流量常数；

e_0、e_{max} ——偏心预调值和最大偏心距；

x ——定子自其最右端位置算起的左移距离。

将式（8.30）代入式（8.29）左端，两侧取拉氏变换，得

$$Q(s) = -(k_q + A_x s)X(s) - \left(k_l + \frac{V}{K}s\right)P(s) \tag{8.31}$$

当不计滑块在支撑处的摩擦力时，定子的受力方程为

$$pA_x = F_0 + k_s x + B\frac{\mathrm{d}x}{\mathrm{d}t} + m\frac{\mathrm{d}^2 x}{\mathrm{d}t^2} \tag{8.32}$$

式中 k_s ——弹簧刚度；

B ——泵的黏性阻尼系数；

m ——移动部分（包括定子、反馈柱塞等）的质量。

对式（8.32）两侧取拉氏变换，得

$$(ms^2 + Bs + k_s)X(s) = A_x P(s) \tag{8.33}$$

则变量叶片泵的传递函数模型为

$$\begin{aligned} Q(s) &= -(k_q + A_x s)X(s) - \left(k_l + \frac{V}{K}s\right)P(s) \\ (ms^2 + Bs + k_s)X(s) &= A_x P(s) \end{aligned} \tag{8.34}$$

将方程中的 $X(s)$ 消去，可得变量叶片泵的传递函数为

$$\begin{aligned} G(s) = \frac{P(s)}{Q(s)} &= \frac{-(ms^2 + Bs + k_s)K}{KA_x(k_q + A_x s) + (Kk_l + Vs)(ms^2 + Bs + k_s)} \\ &= \frac{-(ms^2 + Bs + k_s)K}{Vms^3 + (VB + Kk_l m)s^2 + (Kk_l B + KA_x^2 + Vk_s)s + (KA_x k_q + Kk_l k_s)} \end{aligned} \tag{8.35}$$

根据式（8.34）可画出变量叶片泵的传递函数方框图，如图 8-28 所示。

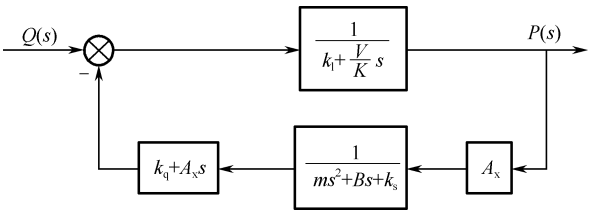


图 8-28　限压式变量叶片泵的传递函数方框图

式（8.35）表明，限压式变量叶片泵在其变量区段内是一个三阶系统。由 Routh 判据可知，这时使泵工作稳定的必要条件是（就用正常的 Routh 判据可以判断）

$$\begin{array}{c|cc} s^3 & Vm & Kk_lB+KA_x^2+Vk_s \\ s^2 & VB+Kk_lm & KA_xk_q+Kk_lk_s \\ s^1 & -\dfrac{1}{VB+Kk_lm}\begin{vmatrix} Vm & Kk_lB+KA_x^2+Vk_s \\ VB+Kk_lm & KA_xk_q+Kk_lk_s \end{vmatrix} & 0 \end{array}$$

可求得使泵稳定工作的必要条件为

$$k_s > \frac{KA_xk_qVm-VB^2Kk_l-K^2k_l^2Bm-V^2Bk_s+A_x^2K(VB+Kk_lm)}{V^2B} \tag{8.36}$$

上式表明，限压式变量叶片泵中的调压弹簧不但影响泵的静态特性，还影响其动态特性。为此，设计中必须使这个 k_s 值满足式（8.36）的要求。从式（8.36）可以看出，黏性系数 B 大则稳定性好，一般可在反馈柱塞缸入口处设置阻尼小孔，以提高 B 值。

8.2.6　液压缸的动态特性仿真

在 8.1.2 节中，我们采用节点容腔法，对液压缸系统进行了建模。事实上，还可以应用传递函数法，对液压缸系统进行建模。

液压缸工作时，当推力与负载相等，且输入油液流量恒定时，缸筒或活塞进行匀速运动；液压缸在输入流量不变、负载发生变化或负载不变、输入流量发生变化时，活塞或缸筒的运动就会出现加速或减速的瞬态过程。液压缸的动态特性就是对瞬态过程中这些变化关系的说明。

下面以一推动负载运动的液压缸为例，其原理图如图 8-29 所示。

液压缸上的受力方程为

$$Ap = m\frac{\mathrm{d}v}{\mathrm{d}t} + Bv + F_L \tag{8.37}$$

式中　A ——活塞有效工作面积；

p ——液压缸工作腔压力；

m ——液压缸所驱动的工作部件质量（包括活塞、活塞杆等移动部件质量在内）；

B ——黏性阻尼系数；

v ——活塞运动速度；

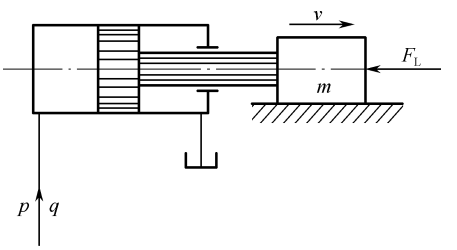


图 8-29　液压缸原理图

F_L ——外负载力。

液压缸工作腔的流量连续性方程为

$$q = Av + k_l p + \frac{V}{K}\frac{\mathrm{d}p}{\mathrm{d}t}\frac{1}{2} \tag{8.38}$$

式中 k_l ——液压缸工作腔的泄漏系数；

V ——液压缸工作腔和进油管内的油液体积；

K ——油液的体积弹性模量。

将上两式进行拉氏变换，有

$$AP(s) = (ms + B)V(s) + F_L(s) \tag{8.39}$$

$$Q(s) = AV(s) + \left(k_l + \frac{V}{K}s\right)P(s) \tag{8.40}$$

利用上两式可作出液压缸的动态结构框图，如图 8-30 所示。

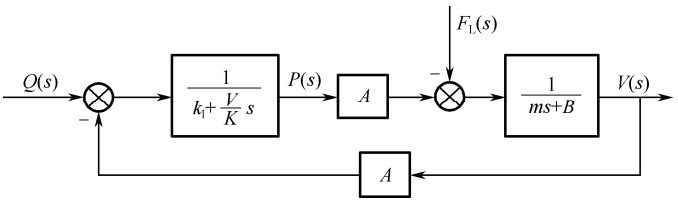


图 8-30 液压缸的动态结构方框图

将式（8.39）和式（8.40）联合，得传递函数表达式为

$$\begin{aligned} V(s) &= \frac{AQ(s) - \left(k_l + \frac{V}{K}s\right)F_L(s)}{\frac{V}{K}ms^2 + \left(k_l + \frac{V}{K}B\right)s + \left(A^2 + k_l B\right)} \\ &= \frac{1}{A^2 + k_l B}\frac{AQ(s) - \left(k_l + \frac{V}{K}s\right)F_L(s)}{\frac{s^2}{\omega_n^2} + \frac{2\xi}{\omega_n}s + 1} \end{aligned} \tag{8.41}$$

外负载 F_L 恒定时的液压缸传递函数为

$$G(s) = \frac{V(s)}{Q(s)} = \left(\frac{A}{A^2 + k_l B}\right)\frac{1}{\left(\frac{s}{\omega_n}\right)^2 + 2\frac{\xi}{\omega_n}s + 1} \tag{8.42}$$

输入流量 q 恒定时的液压缸传递函数为

$$G_2(s) = \frac{V(s)}{F_L(s)} = \left(\frac{-1}{A^2 + k_l B}\right)\frac{k_l + \frac{V}{K}s}{\left(\frac{s}{\omega_n}\right)^2 + 2\frac{\xi}{\omega_n}s + 1} \tag{8.43}$$

以上三式中的 ω_n 和 ξ_n 分别代表液压缸的固有角频率和阻尼比，其表达式为

$$\omega_n = \sqrt{\frac{(A^2 + k_l B)K}{Vm}} \tag{8.44}$$

$$\xi_{\mathrm{n}}=\frac{\omega_{\mathrm{n}}}{2K}\frac{Kk_{\mathrm{l}}m+VB}{A^2+k_{\mathrm{l}}B} \tag{8.45}$$

由以上的一些图和公式可以看到：

（1）带管道的液压缸可以简化成一个二阶系统，它的特征方程中的系数都是正值，因此一般说来它是能够稳定工作的。

（2）液压缸进油腔和进油管中的泄漏通常是很小的，即 $k_{\mathrm{l}}B/A^2\ll 1$，所以式（8.44）中的 ω_{n} 可以近似地用 $\sqrt{A^2K/(Vm)}$ 来表示。这就是说，油液的体积弹性模量 K 越小（油中混入空气越多），活塞有效工作面积 A 越小，液压缸移动时推动的质量越大，进油管越长（V 越大），液压缸的固有角频率 ω_{n} 就越低。同时，活塞移动过程中 V 值也在不断地变化，因此 ω_{n} 不是一个定值，而是一段频率范围，液压缸的频率特性曲线也是随着活塞的移动而变化的。

现从两个方面来讨论液压缸的瞬态响应特性：①负载恒定，输入流量变化时（例如液压缸由静止状态启动，或流入流量突然变化），液压缸的运动速度会产生波动；②输入流量恒定，外负载突然增加或减少时也会使液压缸产生速度不稳定。这两方面的理论分析分别利用式（8.42）和式（8.43）进行。

由式（8.42），在外负载不变情况下，如果对液压缸输入一阶跃流量 $\Delta q=q_0$，即 $Q(s)=\dfrac{q_0}{s}$，则得

$$\begin{aligned}V(s)&=G(s)Q(s)=G(s)\frac{q_0}{s}\\&=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\frac{\omega_{\mathrm{n}}^2}{s(s^2+2\xi_{\mathrm{n}}\omega_{\mathrm{n}}+\omega_{\mathrm{n}}^2)}\end{aligned} \tag{8.46}$$

即

$$\begin{aligned}V(s)&=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\frac{\omega_{\mathrm{n}}^2}{s(s^2+2\xi_{\mathrm{n}}\omega_{\mathrm{n}}s+\xi_{\mathrm{n}}^2\omega_{\mathrm{n}}^2+\omega_{\mathrm{n}}^2-\xi_{\mathrm{n}}^2\omega_{\mathrm{n}}^2)}\\&=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\left(\frac{\omega_{\mathrm{n}}^2}{s\left((s+\xi\omega_{\mathrm{n}})^2+\left(\omega_{\mathrm{n}}\sqrt{1-\xi_{\mathrm{n}}^2}\right)^2\right)}\right)\\&=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\left(\frac{A_{\mathrm{l}}}{s}+\frac{Ms+N}{(s+\xi_{\mathrm{n}}\omega_{\mathrm{n}})^2+\left(\omega_{\mathrm{n}}\sqrt{1-\xi_{\mathrm{n}}^2}\right)^2}\right)\\&=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\left(\frac{1}{s}+\frac{-s-\xi_{\mathrm{n}}\omega_{\mathrm{n}}-\xi_{\mathrm{n}}\omega_{\mathrm{n}}}{(s+\xi_{\mathrm{n}}\omega_{\mathrm{n}})^2+\left(\omega_{\mathrm{n}}\sqrt{1-\xi_{\mathrm{n}}^2}\right)^2}\right)\end{aligned} \tag{8.47}$$

