MATLAB 编程应用

数学物理方程的 MATLAB 解法与可视化

彭芳麟 著

清华大学出版社



本书介绍用科学计算软件MATLAB数值求解数学物理方程并将结果可视化。书中展示了在教材中难得一见的复变函数图形、特殊函数图形和各类本征函数图形，还有拉普拉斯方程、热传导方程和波动方程的各种题型的数值求解与可视化的结果，内容新颖，方法独特，让枯燥的公式伴之以生动的图像，让深奥的内容有了鲜明的物理图像，是学习数学物理方法的极有价值的参考书。

本书也详细地介绍了MATLAB的偏微分方程工具箱与解偏微分方程和本征值方程的其他指令，还介绍了差分方法和有限元方法，对学习数值计算或计算物理课程而言，也是一本很实用的参考教材。

本书的程序来之于教学实践，有许多经验心得体现在编程的技巧中，例如特殊函数的计算及矢量场线的画法，这些技巧不仅实用，也很有特色。书中提供了全部例题的程序，读者可以将这些程序直接当作多媒体课件来使用。本书可供大学生、研究生和科技工作者使用。

ISBN 7-302-09884-0

9 787302 098843 >

定价：33.00元

MATLAB 编程应用

数学物理方程的 MATLAB 解法与可视化

彭芳麟 著

清华大学出版社
北京

内 容 简 介

本书介绍如何用科学计算软件MATLAB数值求解数学物理方程并将结果可视化。书中展示了在教材中难得一见的复变函数图形、特殊函数图形和各类本征函数图形，还有拉普拉斯方程、热传导方程和波动方程的各种题型的数值求解与可视化的结果，内容新颖，方法独特，让枯燥的公式伴之以生动的图像，让深奥的内容有了鲜明的物理图像，是学习数学物理方法极有价值的参考书。本书也详细地介绍了MATLAB的偏微分方程工具箱与解偏微分方程和本征值方程的其他指令，还介绍了差分方法和有限元方法。对学习数值计算或计算物理课程而言，这也是一本很实用的参考教材。本书的程序来之于教学实践，有许多经验心得体现在编程的技巧中，例如特殊函数的计算、矢量场线的画法，这些技巧不仅实用，也很有特色。书中提供了全部例题的程序，可以将这些程序直接当作多媒体课件来使用。本书可供大学生、研究生和科技工作者使用。

版权所有，翻印必究。举报电话：010 62782989　13901104297　13801310933
本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
本书防伪标签采用清华大学核研院专有核径迹膜防伪技术，用户可通过在图案表面涂抹清水，图案消失，水干后图案复现；或将表面膜揭下，放在白纸上用彩笔涂抹，图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

数学物理方程的MATLAB解法与可视化/彭芳麟著. 北京：清华大学出版社，2004.11
ISBN 7-302-09884-0

Ⅰ.数… Ⅱ.彭… Ⅲ.数学物理方程－数值计算：计算机辅助计算－软件包，MATLAB
Ⅳ.O175.24

中国版本图书馆CIP数据核字(2004)第114657号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**：100084
社 总 机：010 62770175　　**客户服务**：010-62776969
责任编辑：宋成斌
印 装 者：北京嘉实印务有限公司
发 行 者：新华书店总店北京发行所
开　　本：175×245　**印张**：20.25　**插页**：2　**字数**：410千字
版　　次：2004年11月第1版　2004年11月第1次印刷
书　　号：ISBN 7-302-09884-0/O·121
印　　数：1～3000
定　　价：33.00元

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770175 3103或(010)62795704

前　言

本书介绍如何用科学计算软件 MATLAB 数值求解数学物理方程及将结果可视化，这是我们在进行数学物理方法课程的数字化教学中做的一些工作。书中提供了全部程序，因而读者不仅可以从中学到解题的方法，还可以将这些程序直接当作多媒体课件来使用。

数学物理方程主要是偏微分方程，考虑到有的读者可能不熟悉 MATLAB 在这方面的应用，本书用附录的形式详细地介绍了 MATLAB 偏微分方程工具箱的使用方法，以及其他一些可用于解偏微分方程的指令和解常微分方程本征值问题的指令，此外还介绍了差分方法、有限元方法解偏微分方程。从这一点来看，本书也是一本介绍用 MATLAB 解偏微分方程的很实用的参考书。为了将结果可视化，书中涉及对多种特殊函数的计算以及物理场如矢量场线的画法，这些技巧不仅实用，也很有特色。

数学物理方法是众多理工科学生的基础课程之一，也是科研人员常用的基本方法。但是，数学物理方法课程的内容多而难，题目繁而杂，是一门公认的较难的课程。教学基本是用公式推导，所得的结果往往是一个复杂的积分或级数，其中还免不了使用特殊函数。**尽管数学物理方法的习题通常都有明确的物理意义，可是怎么能够使学生从让人眼花缭乱的数学表达式中看出其中所表达的物理图像，恐怕不仅学生会感觉困惑，教师也难免觉得棘手。由此不难理解，为什么许多学生都觉得数学物理方法太枯燥乏味了。如何让这些公式中的物理图像展现出来，让这些无言的公式“开口说话”，这正是本书所追求的目标。**

显然，这个工作离不开数值计算及其可视化。原则上，可以用 FORTRAN 或 C 语言来完成这个任务，但并没有人实际去这样做，原因是成本太高。幸运的是，高性能数学软件与有强大计算能力的个人计算机让这件事变得容易起来，本书就是一个明证。书中，**我们选择了数学物理方法教材各章节中的各种题型作为求解实例，并将结果用图形甚至动画表现出来，以此来展现问题的物理图像。**

例如，书中有个例子是研究环形电流的磁场。该题可以用三种不同方法计算，所得结果分别用连带勒让德函数的广义傅里叶级数表示，用椭圆函数的积分表示或者直接用数值积分表示，那么这些形式各异的结果是一致的吗？它跟实验的结果相同吗？谁都会对这个问题感到好奇，可是又觉得难以定论。但是，当我们画出它们的电力线以后，就会看到什么叫殊途同归。虽在意料之中，还是不得不惊叹数学的神奇。尤其对初学本课程的学生来说，留给他们的影响是极其深刻的，甚至也许是终生难忘的。

书中还有许多这类例子，比如平面波展开为球面波，平面波展开为柱面波，达朗贝尔行波公式与有界弦振动之间的联系，当我们用图形和动画模拟出这种结论时，学生对广义傅里叶级数的公式无疑会有更深的理解，而且对不同章节，不

同领域的内容之间的联系也有了切身的体会。所以这些都会给人以面貌一新的感觉，它改变了只能用公式符号教学的模式，对教学效果肯定会有提高。

编写本书的动机起源于教学。早在 2001 年，我们在教改中为了帮助学生学习数值计算，给本科生尝试开设了选修课 理论力学计算机模拟。这是首次在物理系的教学中引进数学软件。试验十分成功，这届学生不仅成为我系第一批会用数值求解常微分方程的学生，而且表现出巨大的学习热情，做出了许多出色的、有创造性的工作。我们将这些成果总结成为《理论力学计算机模拟》一书，在书中介绍了用计算机软件 MATLAB 解常微分方程的方法，并且用它求解了理论力学中一些典型的线性与非线性问题，还将结果做成模拟系统运动的动画，随光盘与书一起发行。这个工作在探索理论力学教学改革、计算物理教学和多媒体教学结合方面，获得了一些有益的经验。该书受到高校教师的广泛好评，该书的工作曾获首届全国大学物理教学优秀论文一等奖，以及中国物理学会举办的第三届全国多媒体与网络物理教学成果评比一等奖。

随着教改的深入、经验的积累以及学生能力的提高，原来安排的只学习数值求解常微分方程的内容已经不能满足学生的需求。于是我们在课程中又增加了数值求解偏微分方程的内容，同样受到学生的热烈欢迎。而这些内容正好是数学物理方法课程多媒体教学所需要的。所以我们因势利导，开展了数学物理方法多媒体教学的一些工作。本书中的程序就是这些新的教学内容的总结与系统化。这个工作获得北京市教改立项项目“理论物理数字化教学的探索”的资助。**书中部分内容曾经在 2002 年兰州的全国数学物理方法教学会议和 2003 年南宁的全国理论力学教学会议上作过介绍，引起了同行们的广泛兴趣。另外，这方面最初的工作曾在 2002 年在澳门举办的第四届全国多媒体与网络物理教学成果评比中获一等奖。**

本书内容安排的顺序与常规教材有所不同，主要是按主题内容或方程分类，这是因为我们考虑的是数字化教学的需要，侧重介绍数值解法或者将解析结果可视化。它区别于教材中以方法分类的知识体系，是以问题分类的知识体系，这对学生学过的知识能起到复习巩固提高的作用。六章的内容分别是函数图形，傅里叶变换，本征函数，拉普拉斯方程，热传导方程和波动方程。为了节约篇幅，在讨论方程的各章中，没有按照解法分类，也没有详细讨论求解过程，而是直接引用一些常见的教材如梁昆淼的教材中习题的题解。读者在使用时，需要参考有关的教材。在选题时，我们也注意尽量找有物理内容、可视化效果明显的习题。书中不仅给出了各题的计算结果，还公布了源程序，直接运行这些程序就能让教师和学生亲自领略将数理方程可视化的乐趣。这些程序曾经花费了我们不少的时间与精力，在教学中经过师生们的反复讨论和比较，有许多经验心得体现在编程的技巧与语句的运用中。这种做法与国外许多同类的书籍做法是一致的，目的是让读者

不仅能知道结果，而且能学会方法，同时也避免在类似的工作中因从头探索而走一些不必要的弯路。

作者水平有限，而所完成的又是一项探索性的工作，所以书中错漏之处，必然存在，敬请专家与读者指正。

最后，作者十分感谢清华大学出版社对本书出版所给予的大力支持，尤其是责任编辑宋成斌的热情支持与帮助。

本书所列出的源程序之电子文件版权为笔者所有，如有需要者，请用电子邮件联系。 Email:pcnf1@bnu.edu.cn 。

作　者

2004 年 10 月

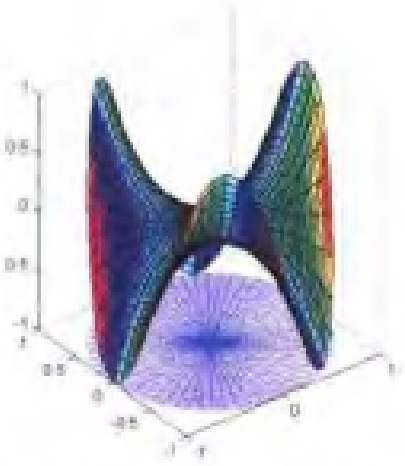


图 1.2　复变函数 z^3 的图形

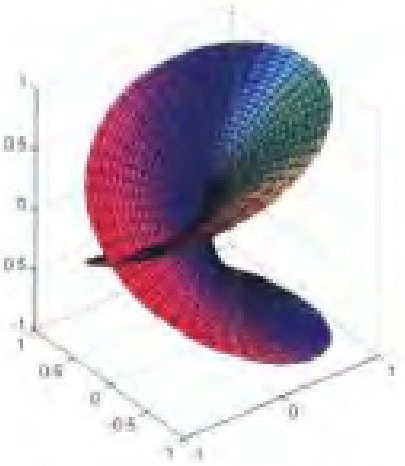
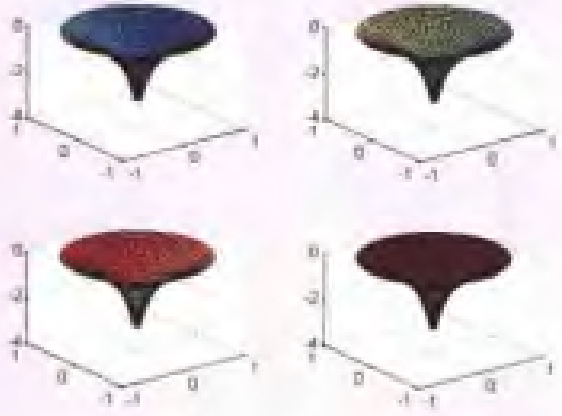
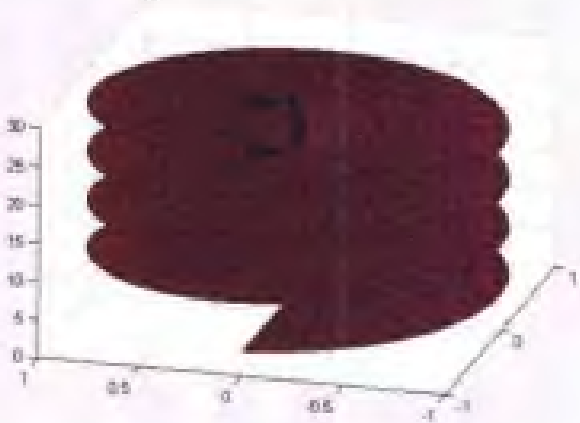


