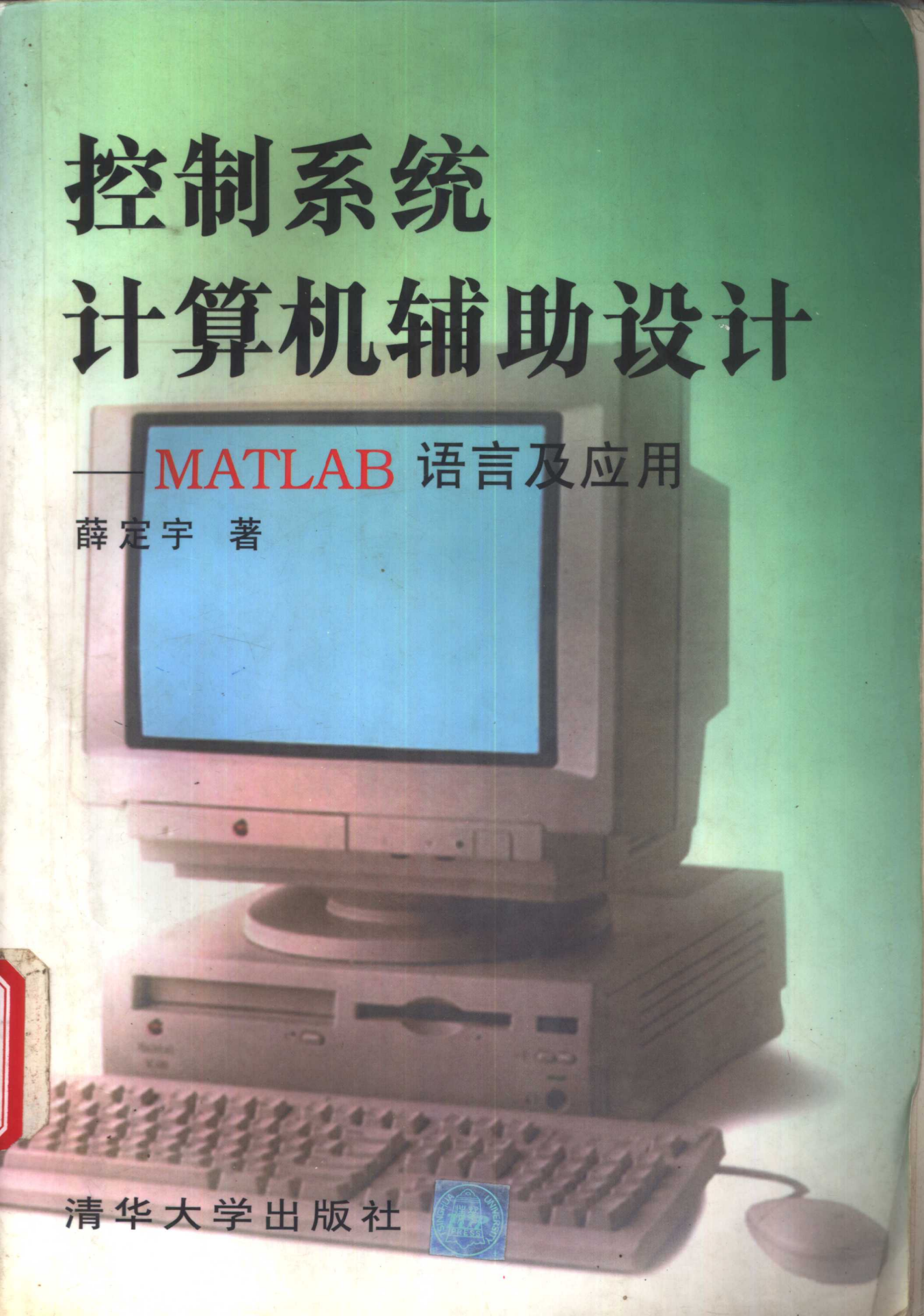
控制系统计算机辅助设计

——MATLAB 语言及应用

薛定宇 著



清华大学出版社

控制系统计算机辅助设计
——MATLAB 语言及应用

薛定宇 著

清华大学出版社

（京）新登字 158 号

内 容 简 介

本书系统地介绍了国际控制界最流行的控制系统计算机辅助设计语言 MATLAB，侧重于介绍 MATLAB 语言编程基础与技巧、数值分析算法的 MATLAB 实现、动态系统的数学模型及仿真工具 SIMULINK 等，并以之为基础介绍了经典的和当前最新的控制系统计算机辅助设计方法，包括多变量系统的频域设计、自整定 PID 控制方法、定量反馈理论、状态空间经典设计方法、LQ 及 LQG/LTR 设计、H_∞ 最优控制等。本书还以一个控制系统计算机辅助教学软件 Control Kit 为例介绍利用 MATLAB 进行 Windows 图形界面设计的方法。

本书可作为自动控制类专业的研究人员参考，也可作为高校该类专业的研究生与高年级本科生控制系统计算机辅助设计课程的教材和参考书，还可供其它专业的学生和科技工作者、教师作为科学计算与图形绘制等方向的工具和参考书。

图书在版编目(CIP)数据

控制系统计算机辅助设计：MATLAB 语言及应用/薛定宇著．—北京：清华大学出版社，1996.7
ISBN 7-302-02194-5

Ⅰ．控… Ⅱ．薛… Ⅲ．自动控制系统-计算机辅助设计 Ⅳ．TP273

中国版本图书馆 CIP 数据核字(96)第 09021 号

出版者：清华大学出版社(北京清华大学校内，邮编 100084)
印刷者：北京市海淀区清华园印刷厂
发行者：新华书店总店北京科技发行所
开 本：787×1092 1/16 印张：24.25 字数：555 千字
版 次：1996 年 6 月第 1 版 1996 年 6 月第 1 次印刷
书 号：ISBN 7-302-02914-5/TP・1054
印 数：0001—3000
定 价：28.00 元

前　言