对上式取拉氏反变换，有

$$\Delta v=\frac{Aq_0}{A^2+k_{\mathrm{l}}B}\left[1-\frac{1}{\sqrt{1-\xi_{\mathrm{n}}^2}}\mathrm{e}^{-\xi_{\mathrm{n}}\omega_{\mathrm{n}}t}\sin\left(\omega_{\mathrm{n}}\sqrt{1-\xi_{\mathrm{n}}^2}t+\arctan\frac{\sqrt{1-\xi_{\mathrm{n}}^2}}{\xi_{\mathrm{n}}}\right)\right] \tag{8.48}$$

其特性如图 8-31（a）所示。从图中可以看出，速度 Δv 围绕稳态值 Δv_0 上下波动，并逐渐

衰减趋向于稳态值。阻尼比 ξ_n 如较大，则波动较小。根据式（8.48），当 $t=0$ 时，$\Delta v=0$；$t\to\infty$，$\Delta v=\Delta v_0=\dfrac{Aq_0}{A^2+k_1B}$。

由式（8.43）可知，当输入流量恒定时，如果作用在活塞杆上的外负载突然减少了 F_{L0}，即 $\Delta F_L=-F_{L0}$，$F_L(s)=-\dfrac{F_{L0}}{s}$，则有

$$V(s)=G_2(s)F_L(s)=-G(s)\frac{F_{L0}}{s} \tag{8.49}$$

即

$$V(s)=\frac{F_{L0}k_1}{A^2+k_1B}\frac{\omega_n^2}{s(s^2+2\xi_n\omega_n s+\omega_n^2)}+\frac{F_{L0}}{A^2+k_1B}\frac{V}{K}\frac{\omega_n^2}{s^2+2\xi_n\omega_n s+\omega_n^2} \tag{8.50}$$

利用拉氏反变换，得

$$\begin{aligned}\Delta v=&\frac{F_{L0}k_1}{A^2+k_1B}\left[1-\frac{1}{\sqrt{1-\xi_n^2}}e^{-\xi_n\omega_n t}\sin\left(\omega_n\sqrt{1-\xi_n^2}\,t+\arctan\frac{\sqrt{1-\xi_n^2}}{\xi_n^2}\right)\right]\\&+\frac{F_{L0}}{A^2+k_1B}\frac{V}{K}\frac{\omega_n}{\sqrt{1-\xi_n^2}}e^{-\xi_n\omega_n t}\sin\omega_n\sqrt{1-\xi_n^2}\,t\end{aligned} \tag{8.51}$$

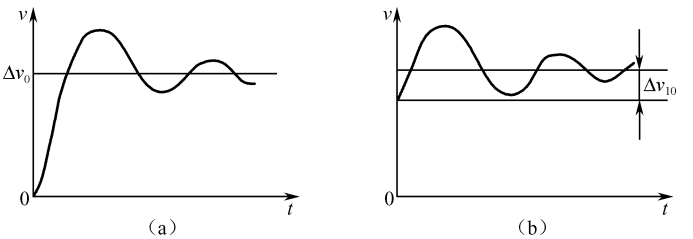


图 8-31　液压缸瞬态响应特性分析图

其特性如图 8-31（b）所示。当负载突然减小时，液压缸的速度突然增加，产生所谓前冲现象。以后又产生速度波动，逐步衰减趋近于新的稳态值。由于负载减小，系统泄漏减小，速度增大了 Δv_{10}。由式（8.51）可知，当 $t=0$ 时，$\Delta v=0$；当 $t\to\infty$ 时，$\Delta v=\Delta v_{10}=\dfrac{k_1F_{L0}}{A^2+k_1B}$。

如果液压缸的泄漏可以忽略不计，$k_1=0$，式（8.51）可简化为

$$\Delta v=\frac{F_{L0}V}{A^2K}\frac{\omega_n}{\sqrt{1-\xi_n^2}}e^{-\xi_n\omega_n t}\sin\omega_n\sqrt{1-\xi_n^2}\,t \tag{8.52}$$

上式中，$t\to\infty$ 时，$\Delta v=0$，也就是速度经波动后仍回复到原来的稳态值。

8.2.7　液压泵-蓄能器组合的动态特性

当蓄能器通过一段管道连接在液压泵输出管道的支路上时，它能减弱（吸收）液压泵出口处的压力脉动，脉动被减弱的程度视蓄能器容量的大小而定。下面讨论图 8-32 所示液压泵-蓄能器组合的动态作用情况。为了简化分析起见，这里假定所有的连接管道都较短，可以用集中参数法进行处理；且液压泵输出管道的液阻可以用 R 表示。

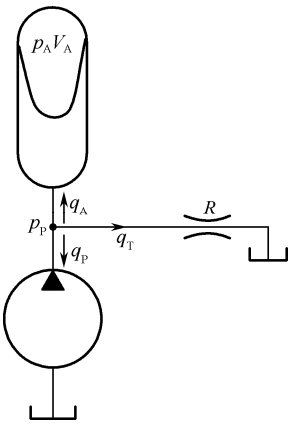


图 8-32　液压泵–蓄能器组合

液压泵输出管道分支点处的流量连续性方程如下。

$$q_p = q_A + q_T \tag{8.53}$$

式中　q_p ——液压泵输出流量；

q_A ——进入蓄能器的瞬时流量；

q_T ——通过输出管的流量。

式（8.53）取增量、进行拉氏变换后可写成

$$Q_p(s) = Q_A(s) + Q_T(s) \tag{8.54}$$

对液压泵的输出管道来说，按图示情况有

$$p_p = Rq_T \tag{8.55}$$

式中　p_p ——液压泵的输出压力，亦即是管道分支点处的压力；

R ——液阻。

由此得

$$P_p(s) = RQ_T(s) \tag{8.56}$$

对蓄能器的连接短管来说，受力平衡方程为

$$(p_p - p_A)A = \rho l \frac{\mathrm{d}q_A}{\mathrm{d}t} + R_A q_A A = \rho l A \frac{\mathrm{d}v}{\mathrm{d}t} + R_A q_A A \tag{8.57}$$

式中　p_A ——蓄能器内的气体压力；

ρ ——油液密度；

l ——短管长度；

A ——短管截面积；

R_A ——短管液阻。

蓄能器入口处的流量连续性方程为

$$q_A = \kappa_A V_A \frac{\mathrm{d}p_A}{\mathrm{d}t} \tag{8.58}$$

式中　κ_A ——气体的压缩系数，当蓄能器内气体的稳定压力为 p_{A0}，气体定律中的指数为 n 时，$\kappa_A = 1/(np_{A0})$；

V_A ——蓄能器内气体体积。

将式（8.58）代入式（8.57）中，并整理，有

$$p_{\mathrm{p}}=\frac{\rho l\kappa_{\mathrm{A}}V_{\mathrm{A}}}{A}\frac{\mathrm{d}^2p_{\mathrm{A}}}{\mathrm{d}t^2}+R_{\mathrm{A}}\kappa_{\mathrm{A}}V_{\mathrm{A}}\frac{\mathrm{d}p_{\mathrm{A}}}{\mathrm{d}t}+p_{\mathrm{A}} \tag{8.59}$$

对上式两侧取增量、进行拉氏变换，代入整理并用蓄能器的固有角频率 ω_{A} 和阻尼比 ξ_{A} 来表达时，可得

$$P_{\mathrm{p}}(s)=P_{\mathrm{A}}(s)\left[\left(\frac{s}{\omega_{\mathrm{A}}}\right)^2+2\xi_{\mathrm{A}}\left(\frac{s}{\omega_{\mathrm{A}}}\right)+1\right] \tag{8.60}$$

式中

$$\omega_{\mathrm{A}}=\sqrt{\frac{A}{\rho l\kappa_{\mathrm{A}}V_{\mathrm{A}}}} \tag{8.61}$$

$$\xi_{\mathrm{A}}=\frac{R_{\mathrm{A}}}{2}\sqrt{\frac{\kappa_{\mathrm{A}}V_{\mathrm{A}}A}{\rho l}} \tag{8.62}$$

由式（8.58），得

$$Q_{\mathrm{A}}(s)=\kappa_{\mathrm{A}}V_{\mathrm{A}}sP_{\mathrm{A}}(s)=\frac{s}{R\omega_{\mathrm{c}}}P_{\mathrm{A}}(s) \tag{8.63}$$

式中 ω_{c} ——转折角频率，$\omega_{\mathrm{c}}=1/(R\kappa_{\mathrm{A}}V_{\mathrm{A}})=(q_{\mathrm{p0}})/(p_{\mathrm{p0}}\kappa_{\mathrm{A}}V_{\mathrm{A}})$，在这里 q_{p0} 和 p_{p0} 为 q_{p} 和 p_{p} 的稳态值。

由式（8.54）、式（8.56）、式（8.60）和式（8.63）可作出液压泵-蓄能器组合的动态结构框图，如图 8-33 所示。

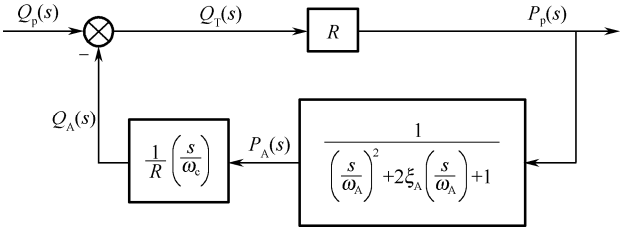


图 8-33 液压泵-蓄能器组合的动态结构框图

得出的传递函数式为

$$\boldsymbol{\Phi}(s)=\frac{P_{\mathrm{p}}(s)}{Q_{\mathrm{p}}(s)}=R\left[\frac{\left(\frac{s}{\omega_{\mathrm{A}}}\right)^2+2\xi_{\mathrm{A}}\left(\frac{s}{\omega_{\mathrm{A}}}\right)+1}{\left(\frac{s}{\omega_{\mathrm{A}}}\right)^2+\left(\frac{2\xi_{\mathrm{A}}}{\omega_{\mathrm{A}}}+\frac{1}{\omega_{\mathrm{c}}}\right)s+1}\right] \tag{8.64}$$