图 1.3　复变函数 $z^{1/3}$ 的图形



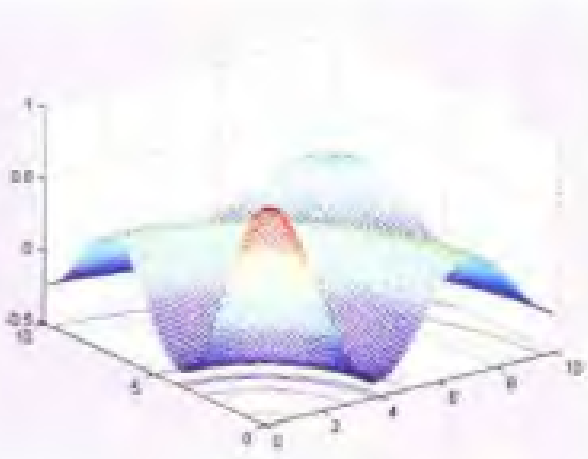
z 轴为 $\ln z$ 的实部

图 1.9　复变函数 $\ln z$ 的图形

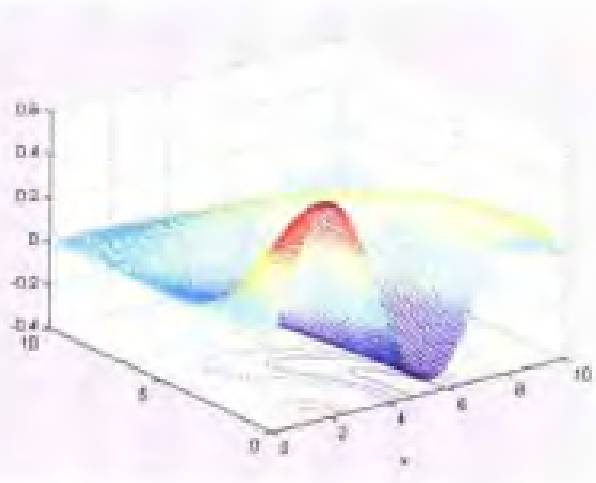


z 轴为 $\ln z$ 的虚部

图 1.10　复变函数 $\ln z$ 的图形

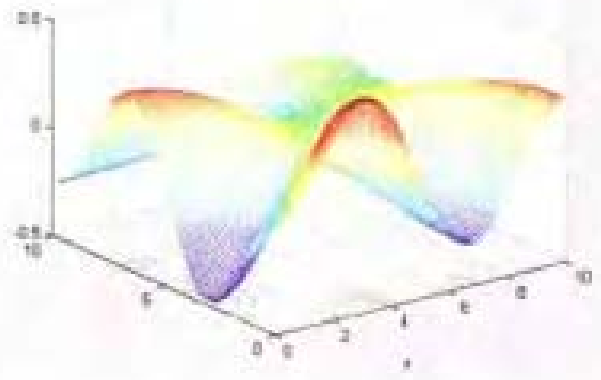


$m=0$

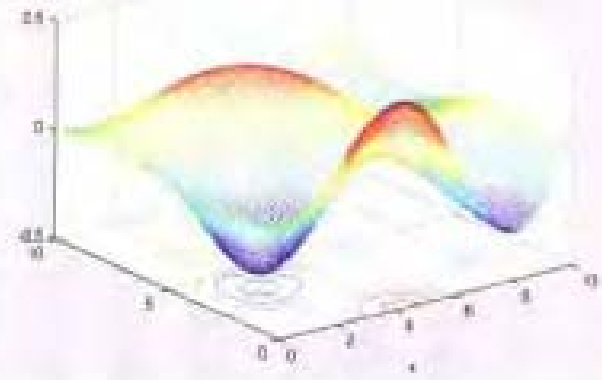


$m=1$

图 2.14　柱面波之一

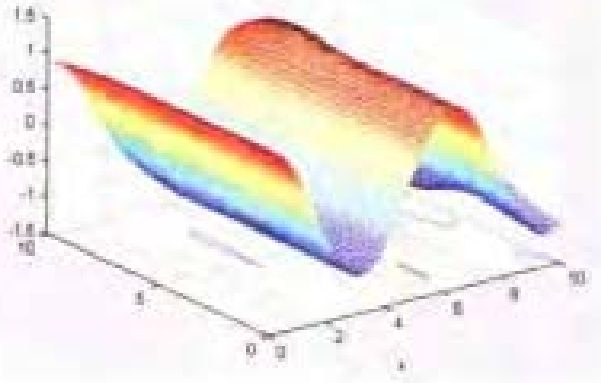


m=2

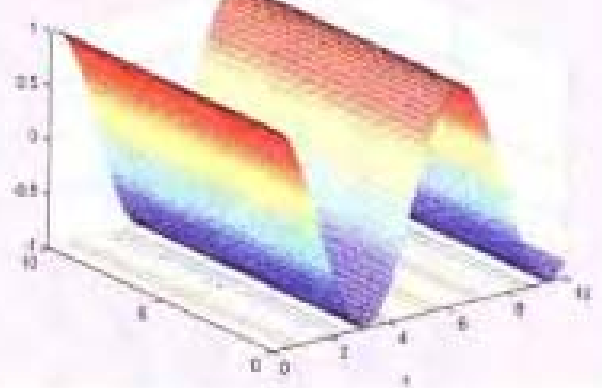


m=3

图 2.15　柱面波之二

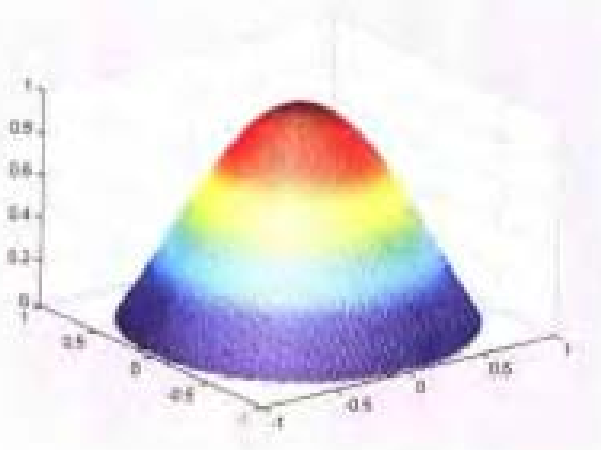


10 个柱面波的迭加效果

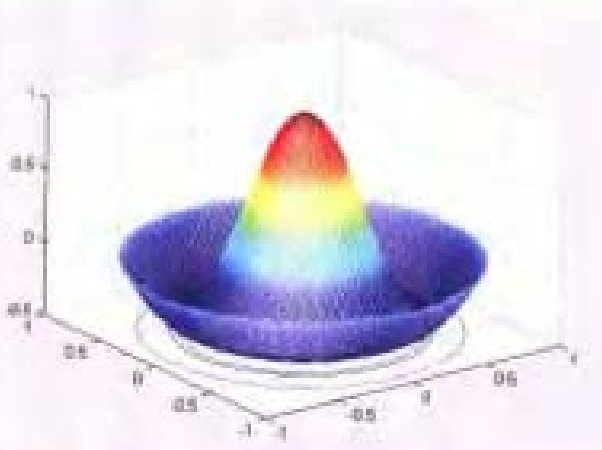


20 个柱面波的迭加效果

图 2.17　柱面波迭加效果

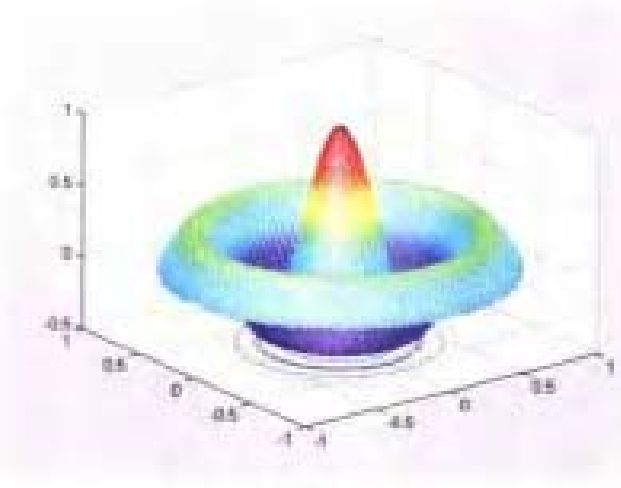


m=0, n=0

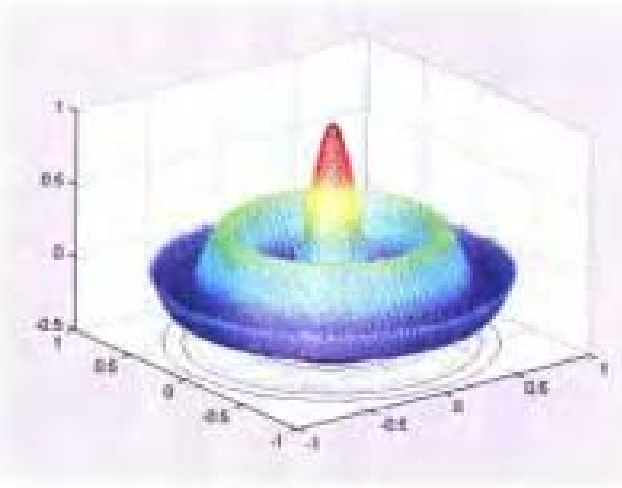


m=0, n=1

图 3.10　圆域本征函数之一

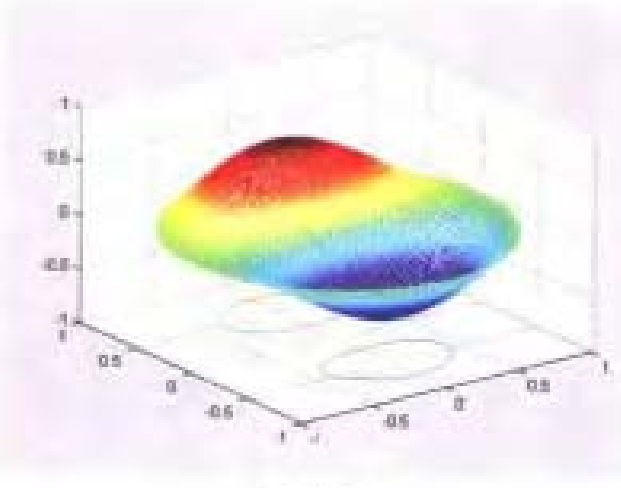


m=0,n=2

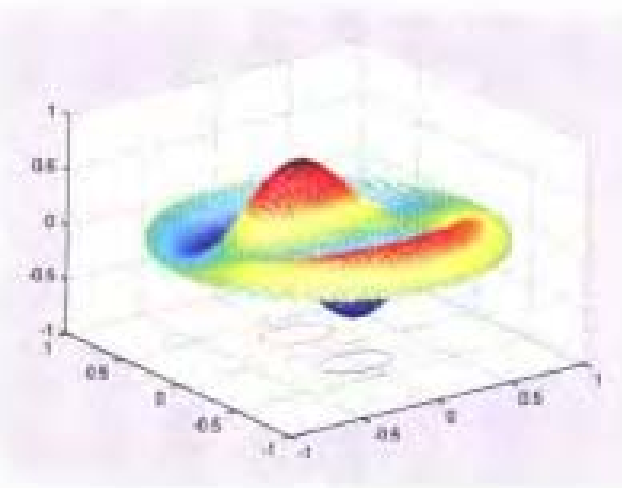


m=0,n=3

图 3.11　圆域本征函数之二

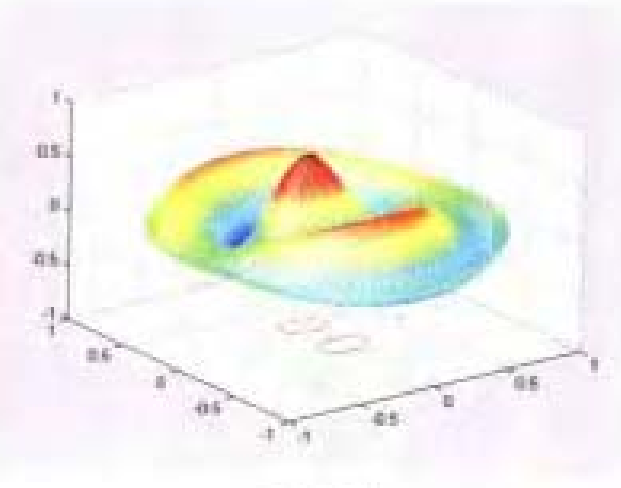


m=1,n=0

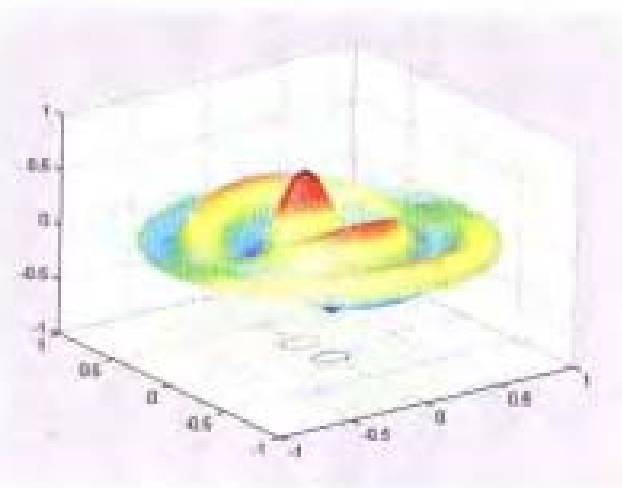


m=1,n=1

图 3.13　圆域本征函数之三

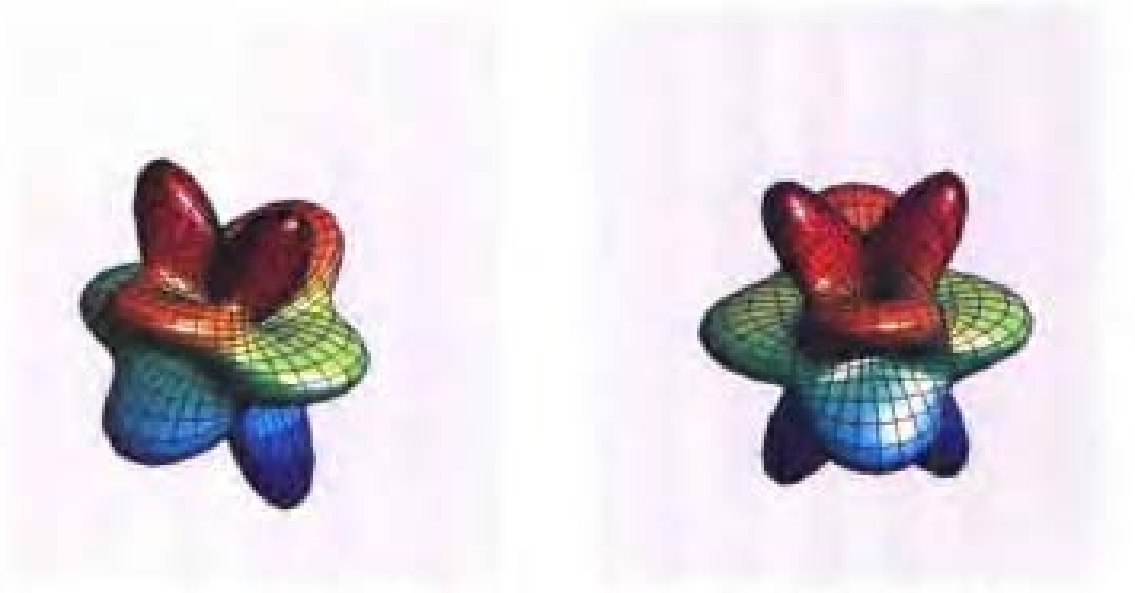


m=1,n=2



m=1,n=3

图 3.14　圆域本征函数



球谐函数($l=6, m=2$)的实部　　球谐函数($l=6, m=2$)的虚部

图 3.25　球谐函数图形

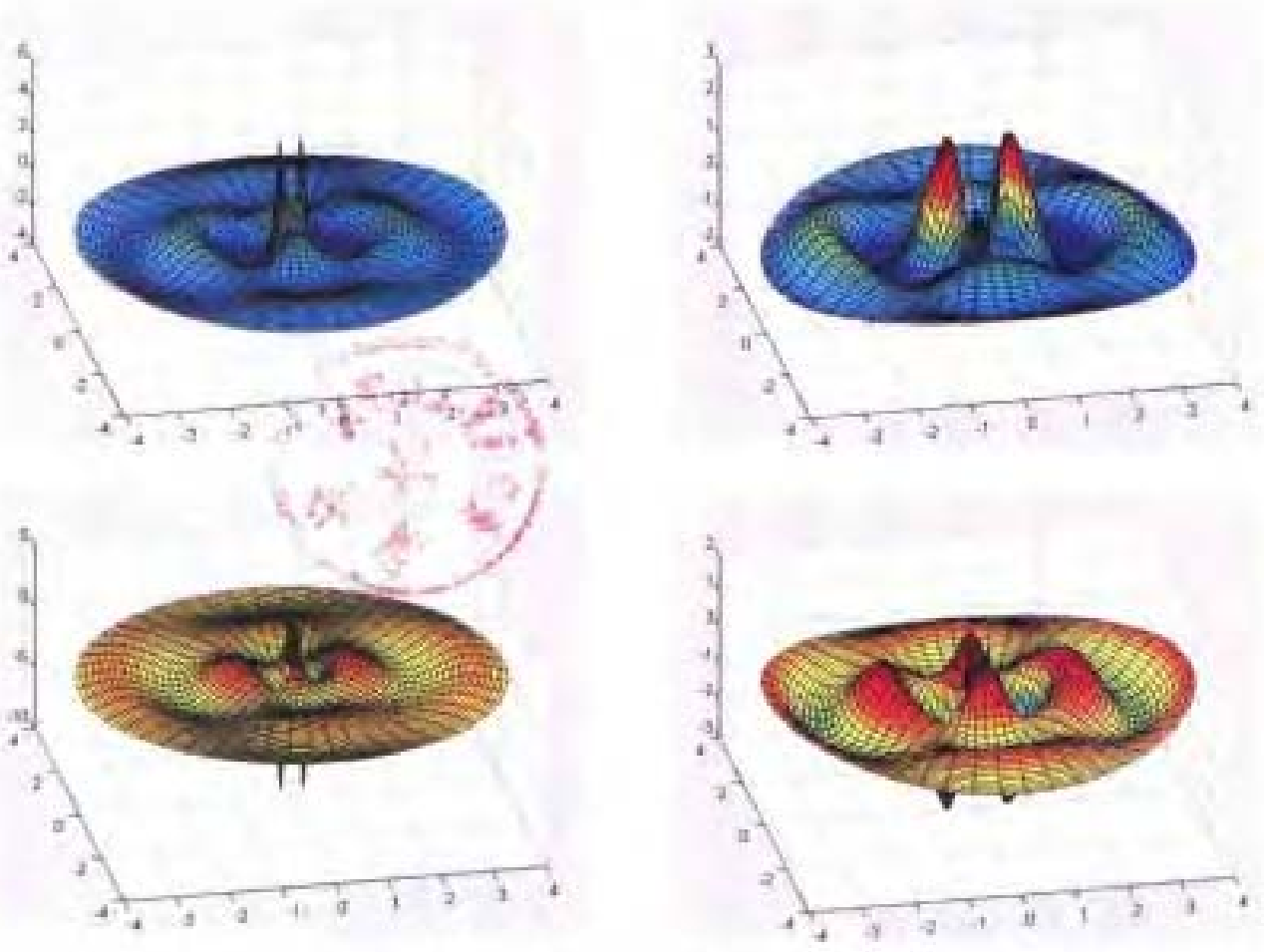


图 6.32　四极声源发射的声波

目　录

第 1 章　函 数 图 形 1

1.1 复变函数图形 1

1.2 特殊函数图形 16

1.2.1 Γ 函数 18

1.2.2 勒让德函数 19

1.2.3 贝塞尔函数 23

1.3 用 MAPLE 指令计算特殊函数 34

第 2 章　傅里叶级数与傅里叶变换 42

2.1 傅里叶级数、傅里叶积分与离散傅里叶变换 42

2.2 傅里叶变换的例题 48

2.3 广义傅里叶级数 53

2.3.1 勒让德函数的母函数 53

2.3.2 贝塞尔函数的母函数 55

2.3.3 平面波展开为球面波的叠加 56

2.3.4 平面波展开为柱面波的叠加 59

第 3 章　本征值函数系与本征振动 62

3.1 一维本征值问题 62

3.1.1 四种常见的本征函数系 62

3.1.2 本征函数系的图像及其运动图像 64

3.2 二维本征值问题 69

3.2.1 矩形区域的本征模与本征振动 69

3.2.2 圆形区域的本征模与本征振动 73

3.2.3 球函数的图形 80

第 4 章　拉普拉斯方程与泊松方程 85

4.1 二维拉普拉斯方程 85

4.1.1 矩形区域的拉普拉斯方程 85

4.1.2 阳光照射的圆柱 90

4.1.3 云与大地之间的电缆 92

4.2 三维拉普拉斯方程 93

4.2.1 静电场中的介质球 93

4.2.2 带有电荷的细圆环的电场 94
4.2.3 均匀圆盘的引力势 99
4.2.4 环形电流的磁感应强度 103
4.2.5 柱体内温度场分布之一 (J_ν 的应用) 115
4.2.6 柱体内温度场分布之二 (J_ν 的应用) 117
4.2.7 柱体内温度场分布之三 (I_0 的应用) 119
4.2.8 圆柱内的温度场 (I_0 的应用) 122
4.2.9 电子透镜 (J_0, I_0 的应用) 124
4.2.10 柱体外的电势 (K_0 的应用) 128
4.3 泊松方程与格林函数 130
4.3.1 矩形区泊松方程 130
4.3.2 球域的格林函数 132
4.3.3 圆域的格林函数 135

第 5 章 热传导方程 **139**
5.1 一维热传导问题 139
5.1.1 无限长细杆的热传导 139
5.1.2 有限长细杆的热传导 141
5.1.3 输运问题 (非齐次方程) 143
5.1.4 第二类边界条件下的细杆导热问题 146
5.2 二维热传导问题 147
5.3 三维热传导问题 149
5.3.1 球体内的热传导问题之一 (j_0 的应用) 149
5.3.2 柱体内的热传导 152
5.3.3 球体内的热传导问题之二 (j_1 的应用) 153

第 6 章 波 动 方 程 **156**
6.1 一维波动问题 156
6.1.1 无限长的弦的自由振动 156
6.1.2 两端固定的弦振动问题之一 (初位移不为零初速为零) 160
6.1.3 两端固定的弦振动问题之二 (初位移为零初速不为零) 167
6.1.4 两端固定的弦振动问题之三 (有阻尼) 170
6.1.5 两端固定的弦振动问题之四 (有驱动力) 172
6.1.6 两端固定的弦振动问题之五 (质量不均匀的弦) 175
6.1.7 非齐次边界条件下弦的振动 177
6.1.8 杆的纵振动 180

6.2 二维波动问题 183
6.2.1 矩形膜的振动 183
6.2.2 圆膜的振动 186
6.3 三维振动问题 187
6.3.1 柱体内的振动 187
6.3.2 柱体外的振动问题之一 188
6.3.3 柱体外的振动问题之二 189
6.3.4 偶极声源 190
6.3.5 四极声源 196

第 7 章 MATLAB 的偏微分方程工具箱 200
7.1 偏微分方程工具箱的功能演示 200
7.1.1 泊松方程 201
7.1.2 亥姆霍兹方程 202
7.1.3 最小表面问题 203
7.1.4 区域分解方法 203
7.1.5 热传导方程 204
7.1.6 波动方程 205
7.1.7 椭圆型方程自适应解法 205
7.1.8 泊松方程快速解法 206
7.2 偏微分方程工具箱的功能 207
7.2.1 可解方程的类型 207
7.2.2 边界条件 208
7.3 工具箱的用户界面窗口 208
7.4 用工具箱解偏微分方程的步骤 210
7.4.1 设置定解问题 210
7.4.2 解方程 210
7.4.3 将结果可视化 211
7.4.4 有特定解法的问题类型 211
7.4.5 如何解三维问题 212
7.5 用图形用户界面窗口的工具栏解方程 212
7.5.1 解方程的步骤 212
7.5.2 一个简单的例子 217
7.6 图形用户界面窗口的菜单 223
7.7 工具箱的指令 243
7.8 例 题 248

7.8.1 散射问题 248
7.8.2 放射性杆中的热传导问题 253
7.8.3 薄膜振动问题 254
7.8.4 本征值问题 256
7.8.5 导电介质中的直流电流 258
7.8.6 热传导问题 259

第 8 章 解微分方程的其他方法 262

8.1 指令 bvp4c 解本征值问题 262
8.1.1 用指令 bvp4c 解一维本征值问题 262
8.1.2 有两个解的边值问题 266
8.1.3 Mathieu 方程的本征值 268
8.1.4 艾登 (emden) 方程 270
8.1.5 Falkner-Skan 边值问题 272
8.1.6 在 $x=0$ 处有突变的问题 274
8.2 用 pdepe 解一维初值边值问题 276
8.2.1 指令 pdepe 的用法 276
8.2.2 例题 279
8.3 差 分 法 292
8.4 有 限 元 法 295
8.4.1 变分原理 295
8.4.2 计算格式 296
8.4.3 强加边界条件处理 301
8.4.4 有限元方法的例子 302

参考文献 312

第1章 函 数 图 形

在学习初等函数时，我们总是先画出它们的图形，因为图形能帮助我们了解函数的性质。复变函数和特殊函数是数学物理方法课程中常用的基本函数，原则上也应该了解这些函数图形的画法。可是画出它们的图形并不容易。因为复变函数的自变数是复数，函数值也是复数，如果要画复变函数的图像，就有四个量需要表示，显然三维图形是不能完整地表现复变函数的图像。因此一般很少看到复变函数的图形。另外画特殊函数的图形涉及到特殊函数的函数值的计算，所以作图也不容易。尤其特殊函数值的计算在数值计算中也是一个很重要的内容。所以本章我们介绍如何用 MATLAB 计算复变函数和特殊函数并画出它们的图形。

1.1 复变函数图形

复变函数与实变函数的的计算在 MATLAB 中是相同的，因为 MATLAB 可以作复数运算。所以不管自变量是实数还是复数，都是将自变量的值直接代入函数表达式中去进行计算。要强调的是，对于多值函数，MATLAB 仅仅是对主值进行计算。

下面是几个计算例子：

(1) 3+5i/(1+i)

(2) (2+3i)∧(1+2i)

(3) sin (4+5i)

(4) ln (3+2i)

(5) log(3+2i)

(6) sinh(3+2i)

(7) e(4 + 3i)

在指令窗中，按下面的格式输入有关的指令，就可以看到结果。在这里，>> 是指令提示符， ans 后面是计算机输出的结果，在它的前面就不会出现指令提示符 >> 。注意自然对数的计算指令是 log，而以 10 为底的对数的计算指令是 log10 。双曲函数的计算指令是 sinh 而不是 sh 。计算以 e 为底的指数的指令是 exp 。

```
>> 3+5i/1+i
ans = 3.0000 + 6.0000i
>>(2+3i)∧(1+2i)
ans = -0.4640 - 0.1995i
>> sin(4+5i)
ans = -56.1623 -48.5025i
```

```
>> log(3+2i)
ans = 1.2825 + 0.5880i
>> log10(3+2i)
ans = 0.5570 + 0.2554i
>> sinh(3+2i)
ans = -4.1689 + 9.1545i
>> exp(4+3i)
ans = -54.0518 + 7.7049i
```

为了能形象地表示复变函数的特性，可以像研究实变函数一样，画出它的图像。 MATLAB 表现四维数据的方法是用三个空间坐标再加上颜色，类似于地球仪用颜色表示海洋与高山一样. 用这个方法可以画出复变函数的图形，从图形上很容易看出复变函数的某些基本性质，例如单值函数是单叶的，而多值函数则是多叶的；复三角函数的绝对值可以大于 1 等。 MATLAB 称这个方法为复变函数的映射。具体的画法是以 xy 平面表示自变量所在的复平面，以 z 轴表示复变函数值的实部，而用颜色来表示复变函数值的虚部. 为了能明确地表示颜色与数值的对应关系，通常还要使用指令 colorbar 来标明各种颜色所代表的数值.