控制系统计算机辅助设计 (CACSD) 从成为一门单独的学科以来至今已经有 20 多年的历史，在其发展过程中出现了各种各样的实用工具和理论成果。CACSD 课程是高校自动控制类专业研究生的一门重要课程，可选用的教材也很多，但由于其中大部分教材出现得较早，已经不能反映当代 CACSD 领域的最新成果。

MATLAB 语言的出现不但对 CACSD 算法的研究，也对其它 CACSD 软件环境的开发起到了巨大的推动作用，它已经成为国际控制界应用最广的语言和工具了。该软件早期版本 80 年代末传入我国以来，在高校中已经有了一些应用，但大部分用户苦于没有该软件相应的资料，难于系统地掌握该语言，有效地解决自己遇到的实际问题。

作者从 1988 年开始接触 MATLAB，使用过早期和现代的各个版本，曾用 MATLAB 为基础开发过几个商品软件，并在研究中一直使用 MATLAB 作为主要工具，所以熟悉 MATLAB 的特点及编程。

1995 年作者受辽宁省系统仿真学会邀请，在 '95 中国自动化教育学术年会后于秦皇岛举办“MATLAB 语言与控制系统计算机辅助设计新技术”讨论班，并为该讨论班编写了试用讲义，这就是本书的雏形。在该讲义的编写和整理过程中作者还在东北大学自动控制系研究生的“控制系统计算机辅助设计”课程中试用过其中的大部分章节，并在自控系本科生“系统仿真”课程中也试用过其中部分的内容，得到了较好的反映。

本书大致分为两个部分，前一个部分系统地介绍了 MATLAB 语言编程与应用，侧重于介绍 MATLAB 语言编程基础与技巧、数值分析算法及 MATLAB 实现、动态系统的数学模型及仿真工具 SIMULINK 等，最后还以作者开发的一个控制系统计算机辅助教学软件 Control Kit 为例，介绍利用 MATLAB 进行 Windows 图形界面设计的方法，其中既包含了 MATLAB 软件的入门知识，也介绍了其应用的高级技术，融合了作者多年来的实际编程经验和体会。第二个部分以 MATLAB 语言及其相应工具箱为主要手段介绍并探讨了经典的和当前最新的控制系统计算机辅助设计方法，包括多变量系统的频域设计、自整定 PID 控制方法、定量反馈理论、状态空间经典设计方法、LQ 及 LQG/LTR 设计、H_∞ 最优控制等。