于是可求得这种液压泵-蓄能器组合的频率特性的模为

$$|\boldsymbol{\Phi}(\mathrm{j}\omega)|=\left|\frac{P_{\mathrm{p}}(\mathrm{j}\omega)}{Q_{\mathrm{p}}(\mathrm{j}\omega)}\right|=R\sqrt{\frac{\left(1-\frac{\omega^2}{\omega_{\mathrm{A}}^2}\right)^2+\left(\frac{2\xi_{\mathrm{A}}\omega}{\omega_{\mathrm{A}}}\right)^2}{\left(1-\frac{\omega^2}{\omega_{\mathrm{A}}^2}\right)^2+\left(\frac{2\xi_{\mathrm{A}}\omega}{\omega_{\mathrm{A}}}+\frac{\omega}{\omega_{\mathrm{c}}}\right)^2}} \tag{8.65}$$

此值就是液压泵输出管道分支点处压力脉动与流量脉动之比。它与 ω 的关系如图 8-34

所示。

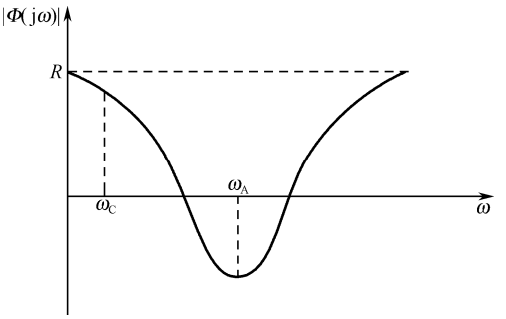


图 8-34　蓄能器吸收脉动时的特性曲线

由图中可见，当 $\omega<\omega_{\mathrm{e}}$ 时，蓄能器对吸收脉动几乎没有什么作用，这时的 $|\boldsymbol{\Phi}(\mathrm{j}\omega)|\approx R$，当 $\omega=\omega_{\mathrm{A}}$ 时，$|\boldsymbol{\Phi}(\mathrm{j}\omega)|$ 有最小值，即

$$|\boldsymbol{\Phi}(\mathrm{j}\omega)|_{\min}=\frac{2\xi_{\mathrm{A}}R}{2\xi_{\mathrm{A}}+\dfrac{\omega_{\mathrm{A}}}{\omega_{\mathrm{e}}}}\approx\frac{2\xi_{\mathrm{A}}R\omega_{\mathrm{e}}}{\omega_{\mathrm{A}}} \tag{8.66}$$

将式（8.61）、式（8.62）代入式（8.66）中，得

$$|\boldsymbol{\Phi}(\mathrm{j}\omega)|_{\min}\approx\frac{2\xi_{\mathrm{A}}R\omega_{\mathrm{e}}}{\omega_{\mathrm{A}}}=R_{\mathrm{A}} \tag{8.67}$$

这时，蓄能器使液压泵流量脉动在管道分支点处所引起的压力脉动达到最小。短管的液阻 R_{A} 越小，压力脉动的振幅越小，蓄能器吸收脉动的效果也越好。

由此可见，在液压泵压力脉动的角频率 ω_{p} 已知的情况下，正确选择蓄能器的容量 V_{A}、连接短管的结构尺寸 l 和 A，并使按式（8.61）算出的固有角频率 ω_{A} 等于 ω_{p}，就能使蓄能器具有最佳的脉动吸收效果。

8.2.8　带管道的溢流阀的动态特性

在定量泵供油的液压系统中，对系统压力起调节作用的溢流阀总是装在压力管道的分支路上，为此在分析溢流阀的动态特性时，采用了如图 8-35 所示的简图，把压力管道包括进去。图 8-35 所示为直动式溢流阀，简图中把阀内的阻尼孔也画了出来。分析中假定溢流阀弹簧腔内和回油口的压力都为零。

当不计阀芯自重时，阀芯的受力平衡方程为

$$p_{\mathrm{a}}A=m\frac{\mathrm{d}^2x}{\mathrm{d}t^2}+B\frac{\mathrm{d}x}{\mathrm{d}t}+k_0(x_0+x) \tag{8.68}$$

式中　p_{a} ——压力腔 a 中的压力；

k_0 ——包括稳态液动力和弹簧在内的等效弹簧刚度；

x_0 ——弹簧的预压缩量；

x ——阀口开度；

m ——包括阀芯、弹簧和液柱等在内的等效质量；

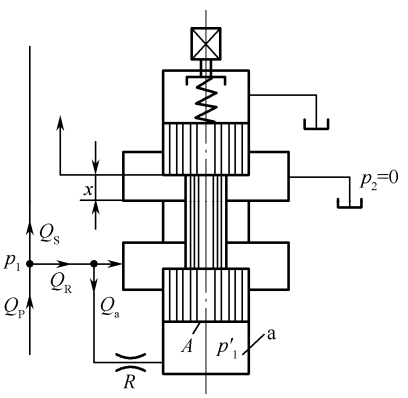


图 8-35　带管道的直动式溢流阀

B ——包括瞬态液动力在内的等效阻尼系数。

对式（8.68）取增量并进行拉氏变换，得

$$P_a(s)A=(ms^2+Bs+k_0)X(s) \tag{8.69}$$

流经阀口的流量 q_l 为

$$q_l=C_d wx\sqrt{\frac{2}{\rho}p_l} \tag{8.70}$$

式中 C_d ——流经阀口的流量系数；

w ——阀口的面积梯度；

p_l ——系统压力；

ρ ——油液的密度。

对式（8.70）进行线性化，并进行拉氏变换，得

$$Q_l(s)=K_{qv}X(s)+K_{cv}P_l(s) \tag{8.71}$$

式中 K_{qv} ——阀的流量增益，$K_{qv}=C_d w\sqrt{\frac{2}{\rho}p_{l0}}$；

K_{cv} ——阀的流量压力系数，$K_{cv}=\frac{C_d wx_0}{\sqrt{2\rho p_{l0}}}$，$p_{l0}$、$x_0$ 为 p_l 和 x 的稳态值。

通过阻尼孔的流量 q_a 为

$$q_a=\frac{p_l-p_a}{R}=A\frac{\mathrm{d}x_R}{\mathrm{d}t} \tag{8.72}$$

式中 R ——阻尼孔液阻；

A ——阀芯面积。

对式（8.72）取增量并进行拉氏变换，得

$$Q_a(s)=\frac{1}{R}[P_l(s)-P_a(s)]=AsX_R(s) \tag{8.73}$$

阀-管道的流量连续性方程为

$$q_p-q_s-k_l p_l-q_a-\frac{V}{K}\frac{\mathrm{d}p_l}{\mathrm{d}t}=q_l \tag{8.74}$$

式中 q_p ——上游来的流量；

q_s ——去下游的流量；

k_l ——泄漏系数；

V ——下游元件和管道内的液体体积；

K ——油液的体积弹性模量。

将式（8.74）取增量并进行拉氏变换，再把式（8.71）和式（8.73）代入，得

$$-Q_s(s)-k_l P_l(s)-\frac{1}{R}[P_l(s)-P_a(s)]-\frac{V}{K}sP_l(s)=K_{qv}X(s)+K_{cv}P_l(s) \tag{8.75}$$

将式（8.73）代入式（8.69）中，消去 $P_a(s)$，得

$$AP_l(s)=[ms^2+(B+A^2R)s+k_a]X(s) \tag{8.76}$$

由式（8.76）可以看出，由于阻尼孔的作用，在阻尼项中，阻尼系数增加了 A^2R，因而也就增加了系统的阻尼比。溢流阀中的阻尼孔具有抑制振荡和提高稳定性的作用。

令 $B_a = B + A^2 R$，式（8.76）可写成

$$AP_l(s) = k_a\left(\frac{m}{k_a}s^2 + \frac{B_a}{k_a}s + 1\right)X(s) = k_a\left(\frac{s^2}{\omega_m^2} + \frac{2\xi_m}{\omega_m}s + 1\right)X(s) \tag{8.77}$$

式中　ω_m ——阀芯无阻尼自然频率，$\omega_m = \sqrt{k_a / m}$；

ξ_m ——阻尼比，$\xi_m = \dfrac{\omega_m}{2}\dfrac{B_e}{k_a}$。

将式（8.73）代入式（8.75）中，消去 $p_a(s)$，得

$$-Q_e(s) - K_{qv}\left(1 + \frac{A}{K_{qv}}s\right)X(s) = K_{ce}\left(1 + \frac{V}{K_{ce}K}s\right)P_l(s) \tag{8.78}$$

式中　K_{ce} —— $K_{ce} = K_{cv} + k_l$。

根据式（8.77）和式（8.78），画成动态结构框图，如图 8-36 所示。

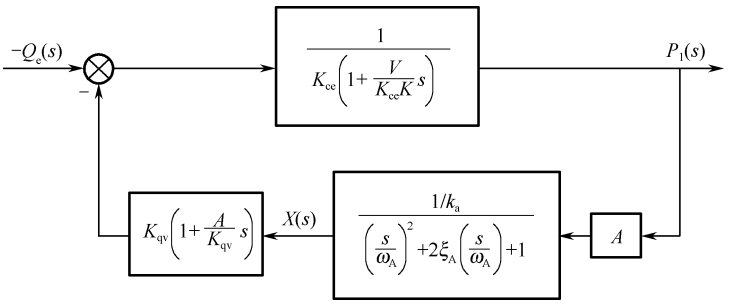


图 8-36　直动式溢流阀的动态结构框图

由式（8.77）和式（8.78）可得，带管道的直动式溢流阀的传递函数为

$$\frac{P_l(s)}{Q_e(s)} = -\frac{\dfrac{s^2}{\omega_m^2} + \dfrac{2\xi_m}{\omega_m}s + 1}{\dfrac{V}{\omega_m K}s^3 + \left(\dfrac{2\xi_m V}{\omega_m K} + \dfrac{K_{ce}}{\omega_m^2}\right)s^2 + \left(\dfrac{V}{K} + \dfrac{2\xi_m K_{ce}}{\omega_m} + \dfrac{A^2}{k_a}\right)s + \left(\dfrac{AK_{qv}}{k_a} + K_{ce}\right)} \tag{8.79}$$

式中，等式右边的“−”号表示输入流量 q_e 增大，则输出压力 p_l 要减小。上式是溢流阀装在液压系统中，以流量为输入，以系统压力为输出的传递函数。即使是简单的直动式溢流阀，其传递函数还是个较复杂的三阶系统。如果不考虑油液压缩性（令 $K = \infty$），系统就可以降为二阶。但这种假设与实际情况差别太大，难以成立。根据式（8.78）的传递函数，运用古典控制理论的工具，可进行稳定性分析和瞬态响应分析，以合理地确定有关参数。