1. MATLAB 画复变函数图形的指令

MATLAB 画复变函数的图形的指令有以下 3 个：

1) **CPLXGRID** 构建一个极坐标的复数数据网格

指令的使用格式是

Z = CPLXGRID(m)

这是一个 $(m+1)\times(2*m+1)$ 的复数的极坐标下的数据网格。

在指令窗输入 edit cplxgrid, 屏幕将显示它的源程序如下，可以参考它来编写自己专用的程序。

```
function   z = cplxgrid(m)
r = (0:m)'/m;
theta = pi*(-m:m)/m;
z = r * exp(i*theta);
```

2) **CPLXMAP** 对复变函数作图

指令的使用格式如下：

CPLXMAP(z,f(z),(optional bound))

它画出复变函数 $f(z)$ 的图形。选项 optional bound 用以选择变量的范围.
源程序如下:

```
function   cplxmap(z,w,B)
  %%z,w,B 分别是自变量, 函数值和函数值的作图范围
blue = 0.2;
x = real(z);
y = imag(z);
u = real(w);
v = imag(w);
  %%如果输入变量数大于两个, 即指定了函数值的范围, 就将不需要的函数值去掉.
if nargin > 2   %%指令 nargin 是输入的变量数目
   k = find((abs(w) > B) | isnan(abs(w)));
  %%找出绝对值大于 B 或者为非数的函数值的元素足标.
   if length(k) > 0   %%如果存在这样的元素, 就要作如下处理.
      u(k) = B*sign(u(k));   %%将范围以外的函数值实部都设为 B
      v(k) = zeros(size(k));   %%将范围以外的函数值虚部都设为 0
      v = v/max(max(abs(v)));   %%函数值虚部归一化
      v(k) = NaN*ones(size(k));   %%设为非数就可以不对它们作图
   end
end

M = max(max(u));   %% 求出函数值的大小范围
m = min(min(u));
axis([-1 1 -1 1 m M]);
caxis([-1 1]);   %%指定颜色轴的范围
s = ones(size(z));
mesh(x,y,m*s,blue*s);   %%画投影图
hold on
surf(x,y,u,v);   %%画表面图
hold off
```

```
colormap(hsv(64))   %%画色轴
```

3) **CPLXROOT** 画复数 n 次根的函数曲面

指令的使用格式有两种：

CPLXROOT(n)

是画复数 n 次根的函数曲面，如果不指定 n 值，则使用默认值 3 画复数的立方根的图形.

CPLXROOT(n,m)

是使用 $m \times m$ 数据网格画复数 n 次根的函数曲面. 如果不指定 m 值，则使用默认值 $m = 20$.

源程序如下：

```
function   cplxroot(n,m)
if nargin < 1,   n = 3; end
  %%输入变量数小于 1，取默认值为 n=3
if nargin < 2,   m = 20; end
  %%输入变量数小于 2，取默认值为 m=20
r = (0:m)'/m;
theta = pi*(-n*m:n*m)/m;
z = r * exp(i*theta);   %%自变量
s = r.^(1/n) * exp(i*theta/n);   %%函数值

surf(real(z),imag(z),real(s),imag(s));   %%画表面图
```

2. 复变函数的图形

下面是几个例子：

1) 幂函数 $z, z^3, z^{1/2}, z^{1/3}$ 的图形

在指令窗输入以下指令：

```
>> z = cplxgrid(30);
>>cplxmap(z,z);
>>colorbar('vert');
>>title('z')
```

所得的复变函数 z 图形如图 1.1 所示。

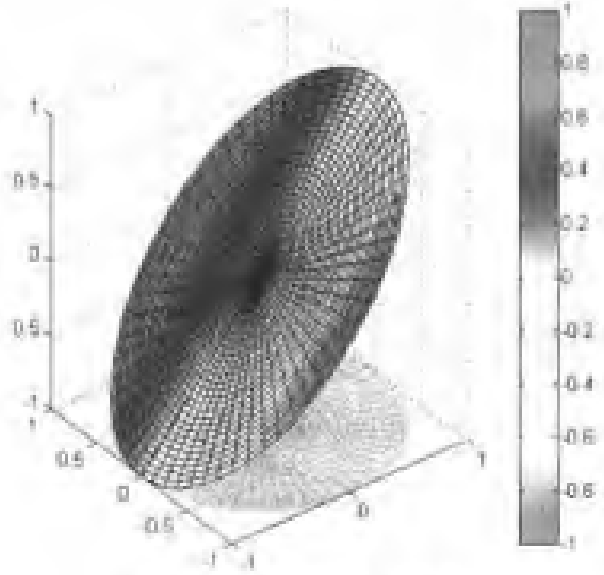


图 1.1 复变函数 z 图形

从图形可看到，自变量 z 的取值在水平面的单位圆内，x 轴是实轴，y 轴是虚轴。画函数 z 时，是以坐标系的 z 轴表示函数的实部，其大小变化范围为 $-1 \sim +1$，形成一个倾斜的圆平面，上面的每一个横条都有相同的实部值。因为平面上的颜色表示虚部，从颜色轴对应的数值看出变化范围也是 $-1 \sim +1$，所以在倾斜的圆平面上从左到右形成了条状的颜色带，正好对应复变函数的虚部是 $+1 \sim -1$，即与平面上 y 轴的正负相对应。

在指令窗输入以下指令：

```
>> z = cplxgrid(30);
>>cplxmap(z,z.^ 3);
>>colorbar('vert');
>>title('z^3')
```

所得的复变函数 z^3 图形如图 1.2 所示。

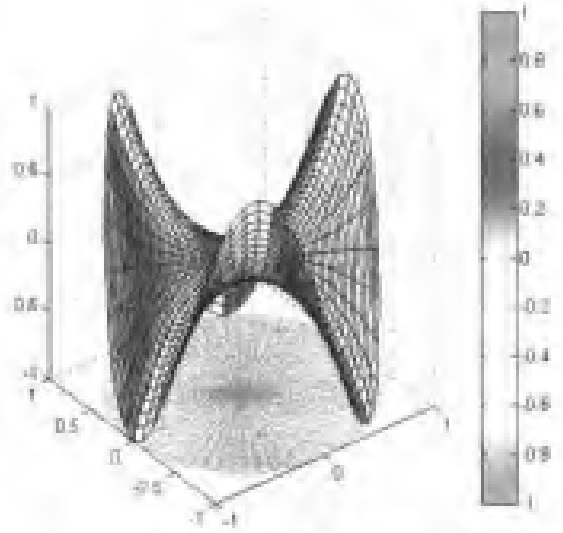


图 1.2 复变函数 z^3 图形

从图形可以看出，函数 z^3 是一个单值函数，它所形成的曲面有三个高峰和三个低谷，对应函数的实部有三个极大值和三个极小值。因为

$$(\rho e^{i\theta})^3 = \rho^3(\cos 3\theta + i\sin 3\theta)$$

在单位圆内，ρ 的最大值 $\rho = 1$，θ 的取值范围是 $-\pi \leqslant \theta \leqslant \pi$，所以，三个高峰对应的是实部的最大值：

$$\cos 3\theta = 1,\ \ \theta = -2\pi/3, 0, 2\pi/3$$

而三个低谷对应的是实部的最小值：

$$\cos 3\theta = -1,\ \ \theta = -\pi/3, \pi/3, \pi$$

虚部的最大值对应于

$$\sin 3\theta = 1,\ \ \theta = -\pi/2, \pi/6, 5\pi/6$$

虚部的最小值对应于

$$\cos 3\theta = -1,\ \ \theta = -5\pi/6, -\pi/6, \pi/2$$

根据颜色轴的颜色，读者很容易找到它们所在的位置。

在指令窗输入以下指令，可得到复变函数 $z^{1/2}$ 图形如图 1.3 所示。

```
>>z = cplxgrid(30);
```

```
>>cplxroot(2);
>>colorbar('vert');
>>title('z^{1/2 }')
```

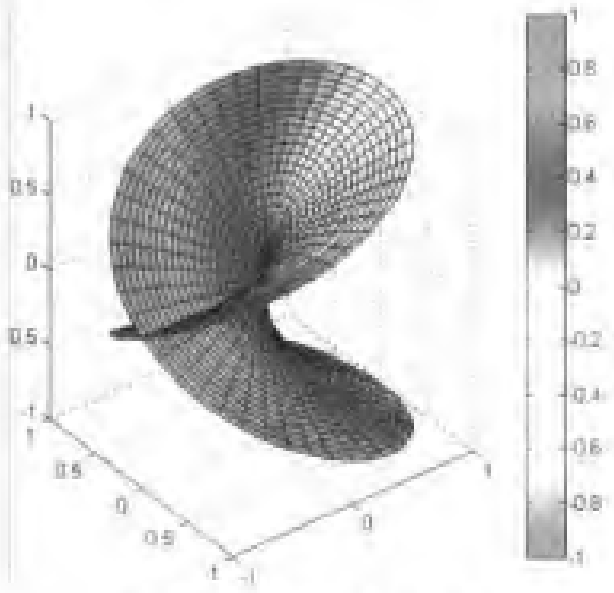


图 1.3 复变函数 $z^{1/2}$ 图形

这是一个双值的函数，函数值的实部变化范围为 $-1 \sim 1$，函数的图像是个二叶的曲面，两个支点分别是 $z=0, z=\infty$，由于单位圆的幅角变化范围是取 $-\pi \sim \pi$，所以曲面的割线是沿 x 轴的负方向。

2) 根式函数 $(z-0.5)^{1/2}$ 的图形

仿照指令 cplxroot 的编程方法，编写如下程序，可以画出根式函数 $(z-0.5)^{1/2}$ 的图形。

```
m=20;    n=2;    r = (0:m)'/m;
theta = pi*(-m:m)/m;
z = r * exp(i*theta)-0.5;
rho=abs(z); theta=angle(z);
w=rho.^(1/n).*exp(i*theta./n);
surf(real(z),imag(z),real(w),imag(w));
hold on
w=rho.^(1/n).*exp(i*pi*i*theta./n);
surf(real(z),imag(z),real(w),imag(w));
```

```
colorbar('vert')
title('(z-0.5)^1/2')
```

所得的复变函数 $(z-0.5)^{1/2}$ 图形如图 1.4 所示.

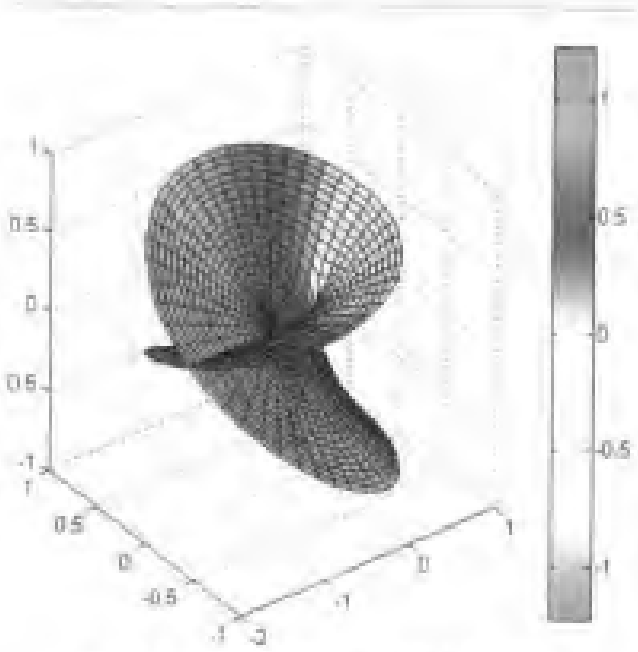


图 1.4 复变函数 $(z-0.5)^{1/2}$ 图形

可以看到函数的图像也是个二叶的曲面，但是，与上一个例子不同的是，从 x 的坐标范围可以看出，两个支点分别是 $z=0.5, z=\infty$，由于单位圆的幅角变化范围是 $-\pi \sim \pi$，所以曲面的割线是沿 x 轴的负方向.

3) 倒数 $1/z$ 的图像

在指令窗输入以下指令

```
>>z = 5*cplxgrid(30);
>>cplxmap(z,1./(z+eps*(abs(z)==0)),5*pi);
>>colorbar('vert');
>>title('1/z')
```

注意，在程序中将分母为零的值换成了一个 MATLAB 所定义的一个极小的数 eps，语句 abs(z)==0 是一个逻辑判断语句，条件为真时其值为 1，否则为零. 所得的复变函数 $1/z$ 图形如图 1.5 所示.

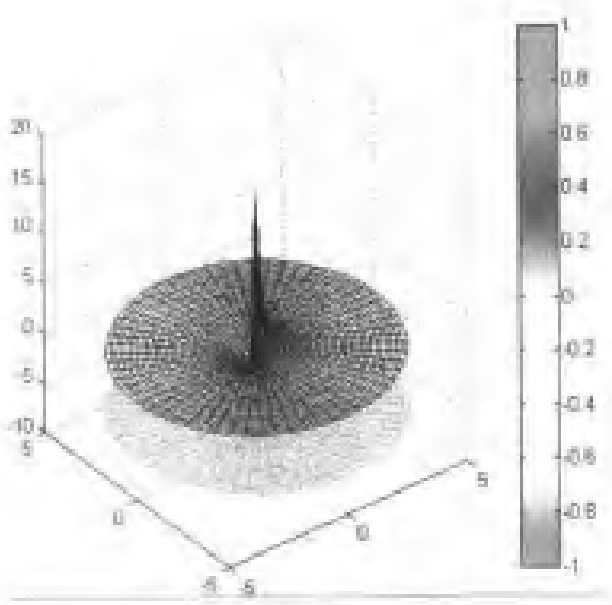


图 1.5 复变函数 $1/z$ 图形

注意，在实轴的正方向，曲面的表现就是我们熟悉的实数的倒数曲线的图像。无论从那个方向 $z \to 0$，函数值都会 $\to \infty$。

复变函数 $\frac{1}{1-z}$ 是级数展开中常用的一个函数，这也是一个倒数函数。图 1.6 是复变函数 $\frac{1}{1-z}$ 的图形。

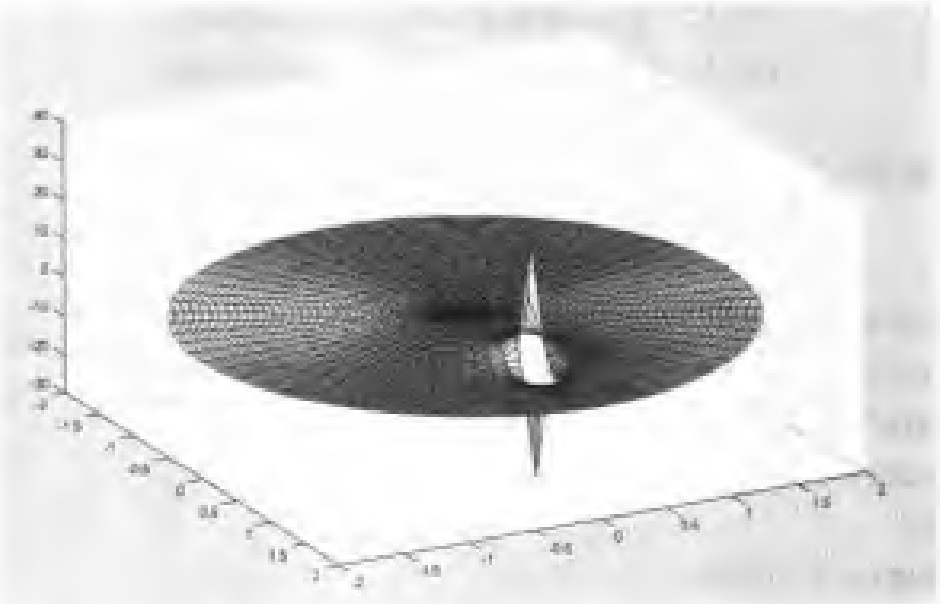


图 1.6 复变函数 $1/(1-z)$ 图形

当 $|z|<1$ 时，它的泰勒展开式是

$$\frac{1}{1-z}=\sum_{k=0}^{\infty}z^k \qquad (|z|<1)$$

当 $|z|>1$ 时，它的罗朗展开式是

$$\frac{1}{z}=-\sum_{k=-\infty}^{-1}z^k \qquad (|z|>1)$$

图 1.7(a) 为 $\frac{1}{1-z}$ 泰勒展开式图形，图 (b) 为 $\frac{1}{1-z}$ 罗朗展开式图形。

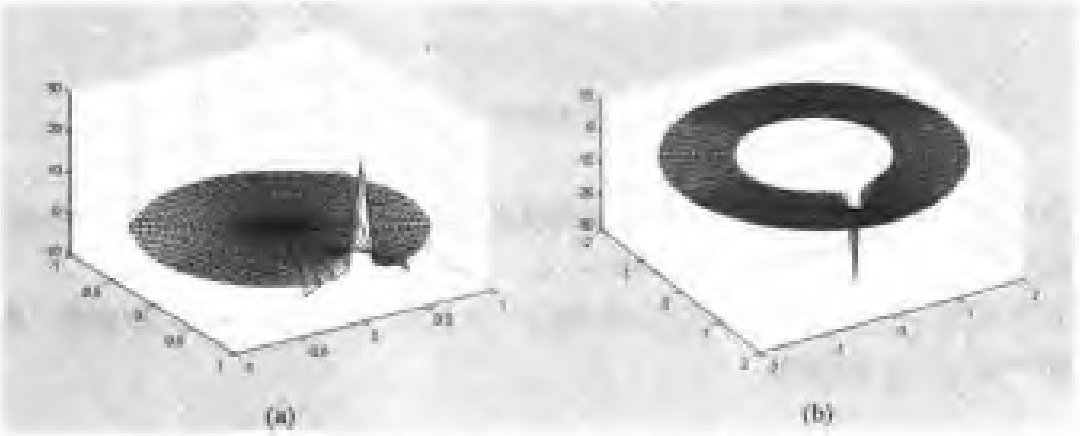


图 1.7 $\frac{1}{1-z}$ 泰勒展开式及罗朗展开式图形
(a) $\frac{1}{1-z}$ 泰勒展开式图形；(b) $\frac{1}{1-z}$ 罗朗展开式图形