本书可作为自动控制类专业的研究人员参考，也可作为高校该类专业的研究生与高年级本科生控制系统计算机辅助设计课程的教材和参考书，还可供其它专业的学生和科技工作者、教师作为自动控制原理、系统仿真等课程的实验辅助教材，以及科学计算与图形绘制等方向的工具和参考书。

本书由东北大学研究生院副院长徐心和教授主审。本书从酝酿到整个写作过程始终得到徐老师的鼓励和支持，他仔细地阅读了全书原稿，并提出了许多建设性的宝贵意见。

作者还感谢他的导师，原 IEEE 控制系统委员会主席，英国 Sussex 大学 Derek Atherton 教授，是他将作者引入 MATLAB 编程的乐园，并指导作者涉足先进的 CACSD 方法。几年来和他们的合作与学术交流使作者受益匪浅，他们严谨的学风与敬业精神亦对作者有很深的影响。

作者在国外学习工作期间的一些同事和朋友也给予作者许多建议和鼓励，使作者获得许多有益的信息与材料，在这当中包括现英国威尔士 Swansea 大学的庄敏霞博士、上海同济大学的赵之凡副研究员、英国 Sussex 大学的姚莉华博士等。本书试印本完成以来还得到很多国内外同行的建议和意见，在此一并表示最诚挚的谢意。

本书写作过程中承蒙东北大学控制仿真研究中心主任李彦平博士等同事的大力支持和鼓励，在此作者表示衷心的感谢。

本书承蒙清华大学自动化系主任、中国自动化学会教育委员会主任胡东成教授的大力推荐，在出版过程中又得到清华大学出版社蔡鸿程副编审的关怀和帮助，在此作者深表谢意。

本书写作与出版部分受国家教委留学回国人员基金和辽宁省博士启动基金资助。

几年来，作者的妻子杨军在生活和事业上给予了作者莫大的帮助与鼓励，作者谨以此书献给她和女儿薛杨。

由于作者水平有限，书中的缺点错误在所难免，欢迎读者批评指教。

薛　定　宇

1996 年 3 月于东北大学

目　录

第 1 章 CACSD 软件环境与新技术概述

1.1 绪 论

随着科学技术的发展，控制理论和系统的研究显得越来越重要。在 40 年代控制理论作为一门独立的学科出现以来，已经得到了迅速的发展。开始时控制系统的设计可以由纸笔等工具容易地计算出来，如 Ziegler 与 Nichols 于 1942 年提出的 PID 经验公式就可以十分容易地设计出来。随着控制理论的迅速发展，控制的效果要求得越来越高，控制算法越来越复杂，控制器的设计也越来越困难，这样光利用纸笔以及计算器等简单的运算工具难以达到预期的效果，加之在计算机领域取得了迅速的发展，于是很自然地出现了控制系统的计算机辅助设计 (Computer-Aided Control System Design，简称为 CACSD) 方法。

控制系统的计算机辅助设计技术的发展目前已达到了相当高的水平，并一直受到控制界的普遍重视。1982 年 12 月和 1984 年 12 月，控制系统领域在国际上最权威的 IEEE 控制系统学会 (Control Systems Society, 简称为 CSS) 的控制系统杂志 (Control Systems Magazine) 和 IEEE 学会的科研报告 (Proceedings of IEEE) 分别第一次出版了关于 CACSD 的专刊 [9, 10]，美国的著名学者 Jamshidi 与 Herget 分别于 1985 年和 1992 年出版了两本著作来展示 CACSD 领域的最新进展 [11, 12]。在如国际自动控制联合会世界大会 (IFAC World Congress)、美国控制会议 (American Control Conference, 简称为 ACC) 及 IEEE 的控制与决策会议 (Conference on Control and Decision, 简称为 CDC) 等各种国际控制界的重要学术会议上都有有关 CACSD 的专题会议及各种研讨会，可见该领域的发展是异常迅速的，控制系统计算机辅助设计又常常称作计算机辅助控制系统工程 (Computer-Aided Control Systems Engineering, 简称 CACSE)。