8.3　常用液压回路的建模

在 8.2 节中，我们主要介绍了仿真液压元件动态特性的建模原理，采用的主要方法是传递函数法。在这一节中，我们将以典型系统为例，介绍常用液压回路的建模方法。采用的建模工具主要是 MATLAB 的 Simulink 工具箱。

8.3.1 节主要采用线性化方法对孔口流量公式进行了线性化，然后采用数值方法进行仿真；而 8.3.2 节则采用数值方法对拥有非线性节流孔的节流调速回路进行了仿真，具有一定的代表性。

8.3.1 液压节流调速回路的线性化仿真

1．进油节流调速回路的仿真

进油节流调速回路原理图如图 8-37 所示。

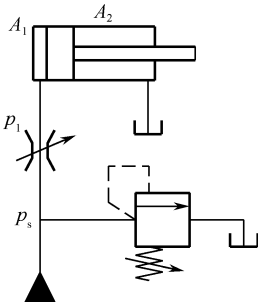


图 8-37 进油节流调速回路原理图

下面列写进油节流调速回路的运动学方程。将 p_1 定义为负载压力，即

$$p_L = p_1 \tag{8.80}$$

通过节流孔的流量方程为

$$Q = q_1 = C_d w x\sqrt{\frac{2}{\rho}(p_s - p_1)} = C_d w x\sqrt{\frac{2}{\rho}(p_s - p_L)} \tag{8.81}$$

式中 C_d ——流量系统，取 0.62；

w ——节流孔开口面积的面积梯度（m）；

x ——节流阀的开口量（m）；

ρ ——液体的密度（kg/m^3）；

p_s ——系统的供油压力（Pa）；

q_1 ——通过节流阀的流量（m^3/s）。

我们将节流孔的流量方程在小范围内进行线性化，有

$$\begin{aligned}\Delta q_1 &= \frac{\partial q_1}{\partial x}\Delta x + \frac{\partial q_1}{\partial p_L}\Delta p_L \\ &= C_d w\sqrt{\frac{2}{\rho}(p_s - p_L)}\Delta x - C_d w x\sqrt{\frac{1}{2\rho(p_s - p_L)}}\Delta p_L\end{aligned} \tag{8.82}$$

去掉增量符号得

$$q_1 = K_q x_v - K_c p_L \tag{8.83}$$

式中 K_q ——滑阀的流量增益；

K_c ——滑阀流量-压力系数。

进油节流调速回路的流量连续性方程为

$$q_1 = A_1 v \tag{8.84}$$

液压缸的运动学方程为

$$p_1 A_1 - F = m\dot{v} \tag{8.85}$$

式中　p_1——液压缸进油腔的压力（Pa）；

A_1——液压缸无杆腔的面积（m^2）；

F——液压缸所受到的负载力（N）；

m——液压缸所推动的负载的质量（kg）；

v——液压缸的运动速度（m/s）。

联立式（8.83）、式（8.84）和式（8.85），可得

$$\begin{aligned} q_1 &= A_1 v \\ q_1 &= K_q x_v - K_c p_L \\ p_L A_1 - F &= m\dot{v} \end{aligned} \tag{8.86}$$

根据式（8.86）可以建立仿真模型，如图 8-38 所示。

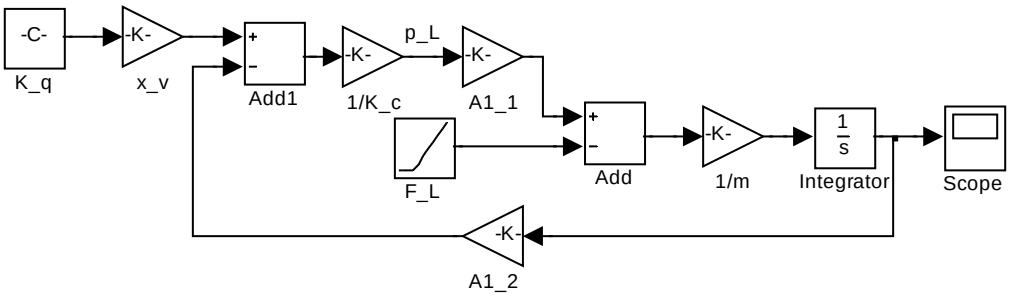


图 8-38　进油节流调速回路线性化仿真模型

2．回油节流调速回路的仿真

回油节流调速回路（见图 8-39）通过节流阀的流量方程式为

$$q_2 = C_d w x \sqrt{\frac{2}{\rho} p_2} \tag{8.87}$$

式中　p_2——节流阀进油口的压力（Pa）；

q_2——通过节流阀的流量（m^3/s）。

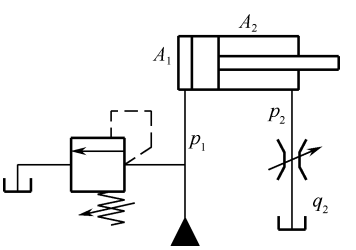


图 8-39　回油节流调速回路原理图

节流阀的线性化流量方程变为

$$\begin{aligned} \Delta q_2 &= \frac{\partial q_2}{\partial x_v}\Delta x_v + \frac{\partial q_2}{\partial p_2}\Delta p_2 \\ &= C_d w \sqrt{\frac{2}{\rho} p_2}\Delta x_v + C_d w x \sqrt{\frac{1}{2\rho p_2}}\Delta p_2 \\ &= K_q \Delta x_v + K_c \Delta p_2 \end{aligned} \tag{8.88}$$

即此时系统的负载压力为

$$p_L = p_2 \tag{8.89}$$

液压缸的运动学方程为

$$p_1 A_1 - p_2 A_2 - F = m\dot{v} \tag{8.90}$$

回油节流调速回路的流量连续性方程为

$$q_2 = A_2 v \tag{8.91}$$

联立上述方程，可得回油节流调速回路的方程组为

$$\begin{aligned} q_2 &= K_q x_v + K_c p_2 \\ q_2 &= A_2 \dot{x} \\ p_1 A_1 - p_2 A_2 - F &= m\ddot{x} \end{aligned} \tag{8.92}$$

将式（8.92）进行拉氏变换，得到

$$\begin{aligned} Q_2 &= K_q X_v + K_c P_2 \\ Q_2 &= A_2 sX(s) \\ P_1 A_1 - P_2 A_2 - F &= ms^2 X(s) \end{aligned} \tag{8.93}$$

根据式（8.93）可以建立用传递函数描述的回油节流调速回路，如图 8-40 所示。

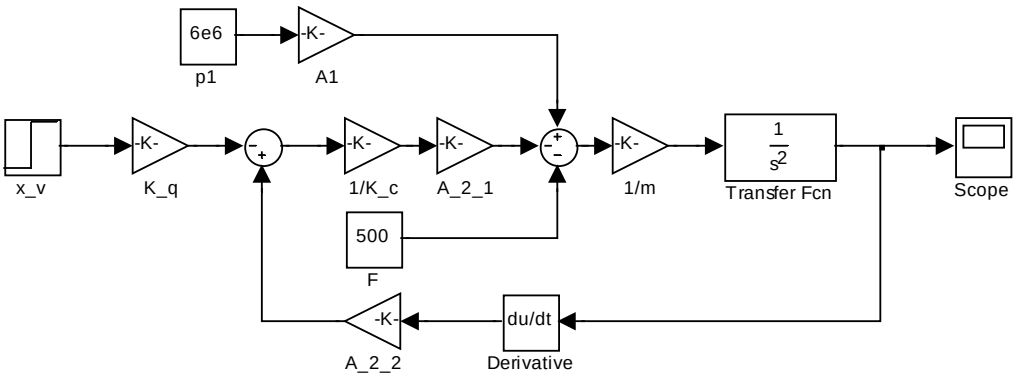


图 8-40　回油节流调速回路线性化仿真模型

3．旁路节流调速回路的仿真

根据旁路节流调速回路原理图（见图 8-41），可得通过节流阀的流量方程式为

$$q_2 = C_d w x \sqrt{\frac{2}{\rho} p_1} \tag{8.94}$$

式中　p_1——节流阀进油口的压力（Pa）；

q_2——通过节流阀的流量（m^3/s）。

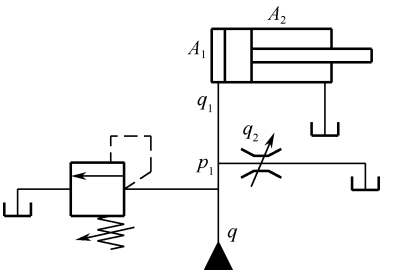


图 8-41　旁路节流调速回路原理图

节流阀的线性化流量方程变为

$$\begin{aligned}\Delta q_2 &= \frac{\partial q_2}{\partial x_v}\Delta x_v + \frac{\partial q_2}{\partial p_2}\Delta p_1 \\ &= C_d w\sqrt{\frac{2}{\rho}p_1}\Delta x_v + C_d wx\sqrt{\frac{1}{2\rho p_1}}\Delta p_1 \\ &= K_q\Delta x_v + K_c\Delta p_1\end{aligned} \tag{8.95}$$

即此时系统的负载压力为

$$p_L = p_1 \tag{8.96}$$

去掉增量符号，最终得线性化流量方程为

$$q_2 = K_q x_v + K_c p_1 \tag{8.97}$$

液压缸的流量连续性方程为

$$q - q_2 = A_1 v \tag{8.98}$$

液压缸的受力平衡方程为

$$p_1 A_1 - F = m\dot{v} \tag{8.99}$$

则可得旁路节流调速回路的方程组为

$$\begin{aligned}q_2 &= K_q x_v - K_c p_1 \\ q - q_2 &= A_1 v \\ p_1 A_1 - F &= m\dot{v}\end{aligned} \tag{8.100}$$

将式（8.100）进行拉氏变换，有

$$\begin{aligned}Q_2 &= K_q x_v + K_c P_1 \\ Q - Q_2 &= A_1 sX(s) \\ P_1 A_1 - F &= ms^2 X(s)\end{aligned} \tag{8.101}$$