画图的程序如下：

```
m=30;
r=(0:2*m)'/m;
theta=pi*(-m:m)/m;
z=r*exp(i*theta);
z(find(z==1))=NaN;
figure
cplxmap(z,1./(1-z))


z1=z;
z1( abs(z1)>=1)=NaN;
```

```
w1=1; u1=1;
for k=1:100
    u1=u1.*z1;
    w1=w1+u1;
end
figure
cplxmap(z1,w1)


z2=z;
z2( abs(z2)<=1)=NaN;
w2=1./z2; u2=1./z2;
for k=1:100
    u2=u2./z2;
    w2=w2+u2;
end
figure
cplxmap(z2,-w2)
```

4) 指数函数 e^z

下面的程序画出了指数函数 e^z 及其泰勒展开式的图形：

```
figure(1)
z=5*cplxgrid(30);
cplxmap(z,exp(z))
figure(2)
w=1; u=1;
for k=1:15
    u=u.*z./k;
    w=w+u;
end
cplxmap(z,w)
```

所得的指数函数 e^z 图形为图 1.8(a)。

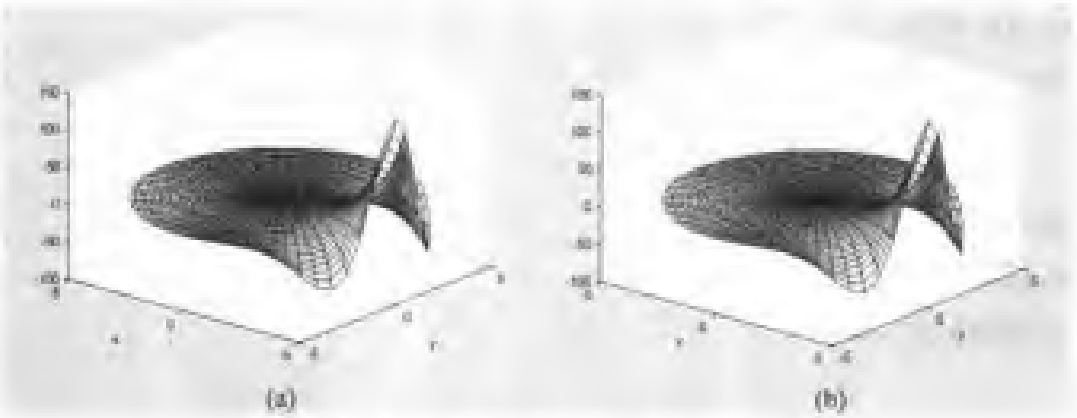


图 1.8 e^z 及其泰勒展开式的图形

(a) 复变函数 e^z 图形; (b) e^z 泰勒展开式的图形

注意，在实轴的正方向上的曲面就是我们熟悉的实数的指数函数曲线的图像。众所周知，e^z 的泰勒展开式是

$$e^z = \sum_{k=0}^{\infty} \frac{1}{k!} z^k$$

图 1.8(b) 图是等式右边 15 项求和的结果。

5) 对数函数 $\ln z$

这是一个多值函数，为了更好地看出函数的变化趋势，下面使用了两个不同程序采用了不同的表现方法，第一个程序是用 z 轴作实部，颜色作虚部，自变量的取值范围是

$$z = \rho e^{i(\theta+2k\pi)} \qquad (k = 0, 1, 2, 3)$$

```
z = cplxgrid(20);
w=log(z+eps*(abs(z)==0));
for k=0:3
   subplot(2,2,k+1);
   w=w+i*2*pi;
   surf(real(z),imag(z),real(w),imag(w));
   caxis([0,8*pi])
   title('ln z')
end
```

所得的复变函数 $\ln z$ 图形如图 1.9 所示。

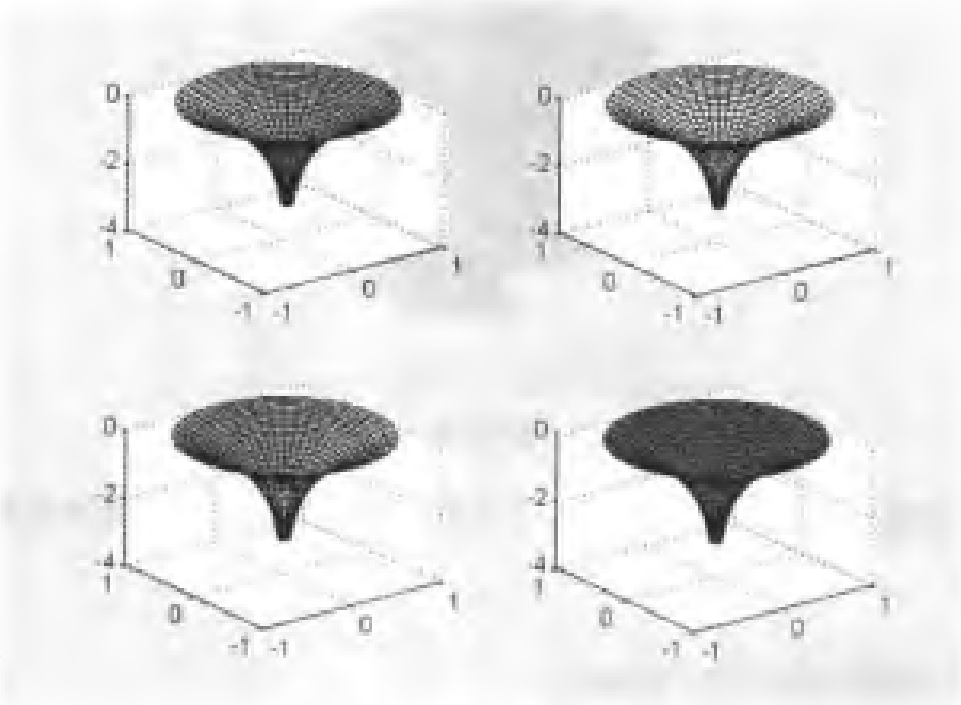


图 1.9 复变函数 $\ln z$ 图形

在这个图像中，为了把不同虚部表示出来，我们将它画成了 4 个图像，它们分别具有不同的颜色，也就是虚部的值是不同的，而实部的形状则相同。注意，在实轴的正方向，曲面的表现就是我们熟悉的实数的对数函数曲线的图像。

第二个程序是用 z 轴作虚部，颜色作实部。

```
z = cplxgrid(20);
w=log(z);
for k=0:3
   w=w+i*2*pi;
   surf(real(z),imag(z),imag(w),real(w));
   hold on
   title('ln z')
end
view(-75,30)
```

所得的复变函数 $\ln z$ 图形如图 1.10 所示。

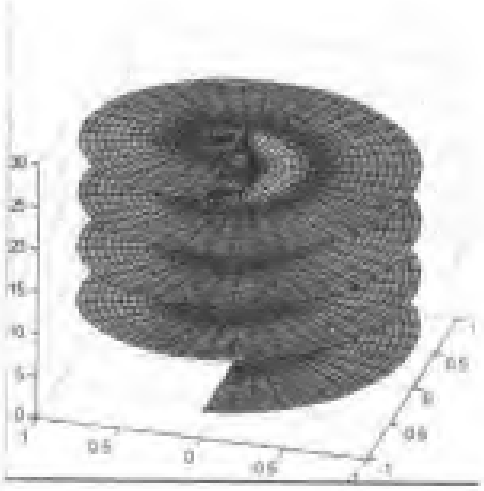


图 1.10 复变函数 $\ln z$ 图形

这个图像很像一个螺旋，和上一个图像完全不同，更能形象地表现多值函数的图像。

6) 三角函数 $\sin z, \cos z, \arctan z$

在指令窗输入以下指令：

```
>>z = 5*cplxgrid(30);
>>cplxmap(z,sin(z));
>>colorbar('vert');
>>title('sin(z)')
```

所得的复变函数 $\sin z$ 图形如图 1.11 所示。

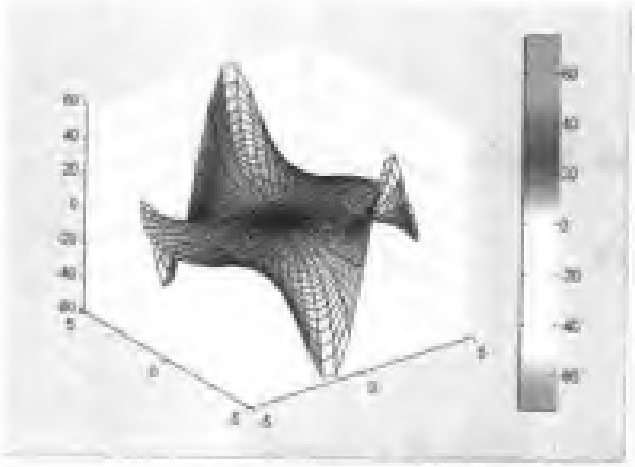


图 1.11 复变函数 $\sin z$ 图形

这些都是单值函数，从图形可看到一个众所周知的结论，$\sin z$ 的绝对值可以大于 1，在图形上 z 轴所表示的函数的实部已经几乎达到 60 。

在指令窗输入以下指令：

```
>> z = 5*cplxgrid(30);
>>cplxmap(z,cos(z));
>>colorbar('vert');
>>title('cos(z)')
```

所得的复变函数 $\cos z$ 图形如图 1.12 所示。

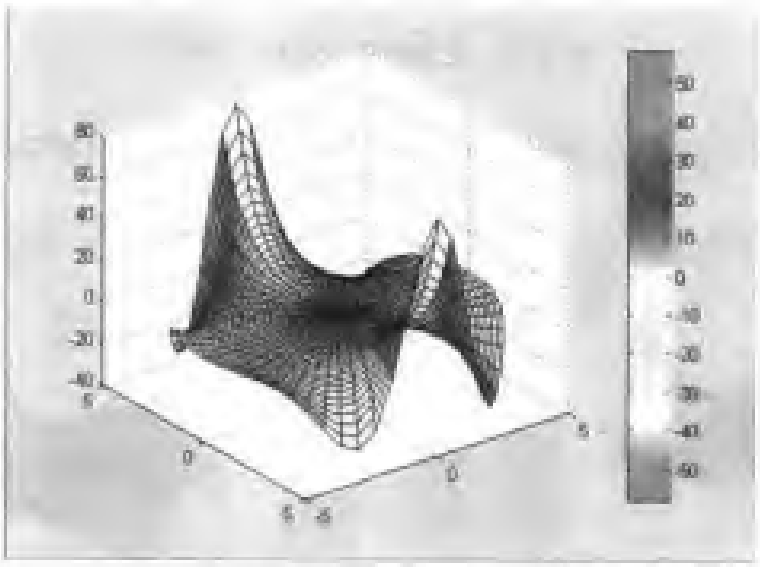


图 1.12 复变函数 $\cos z$ 图形

从图形同样可看到，$\cos z$ 的绝对值可以大于 1，在图形上 z 轴所表示的函数的实部已经几乎达到 80 。

在指令窗输入以下指令：

```
>> z = 5*cplxgrid(30);
>>cplxmap(z,atan(z));
>>colorbar('vert');
>>title('atan(z)')
```

所得的复变函数 $\arctan z$ 图形如图 1.13 所示。

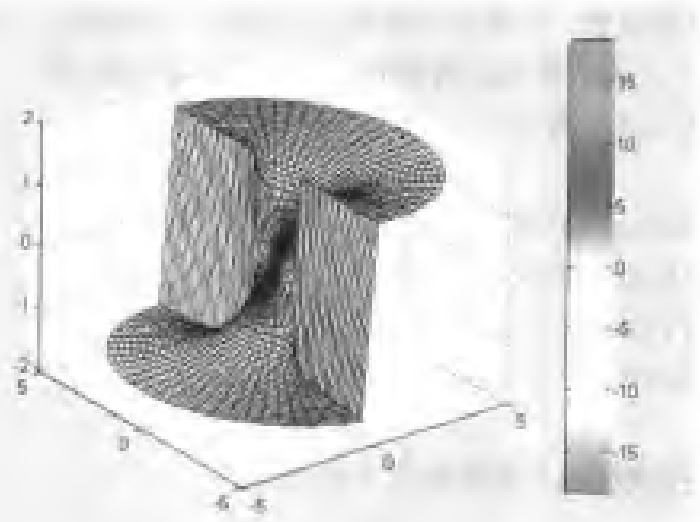


图 1.13 复变函数 arctan z 图形

类似地，可以画出双曲函数 $\sinh z, \cosh z$ 的图像如图 1.14 所示。

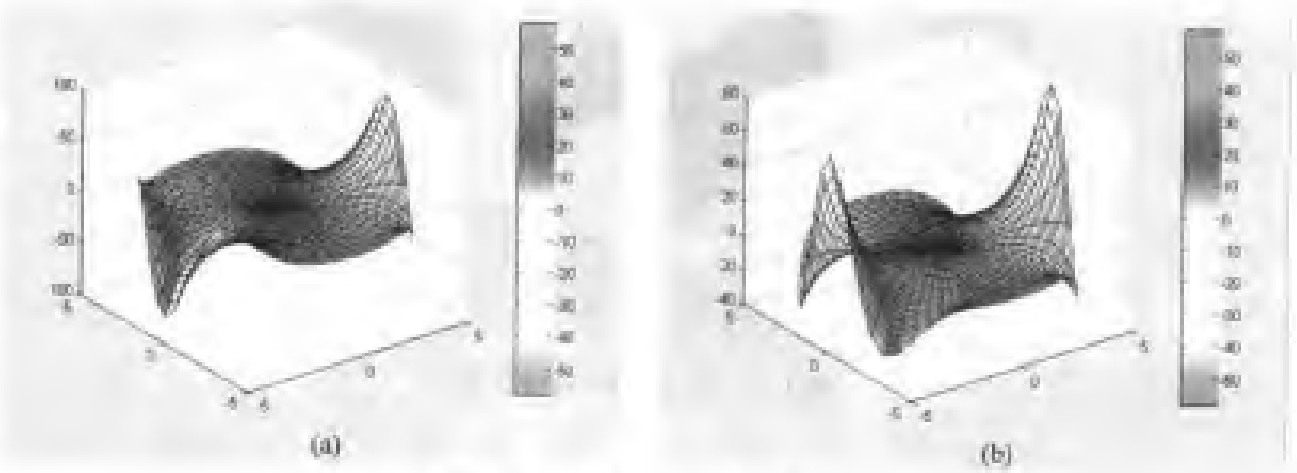


图 1.14 $\sinh z$ 、 $\cosh z$ 图形

(a) 复变函数 $\sinh z$ 图形; (b) 复变函数 $\cosh z$ 图形

1.2 特殊函数图形

在 MATLAB 中，有许多现成的特殊函数可以调用，在指令窗中键入

```
>> help matlab\specfun
```

在屏幕上就会显示已有的特种函数名称如下所示。

```
Specialized math functions.
```

```
airy        - Airy functions.
besselj     - Bessel function of the first kind.
bessely     - Bessel function of the second kind.
besselh     - Bessel functions of the third kind (Hankel function).
besseli     - Modified Bessel function of the first kind.
besselk     - Modified Bessel function of the second kind.
beta        - Beta function.
betainc     - Incomplete beta function.
betaln      - Logarithm of beta function.
ellipj      - Jacobi elliptic functions.
ellipke     - Complete elliptic integral.
erf         - Error function.
erfc        - Complementary error function.
erfcx       - Scaled complementary error function.
erfinv      - Inverse error function.
expint      - Exponential integral function.
gamma       - Gamma function.
gammainc    - Incomplete gamma function.
gammaln     - Logarithm of gamma function.
legendre    - Associated Legendre function.
...
```

这些函数的中文名称如下：

airy	爱里函数
besselj	第一类贝塞尔函数
bessely	第二类贝塞尔函数 (诺伊曼函数)
besselh	第三类贝塞尔函数 (汉开尔函数)
besseli	第一类虚宗量贝塞尔函数
besselk	第二类虚宗量贝塞尔函数
beta	B 函数
betainc	不完全的 B 函数
betaln	B 函数的对数
ellipj	雅可比椭圆函数
ellipke	完全的椭圆积分

erf	误差函数
erfc	余误差函数
erfcx	标度的余误差函数
erfinv	逆误差函数
expint	指数积分函数
gamma	Γ 函数
gammainc	不完全的 Γ 函数
gammaln	Γ 函数的对数
legendre	连带勒让德函数

利用 MATLAB 的帮助系统，就可以知道这些特殊函数的用法，从而计算各种特殊函数的值. 借助于 MATLAB 的符号运算工具箱，还可以调用另一个优秀的数学软件 MAPLE 中更多的特殊函数. 下面是一些例子.

1.2.1 Γ 函数

Γ 函数是常见的一种特殊函数，常用的定义是

$$\Gamma(z)=\int_0^{\infty}\mathrm{e}^{-t}t^{z-1}\mathrm{d}t \qquad (\mathrm{Re}z>0)$$

约定其中的积分变量 $t>0$.

定义

$$\Gamma(z+1)=z\Gamma(z)$$

则可以将 Γ 函数在全平面上作解析延拓，这样定义的 $\Gamma(z)$ 在整个复平面上除去 $z=0,-1,-2,\cdots$ 之外解析。

取 Γ 函数中的自变量为实数，画出它的图像，可以从实数范围直观地看出，Γ 函数在全平面无零点，还可以看出它的奇点分布为 $z=0,-1,-2,\cdots$.

计算 Γ 函数的指令是 **gamma**, 作图的方法很简单，在指令窗中键入以下指令：

```
>> x=-3:0.01:3;
>>y=gamma(x);
>>plot(x,y,'linewidth',4);
>>grid on
>>axis([-3 3 -5 5])
```

可以得到这个图像如图 1.15 所示。

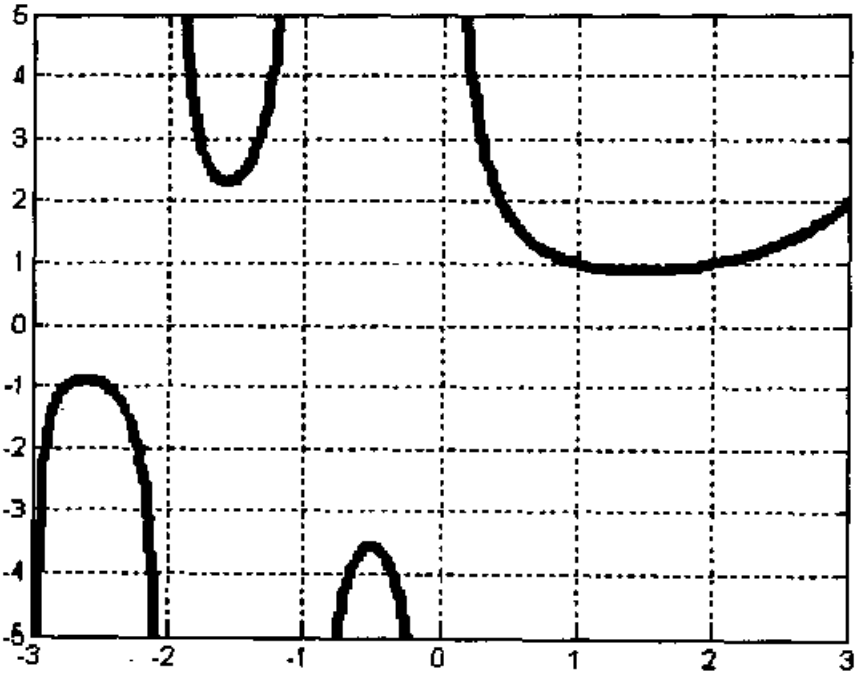


图 1.15 Γ 函数的图形

1.2.2 勒让德函数

l 阶勒让德多项式 $\mathrm{P}_l(x)$ 的定义是

$$\mathrm{P}_l(x)=\sum_{k=0}^{[l/2]}(-1)^k\frac{(2l-2k)!}{2^l k!(l-k)!(l-2k)!}x^{l-2k}\qquad(-1\leqslant x\leqslant 1)$$

其中

$$[l/2]=\begin{cases}l/2, & l=2n\\ (l-1)/2, & l=2n+1\end{cases}\qquad(n=0,1,2,\cdots)$$

当 l 为偶数时，$\mathrm{P}_l(x)$ 中的最低幂项为 x^0 即 $l-2k_m=0$，所以 $k_m=l/2$;
当 l 为奇数时，$\mathrm{P}_l(x)$ 中的最低幂项为 x^1 即 $l-2k_m=1$，所以 $k_m=(l-1)/2$.
连带勒让德函数 $\mathrm{P}_l^m(x)$ 的定义是

$$\mathrm{P}_l^m(x)=(1-x^2)^{\frac{m}{2}}\mathrm{P}_l^{[m]}(x)$$

其中 $l=0,1,2,\cdots$, $m=0,1,2,\cdots,l$, 而 $\mathrm{P}_l^{[m]}(x)$ 是 $P_l(x)$ 的 m 阶导数.