近 30 年来，随着计算机技术的飞速发展，出现了很多的优秀计算机应用软件，在控制系统的计算机辅助设计领域更是如此，各类 CACSD 软件频繁出现且种类繁多，有的是用 FORTRAN 语言编写的软件包，有的是人机交互式软件系统，还有专用的仿真语言，在国际控制界广泛使用的这类软件就有几十种之多。

在国内流行的大多数有关教材中，控制系统计算机辅助设计主要讲述“经典”的 CACSD 方法，如串联校正、简单的二次型最优控制策略等，而对当前国际上比较热门的 CACSD 方法诸如自整定 (autotuning) PID 控制方法、定量反馈理论 (quantitative feedback theory, 简称为 QFT) 方法、以及 H_∞ 鲁棒控制方法等并无系统的介绍。本书的目的之一就是系统地引入这些先进的方法，使得读者能更开阔自己的眼界。

以前国内外在介绍控制理论及CACSD技术的教材中，都采用BASIC语言[13, 28]和FORTRAN语言[18]，并有少部分教材中采用C语言作为主要的程序设计语言。这固然可以锻练学生，使之可以从最底层了解有关的CACSD程序设计的方法与技巧，但难以使之能有一个对整个方法的全面了解，容易产生“只见树木，不见森林”的认识偏差。另外由于程序往往是用户自己编写的，所以难免会造成数值不稳定、计算结果错误等不该发生事件的出现，此外，这种从最底层进行编程的方式在效率上来讲是相当低的，大部分时间将花费在没有太大价值的重复性机械性劳动上了。

孟子云：“工欲善其事，必先利其器”。跟踪国际最先进的CACSD软件环境及发展，以当前国际上最流行的CACSD软件环境MATLAB为基本出发点来系统地介绍控制系统计算机辅助设计技术及软件实现，从而大大提高CACSD算法研究与实际应用的效率和可靠性，这也是本书的另外一个目的。

1.2 国外CACSD软件环境综述

1973年美国学者Melsa教授和Jones博士出版了一本专著[18]，书中给出了许多当时流行的控制系统计算机辅助分析与设计的程序，包括求取系统的根轨迹、频率响应、时间响应、以及各种控制系统设计的子程序如Luenberger观测器、Kalman滤波等。瑞典Lund工学院Karl Åström教授主持开发的一套交互式CACSD软件INTRAC (IDPAC, MODPAC, SYNPAC, POLPAC等，以及仿真语言SIMNON)[2]，其中的SIMNON仿真语言要求用户依照它所提供的语句编写一个描述系统的程序，然后才可以对控制系统进行仿真。日本的古田胜久 (Katsuhisa Furuta) 教授主持开发的DPACS-F软件[6]，在处理多变量系统的分析和设计上还是很有特色的。在国际上流行的仿真语言ACSL, CSMP, TSIM, ESL等也同样要求用户编写模型程序，并提供了大量的模型模块。在这一阶段还出现了很多的专用程序，如英国剑桥大学推出的线性系统分析与设计软件CLADP (Cambridge linear analysis and design programs)[5, 17]与美国国家宇航局 (NASA) Langley研究中心的Armstrong开发的LQ控制器设计的ORACLS (optimal regulator algorithms for the control of linear systems)[1]等。

1980年美国学者Cleve Moler等人推出的交互式MATLAB语言逐渐受到了控制界研究者的普遍重视，从而陆续出现了许多专门用于控制理论及其CAD的工具箱。图形交互式的模型输入计算机仿真环境SIMULINK的出现为MATLAB应用的进一步推广起到了积极性的推动作用。现在，MATLAB已经风靡了全世界，成为控制系统CAD领域最普及也是最受欢迎的软件环境。

在MATLAB迅速发展的同时，很多软件开发者针对控制系统领域的实际问题开发了专用的CACSD计算机辅助设计软件，如美国系统控制技术公司 (Systems Control Technology, Inc) 的John Little等人研制的CTRL-C, Boeing公司的EASY 5及EASY5x，Integrated Systems公司的MATRIXx及Xmath，Systems Technology Incorporated公司的CC程序，Visual Simulation公司的VisSim, 日本东京工学院的MaTX等，其中很多软件是并行于MATLAB而独立开发的，但或多或少都会从这些软件的语句结构或使