根据式（8.101）可以建立旁路节流调速回路线性化仿真模型，如图 8-42 所示。

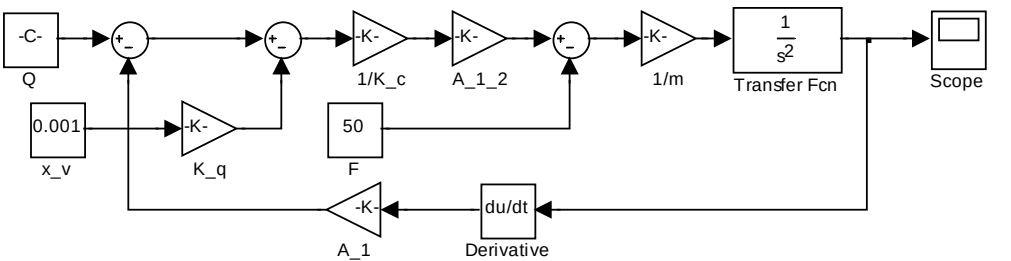


图 8-42　旁路节流调速回路线性化仿真模型

8.3.2　液压节流调速回路非线性化仿真

1．进油节流调速回路的仿真

进油节流调速回路原理图如图 8-43 所示。

下面来列写进油节流调速回路的运动学方程。

液压缸的运动学方程为

$$p_1 A_1 - F = m\dot{v} \tag{8.102}$$

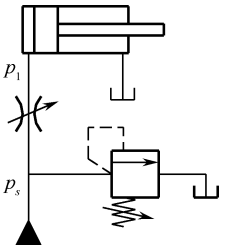


图 8-43 进油节流调速回路原理图

式中 p_1——液压缸进油腔的压力（Pa）；

A_1——液压缸无杆腔的面积（m^2）；

F——液压缸所受到的负载力（N）；

m——液压缸所推动的负载的质量（kg）；

v——液压缸的运动速度（m/s）。

经整理得

$$p_1 = \frac{F + m\dot{v}}{A_1} \tag{8.103}$$

通过节流孔的流量方程为

$$q_1 = C_d w x \sqrt{\frac{2}{\rho}(p_s - p_1)} \tag{8.104}$$

式中 C_d——流量系统，取 0.62；

w——节流孔开口面积的面积梯度（m）；

x——节流阀的开口量（m）；

ρ——液体的密度（kg/m^3）；

p_s——系统的供油压力（Pa）；

q_1——通过节流阀的流量（m^3/s）。

进油节流调速回路的流量连续性方程为

$$q_1 = A_1 v \tag{8.105}$$

将式（8.103）、式（8.105）代入式（8.104）中，得到

$$A_1 v = C_d w x \sqrt{\frac{2}{\rho}\left(p_s - \frac{F + m\dot{v}}{A_1}\right)} \tag{8.106}$$

根据式（8.106）即可以构建进油节流调速回路的传递函数仿真模型，如图 8-44 所示。

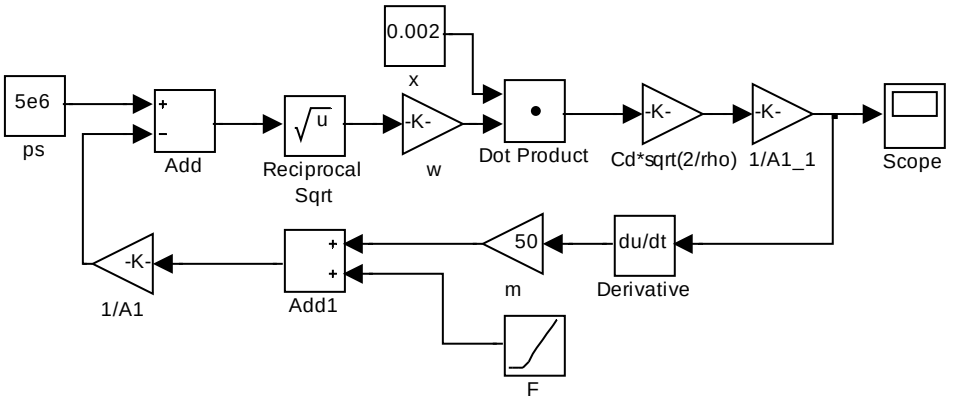


图 8-44 进油节流调速回路非线性化仿真模型

值得说明的是，如果按上述回路进行仿真，Simulink 会提示错误，如图 8-45 所示。该提示表明回路中出现代数环。其解决方法可以参考本书 6.7.3 节内容。

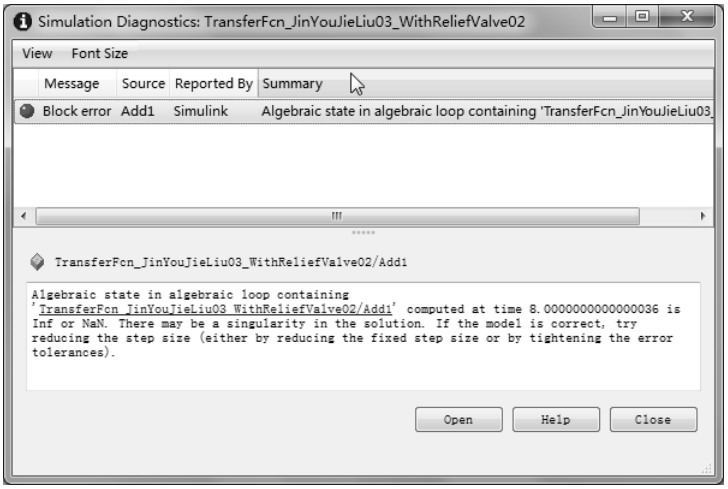


图 8-45　代数环错误提示

参考 6.7.3 节的内容，在本例中，我们采用添加记忆元件的方法来消除代数环，修改后的仿真草图如图 8-46 所示（如图中画圈处所示，添加了一个记忆模块）。

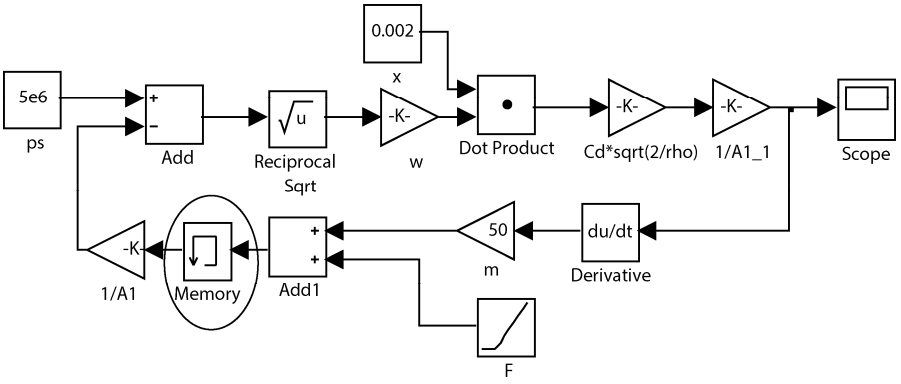


图 8-46　添加了记忆元件的进油节流调速回路仿真模型

2．回油节流调速回路的仿真

回油节流调速回路原理图如图 8-47 所示。

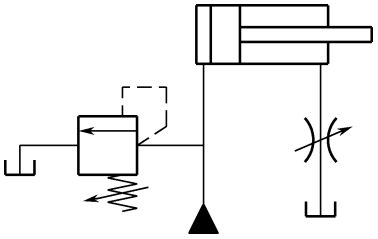


图 8-47　回油节流调速回路原理图

下面来列写回油节流调速回路的运动学方程。

液压缸的运动学方程为

$$p_1 A_1 - p_2 A_2 - F = m\dot{v} \tag{8.107}$$

式中 p_2 ——液压缸回油腔压力（Pa）；

A_2 ——液压缸有杆腔有效面积（m^2）。其余符号意义同式（8.102）。

将式（8.107）整理成关于 p_2 的表达式，有

$$p_2 = \frac{p_1 A_1 - F - m\dot{v}}{A_2} \tag{8.108}$$

通过节流阀的流量方程为

$$q_2 = C_d w x \sqrt{\frac{2}{\rho} p_2} \tag{8.109}$$

式中 p_2 ——节流阀进油口的压力（Pa）；

q_2 ——通过节流阀的流量（m^3/s）。

通过节流阀的流量连续性方程为

$$q_2 = A_2 v \tag{8.110}$$

将式（8.108）、式（8.110）代入式（8.109）中，整理得到

$$A_2 v = C_d w x \sqrt{\frac{2}{\rho}\left(\frac{p_1 A_1 - F - m\dot{v}}{A_2}\right)} \tag{8.111}$$

根据式（8.111）可以建立 Simulink 的仿真模型，如图 8-48 所示。

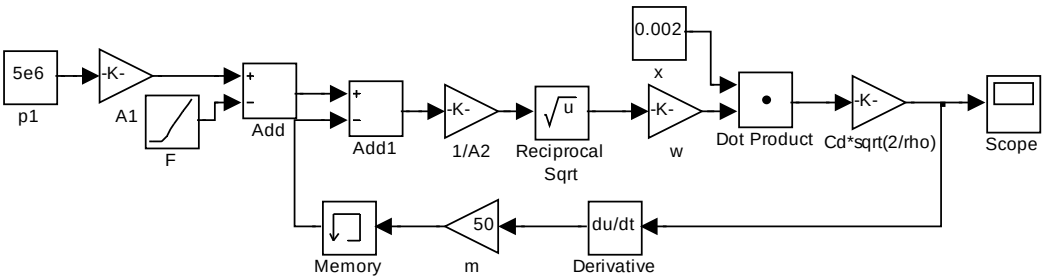


图 8-48 回油节流调速回路非线性化仿真模型

值得注意的是，为了避免在回路中出现代数环问题，在反馈通道中添加了记忆环节。

3．旁路节流调速回路的仿真

旁路节流调速回路原理图如图 8-49 所示。

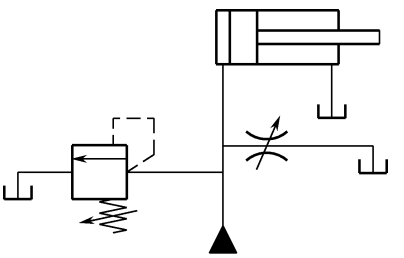


图 8-49 旁路节流调速回路原理图

为了对其进行仿真，需要列写该回路所对应的运动学方程。

液压缸的运动学方程为

$$p_1 A_1 - F = m\dot{v} \tag{8.112}$$

式中各符号意义同前。式（8.112）可以整理为

$$p_1 = \frac{F + m\dot{v}}{A_1} \tag{8.113}$$

通过节流阀的流量的方程为

$$q_2 = C_d w x \sqrt{\frac{2}{\rho} p_1} \tag{8.114}$$

式中　q_2 ——为通过节流阀的流量（m^3/s）。

进入液压缸的流量方程为

$$q_1 = A_1 v \tag{8.115}$$

因为在旁路节流调速回路中，溢流阀所起的作用是安全保护，没有流量通过溢流阀，则液压泵所打出的流量全部进入液压缸和节流阀，则有流量连续性方程为

$$q = q_1 + q_2 \tag{8.116}$$

式中　q ——液压泵的流量（m^3/s）。

将式（8.113）、式（8.114）和式（8.115）代入式（8.116）中，可得

$$q = A_1 v + C_d w x \sqrt{\frac{2}{\rho} \frac{F + m\dot{v}}{A_1}} \tag{8.117}$$