MATLAB 计算连带勒让德函数的指令是

legendre(N,X)

在给定 N,X 的值以后，它将计算所有 N 阶连带勒让德函数在 X 处的函数值。如果 X 是矢量，所得的结果 P 是矩阵，而 P(m+1,i) 则是连带勒让德函数 $P_l^{[m]}(x)$ 在 X(i) 处的函数值。

例如，在指令窗中输入

```
>>legendre(2, 0.0:0.1:0.2)
```

屏幕上显示的矩阵是

```
-0.5000    -0.4850    -0.4400
      0    -0.2985    -0.5879
 3.0000     2.9700     2.8800
```

它表示的结果是

	$x=0$	$x=0.1$	$x=0.2$
$m=0$	$P_2^0(0)=-0.5000$	$P_2^0(0.1)=-0.4850$	$P_2^0(0.2)=-0.4400$
$m=1$	$P_2^1(0)=0$	$P_2^1(0.1)=-0.2985$	$P_2^1(0.2)=-0.5879$
$m=2$	$P_2^2(0)=3.0000$	$P_2^2(0.1)=2.9700$	$P_2^2(0.2)=2.8800$

要画出前 6 个勒让德多项式的图像，可在指令窗中键入

```
>>x=0:0.01:1;
>>y1=legendre(1,x);
>>y2=legendre(2,x);
>>y3=legendre(3,x);
>>y4=legendre(4,x);
>>y5=legendre(5,x);
>>y6=legendre(6,x);
>>plot(x,y1(1,:), x,y2(1,:), x,y3(1,:),...
x,y4(1,:), x,y5(1,:), x,y6(1,:))
>>title('勒让德多项式')
```

所得的图像如图 1.16 所示。

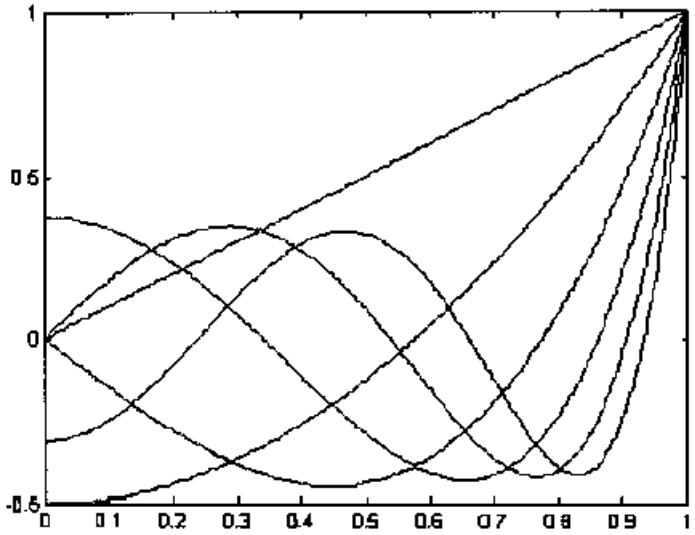


图 1.16 勒让德多项式的图形

同样可以画出所有 3 阶的连带勒让德函数的图形. 在指令窗中键入

```
>>x=0:0.01:1;
>>y=legendre(3,x);
>>plot(x, y(1,:),'-',x, y(2,:),'-.',x, y(3,:),':',x, y(4,:),'--')
>>legend('P_3^0','P_3^1','P_3^2','P_3^3');
```

得到的图形如图 1.17 所示.

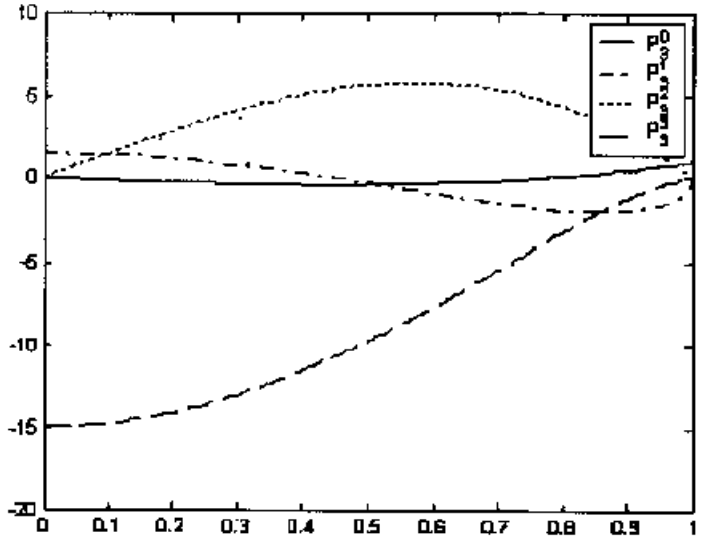


图 1.17 三阶连带勒让德函数的图形

勒让德函数实际上是以 $\cos\theta$ 为变量的函数，所以也可以在极坐标下作图．下面画出 $P_1^{(0,1)}, P_2^{(0,1,2)}, P_3^{(0,1,2,3)}$ 在极坐标下的图形．所用的程序如下：

```
rho=legendre(1,cos(0:0.1:2*pi));
t=0:0.1:2*pi;
rho1=legendre(2,cos(0:0.1:2*pi));
rho2=legendre(3,cos(0:0.1:2*pi));
subplot(3,4,1)
polar(t,rho(1,:))
subplot(3,4,2)
polar(t,rho(2,:))

subplot(3,4,5)
polar(t,rho(2,:))
subplot(3,4,6)
polar(t,rho(2,:))
subplot(3,4,7)
polar(t,rho(2,:))

subplot(3,4,9)
polar(t,rho(2,:))
subplot(3,4,10)
polar(t,rho(2,:))
subplot(3,4,11)
polar(t,rho(2,:))
subplot(3,4,12)
polar(t,rho(2,:))
```

所得的图形如图 1.18，第一行是 P_1 的两个图形，第二行是 P_2 的三个图形，第三行是 P_3 的四个图形．

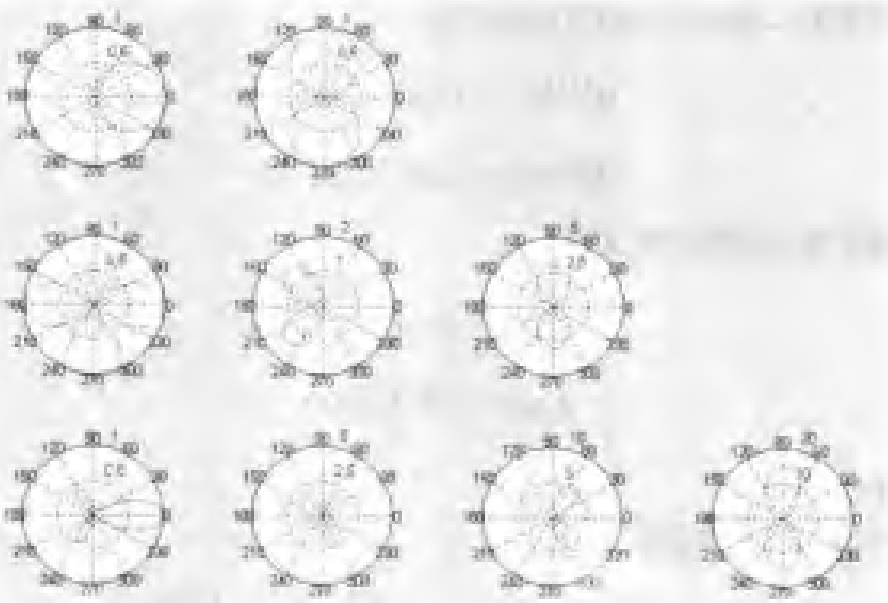


图 1.18 连带勒让德函数的极坐标图形

1.2.3 贝塞尔函数

1. 贝塞尔函数的定义

ν 阶贝塞尔函数的定义是

$$\mathrm{J}_\nu(x)=\sum_{k=0}^{\infty}\frac{(-1)^k}{k!\,\Gamma(\nu+k+1)}\left(\frac{x}{2}\right)^{\nu+2k}$$

其中 $\Gamma(x)$ 是 Γ 函数.

ν 阶诺伊曼函数的定义是

$$\mathrm{N}_\nu(x)=\frac{\mathrm{J}_\nu(x)\cos\nu\pi-\mathrm{J}_{-\nu}(x)}{\sin\nu\pi}$$

当 ν 取整数 m 时，得 m 阶诺伊曼函数

$$\begin{aligned}\mathrm{N}_m(x) &= \lim_{\nu\to m}\mathrm{N}_\nu(x)=\lim_{\nu\to m}\frac{\mathrm{J}_\nu(x)\cos\nu x-\mathrm{J}_{-\nu}(x)}{\sin\nu\pi}\\ &= \frac{2}{\pi}\left(\ln\frac{x}{2}+C\right)\mathrm{J}_m(x)-\frac{1}{\pi}\sum_{n=0}^{m-1}\frac{(m-n-1)!}{n!}\left(\frac{x}{2}\right)^{-m+2n}\\ &\quad -\frac{1}{\pi}\sum_{n=m}^{\infty}\frac{(-1)^{n-m}}{n!(n-m)!}\left(1+\frac{1}{2}+\cdots+\frac{1}{n}\right)\left(\frac{x}{2}\right)^{-m+2n}\\ &\quad -\frac{1}{\pi}\sum_{n=m+1}^{\infty}\frac{(-1)^{n-m}}{n!(n-m)!}\left(1+\frac{1}{2}+\cdots+\frac{1}{n-m}\right)\left(\frac{x}{2}\right)^{-m+2n}\end{aligned}$$

当 $m=0$ 时没有 $\sum_{n=0}^{m-1}$ 项.

第一种和第二种汉开尔函数的定义是

$$\mathrm{H}_{\nu}^{(1)}(x)=\mathrm{J}_{\nu}(x)+\mathrm{i}\mathrm{N}_{\nu}(x)$$

$$\mathrm{H}_{\nu}^{(2)}(x)=\mathrm{J}_{\nu}(x)-\mathrm{i}\mathrm{N}_{\nu}(x)$$

虚宗量贝塞尔函数的定义是

$$\mathrm{I}_{\nu}(x)=\mathrm{i}^{-\nu}\mathrm{J}_{\nu}(\mathrm{i}x)$$

$$\mathrm{I}_{-\nu}(x)=\mathrm{i}^{\nu}\mathrm{J}_{-\nu}(\mathrm{i}x)$$

它们是实变量函数.

定义虚宗量汉开尔函数为

$$\mathrm{K}_{\nu}=\frac{\mathrm{i}\pi}{2}\mathrm{e}^{\mathrm{i}\frac{\pi}{2}\nu}\mathrm{H}_{\nu}^{(1)}(\mathrm{i}x)=\frac{\pi}{2}\frac{\mathrm{I}_{-\nu}(x)-\mathrm{I}_{\nu}(x)}{\sin\nu\pi}$$

当 ν 取整数 m 时，得 m 阶虚宗量贝塞尔函数为

$$\mathrm{K}_{\nu}=\lim_{\nu\to m}\frac{\mathrm{i}\pi}{2}\mathrm{e}^{\mathrm{i}\frac{\pi}{2}\nu}\mathrm{H}_{\nu}^{(1)}(\mathrm{i}x)=\lim_{\nu\to m}\frac{\pi}{2}\frac{\mathrm{I}_{-\nu}(x)-\mathrm{I}_{\nu}(x)}{\sin\nu\pi}$$

注意，存在以下关系：

$$\mathrm{J}_{-m}=(-1)^{m}\mathrm{J}_{m}$$

$$\mathrm{I}_{-m}=\mathrm{I}_{m}$$

2. 计算贝塞尔函数的指令

MATLAB 有 5 种计算贝塞尔函数的指令，

计算指令	所计算的函数
besselj	计算第一类贝塞尔函数，简称贝塞尔 (Bessel) 函数
bessely	计算第二类贝塞尔函数，即诺伊曼 (Neumann) 函数
besselh	计算第三类贝塞尔函数，即汉开尔 (Hankel) 函数
besseli	计算第一类虚宗量贝塞尔函数 (虚宗量贝塞尔函数)
besselk	计算第二类虚宗量贝塞尔函数 (虚宗量汉开尔函数)

分别介绍如下：

1) besselj　计算第一类贝塞尔函数的指令

指令的使用格式是

J = besselj(ν, z)

计算 ν 阶第一类贝塞尔函数 $J_\nu(z)$ 的值， ν 不一定是整数，但必须是实数. 自变量 z 可以是复数，如果 z 是正数，所得结果是实数.

`J = besselj(`$\nu, z, 1$`)` 计算的结果是 $J_\nu(z)$ 除以 $e^{-|Im z|}$.

如果 ν 和 z 是相同大小的矩阵，所得结果也是大小相同的矩阵. 如果其中有一个是标量，那么在结果中它将扩展成与另一个量的相同的大小. 如果一个是行向量而另一个是列向量，则所得结果是二维的函数值表.

`[J,ierr] = besselj(`ν, z`)` 返回的结果是具有误差标志的矩阵. 其中 ierr 的各个值所代表的含义是

ierr = 1	不合理的结果.
ierr = 2	计算溢出，返回 Inf .
ierr = 3	在化简某个量时丧失了某些精度.
ierr = 4	z 和 ν 太大，精度完全丧失.
ierr = 5	计算不收敛，返回的结果是 NaN .

例如 `besselj(3:9,(0:.2:10)')` 可以产生 Abramowitz 和 Stegun 的数学手册 (Handbook of Mathematical Functions) 中第 398 页上的整个函数值表.

MATLAB 的演示程序 `membrane` 也是使用指令 besselj 来计算在 MathWorks Logo (MathWorks 公司的图标，表示 L 形薄膜的振动) 中用到的分数阶的贝塞尔函数. 在指令窗中输入 `membrane` 即可运行这个程序.

2) bessely 计算第二类贝塞尔函数 (诺伊曼函数)

指令的使用格式是

N = bessely(ν, z**)**

计算 ν 阶第二类贝塞尔函数 $N_\nu(z)$ 的值， ν 不一定是整数，但必须是实数. 自变量 z 可以是复数，如果 z 是正数，所得结果是实数.

如果 ν 和 z 是相同大小的矩阵，所得结果也是大小相同的矩阵. 如果其中有一个是标量，那么在结果中它将扩展成与另一个量相同的大小. 如果一个是行向量而另一个是列向量，则所得结果是二维的函数值表.

`N = bessely(`$\nu, z, 1$`)` 计算的结果是 $N_\nu(z)$ 乘以 $e^{-|Im z|}$.

`[N,ieer] = bessely(`ν, z`)` 返回的结果是具有误差标志的矩阵. 其中 ierr 的各个值所代表的含义是

ierr = 1	不合理的结果.
ierr = 2	计算溢出，返回 Inf .
ierr = 3	在化简某个量时丧失了某些精度.
ierr = 4	z 和 ν 太大，精度完全丧失.
ierr = 5	计算不收敛，返回的结果是 NaN .

例如 bessely(3:9,(0:.2:10)') 可以产生 Abramowitz 和 Stegun 的数学手册 (Handbook of Mathematical Functions) 中第 399 页上的整个函数值表.

3) besselh 计算第三类贝塞尔函数 (汉开尔函数)

指令的使用格式是

$$\mathbf{H} = \mathbf{bessekh}(\nu, k, z)$$

其中 $k = 1,2$, 是对复数矩阵 z 的每个元素分别计算第一类汉开尔函数 $\mathrm{H}_\nu^{(1)}(z)$ 或第二类汉开尔函数 $\mathrm{H}_\nu^{(2)}(z)$ 的值. 而 H = bessekh(ν, z) 是默认 $k = 1$.

H = bessekh($\nu, 1, z, 1$) 计算的结果是 $\mathrm{H}_\nu^{(1)}(z)$ 除以 $\mathrm{e}^{-\mathrm{i}z}$.

H = bessekh($\nu, 2, z, 1$) 计算的结果是 $\mathrm{H}_\nu^{(2)}(z)$ 除以 $\mathrm{e}^{\mathrm{i}z}$.

如果 ν 和 z 是相同大小的矩阵, 所得结果也是大小相同的矩阵. 如果其中有一个是标量, 那么在结果中它将扩展成与另一个量相同的大小. 如果一个是行向量而另一个是列向量, 则所得结果是二维的函数值表.

[H,ierr] = besselh(ν, z) 返回的结果是具有误差标志的矩阵. 其中 ierr 的各个值所代表的含义是

ierr = 1 不合理的结果.
ierr = 2 计算溢出, 返回 Inf .
ierr = 3 在化简某个量时丧失了某些精度.
ierr = 4 z 和 ν 太大, 精度完全丧失.
ierr = 5 计算不收敛, 返回的结果是 NaN .

汉开尔函数与贝塞尔函数的关系是

besselh(ν,1,z) = besselj(ν, z) + i*bessely(ν, z)
besselh(ν,2,z) = besselj(ν, z) - i*bessely(ν, z)

例如下面的程序

```
[X,Y] = meshgrid(-4:0.025:2,-1.5:0.025:1.5);
H = besselh(0,1,X+i*Y);
contour(X,Y,abs(H),0:0.2:3.2), hold on
contour(X,Y,(180/pi)*angle(H),-180:10:180); hold off
```

可以产生 Abramowitz 和 Stegun 的数学手册 (Handbook of Mathematical Functions) 中第 359 页上的汉开尔函数 $\mathrm{H}_0^{(1)}(z)$ 的模与相位的等值线. 其图形如图 1.19 所示.

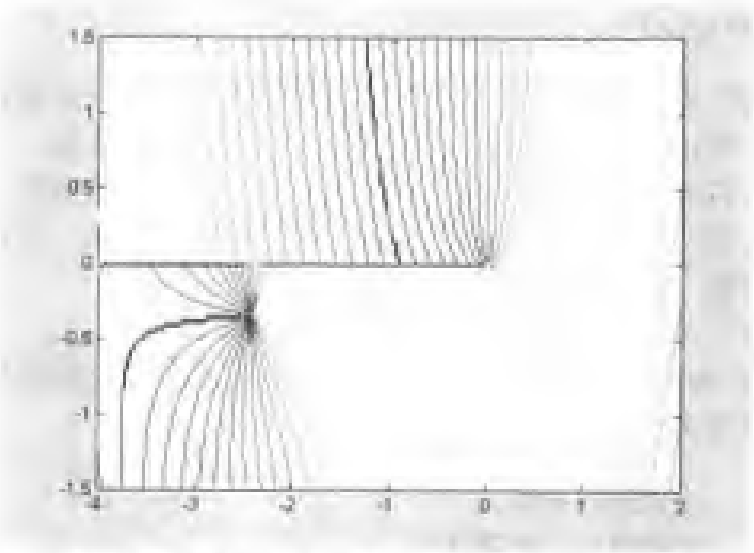


图 1.19 汉开尔函数 $H_0^{(1)}(z)$ 的模与相位的等值线

4) besseli 计算第一类虚宗量贝塞尔函数 (虚宗量贝塞尔函数)

指令的使用格式是

$\mathbf{I = besseli}(\nu, z)$

计算 ν 阶第一类虚宗量贝塞尔函数 $I_\nu(z)$ 的值. ν 不一定是整数, 但必须是实数. 自变量 z 可以是复数. 如果 z 是正数, 所得结果是实数.