用方法中看出受到 MATLAB 影响的痕迹，所以说，从国际上最流行的 MATLAB 出发来介绍控制系统的计算机辅助设计技术是再合适也不过的了。这就是本书在众多 CACSD 软件中挑选 MATLAB 作为基本语言的一个最主要的原因。

国际上控制系统计算机辅助设计软件的发展大致分为几个阶段：软件包阶段、交互式语言阶段及当前的面向对象的程序环境阶段[14]。

在早期的工作中，CACSD 主要集中在软件包的编写上，如前面提及的 Melsa 和 Jones 的著作。从数值算法的角度上也出现了一些著名的软件包，如美国的基于特征值的软件包 EISPACK[7, 26] 和线性代数软件包 LINPACK[4]，英国牛津数值算法研究组 (Numerical Algorithm Group) 开发的 NAG 软件包 [21] 及文献 [23] 中给出的程序集等。在 CACSD 领域的经典软件包作品有英国 Kingston Polytechnic 控制系统研究组开发的 SLICE (subroutine library in control engineering) 软件包，前面提及的 DPACS-F, ORACLS 等。这些软件包大都是由 FORTRAN 语言编写的源程序组成的，例如若想求出 N 阶实矩阵 A 的全部特征值 (用 WR，WI 数组分别表示其实虚部) 和对应的特征向量矩阵 Z，则 EISPACK 软件包给出的子程序建议调用路径为

```
CALL BALANC(NM,N,A,IS1,IS2,FV1)
CALL ELMHES(NM,N,IS1,IS2,A,IV1)
CALL ELTRAN(NM,N,IS1,IS2,A,IV1,Z)
CALL HQR2(NM,N,IS1,IS2,A,WR,WI,Z,IERR)
IF (IERR.EQ.0) GOTO 99999
CALL BALBAK(NM,N,IS1,IS2,FV1,N,Z)
```

由上面的叙述可以看出，要求取矩阵的特征值和特征向量，首先要给一些数组和变量依据 EISPACK 的格式作出定义和赋值，并编写出主程序，再经过编译和连接过程，形成可执行文件，最后才能得出所需的结果。用软件包的形式来编写程序有如下的缺点：

- **使用不方便**: 对不是很熟悉 EISPACK 的用户来说，直接利用软件包来编写程序是相当困难的，也是相当容易出错的，其中一个子程序调用发生微小的错误可能将导致最终得出错误的结果。
- **调用过程繁琐**: 首先需要编写主程序，确定对软件包的调用过程，再经过必要的编译和连接过程，有时还要花大量的时间去调试程序以保证其正确性，而不是想得出什么马上就可以得出的。
- **执行程序过多**: 想求解一个特定的问题就需要编写一个专门的程序，并形成一个可执行文件，如果需要求解的问题很多，就需要在计算机硬盘上同时保留很多这样的可执行文件，这样计算机磁盘空间的利用不是很经济。
- **不利于传递数据**: 通过软件包调用方式针对每个具体问题就能形成一个孤立的可执行文件，在一个程序中产生的数据无法传入另一个程序，更无法使几个程序同时执行来解决所关心的问题。
- **维数指定困难**: 在 CACSD 中最重要的变量是矩阵，如果要求解的问题维数较低，则形成的程序就不能用于求解高阶问题，例如文献 [18] 中的程序均定为 10 阶。所以有

时为使得程序通用，往往将维数设置得很大，这样在解小规模问题是会出现空间的浪费，而大规模问题仍然求解不了。在优秀的软件中往往需要动态地进行矩阵定维。

此外，这里介绍的大多数早期软件包都是由 FORTRAN 语言编写的，由于众所周知的原因，以前 FORTRAN 语言绘图并不是轻而易举的事情，这就需要再调用相应的软件包来做进一步处理，在绘图方面比较实用和流行的软件包是 GINO-F[3]，但这种软件包只给出绘图的基本子程序，所以要绘制较满意的图形需要用户自己去用这些低级命令去编写出合适的绘图子程序来。