根据式（8.117）可以绘制仿真模型，如图 8-50 所示。

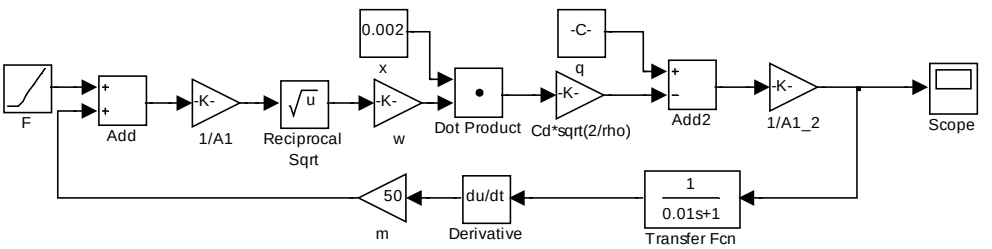


图 8-50　旁路节流调速回路非线性化仿真模型

值得注意的是，为了消除该回路的代数环，使用了高频振荡环节，而没有采用记忆环节。从该实例得知，在仿真过程中，如果采用一种方法无法消除代数环，可以尝试采用另外一种方法，并不断尝试改变求解器参数设置，直到能够仿真为止。

8.3.3　动态系统的方块图与传递函数

下面以一机床进给系统为例，介绍实际液压系统的建模方法。

1．动态系统方块图

图 8-51 所示是机床进给系统的液压原理简图，p_1、p_2、A_1 和 A_2 分别表示液压缸进出油腔的压力和相应油腔的有效面积，Q_1 和 Q_2 分别为流入和流出液压缸的流量，$m\frac{\mathrm{d}v}{\mathrm{d}t}$、$Bv$、$F_L$ 分别为惯性力、与速度有关的阻力和与时间有关的干扰负载力。惯性力、阻力和负载干扰力来自机械方面；压力 p_1、p_2 是液压参数；运动速度 v 是执行机构的运动速度，又是液压缸内

液流速度，它是机械参数和液压参数的相关量。由此可知，一般的液压传动系统是由机械系统和液压系统两部分组成的，两者之间紧密联系。

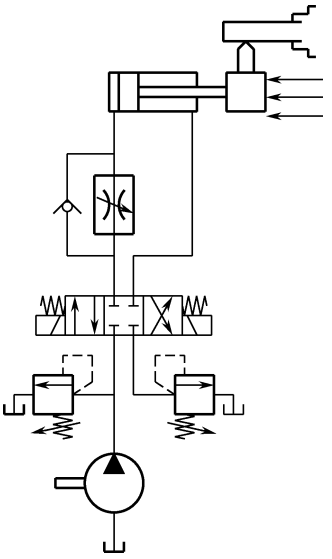


图 8-51 机床进给系统的液压原理简图

当刀架上作用外干扰力 F_L 时，液压缸活塞上力平衡方程为

$$F = m\frac{\mathrm{d}v}{\mathrm{d}t} + Bv + F_L \tag{8.118}$$

式中 F ——液压驱动力，$F = p_1A_1 - p_2A_2$；

B——阻尼系数；

m ——运动部分质量。

因为惯性力和摩擦阻力与速度有关，用 F_{mv} 表示，称为惯性摩擦力，即

$$F_{mv} = m\frac{\mathrm{d}v}{\mathrm{d}t} + Bv \tag{8.119}$$

故

$$F_{mv} = F - F_L \tag{8.120}$$

研究液压传动系统的动态特性，重点放在分析和计算负载干扰力发生变化时，执行机构的速度变化情况。对本调速系统来说，负载干扰力 F_L 为系统的输入干扰量，而执行机构的刀架活塞杆部件运动速度 v 是系统的输出量。输入量 v 随输入量 F_L 的变化过程是：当 F_L 发生变化，在液压驱动力 F 还未来得及变化时，便引起惯性摩擦力 F_{mv} 的变化。由于 F_{mv} 的变化，执行机构的速度 v 也随之变化。反过来引起驱动力 F 的变化，构成具有反馈联系的闭环系统。图 8-52 所示为机床等效动态系统图。

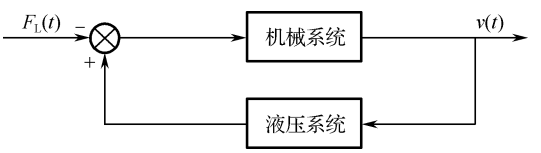


图 8-52 机床等效动态系统图

设初始条件为零，对输入量与输出量取拉氏变换，可得到以下的传递函数。

$$F_{mv}(s) = F(s) - F_L(s) \tag{8.121}$$

机械系统的传递函数可表示为

$$W_m(s) = \frac{V(s)}{F_{mv}(s)} \tag{8.122}$$

液压系统的传递函数可表示为

$$W_h(s) = \frac{F(s)}{U(s)} \tag{8.123}$$

将式（8.121）、式（8.122）和式（8.123）联立，整理得到

$$\frac{F_{\mathrm{mv}}(s)}{F_0}=\frac{F(s)}{F_0}-\frac{F_{\mathrm{L}}(s)}{F_0}$$

$$\bar{W}_{\mathrm{m}}(s)=\frac{U(s)/v_0}{F_{\mathrm{mv}}(s)/F_0} \tag{8.124}$$

$$\bar{W}_{\mathrm{s}}(s)=\frac{F(s)/F_0}{U(s)/v_0}$$

式中　$W_{\mathrm{m}}(s)$、$\bar{W}_{\mathrm{m}}(s)$——机械系统有量纲和无量纲传递函数；

$W_{\mathrm{s}}(s)$、$\bar{W}_{\mathrm{s}}(s)$——液压系统有量纲和无量纲传递函数；

v_0、F_0——v、F 的稳态值。

根据上述方程，可作出机床动态系统方块图，如图 8-53 所示。

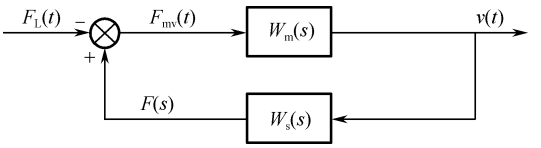


图 8-53　机床动态系统方块图

2．液压系统的传递函数

从图 8-53 可以看出，液压系统的传递函数 $W_{\mathrm{s}}(s)$ 是机床传动系统的反馈环节。传动系统要正常工作，系统的反馈环节必须具有在负载干扰力 $F_{\mathrm{L}}(s)$ 作用下，随着刀架活塞杆速度的下降传动系统驱动力将增加，并趋向于恢复平衡状态。反之，在外干扰力 F_{L} 作用下，随着刀架活塞杆运动速度的降低，传动系统的驱动力减小，系统不能正常工作。因此，要对液压系统的传递函数进行研究。一般液压系统的传递函数取决于组成该系统的液压元件（包括管道）。

如图 8-51 所示的系统，若不考虑液压缸和管道的内外泄漏，则液压系统的输入干扰量速度 v 和输出量驱动力 F 的微分方程组为

$$\begin{aligned}
A_1 v &= Q_1 - c_1\frac{\mathrm{d}p_1}{\mathrm{d}t}\\
A_2 v &= Q_2 + c_2\frac{\mathrm{d}p_2}{\mathrm{d}t}\\
F &= A_1 p_1 - A_2 p_2\\
c_1 &= l_1 A_1 / K\\
c_2 &= l_2 A_2 / (K\lambda_{\mathrm{k}})
\end{aligned} \tag{8.125}$$

式中　l_1、l_2——某一瞬时液压缸进油腔和回油腔的长度；

K——液压油等效体积弹性模量；

λ_{k}——液压油体积弹性模量则算系数。

将式（8.125）取增量，并进行拉氏变换，整理后得

$$\begin{aligned}
A_1 U(s) &= [Q_1(s)/P_1(s) - c_1 s]P_1(s)\\
A_2 U(s) &= [Q_2(s)/P_2(s) + c_2 s]P_2(s)\\
F(s) &= A_1 P_1(s) - A_2 P_2(s)
\end{aligned} \tag{8.126}$$

任何液压元件或组件的动态特性，都可以用压力和流量的传递函数表示。因此，在式

（8.126）中出现的$Q_1(s)/P_1(s)$和$Q_2(s)/P_2(s)$，都可认为它们分别表示液压缸进油路上和回油路上所有元件及管道的综合传递函数。

令

$$W_1(s)=P_1(s)/Q_1(s)$$
$$W_2(s)=P_2(s)/Q_2(s) \tag{8.127}$$

式中 $W_1(s)$——液压缸进油路上液压元件及管道总的传递函数；

$W_2(s)$——液压缸回油路上液压元件及管道总的传递函数。

将式（8.127）代入式（8.126）中，便可绘出图 8-54 下半部分的液压系统方块图。由图中可以清楚地看出，刀架活塞杆运动速度$U(s)$的变化，同时引起液压缸两腔中的压力变化，即驱动力$F(s)$的变化。但液压缸两腔压力变化滞后于速度的变化。$P_1(s)$的变化滞后取决于积分环境$-1/(c_1s)$；$P_2(s)$的滞后取决于积分环节$1/(c_2s)$。液压缸两腔中压力变化会引起液压缸进出流量的变化，这个变化将起反馈作用，其传递函数为$1/W_1(s)$和$1/W_2(s)$。这说明液压缸进出油路上液压元件及管道的综合动特性对系统动态特性是有影响的。

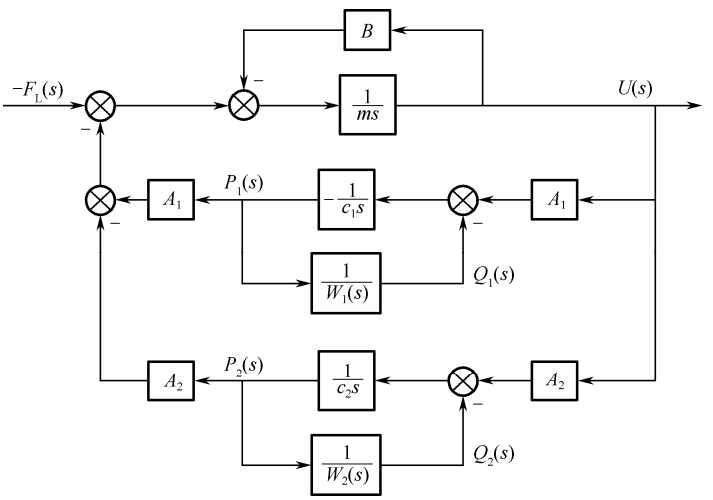


图 8-54 工作机构-液压传动方块图

由图 8-54 下半部分可得液压系统的传递函数为

$$W_s(s)=\frac{F(s)}{U(s)}=A_1^2\frac{W_1(s)}{1-c_1sW_1(s)}-A_2^2\frac{W_2(s)}{1+c_2sW_2(s)} \tag{8.128}$$