如果 ν 和 z 是相同大小的矩阵, 所得结果也是大小相同的矩阵. 如果其中有一个是标量, 那么在结果中它将扩展成与另一个量相同的大小. 如果一个是行向量而另一个是列向量, 则所得结果是二维的函数值表.

`I = besseli(ν,z,1)` 计算的结果是 $I_\nu(z)$ 除以 $e^{-|\mathrm{Re}z|}$.

`[I,ierr] = besseli(ν,z)` 返回的结果是具有误差标志的矩阵. 其中 ierr 的各个值所代表的含义是

ierr = 1 不合理的结果.
ierr = 2 计算溢出, 返回 Inf.
ierr = 3 在化简某个量时丧失了某些精度.
ierr = 4 z 和 ν 太大, 精度完全丧失.
ierr = 5 计算不收敛, 返回的结果是 NaN.

例如 `besseli(3:9,(0:.2:10)')` 可以产生 Abramowitz 和 Stegun 的数学手册 (Handbook of Mathematical Functions) 中第 423 页上的整个函数值表.

5) besselk 计算第二类虚宗量贝塞尔函数 (虚宗量汉开尔函数)

指令的使用格式是

`K = besselk(`ν, z`)`

计算 ν 阶第二类虚宗量贝塞尔函数 $K_\nu(z)$ 的值。ν 不一定是整数，但必须是实数。自变量 z 可以是复数，如果 z 是正数，所得结果是实数。

如果 ν 和 z 是相同大小的矩阵，所得结果也大小相同的矩阵。如果其中有一个是标量，那么在结果中它将扩展成与另一个量相同的大小。如果一个是行向量而另一个是列向量，则所得结果是二维的函数值表。

`K = besselk(`$\nu, z, 1$`)` 计算的结果是 $K_\nu(z)$ 除以 e^z 。

`[K,ierr] = besselk(`ν, z`)` 返回的结果是具有误差标志的矩阵，其中 ierr 的各个值所代表的含义如下：

ierr = 1	不合理的结果。
ierr = 2	计算溢出，返回 Inf 。
ierr = 3	在化简某个量时丧失了某些精度。
ierr = 4	z 和 ν 太大，精度完全丧失。
ierr = 5	计算不收敛，返回的结果是 NaN 。

例如 `besselk(3:9,(0:.2:10)')` 可以产生 Abramowitz 和 Stegun 的数学手册 (Handbook of Mathematical Functions) 中第 424 页上的整个函数值表。

6) 半整数阶的贝塞尔函数

半整数阶的贝塞尔函数常用以下几种形式：

球贝塞尔函数

$$\mathrm{j}_l(x) = \sqrt{\frac{\pi}{2x}}\mathrm{J}_{l+1/2}(x)$$

$$\mathrm{j}_{-l}(x) = \sqrt{\frac{\pi}{2x}}\mathrm{J}_{-l+1/2}(x)$$

球诺伊曼函数

$$\mathrm{n}_l(x) = \sqrt{\frac{\pi}{2x}}\mathrm{Y}_{l+1/2}(x)$$

球汉开尔函数

$$\mathrm{h}_l^{(1)}(x) = \sqrt{\frac{\pi}{2x}}\mathrm{H}_{l+1/2}^{(1)}(x)$$

$$\mathrm{h}_l^{(2)}(x) = \sqrt{\frac{\pi}{2x}}\mathrm{H}_{l+1/2}^{(2)}(x)$$

它们都是用前面的 5 种贝塞尔函数之一构成的，所以很容易利用上面的指令把它们计算出来。

3. 例题

在指令窗中键入

```
>> y=besselj(0:3,(0:.2:10)');
>> figure(1)
>> plot((0:.2:10)',y)
>> legend('J_0 ', 'J_1 ', 'J_2 ', 'J_3 ')
```

可得到 $J_{0,1,2,3}$ 的曲线如图 1.20 所示。

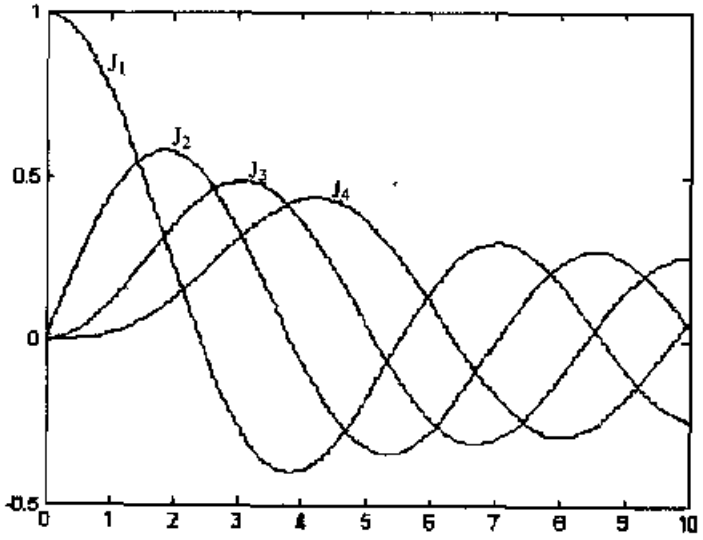


图 1.20 贝塞尔函数 $J_{0,1,2,3}$ 的图形

有时候需要寻找贝塞尔函数的零点，这个任务可以用内插法来解决。例如要找 $0 \leqslant x \leqslant 50$ 之间的零点，可用以下的程序。

```
x=0:0.05:50;
y=besselj(0,x);
LD=[];
for k=1:1000,
   if y(k)*y(k+1)<0
      h=interp1(y(k:k+1), x(k:k+1),0);
      LD=[LD,h]
   end
end
LD
```

程序运行以后会在屏幕上显示 16 个零点的位置如下：

```
LD =
  Columns 1 through 5
    2.4049    5.5201    8.6537   11.7915   14.9309
  Columns 6 through 10
   18.0711   21.2116   24.3525   27.4935   30.6346
  Columns 11 through 15
   33.7758   36.9171   40.0584   43.1998   46.3412
  Column 16
   49.4826
```

贝塞尔函数和其他内建的函数一样，可以用于建造各种函数，下面程序中的函数 y，就是以 x 为变量的零阶贝塞尔函数. 所以也可以用下面的程序来查找贝塞尔函数的前若干个零点，程序中是找 20 个零点. 注意，这时会出现相同的值，只有不同值的零点才是新的零点.

```
j=1;      a=[ ];
y=inline('besselj(0,x)','x'); for k=1:40
    while ~(besselj(0,j)*besselj(0,j+1)>0)
       j=j+1;
    end
    q=fzero(y,j);
    j=j+1;
    a=[a,q]
    k=k+1;
end
```

在指令窗中键入下面的指令，可得到诺伊曼函数 $N_{0,1}$ 的曲线如图 1.21 所示.

```
>> y=bessely(0:1,(0:.2:10)');
>> plot((0:.2:10)',y)
>> grid on
```

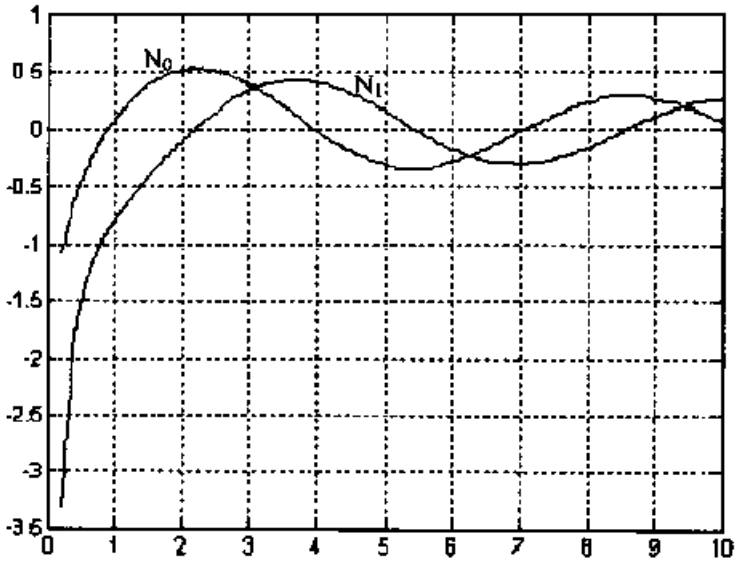


图 1.21 诺伊曼函数 $N_{0,1}$ 的图形

在指令窗中键入

```
>>I=besseli(0:1,(0.1:0.1:3)');
>>plot((0.1:0.1:3)',I)
```

可得到虚宗量贝塞尔函数 $I_{0,1}$ 的曲线如图 1.22 所示。

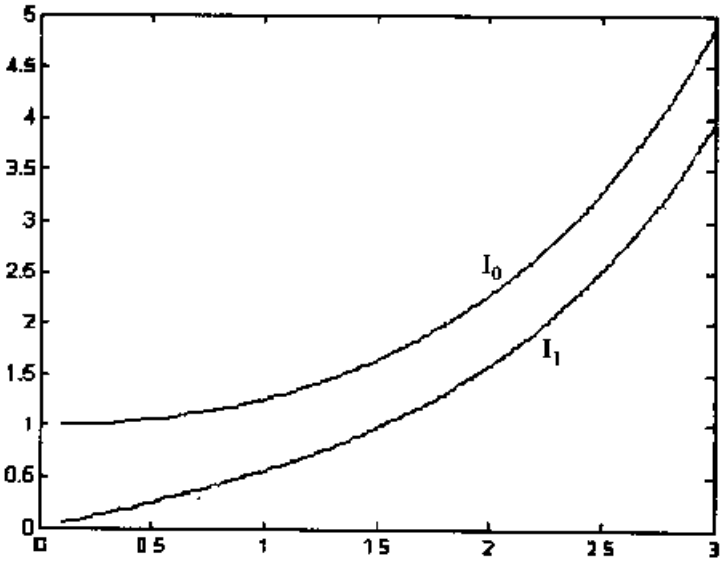


图 1.22 虚宗量贝塞尔函数 $I_{0,1}$ 的图形

在指令窗中键入

```
>> K=besselk(0:1,(0.1:0.1:3)');
>> plot((0.1:0.1:3)',k)
```

可得到虚宗量汉开尔函数 $K_{0,1}$ 的曲线如图 1.23 所示.

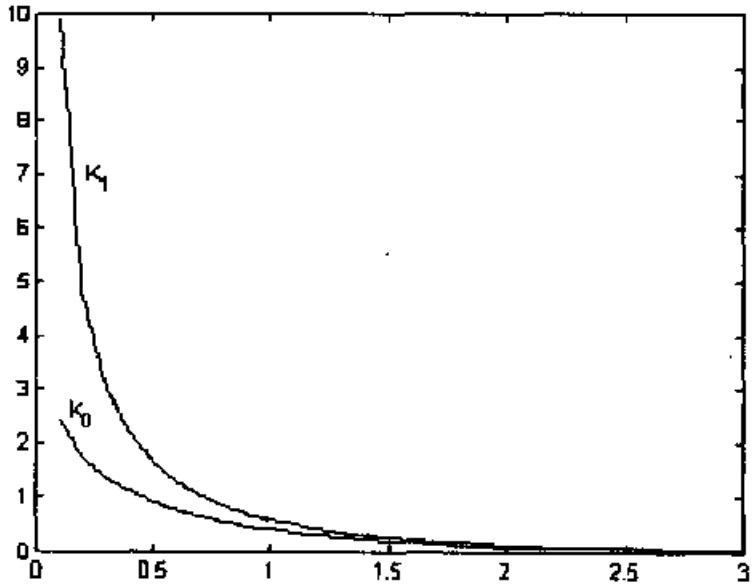


图 1.23 虚宗量汉开尔函数 $K_{0,1}$ 的图形

在指令窗中键入

```
x=eps:0.2:15;
y1=sqrt(pi/2./x).*besselj(1/2,x);
y2=sqrt(pi/2./x).*besselj(3/2,x);
y3=sqrt(pi/2./x).*besselj(5/2,x);
y4=sqrt(pi/2./x).*besselj(7/2,x);
plot(x,y1,x,y2,x,y3,x,y4)
```

可得球贝塞尔函数 $j_{0,1,2,3}$ 的图形如图 1.24, 由于在 $x=0$ 处函数值是 0/0 , 所以作图不是从原点出发.

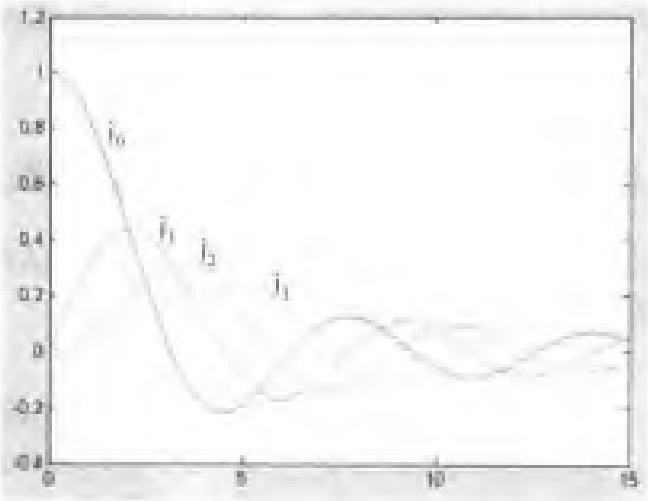


图 1.24 球贝塞尔函数 $j_{0,1,2,3}$ 的图形

```
x=0.8:0.2:15;
y1=sqrt(pi/2./x).*bessely(1/2,x);
y2=sqrt(pi/2./x).*bessely(3/2,x);
y3=sqrt(pi/2./x).*bessely(5/2,x);
y4=sqrt(pi/2./x).*bessely(7/2,x);
plot(x,y1,x,y2,x,y3,x,y4)
axis([1 10 -0.5 0.4])
grid on
```

可得球诺伊曼函数 $n_{0,1,2,3}$ 的图形如图 1.25 所示。

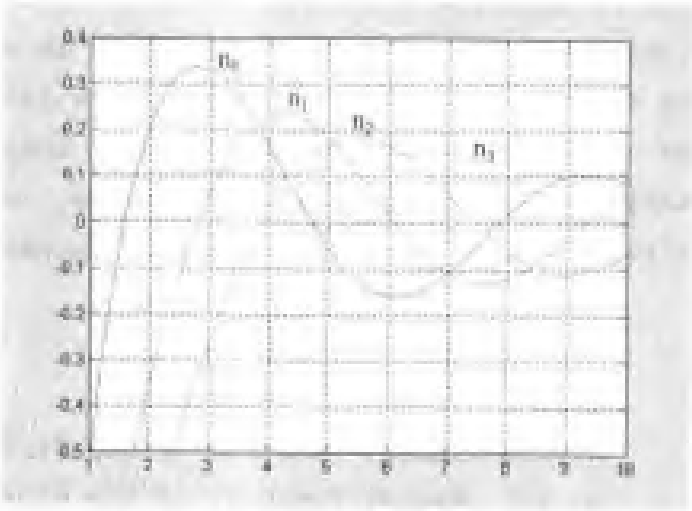


图 1.25 球诺伊曼函数 $n_{0,1,2,3}$ 的图形

球汉开尔函数的实部是球贝塞尔函数，虚部是球诺伊曼函数，所以不必再画它们的图形.

1.3 用 MAPLE 指令计算特殊函数

MAPLE 是一个符号计算功能很强的数学软件,在 MATLAB 中可以调用 MAPLE 的某些功能，调用的指令一共有以下五个：

mfun	对 MAPLE 中的特殊函数作数值计算的指令
mfunlist	列出在 MAPLE 中能用 mfun 计算的特殊函数
mhelp	查阅 MAPLE 中的库函数及其调用方法
maple	调用 MAPLE 的计算指令
procread	读入 MAPLE 程序

这里用这些指令来计算特殊函数.

1. mfunlist 的用法

例如，为了查看在 MATLAB 中可以用 MAPLE 计算的特殊函数，在指令窗中键入

```
>> mfunlist
```

可以在屏幕上看到全部可调用的 MAPLE 的特殊函数如下

```
MFUNLIST Special functions for MFUN.
    The following special functions are listed in alphabetical order
    according to the third column. n denotes an integer argument,
    x denotes a real argument, and z denotes a complex argument. For
    more detailed descriptions of the functions, including any
    argument restrictions, see the Reference Manual, or use MHELP.

 bernoulli       n       Bernoulli Numbers
 bernoulli       n,z     Bernoulli Polynomials
 BesselI         x1,x    Bessel Function of the First Kind
 BesselJ         x1,x    Bessel Function of the First Kind
 BesselK         x1,x    Bessel Function of the Second Kind
```

BesselY	x1,x	Bessel Function of the Second Kind
Beta	z1,z2	Beta Function
binomial	x1,x2	Binomial Coefficients
LegendreKc	x	Complete Elliptic Integral of First Kind
LegendreEc	x	Complete Elliptic Integral of Second Kind
LegendrePic	x1,x	Complete Elliptic Integral of Third Kind
LegendreKc1	x	LegendreKc using Complementary Modulus
LegendreEc1	x	LegendreEc using Complementary Modulus
LegendrePic1	x1,x	LegendrePic using Complementary Modulus
erfc	z	Complementary Error Function
erfc	n,z	Complementary Error Function's Iterated Integrals
Ci	z	Cosine Integral
dawson	x	Dawson's Integral
Psi	z	Digamma Function
dilog	x	Dilogarithm Integral
erf	z	Error Function
euler	n	Euler Numbers
euler	n,z	Euler Polynomials
Ei	x	Exponential Integral
Ei	n,z	Exponential Integral
FresnelC	x	Fresnel Cosine Integral
FresnelS	x	Fresnel Sine Integral
GAMMA	z	Gamma Function
harmonic	n	Harmonic Function
Chi	z	Hyperbolic Cosine Integral
Shi	z	Hyperbolic Sine Integral
hypergeom	X1,X2	(Generalized) Hypergeometric Function
LegendreF	x,x1	Incomplete Elliptic Integral of First Kind
LegendreE	x,x1	Incomplete Elliptic Integral of Second Kind
LegendrePi	x,x2,x1	Incomplete Elliptic Integral of Third Kind
GAMMA	z1,z2	Incomplete Gamma Function
W	z	Lambert's W Function
W	n,z	Lambert's W Function
lnGAMMA	z	Logarithm of the Gamma function

```
Li          x       Logarithmic Integral
Psi         n,z     Polygamma Function
Ssi         z       Shifted Sine Integral
Si          z       Sine Integral
Zeta        z       (Riemann) Zeta Function
Zeta        n,z     (Riemann) Zeta Function
Zeta        n,z,x   (Riemann) Zeta Function

      Orthogonal Polynomials (Extended Symbolic Math Toolbox only)
T     n,x           Chebyshev of the First Kind
U     n,x           Chebyshev of the Second Kind
G     n,x1,x        Gegenbauer
H     n,x           Hermite
P     n,x1,x2,x     Jacobi
L     n,x           Laguerre
L     n,x1,x        Generalized Laguerre
P     n,x           Legendre

   See also MFUN, MHELP.
```