英国剑桥大学学者 John Edmunds 和 Jan Maciejowski 等人开发的 CLAPD 在控制界享有盛誉，它包括了多变量系统分析与设计的多种方法，其中有 Nyquist 类以及特征轨迹等多变量频域设计方法，也有 LQG 与 Kalman 滤波等时域设计方法，还可以处理时间延迟及分布系统等非有理问题。日本东京工学院的古田胜久教授主持开发的 DPACS-F 软件是由 FORTRAN 语言编写的，它可以由状态空间和频域方法来分析多变量线性系统，并可以由极点配置和 LQG 等方法来设计控制器，此外还可以进行多变量系统辨识等工作。NASA 的 Armstrong 教授编写的 ORACLS 则是一个十分专用的软件，它可以用于多变量系统的 LQG 设计，该软件也是由 FORTRAN 语言编写的。

70 年代末期和 80 年代初期出现了很多实用的具有良好人机交互功能的软件，MATLAB 就是其中的一个成功的范例，此外前面提及的 INTRAC 和 CTRL-C 等也是优秀的人机交互式软件，例如在 CTRL-C 下用户可以输入下面的命令来给出矩阵参数并求取该矩阵的特征值与特征向量

```
[>  A=[1,2,3; 4,5,6; 7,8,0];
[>  [x,d]=eig(A)
```

这里 [> 为 CTRL-C 的专用提示符，在这一提示符下可以容易地输入 A 矩阵，并由一条命令直接求出 A 矩阵的特征值和特征向量，这无疑使得问题的解法得到了极大的简化。这种输入方式和后面将介绍的 MATLAB 几乎是一致的，另外若已经获得了数据，则利用 CTRL-C 可以容易地绘制出相应的曲线来。可以看出，使用 CTRL-C 这样的交互式软件系统比起软件包来说上了一个台阶。交互式软件的一个显著特点是其使用方便、集成度高，由简单的几条规范命令就可以实现以前 FORTRAN 类语言成百上千条语句的功能，且结果稳定可靠。

正因为存在多种多样的 CACSD 软件，而它们之间又各有所长，所以在 CACSD 技术的发展过程中曾有过几次将若干常用软件集成在一起的尝试，例如 1984 年前后美国学者 Spang, III 教授曾将当时流行的 SIMNON, CLAPD, IDPAC 及他自己研制的 SSDP(state space design program) 集成在一起，形成了一个强大的软件 [27]，各个组成软件之间是靠读写文件的方式来传递数据的，这多少可以解决前面提及的程序之间不能传递数据的弊病。1986 年前后由英国科学与工程委员会 (SERC) 资助，Harold Rosenbrock 教授和 Neil Munro 教授主持的，英国多所大学和研究机构参与的 ECSTASY (environment for control system theory and synthesis) 软件环境的开发项目 [20]，在该软件中试图将流行的新一代软件如 MATLAB, ACSL, TSIM 甚至新近出现的 Mathematica 等集成到一个框

架之下，该软件还可以同时自动采用 LaTeX和 Framemaker 等来输出专业的排版结果，并取得了一些成效。各个软件之间的数据传递是通过数据库来实现的，ECSTASY 定义了方便实用的 CACSD 新命令，比 MATLAB 更为简洁。ECSTASY 这样的软件是一个有益的尝试，但该软件当时只可以在 Sun 工作站上运行，并没有考虑 PC 机的兼容性。

依作者之见，这些集成出来的软件并不是很成功的，因为它们并没有达到预期的效果，事实上，从那以后每个软件的功能都有了明显的改善，MATLAB 语言有了自己的仿真功能，SIMULINK 从某种意义上来讲其功能和接口更优于 ACSL, MATLAB 和 Mathematica 之间也有了较好的接口，它们的优势可以得到充分地互补。

1.3 ACSL 仿真语言及其应用

ACSL 是一类有代表性意义的仿真语言，它是由美国 Mitchell and Gauthier Associate 公司推出的仿真语言，其全名为 Advanced Continuous Simulation Language (高级连续仿真语言) [19]。ACSL 的语句规则是建立在 1967 年由仿真委员会 (Simulation Councils Inc, 简称为 SCi) 规定的仿真语言规范的基础上的。

ACSL 要求用户用它所提供的模块编写一个描述系统的程序，这一程序和 FORTRAN 语言的程序十分相似，其显式结构如表 1-1 所示。ACSL 首先要求用户依照其语言规