3．等效机械系统的传递函数

将式（8.119）取增量，并进行拉氏变换得

$$F_{mv}(s)=(ms+B)U(s) \tag{8.129}$$

由式（8.129）画出的机械系统方块图如图 8-54 的上半部分，其传递函数一般表达式则为

$$W_m(s)=U(s)/F_{mv}(s)=1/(ms+B) \tag{8.130}$$

无量纲传递函数为

$$\bar{W}_m(s)=U(s)F_0/F_{mv}(s)v_0=\frac{m/B}{(ms/B+1)T_0} \tag{8.131}$$

式中 T_0——执行机构的时间常数，$T_0=mv_0/F_0=mv_0/(p_{10}A_1-p_{20}A_2)$。

若机械系统没有与速度有关的阻尼力，则 $B=0$，其传递函数为

$$W_{\mathrm{m}}(s)=U(s)/F_{\mathrm{mv}}(s)=1/ms \tag{8.132}$$

$$\bar{W}_{\mathrm{m}}(s)=W_{\mathrm{m}}F_0/v_0=1/(T_0 s) \tag{8.133}$$

若机械系统有与速度有关的阻尼力，则摩擦特性与实际条件有关。当工作部件高速运动时，可以认为无此力；工作部件以较低速度运动时，摩擦面间出现液体摩擦状态，摩擦力与速度成正比，其机械传递函数为

$$W_{\mathrm{m}}(s)=\frac{1}{ms+B_{\mathrm{a}}} \tag{8.134}$$

$$W_{\mathrm{m}}(s)=\frac{m/B_{\mathrm{a}}}{[(ms/B_{\mathrm{a}})+1]T_0} \tag{8.135}$$

式中　B_{a} ——黏性阻力系数，其值为正，一般很小可以忽略。

当工作部件极低速移动时，摩擦面间出现边界摩擦状态。摩擦力具有降落特性，此时极易出现爬行。若从运动稳定性出发，则应取最坏的条件研究，用不稳定环节的传递函数，即

$$W_{\mathrm{m}}(s)=\frac{1}{ms-|B_{\mathrm{f}}|} \tag{8.136}$$

$$W_{\mathrm{m}}(s)=\frac{m/|B_{\mathrm{f}}|}{[(ms/|B_{\mathrm{f}}|)-1]T_0} \tag{8.137}$$

式中　B_{f} ——摩擦降落特性阻力系数，其值为负。

4．传动系统传递函数的一般表达式

由式（8.128）和式（8.130）可得干扰力 $F_{\mathrm{L}}(s)$ 为输入，执行机构液压缸速度 $U(s)$ 为输出，传动系统开环传递函数为

$$W_1(s)=W_{\mathrm{m}}(s)W_{\mathrm{s}}(s)=\frac{1}{ms+B}\left[\frac{A_1^2W_1(s)}{1-c_1sW_1(s)}-\frac{A_2^2W_2(s)}{1+c_2sW_2(s)}\right] \tag{8.138}$$

闭环传递函数为

$$\Phi_1(s)=\frac{U(s)}{F_{\mathrm{L}}(s)}=-\frac{W_{\mathrm{m}}(s)}{1-W_1(s)} \tag{8.139}$$

8.4　液压仿真的例子

再举一个液压仿真的例子。本例采用传递函数法、微分方程的数值解法和 Simulink 建模方法对同一回路进行仿真。

如图 8-55 所示的液压系统。一个柱塞缸支撑一个重物，假设系统初始处于平衡状态。设重物的质量为 m，柱塞的直径为 D，油液的弹性模量为 k，通入的液压油的流量为 Q_{s}，柱塞缸中初始的液体体积为 V_0，柱塞运动时的黏性阻尼系数为 B。试仿真活塞的运动速度（v）随时间的变化过程。

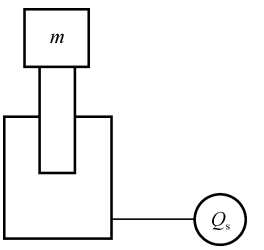


图 8-55　柱塞缸支撑系统

可以用传递函数法、微分方程的数值解法和 MATLAB 的 Simulink 工具箱，来求解这个问题。

8.4.1 基本原理

根据牛顿第二定律，分析柱塞缸的受力平衡方程，有

$$pA - Bv - mg = m\frac{\mathrm{d}v}{\mathrm{d}t} \tag{8.140}$$

又根据油液的弹性模量公式，根据 2.1.2 节中的式（2.2），有

$$k = -\frac{\Delta p}{\frac{\Delta V}{V_0}} = -V_0\frac{\Delta p}{\Delta V} = -V_0\frac{\Delta p/\Delta t}{\Delta V/\Delta t} = -V_0\frac{\mathrm{d}p/\mathrm{d}t}{\Delta Q} \tag{8.141}$$

得到由于油液的可压缩性而造成的流量损失为

$$\Delta Q = -\frac{V_0}{k}\frac{\mathrm{d}p}{\mathrm{d}t} \tag{8.142}$$

式中的负号表示由于压力的增加造成流量减少。

根据油液的流量连续性方程可知，通入的流量减去由于油液的可压缩性而损失的流量等于柱塞缸的有效面积乘以柱塞缸的运动速度，有

$$Q - |\Delta Q| = Av \tag{8.143}$$

即

$$Q - \frac{V_0}{k}\frac{\mathrm{d}p}{\mathrm{d}t} = Av \tag{8.144}$$

最终得到该系统的动态特性的微分方程组为

$$\begin{cases} pA - Bv - mg = m\dfrac{\mathrm{d}v}{\mathrm{d}t} \\ Q - \dfrac{V_0}{k}\dfrac{\mathrm{d}p}{\mathrm{d}t} = Av \end{cases} \tag{8.145}$$

8.4.2 传递函数法

如式（8.145）所示，已经得到了描述系统动态特性的微分方程，要想求通入的流量和液压缸速度之间的传递函数，只需消去变量 p。将式（8.145）的第一式两侧对 t 求导，整理，有

$$A\frac{\mathrm{d}p}{\mathrm{d}t} = B\frac{\mathrm{d}v}{\mathrm{d}t} + m\frac{\mathrm{d}^2v}{\mathrm{d}t^2} \tag{8.146}$$

将式（8.146）代入式（8.145）的第二式，整理得

$$\frac{V_0 m}{kA}\frac{\mathrm{d}^2v}{\mathrm{d}t^2} + \frac{V_0 B}{kA}\frac{\mathrm{d}v}{\mathrm{d}t} + Av = Q \tag{8.147}$$

对上式两侧进行拉氏变换，整理成传递函数的形式，有

$$G(s) = \frac{V(s)}{Q(s)} = \frac{Q_k}{\frac{V_0 m}{kA}s^2 + \frac{V_0 B}{kA}s + A} \tag{8.148}$$

可以依据上面的推导过程，编写 MATLAB 程序，对该系统进行仿真。

编写的 MATLAB 程序代码如下。

```
%用传递函数来验证柱塞缸的动态特性
g = 9.8;                    %重力加速度
```

```
D = 0.1;                    %柱塞直径 0.1m
k = 1700000000;             %油液的弹性模量为 1700000000Pa
Q = 100*10^-3/60.0;         %通入的流量为 100L/min
V0 = 50*10^-6;              %液压缸中的初始体积为 50mL
B = 5000;                   %黏性阻尼系数为 N/m/s
m = 100;                    %物体的质量为 100kg/m^3
A = (pi/4)*D^2;             %柱塞缸的有效面积

num = [Q];
den = [(V0*m)/(k*A) (V0*B)/(k*A) A];
Gs = tf(num,den);
t = 0:0.05:2;
y = step(Gs,t);
plot(t,y)
```

执行上述程序，得到的仿真结果如图 8-56 所示。

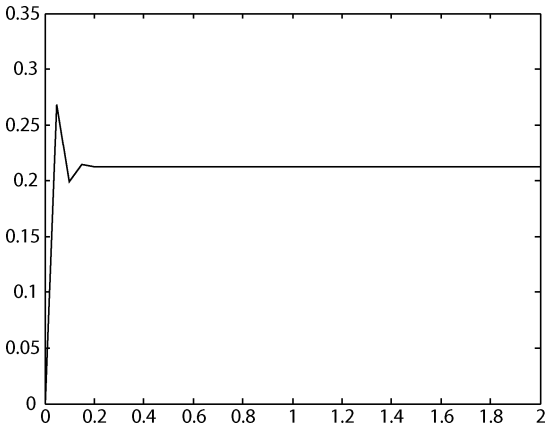


图 8-56　液压缸速度的仿真曲线

8.4.3　微分方程的数值解法

事实上，上述仿真，也可以用微分方程的数值解法来求得仿真结果。将式（8.145）整理成微分方程组的形式，即

$$\begin{cases} \dfrac{\mathrm{d}v}{\mathrm{d}t} = \dfrac{1}{m}(pA - Bv - mg) \\ \dfrac{\mathrm{d}p}{\mathrm{d}t} = \dfrac{k}{V_0}(Q - Av) \end{cases} \tag{8.149}$$

根据本章讲解的内容，就可以编写微分方程组的数值算法求解程序，来对这个动态系统进行仿真了。

MATLAB 程序如下。

```
%测试液压系统的仿真，一个柱塞缸的例子
g = 9.8;                    %重力加速度
```

```
D = 0.1;                %柱塞直径 0.1m
k = 1700000000;         %油液的弹性模量为 1700000000Pa
Q = 100*10^-3/60.0;     %通入的流量为 100L/min
V0 = 50*10^-6;          %液压缸中的初始体积为 50mL
m = 100;                %物体的质量为 100kg/m^3
B = 5000;               %黏性阻尼系数为 N/m/s
A = (pi/4)*D^2;         %柱塞缸的有效面积

odefun = @(t,x)[(x(2)*A - B*x(1) - m*g)/m;
                k*(Q - A*x(1))/V0];      %微分方程组
x0 = [0 0];                              %初始条件
options = odeset('RelTol',1e-2, 'AbsTol', 1e-3);
[t,x] = ode23(odefun, [0 2], x0,options);
plot(t,x(:,1))
```