这些函数的中文名称如下：

bernoulli(n)	伯努利数
bernoulli(n,z)	伯努利多项式
BesselI(x1,x)	虚宗量贝塞尔函数
BesselJ(x1,x)	第一类贝塞尔函数
BesselK(x1,x)	虚宗量汉开尔函数
BesselY(x1,x)	第二类贝塞尔函数 (诺伊曼函数)
Beta(z1,z2)	B 函数
binomial(x1,x2)	二项式系数
LegendreKc(x)	第一类完全椭圆积分
LegendreEc(x)	第二类完全椭圆积分
LegendrePic(x1,x)	第三类完全椭圆积分
LegendreKc1(x)	使用补充系数的第一类完全椭圆积分
LegendreEc1(x)	使用补充系数的第二类完全椭圆积分

LegendrePic1(x1,x)	使用补充系数的第三类完全椭圆积分
erfc(z)	余误差函数
erfc(n,z)	余误差函数的内插积分
Ci(z)	余弦积分
dawson(x)	Dawson's 积分
Psi(z)	Ψ 函数
dilog(x)	Dilogarithm 积分
erf(z)	误差函数
euler(n)	欧拉数
euler(n.z)	欧拉多项式
Ei(x)	指数积分
Ei(n,z)	指数积分
FresnelC(x)	菲涅尔余弦积分
FresnelS(x)	菲涅尔正弦积分
GAMMA(z)	Γ 函数
harmonic(n)	谐函数
Chi(z)	双曲余弦积分
Shi(z)	双曲正弦积分
hypergeom(X1,X2)	(广义的) 超几何函数
LegendreF(x,x1)	第一类不完全椭圆积分
LegendreE(x,x1)	第二类不完全椭圆积分
LegendrePi(x,x2,x1)	第三类不完全椭圆积分
GAMMA(z1,z2)	不完全 Γ 函数
W(z)	Lambert's W 函数
W(n,z)	Lambert's W 函数
lnGAMMA(z)	Γ 函数的对数
Li(x)	对数积分
Psi(n,z)	Ψ 函数
Ssi(z)	转换正弦积分
Si(z)	正弦积分
Zeta(z)	黎曼 ζ 函数
Zeta(n,z)	黎曼 ζ 函数
Zeta(n,z,x)	黎曼 ζ 函数

(以下的正交多项式只在扩展的符号工具箱中使用)

T(n, x)	生成 n 阶的第一类切比雪夫多项式

U(n, x)	生成 n 阶第二类切比雪夫多项式
G(n, x1, x)	生成 n 阶 Gegenbauer 多项式
H(n, x)	生成 n 阶厄米特多项式
P(n, x1, x2, x)	生成 n 阶雅可比多项式
L(n, x)	生成 n 阶拉盖多项式
L(n, x1, x)	生成 n 阶广义拉盖多项式
P(n, x)	生成 n 阶勒让德多项式

从中可以看到，有一部分特殊函数是可以用指令 mfun 来调用，而另一些正交多项式 T,U,G,H,P,L 等则只能在扩展的符号工具箱中使用．如果想知道某个具体的函数的用法，可以用 mhelp 查询．例如下面的操作是查阅误差函数的用法．

```
>> mhelp erf
erf - The Error Function
erfc - The Complementary Error Function and its Iterated Integrals
erfi - The Imaginary Error Function
Calling Sequence:
     erf(x)
     erfc(x)
     erfc(n, x)
     erfi(x)
     ...
```

屏幕上显示了误差函数的 3 种类型及其 4 种调用的方法，还举了一些例子．

2. mfun 的用法

用 mfun 对特殊函数作数值计算的格式是

MFUN('fun',MFUN('fun',p1,p2,...,pk))

这里， fun 是特殊函数， p1,p2,...,pk 是函数中的参数．例如要计算误差函数的值所用的操作是

```
>>x = 0:0.1:5.0;
>>y=mfun('erf',x)
```

3. mhelp 的用法

指令 mhelp 是用来查找帮助文件. 例如要查找调用正交多项式厄米特函数的方法, 操作如下:

```
>>mhelp H
orthopoly[H] - Hermite polynomial
Calling Sequence:
     H(n, x)
     orthopoly[H](n, x)

Parameters:
     n - a non-negative integer
     x - any algebraic expression

Description:
- The function H will compute the nth Hermite polynomial, evaluated
  at x.
- These polynomials are orthogonal on the interval (-infinity,
  infinity), with respect to the weight function w(x) = exp(-x^2).
  Thus
    ∫_{-∞}^{∞} w(x)H(n,x)H(m,x)dx = 0
  if m is not equal to n.
- They satisfy the following recurrence relation:
       H(0,x) = 1,
       H(1,x) = 2*x,
       H(n,x) = 2*x*H(n-1,x) - 2*(n-1)*H(n-2,x), for n>1.
- The command with(orthopoly,H) allows the use of the abbreviated form
  of this command.

Examples:
>> with(orthopoly);
    [G, H, L, P, T, U]

>> H(2,x);
    4x^2 - 2
```

```
>> H(3,x);
```

$$8x - 12x^3$$

```
See Also:
linalg[hermite]
```

从帮助信息中知道，调用 H 的格式有两种，H(n, x), 和 orthopoly[H](n, x), 而第一种简略格式要在使用了指令 with(orthopoly,H) 才能使用。此外，帮助信息还给出了正交关系，递推关系以及几个实例。但是，如果直接将例子中的指令键入指令窗，却发现它们并不能执行。那么问题在那里呢？问题就在于，这些指令都是软件 MAPLE 的指令，要在 MAPLE 的环境下使用。所以正确的用法是使用指令 maple 。

4. maple 的用法

```
>> yy = maple('orthopoly[H](2, x)')
yy = 4*x^2-2
```

或者是

```
>> maple('with(orthopoly)')
ans = [G, H, L, P, T, U]
>> yy = maple('H(2,x)')
yy = 4*x^2-2
```

这样得到的 yy 就是二阶的厄米特多项式。如果要画出它的图形，指令是

```
>> ezplot(yy)
```

所得的图形如图 1.26 。

如果要计算 $x = 1 : 0.2 : 2$ 时 yy 的值，要使用给符号变量赋值的指令 sub 。

```
>> subs(yy,1:0.2:2)
ans =
2.0000    3.7600    5.8400    8.2400    10.9600    14.0000
```

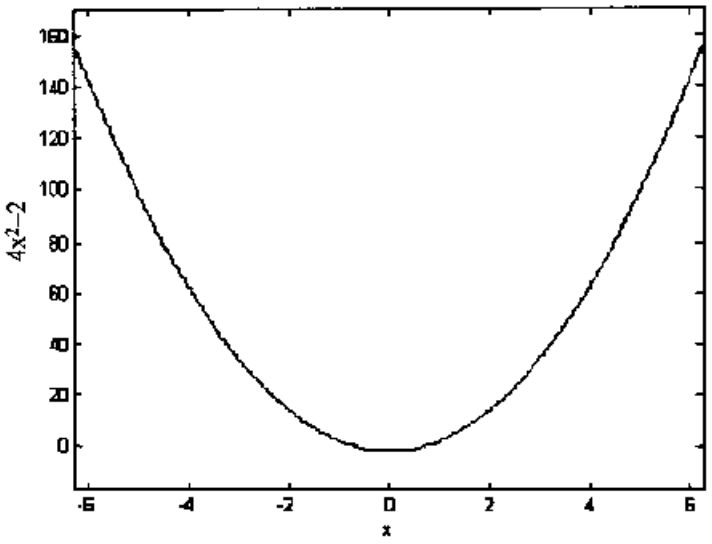


图 1.26 二阶的厄米特多项式的图形

第 2 章　傅里叶级数与傅里叶变换

本章介绍傅里叶级数、傅里叶积分及其之间的关联，还介绍了教材中一般不介绍的离散傅里叶变换及其快速算法 —— 快速傅里叶变换. 最后给出了几个常见的广义傅里叶级数的图像。

2.1　傅里叶级数、傅里叶积分与离散傅里叶变换

1. 傅里叶级数

定义在 $[-l,l]$ 上的任意平方可积函数 $f(x)$ 可展成傅里叶级数

$$f(x)=C_0+\sum_{n=1}^{\infty}\left(C_n\cos\frac{n\pi x}{l}+D_n\sin\frac{n\pi x}{l}\right)$$

其中傅里叶系数 C_0,C_n,D_n 分别为

$$\begin{aligned}C_0&=\frac{1}{2l}\int_{-l}^{l}f(x)\mathrm{d}x\\C_n&=\frac{1}{l}\int_{-l}^{l}f(x)\cos\frac{n\pi x}{l}\mathrm{d}x\quad(n\neq0)\\D_n&=\frac{1}{l}\int_{-l}^{l}f(x)\sin\frac{n\pi x}{l}\mathrm{d}x\quad(n\neq0)\end{aligned}$$

在 $x=\pm l$ 的边界上，$f(x)$ 的傅里叶级数与 $f(x)$ 在边界上的值 $f(l),f(-l)$ 满足如下关系：

$$\left[f(x)=C_0+\sum_{n=1}^{\infty}\left(C_n\cos\frac{n\pi x}{l}+D_n\sin\frac{n\pi x}{l}\right)\right]_{x=\pm l}=\frac{1}{2}[f(l)+f(-l)]$$

因此，$f(x)$ 的傅里叶级数在边界上并不一定收敛于 $f(x)$，但是当 $f(l)=f(-l)$ 时，$f(x)$ 的傅里叶级数在边界上收敛于 $f(x)$ 。

$f(x)$ 也可以展成如下复数形式的傅里叶级数：

$$\begin{aligned}f(x)&=\sum_{n=-\infty}^{\infty}C_n\mathrm{e}^{\mathrm{i}\frac{n\pi x}{l}}\\C_n&=\frac{1}{2l}\int_{-l}^{l}f(x)\mathrm{e}^{-\mathrm{i}\frac{n\pi x}{l}}\mathrm{d}x\end{aligned}$$

即使 $f(x)$ 是实函数，其复数形式的傅里叶级数的系数却可能是复数。

这两种形式的傅里叶级数是等价的，可以根据需要选用。

2. 傅里叶积分

若定义在 $(-\infty,\infty)$ 上的函数 $f(x)$ 满足：

① 在任一有限区间上连续或有有限个第一类间断点，并且有有限个极大值、极小值；

② $\sum_{-\infty}^{\infty}|f(x)|\mathrm{d}x$ 为有限值；

则 $f(x)$ 可以展成如下形式的傅里叶积分：

$$f(x)=\int_0^{\infty}A(\omega)\cos\omega x\mathrm{d}\omega+\int_0^{\infty}B(\omega)\sin\omega x\mathrm{d}\omega$$

其中

$$A(\omega)=\frac{1}{\pi}\int_{-\infty}^{\infty}f(\xi)\cos\omega\xi\mathrm{d}\xi$$

$$B(\omega)=\frac{1}{\pi}\int_{-\infty}^{\infty}f(\xi)\sin\omega\xi\mathrm{d}\xi$$

这是实数形式的傅里叶积分，还有复数形式的傅里叶积分，形式为

$$G(\omega)=\frac{1}{2\pi}\int_{-\infty}^{\infty}g(x)\mathrm{e}^{-\mathrm{i}\omega x}\cdot\mathrm{d}x$$

$$g(x)=\int_{-\infty}^{\infty}G(\omega)\mathrm{e}^{\mathrm{i}\omega x}\cdot\mathrm{d}\omega$$

可将两式写成对称形式

$$f(x)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}F(\omega)\mathrm{e}^{\mathrm{i}\omega x}\cdot\mathrm{d}\omega$$

$$F(\omega)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}f(x)\mathrm{e}^{-\mathrm{i}\omega x}\cdot\mathrm{d}x$$

这两个式子分别叫傅里叶逆变换式和傅里叶变换式。

傅里叶级数是对变量范围为无限的连续的周期函数所作的变换，傅里叶积分是对变量范围为无限的连续的非周期函数所作的变换，在实际中还有两种情况，一个是无限长离散的函数的变换，另一个是有限长的离散的函数的变换。

对无限长的函数按照间隔 b 取样，就得到无限长的离散的函数，按照取样定理，它的傅里叶变换不同于原来的连续函数的傅里叶变换，是一个以 $1/b$ 为周期的函数，在每个周期内，则是原来的连续函数的傅里叶变换的频谱。如果连续函数是有一定的带宽 s_c，只要 $s_c\leqslant\frac{1}{2b}$，就可以用一个周期的频谱去代替原来的连续函数的频谱，反之，如果 $s_c\geqslant\frac{1}{2b}$，则各个周期的频谱会发生重叠，导致高端的频谱失真。

如果是对有限长的离散的函数作傅里叶变换，得到的也是有限长的离散的频谱，这就是下面要讨论的离散的傅里叶变换。结论是，N 个数据点的傅里叶变换得到的是 N 个离散的频谱数据，它是上面讲的无限长离散的函数的频谱在一个周期 $[0,1/b]$ 上 N 个等分点的值.

3. 离散傅里叶变换

考虑到教材中，一般不介绍离散傅里叶变换 (DFT) 与快速傅里叶变换 (FFT)，下面作一简单说明. 先介绍离散傅里叶变换.

记

$$\omega_n = \mathrm{e}^{-\mathrm{i}2\pi/n} = \cos(2\pi/n) - \mathrm{i}\sin(2\pi/n)$$

它就是 1 的 n 次根，即

$$\omega_n^n = 1$$

N 个数据点 $x_l(l = 0, 1, \cdots, N-1)$ 的离散傅里叶变换是

$$y_m = \sum_{k=0}^{n-1} x_k \omega_n^{mk} \quad (m = 0, 1, 2, \cdots, n-1)$$

用矩阵表示为

$$\boldsymbol{y}=\boldsymbol{F}_n\boldsymbol{x}$$

在这里

$$\{\boldsymbol{F}_n\}_{mk} = \omega_n^{mk}$$

当 $n = 4$ 时，所得的矩阵是

$$\boldsymbol{F}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -\mathrm{i} & -1 & \mathrm{i} \\ 1 & -1 & 1 & -1 \\ 1 & \mathrm{i} & -1 & -\mathrm{i} \end{bmatrix}.$$

显然有

$$\frac{1}{n}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \omega^{-3} \\ 1 & \omega^{-2} & \omega^{-4} & \omega^{-6} \\ 1 & \omega^{-3} & \omega^{-6} & \omega^{-9} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

所以很容易得到逆变换是

$$x_k = \frac{1}{n}\sum_{k=0}^{n-1} y_m \omega_n^{-mk} \quad (k = 0, 1, \cdots, n-1)$$

离散傅里叶变换把输入的数据 x 分解为它的基本频率的组合，也就是变换后所得的 y 的各个分量。由表达式

$$\omega_n^{-mk} = \left(\mathrm{e}^{-\mathrm{i}\frac{2\pi}{n}}\right)^{-mk} = \mathrm{e}^{\mathrm{i}k\frac{2\pi}{n}m}$$

可知，$k2\pi/n$ 相当于频率，y_m/n 相当于各频率的振幅，常把振幅的平方称之为功率谱。其中两个特殊的分量是 y_0，即零频分量也就是一个常数，其值为 x 的各个分量之和，还有一个是 $y_{n/2}$，叫做奈奎斯特频率 (Nyquist frequency)，是由取样频率所表达的最高频率。由于离散傅里叶变换在频率上有周期性，所以在奈奎斯特频率以外的频率实际上对应低于奈奎斯特频率的那些频率的负值。

例 1. 任取 8 个数作离散傅里叶变换。

$$\boldsymbol{F}_8\boldsymbol{x} = \boldsymbol{F}_8\begin{bmatrix}4\\0\\3\\6\\2\\9\\6\\5\end{bmatrix} = \begin{bmatrix}35\\-5.07+8.66\mathrm{i}\\-3+2\mathrm{i}\\9.07+2.66\mathrm{i}\\-5\\9.07-2.66\mathrm{i}\\-3-2\mathrm{i}\\-5.07-8.66\mathrm{i}\end{bmatrix} = \boldsymbol{y}$$

可以看到，变换的结果是复数数列，但 y_0 和 y_4 是实的，而 y_5, y_6, y_7 是 y_3, y_2, y_1 的复共轭。这是实数数列变换所预期的，而在频率上，它们是不可区分的。

例 2. 下面是一个周期数列的变换。

$$\boldsymbol{F}_8\boldsymbol{x}=\boldsymbol{F}_8\begin{bmatrix}1\\-1\\1\\-1\\1\\-1\\1\\-1\end{bmatrix}=\begin{bmatrix}0\\0\\0\\0\\8\\0\\0\\0\end{bmatrix}=\boldsymbol{y}$$

对这个数列，我们有意选取了这种取样方式下可能的在 1 与 -1 之间最高的振荡次数. 在变换的结果中，我们看到惟一的一个非零分量是奈奎斯特分量 y_4，而 y_0 仍然等于 $\boldsymbol{x}$ 所有分量之和。

4. 快速傅里叶变换

本节介绍离散傅里叶变换的快速算法，也即快速傅里叶变换 (FFT) .

利用离散傅里叶变换的某种对称性和允余性，可以发展一种很有效的计算离散傅里叶变换的算法. 下面以 $n=4$ 为例加以说明。

$$y_m=\sum_{k=0}^{3}x_k\omega_n^{mk}\qquad(m=0,1,2,3)$$

写出来就是

$$\begin{aligned}y_0&=x_0\omega_n^0+x_1\omega_n^0+x_2\omega_n^0+x_3\omega_n^0\\y_1&=x_0\omega_n^0+x_1\omega_n^1+x_2\omega_n^2+x_3\omega_n^3\\y_2&=x_0\omega_n^0+x_1\omega_n^2+x_2\omega_n^4+x_3\omega_n^6\\y_3&=x_0\omega_n^0+x_1\omega_n^3+x_2\omega_n^6+x_3\omega_n^9\end{aligned}$$

注意到

$$\omega_n^0=\omega_n^4=1,\quad\omega_n^2=\omega_n^6=-1,\quad\omega_n^9=\omega_n^1$$

可以将上式重新组合为

$$y_0 = (x_0 + \omega_n^0 x_2) + \omega_n^0 (x_1 + \omega_n^0 x_3)$$

$$y_1 = (x_0 - \omega_n^0 x_2) + \omega_n^1 (x_1 - \omega_n^0 x_3)$$

$$y_2 = (x_0 + \omega_n^0 x_2) + \omega_n^2 (x_1 + \omega_n^0 x_3)$$

$$y_3 = (x_0 - \omega_n^0 x_2) + \omega_n^3 (x_1 - \omega_n^0 x_3)$$

可以看到，现在只要 8 次加法和 6 次乘法，而不是原来的 $(4-1)*4=12$ 次加法和 $4^2=16$ 次乘法．其实考虑 $\omega_n^0=1$，需要的乘法次数还会更少，不过我们这里主要是为了说明一般的算法．现在计算 4 点的 DFT 已经变成了 2 点的对奇数数列和偶数数列的变换．这个性质一般都成立，如果 n 是偶数，则 n 点的 DFT 可能分解成两个长度为一半的 DFT．

请看下面几个傅里叶矩阵，这种特征更清楚：

$$\boldsymbol{F}_1 = 1, \quad \boldsymbol{F}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad \boldsymbol{F}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -\mathrm{i} & -1 & \mathrm{i} \\ 1 & -1 & 1 & -1 \\ 1 & \mathrm{i} & -1 & -\mathrm{i} \end{bmatrix}$$

令 $\boldsymbol{P}_4$ 是置换矩阵

$$\boldsymbol{P}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

而 $\boldsymbol{D}_2$ 是对角矩阵，

$$\boldsymbol{D}_2 = \mathrm{diag}(1, \omega_4) = \begin{bmatrix} 1 & 0 \\ 0 & -\mathrm{i} \end{bmatrix}$$

可得

$$\boldsymbol{F}_4\boldsymbol{P}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -\mathrm{i} & \mathrm{i} \\ 1 & 1 & -1 & -1 \\ 1 & -1 & \mathrm{i} & -\mathrm{i} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_2 & \boldsymbol{D}_2\boldsymbol{F}_2 \\ \boldsymbol{F}_2 & -\boldsymbol{D}_2\boldsymbol{F}_2 \end{bmatrix}$$