表1-1 ACSL 仿真语言的基本程序结构

PROGRAM	程序名称
INITIAL	
	在运行之前要执行的语句在一般情况下为初始条件等的定义。
END	
DYNAMIC	
DERIVATIVE	
	用积分语句 INTEG 描述的微分方程计算的语句
END	
	其它计算语句。
END	
TERMINAL	
	终止判断语句，通常在 TERMT 语句条件为真时终止
END	

则建立起一个模型文件，然后可以通过 ACSL 本身提供的命令对之进行仿真及辅助分析。ACSL 与 FORTRAN 语言的主要区别在于，ACSL 的语句更简练，内容更丰富，ACSL 语言也可以直接调用由 FORTRAN 编写的子程序。ACSL 编程的结构比相应的 FORTRAN 语言更严格。程序的基本结构必须严格按表 1-1 所给出的格式来编写，否则所得出的仿真结果可能出现意想不到的错误。ACSL 提供了几十个系统子模块 (macros), 其中包括很常用的线性和非线性子模块，如传递函数模块 TRAN, 积分器模块 INTEG, 超前滞后环节 LEDLAG, 延迟模块 DELAY, 死区非线性模块 DEAD, 磁滞回环 BAKLSH, 限幅积分器 LIMINT 等，用户可以利用这些子模块简单地编写出描述给定系统的仿真模型，然后采用 ACSL 提供的功能来对系统进行仿真分析，并绘制出结果的曲线表示。例如，著名的

der Pol 方程的一种特殊形式可以写成

$$\ddot{y} + \mu(y^2 - 1)\dot{y} + y = 0$$

若取 $\mu = 1$, 并取状态变量为 $y_1 = y, y_2 = \dot{y}$，则 Van der Pol 方程可以写成

$$\dot{y}_1 = y_1(1 - y_2^2) - y_2,\ \dot{y}_2 = y_1$$

这时可以由 ACSL 所定义的语言写出此系统的模型如下：

```
PROGRAM VAN DER POL EQUATION
CINTERVAL CINT=0.01
CONSTANT  Y1C=3.0,   Y2C=2.5,  TSTP=15.0
Y1=INTEG(Y1*(1-Y2**2)-Y2, Y1C)
Y2=INTEG(Y1, Y2C)
TERMT (T.GE.TSTP)
END
```

在此程序中把显示步长 CINT 设置为 0.01, 状态变量的初值由 Y1C 和 Y2C 表示，而终止仿真时间 TSTP 设置为 15，该程序中变量 T 为实际仿真时间。这时可以采用 ACSL 的编译命令来编译并将此模型和 ACSL 库连接起来，则可以形成一个可执行文件。这一过程完成之后 ACSL 将自动给出提示：

```
ACSL>
```

用户可以在此提示符下键入 PREPAR T, Y1, Y2 来通知 ACSL 模型在仿真时需要保留 T, Y1, Y2 三个参数，然后键入 START 来开始此模型的数字仿真。如果要得出系统的相平面图 (Y1-Y2)，则可以由 ACSL 的运行时命令 (run-time command) PLOT 来绘制所需的图形，亦即要想得出系统的相平面图，则可以键入如下命令：

```
PLOT Y1, Y2
```

这时在图形窗口上将显示出系统的相平面图如图 1-1 (a) 所示，这一曲线最终趋于封闭的图形。可见，在系统取了一个较远离原点的初始条件时，相平面图将逐渐收缩，最终将