在 MATLAB 中执行上述仿真程序，得到如图 8-57 所示的仿真结果。

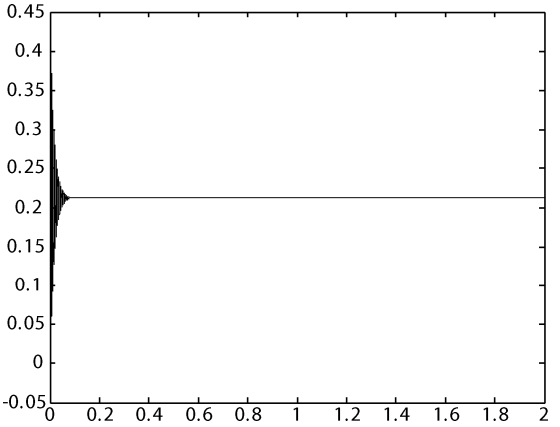


图 8-57　用数值算法求得的液压缸速度的仿真结果

从以上两者的对比可以看出，仿真的终值是一致的，但在初始阶段有一些不同。这是由于所采用的数值算法不同造成的。

8.4.4　用 Simulink 方法

如前面章节所述，也可以用 Simulink 搭建这个柱塞缸控制系统动态特性仿真模型。对照式（8.145），跟随公式的加减乘除以及微分积分关系，可以建立如图 8-58 所示的 Simulink 仿真模型。

在建模的过程中，要注意如下几点。

（1）质量、重力加速度、输入流量、油液弹性模量、油液初始体积可以用常量元件来代表。

（2）要优先选用积分元件来描述方程中的微分项。比如要将速度时间的导数通过积分元件积分成速度来使用；将压力对时间的求导通过积分元件积分成压力来使用。

（3）微分方程中的加减乘除关系都能在“Math Operations”库中找到。值得一提的是，求倒数的元件可以在“Math Operations”中的“Math Function”中找到。

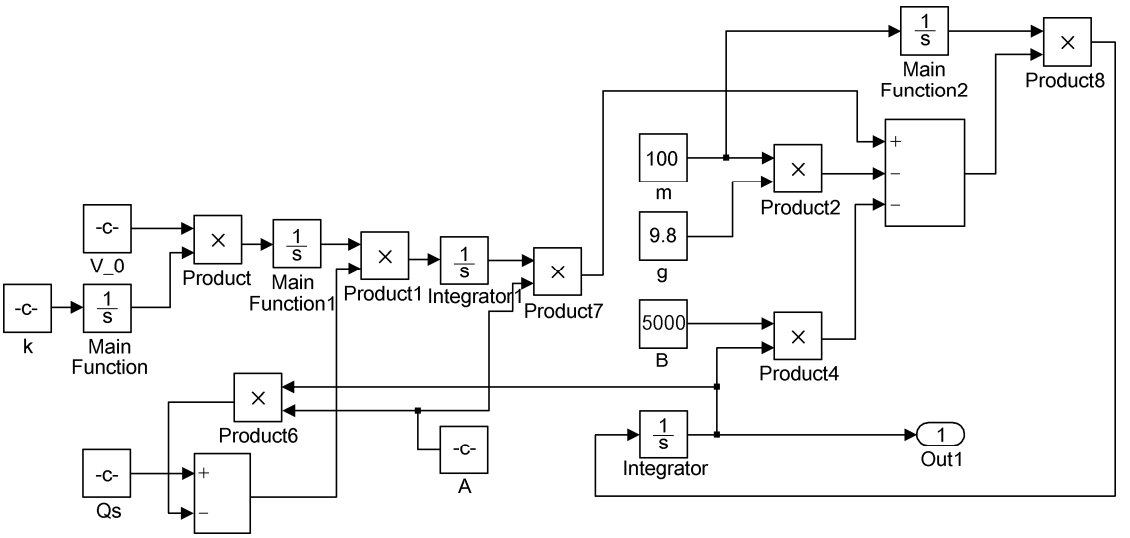


图 8-58　柱塞缸控制的 Simulink 仿真模型

建立好上述模型后，就可以进行仿真了。本例中，设置仿真结束的时间为 2s，仿真结果如图 8-59 所示。

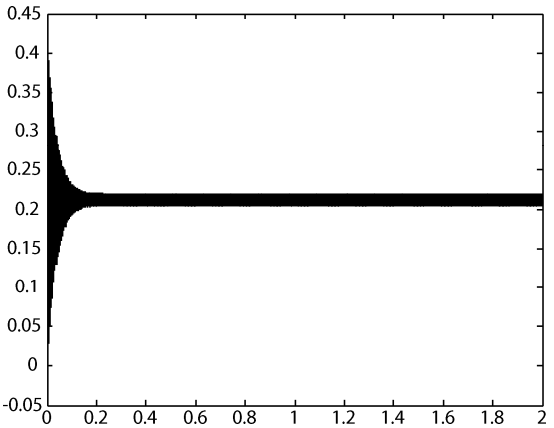


图 8-59　Simulink 建立的柱塞缸仿真模型仿真结果

对比图 8-56、图 8-57 和图 8-59 可见，仿真的终值是接近的，但是瞬态值之间是有细微的差别的，这主要是由于数值计算的方法不同导致的。

课后题

1．试在 Simulink 中用子系统封装法和用户定义函数法编写薄壁孔口的仿真模型，所采用的数据如下：流量系数 $C_{\mathrm{d}}=0.62$，孔口面积 $A=0.05\mathrm{cm}^2$，液压油的密度 $\rho=870\ \mathrm{kg/m}^3$。

2．试采用节点容腔法对含有如下参数的液压缸进行建模。参数：活塞直径 $D=100\mathrm{mm}$，活塞杆直径 $d=63\mathrm{mm}$，双活塞杆对称液压缸，总行程 $L=300\mathrm{mm}$，负载质量为 100kg，进油压力为 10MPa，回油压力为 0MPa。

3．试利用传递函数法，列写输入为流量、输出为位移的液压缸的传递函数。

4．试利用传递函数法，列写输入为压力、输出为位移的液压缸的传递函数，这是带有弹性负载的情况。

5．根据如下给定参数，绘制进油节流调速回路的速度负载特性曲线。分别采用线性化方法和非线性化方法。已知参数如下：流量系数 $C_d=0.62$，面积梯度 $w=0.188\text{m}$，开口量 $x=0.001\text{m}$，液压油的密度 $\rho=870\text{kg/m}^3$，液压缸无杆腔面积 $A_1=50\text{cm}^2$，有杆腔面积 $A_2=25\,\text{cm}^2$，液压缸所受到的负载力从 0～10000N 变化，液压缸所推动的负载为 100kg。

6．图 8-60 所示为某液压阻尼器。设 A 为液压缸无杆腔有效面积，k 为弹簧刚度，R_T 为节流阀液阻，p_1、p_2 为液压缸左、右腔的压力。试求液压缸缸筒位移 x_o 对活塞位移 x_i 的传递函数。

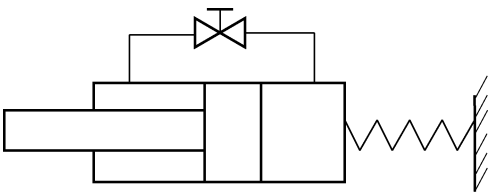


图 8-60　题 6 图

7．定量泵动态特性计算简图如图 8-61 所示。图中用一个容积 V_p 等效地表示泵内压油区的工作容积，用一个液阻 R_p 和一个液感 L_p 等效地表示泵的泄漏效应。试推导泵的输出压力对输出流量的传递函数，画出其动态结构框图，并分析定量泵的动态特性。

8．有一液压缸差动回路。若将液压缸的弹性变形、黏性摩擦、运动部件的质量都折算到一个具有质量 m、黏性摩擦 B 和刚度为 k 的弹簧相连的负载上，如图 8-62 所示。如果不计连接管道和液压缸泄漏的影响，试求活塞位移 x 对供油量 Q_p 的传递函数，画出回路动态结构框图，并对回路的动态特性进行分析讨论。

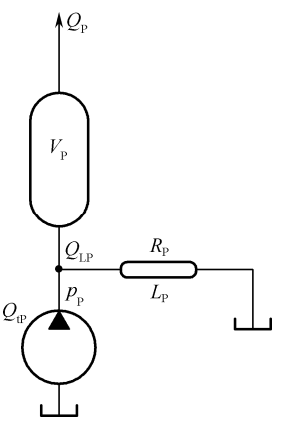


图 8-61　题 7 图

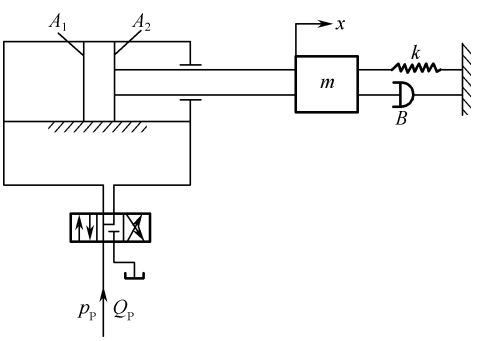


图 8-62　题 8 图

参考文献

[1] 梁全，苏齐莹. 液压系统 AMESim 计算机仿真指南[M]. 北京：机械工业出版社，2014.

[2] 周连山，庄显义. 液压系统的计算机仿真[M]. 北京：国防工业出版社，1986.

[3] W. 霍夫曼. 液压元件及系统的动态仿真[M]. 陈鹰，译. 杭州：浙江大学出版社，1988.

[4] 刘能宏，田树军. 液压系统动态特性数字仿真[M]. 大连：大连理工大学出版社，1993.

[5] 李永堂，雷步芳，高雨茁. 液压系统建模与仿真[M]. 北京：冶金工业出版社，2003.

[6] 李成功，和彦淼. 液压系统建模与仿真分析[M]. 北京：航空工业出版社，2008.

[7] 付永领. LMS Imagine.Lab AMESim 系统建模和仿真实例教程[M]. 北京：北京航空航天大学出版社，2011.

[8] 付永领，祁晓野，李庆. LMS Imagine.Lab AMESim 系统建模和仿真参考手册[M]. 北京：北京航空航天大学出版社，2011.

[9] 高钦和，马长林. 液压系统动态特性建模仿真技术及应用[M]. 北京：电子工业出版社，2014.

[10] 全国液压气动标准化技术委员会. GB/T 786.1—2009 流体传动系统及元件图形符号和回路图 第 1 部分：用于常规用途和数据处理的图形符号[S]. 北京：中国标准出版社，2009.

反侵权盗版声明