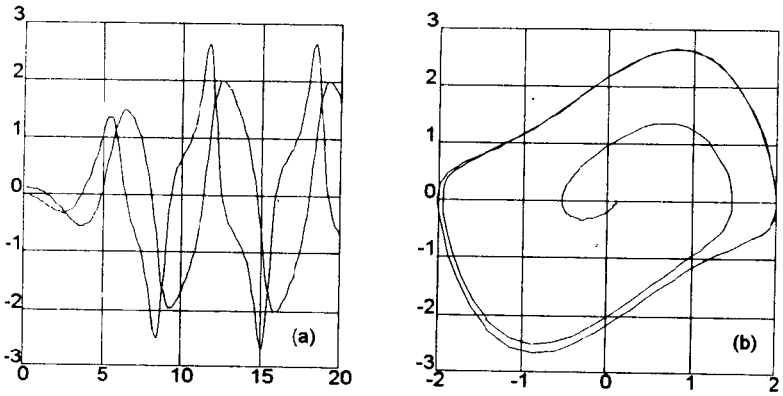


图1-1 Van der Pol 方程的相平面图

稳定地收敛于这一条封闭的曲线。如果改变系统的初始条件，则可以利用 ACSL 的 SET 命令来实现：如 SET Y1C=0, Y2C=0.1, 这时再重复上述过程，则可以得出系统的相平面图如图 1-1 (b) 所示。可见，在系统初始条件取得较接近原点时，系统的相平面曲线将逐渐延伸，最后仍将收敛于一条封闭的曲线。由此可以定性地得出结论：无论系统的初始条件选作何值，Van der Pol 方程描述的系统最终的相平面曲线均将收敛于一条封闭的曲线，这一条曲线称为系统的极限环 (limit cycle) 。

ACSL 仿真语言近来又出现了一些新的进展，例如它提供了一个系统框图编辑界面，用户可以根据自己要研究的系统方框图直接“绘制出”ACSL 可以识别的框图形式，然后可以调用 ACSL 进行仿真，这样就显著地减少了用户使用该软件的难度，因为用户不必要去像以前那样记忆大量的 ACSL 函数，而直接可以通过可视的方法构成系统的模型来。

除了前面介绍的 ACSL 语言外还存在许许多多的其它仿真语言，如 ESL, TSIM, CSMP 等，例如在 ELS 语言下描述 Van der Pol 方程的语句和前面的类似，而得出的曲线如图 1-2 所示 [8] 。在这里是分别对 5 种 μ 值分别进行仿真的结果，其中每条曲线都由不同的标志给出。

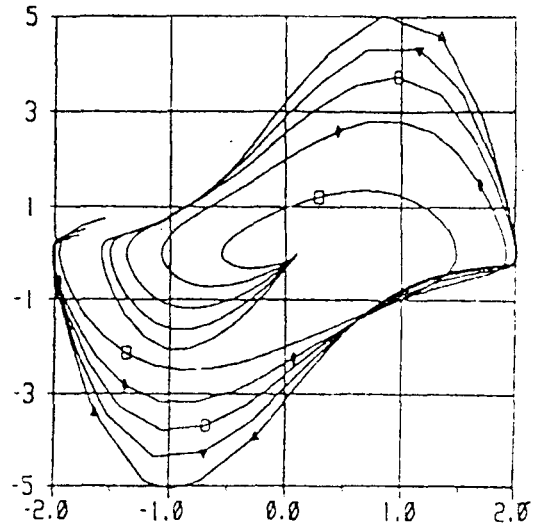


图 1-2 不同 μ 下 Van der Pol 方程相平面图

1.4 MATLAB, SIMULINK 与各种 CACSD 工具箱

1980 年美国的 Cleve Moler 博士研制的 MATLAB 环境 (或语言) 对后来的控制系统的理论及计算机辅助设计技术起到了巨大的推动作用，值得指出的是该语言原本并不是专门为控制理论领域的学者使用的，其最初目的是为线性代数等课程提供一种方便可行的实验手段，该软件出现以后一直在美国的 New Mexico 等大学作为教学辅助软件使用，并作为面向公众的免费软件 (public domain software) 广为流传，该软件的出现对控制界的影响是十分巨大的 [24] 。MATLAB 于 1984 年推出了正式版本。

由于该软件的使用极其容易，且提供了丰富的矩阵处理功能，所以控制理论领域的研究人员很快就注意到了这样的特点，并在它的基础上开发了控制理论与 CAD 专门的应用程序集 (又称为工具箱)，使之很快地在国际控制界流行起来，目前它已经成为国际控制界最流行的语言了。除了流行于控制界以外，MATLAB 还在诸如图像信号处理、生物医学工程等相关的领域中有广泛的应用，并且出现了数以百计的各种实用工具箱，而这些工具箱反过来又进一步促进了 MATLAB 的应用。MATLAB 当前的功能可以说是集可靠的数值运算 (特别是但不局限于矩阵运算)、图像与图形显示及处理、高水平的