对照一下，可以看出，前面交换以后的结果，其实就是 $\boldsymbol{F}_4\boldsymbol{P}_4\boldsymbol{x}$。可见，$\boldsymbol{F}_4$ 可以排列成用 $\boldsymbol{F}_2$ 组成的方块．只要点数是偶数，这种分层分解的步骤就可以逐层进行下去．通常，$\boldsymbol{P}_4$ 是交换矩阵，它将 $\boldsymbol{F}_n$ 的偶数列放到奇数列之前．而

$$\boldsymbol{D}_{n/2} = \mathrm{diag}\left(1, \omega_n, \cdots, \omega_n^{(n/2)-1}\right)$$

因而，把 $\boldsymbol{F}_n$ 应用于长度为 n 的数列，只要将 $\boldsymbol{F}_{n/2}$ 分别作用于它的偶数列和奇数列，然后再用 $\boldsymbol{D}_{n/2}$ 标度一下．用这种递归分解与解决的方法来计算 DFT 就叫 FFT．

2.2 傅里叶变换的例题

本节介绍几个傅里叶变换的例题。

例 1. 矩形波列的傅里叶级数

矩形波列 $f(x)$ 如图 2.1, 它的一个周期可表示为

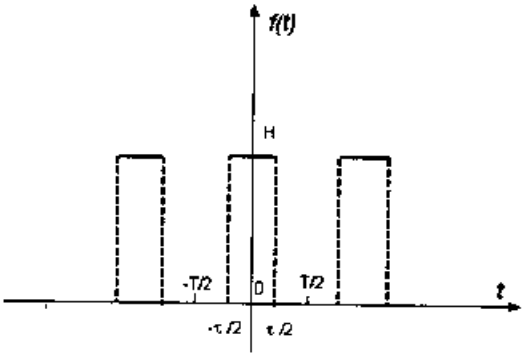


图 2.1 矩形函数的图形

$$f(t) = \begin{cases} H & \left(-\dfrac{\tau}{2} \leqslant t \leqslant \dfrac{\tau}{2}\right) \\ 0 & \left(-\dfrac{T}{2} < t < -\dfrac{\tau}{2}, \dfrac{\tau}{2} < t < \dfrac{T}{2}\right) \end{cases}$$

试把它展为复数形式的傅里叶级数，并研究其频谱。

解：计算系数

$$C_k = \frac{1}{T}\int_{-\tau/2}^{\tau/2} H\mathrm{e}^{-\mathrm{i}2\pi\frac{k}{T}\xi}\mathrm{d}\xi = \frac{H}{k\pi}\sin\frac{k\pi\tau}{T}$$

$$C_0 = \frac{1}{T}\int_{-\tau/2}^{\tau/2} H\mathrm{d}\xi = \frac{H\tau}{T}$$

所得的傅里叶级数为

$$f(t) = \frac{H\tau}{T} + \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \frac{H}{k\pi}\sin\frac{k\pi\tau}{T}\mathrm{e}^{\mathrm{i}2\pi\frac{k}{T}t}$$

将该矩形波的 k 次谐波的频率记为

$$\omega_k = 2k\pi/T$$

则 k 次正谐波的振幅为

$$A_k = |C_k| = \left|\frac{H}{k\pi}\sin\frac{k\pi\tau}{T}\right| = \left|\frac{H}{T}\frac{\sin\left(\frac{k}{2}\tau\right)}{\frac{k}{2}}\right|$$

$$A_0 = |C_0| = \frac{2H\tau}{T}$$

其频谱如图 2.2 所示，圆圈就是频谱的值，曲线是它的包络，图中取 $H=1, T=1, \tau=0.2$ 。

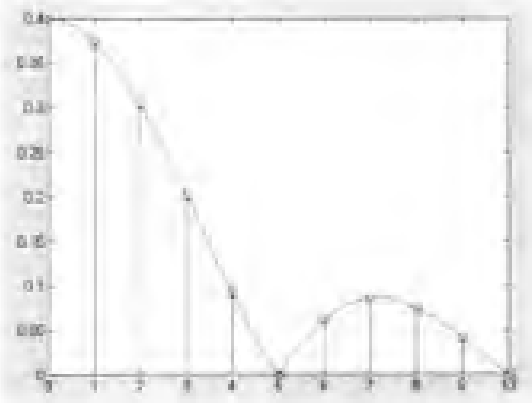


图 2.2 矩形函数的傅里叶频谱

例 2. 单个矩形脉冲的傅里叶变换

研究矩形函数

$$f(t)=\begin{cases} H & \left(-\dfrac{\tau}{2}\leqslant t\leqslant\dfrac{\tau}{2}\right) \\ 0 & \left(t<-\dfrac{\tau}{2},\dfrac{\tau}{2}<t\right) \end{cases}$$

的傅里叶积分。

解：

$$\frac{1}{2\pi}\int_{-\infty}^{\infty}Hf(t)\mathrm{e}^{-\mathrm{i}\omega t}\mathrm{d}t=\frac{H}{2\pi}\int_{-\tau/2}^{\tau/2}\mathrm{e}^{-\mathrm{i}\omega t}\mathrm{d}t$$
$$=-\frac{H}{2\pi\mathrm{i}\omega}\mathrm{e}^{-\mathrm{i}\omega t}\Big|_{-\tau/2}^{\tau/2}=\frac{H}{\pi}\frac{\sin\frac{\omega}{2}\tau}{\omega}$$

可以看出，除了一个常数因子之外，它就是上一题中傅里叶级数频谱振幅的包络。

再对矩阵函数作离散傅里叶变换，方法是直接采用 MATLAB 快速傅里叶变换算法指令 fft 。所用程序如下：

```
x=-1:0.1:1;
y=zeros(1,21);
y(9:13)=1;
Y=fft(y);
Y(1)=[ ];
n=fix(length(Y)/2);
freq=[1:n]./length(Y);

Y1=fft(y,256);
Y1(1)=[ ];
n1=fix(length(Y1)/2);
freq1=[1:n1]./length(Y1);

subplot(2,1,1),
plot(x,y,'*r')
subplot(2,1,2),
plot(freq,abs(Y(1:n)),freq1,abs(Y1(1:n1)),'r-.')
```

在程序中，第一种计算只用了 21 个数据点，第二种计算为了提高精度，用补零的方法，将数据点扩充成 256 个数据点。从图 2.3 可以看出，第二种计算有更高

的精度。另外，图形中只画出了频率小于奈奎斯特频率的振幅，而没有画出全部频率的振幅。

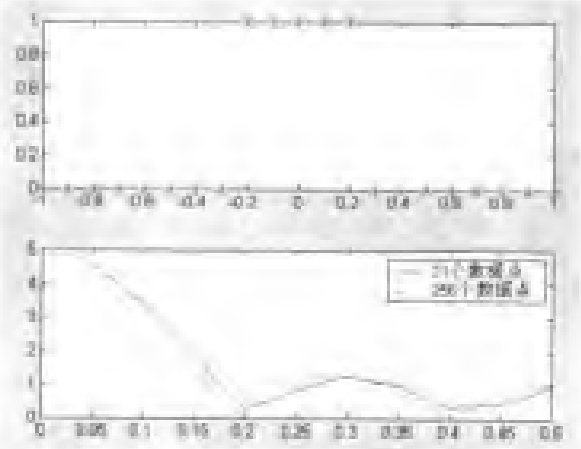


图 2.3 矩形函数的快速傅里叶变换

例 3. 两个频率不同的正弦函数之和的傅里叶变换

对函数 $x = \sin(2 * pi * 15 * t) + \sin(2 * pi * 40 * t)$ 作快速傅里叶变换，所用的程序是

```
t = 0:1/100:10-1/100;
x = sin(2*pi*15*t) + sin(2*pi*40*t);
y = fft(x);
m = abs(y);
p = unwrap(angle(y));
figure(3)
f = (0:length(y)-1)'*100/length(y);
subplot(2,1,1)
plot(f,m),
ylabel('Abs. Magnitude')
grid on
subplot(2,1,2)
plot(f,p*180/pi)
ylabel('Phase [Degrees]')
grid on
xlabel('Frequency [Hertz]')
```

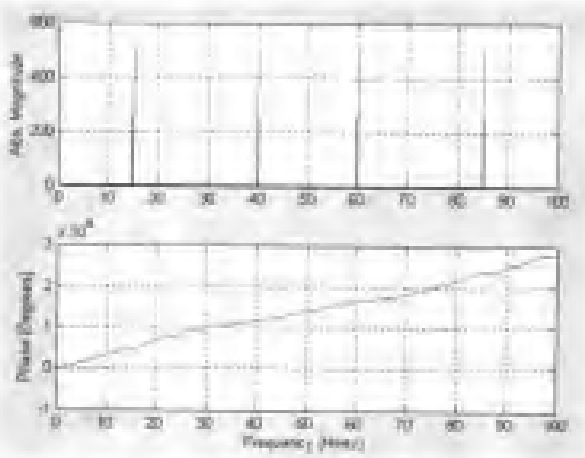


图 2.4 迭加正弦波的快速傅里叶变换

图 2.4 列出了全部的频谱以及它们对应的幅角. 从图中可以看出, 通过频谱分析, 已经找出了两个迭加的频率.

例 4. 二维矩形函数的快速傅里叶变换

二维矩形函数是

$$f(x,y)=\begin{cases} H & (-a/2\leqslant x\leqslant a/2,-b/2\leqslant y\leqslant b/2) \\ 0 & \text{其余} \end{cases}$$

它的快速傅里叶变换所用的程序是

```
f=zeros(30,30);
f(5:24,13:17)=1;
figure(1)
imshow(f,'notruesize')
F=fft2(f,256,256);
FF=fftshift(F);
F2=log(abs(FF));
figure(2)
imshow(F2,[-1,5],'notruesize')
colormap(jet);
colorbar
```

在程序中，指令 imshow 是用图像显示矩阵数据（见图 2.5），fftshift 是将傅里叶变换所得的直流分量移到图像中心，所得的结果为图 2.6 所示。



图 2.5 二维矩形函数的图像

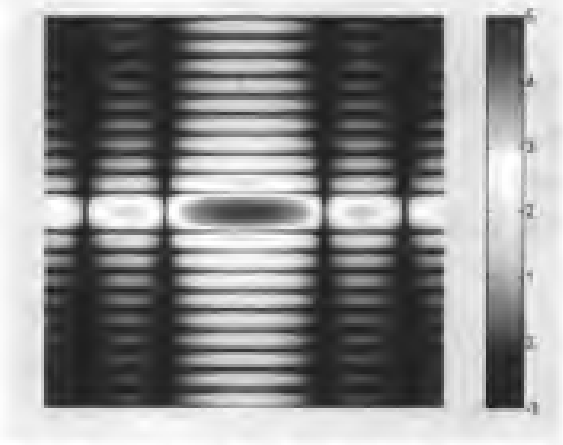


图 2.6 二维矩形函数的快速傅里叶变换

2.3 广义傅里叶级数

2.3.1 勒让德函数的母函数

利用勒让德函数的母函数公式，有

$$\frac{1}{\sqrt{1-2r\cos\theta+r^2}}=\sum_{l=0}^{\infty}r^l p_l(\cos\theta) \qquad (r<1)$$

$$\frac{1}{\sqrt{1-2r\cos\theta+r^2}}=\sum_{l=0}^{\infty}\frac{1}{r^{l+1}}p_l(\cos\theta) \quad (r>1)$$

下面来画出这两个等式两边在第一象限的图形如图 2.7 所示。

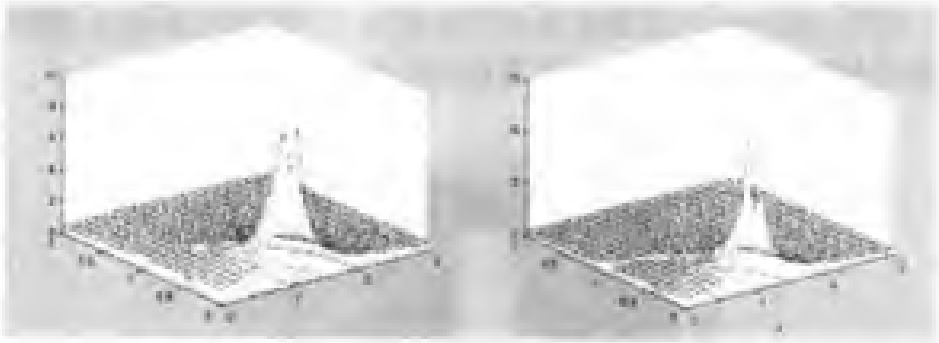


图 2.7 勒让德函数的母函数等式两边的图形

左边可以看成是在极轴上 $z=1$ 处放有电量为 $4\pi\varepsilon$ 的电荷在空间产生的电势. 不难想象, 等势面应该是以极轴上 $z=1$ 为圆心的同心圆. 而右边则可以看成是对 r 作展开, 所得的系数就是勒让德函数. 在作图时, 是分成 $r<1$ 和 $r>1$ 两种情况来画图的, 显然, 只要取足够的项数, 就可以得到与左边非常相近的图形, 这一点从等值线可以看得更清楚.

所用程序是

```
[X,Z]=meshgrid([0:0.1:3],[0:0.1:2]);
[Q,R]=cart2pol(X,Z);
R(find(R==1))=NaN;
 u=1./sqrt(1-2.*R.*cos(Q)+R.^2);
 meshc(X,Z,u)

Rin=R;
Rin(find(Rin>1))=NaN;
Rout=R;
Rout(find(Rout<1))=NaN;
Uin=1;       %p0=1
Uout=1./Rout;
for k=1:20
    Leg=legendre(k,cos(Q));
    Legk=squeeze(Leg(1,:,:));
    uin=Rin.^k.*Legk;
    uout=1./Rout.^(k+1).*Legk;
    Uin=Uin+uin;
    Uout=Uout+uout;
end
figure
```

```
meshc(X,Z,Uin)
hold on
meshc(X,Z,Uout)
xlabel('x')
```

2.3.2 贝塞尔函数的母函数

贝塞尔函数的母函数公式是

$$e^{\frac{x}{2}\left(z-\frac{1}{z}\right)}=\sum_{m=-\infty}^{\infty}J_m(x)z^m \quad (0<|z|<\infty)$$

这是一个以 x 为参数的复函数的展开式，给定一个 x 值，就可以按照前面介绍的复函数作图法画出它们两边的图形. 由于 $z=0$ 是函数的奇点，函数值变化剧烈，为了有较好的效果，我们选择了离原点稍远的区域画图，所用的程序如下，程序中取 $x=2$.

```
m=30;
r=(0.3*m:m)'/m;
theta=pi*(-m:m)/m;
z=r*exp(i*theta);
z(find(z==0))=NaN;
figure
cplxmap(z,exp(z-1./z))
view(34,44)

w=0;
for k=-20:20
    u=besselj(k,2).*z.^k;
    w=w+u;
end
figure
cplxmap(z,w)
view(34,44)
```

所得的图形如图 2.8 所示.

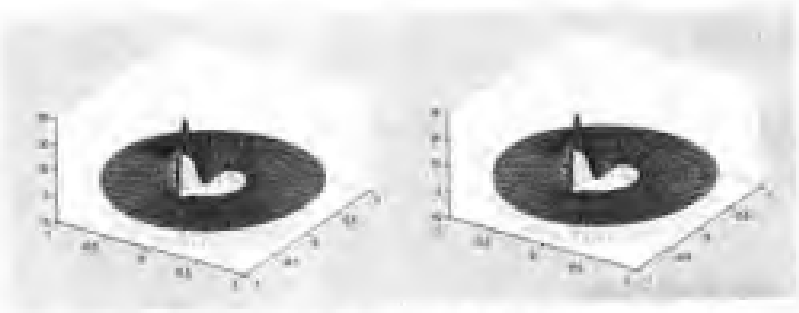


图 2.8 勒让德函数的母函数等式两边的图形

2.3.3 平面波展开为球面波的叠加

平面波展开为球面波的叠加是广义傅里叶级数的一个很好的例子，这个公式在研究量子力学散射问题时很有用处。展开公式是*

$$e^{-ikr\cos\theta}=\sum_{l=0}^{\infty}(2l+1)i^{l}j_{l}(kr)P_{l}(\cos\theta)$$

等式左边的各个分量代表的是 $m=0$，l 取不同值时不同的球面波。这时球面波与 φ 无关。以水平面作为 XOZ 面，画出 $l=0,1,2,\cdots,5$ 的分量的图形如图 2.9、图 2.10 以及图 2.11 所示。

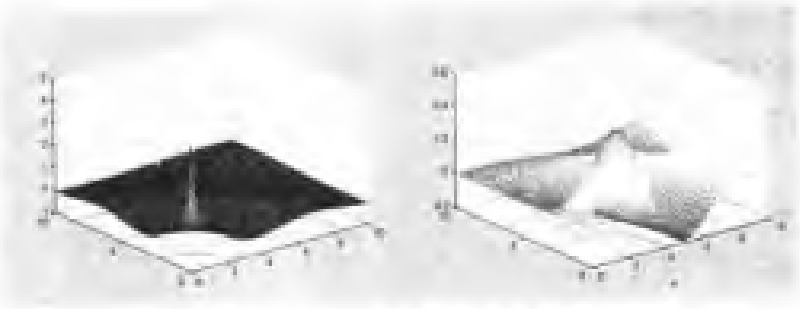


图 2.9 $l=0,1$ 球面波分量图

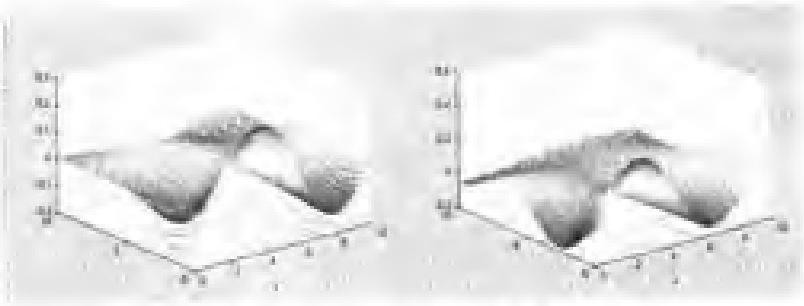


图 2.10 $l=2,3$ 球面波分量图

*见参考文献 [1]，371 页。

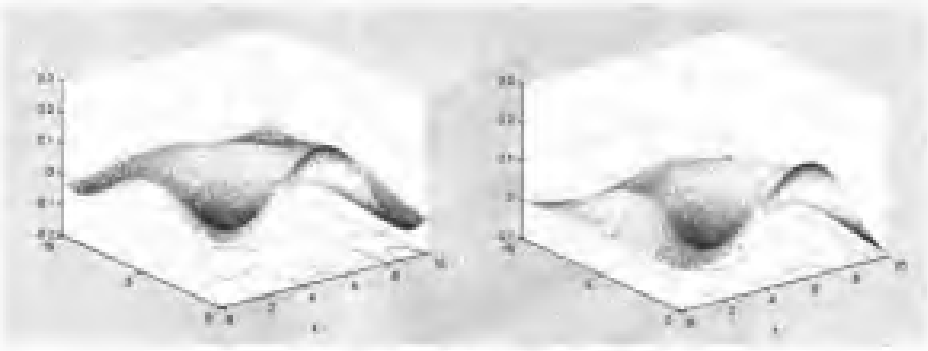


图 2.11 $l=4,5$ 球面波分量图

所用的程序如下：

```
[X,Z]=meshgrid(0.05:0.1:10);
figure
[Q,R]=cart2pol(X,Z);
sqrtR=sqrt(pi/2./R);
Leg0=legendre(0,cos(Q));
Bes0=sqrtR.*besselj(0,R);
qiu=Bes0.*Leg0;
surfc(X,Z,qiu)
for k=1:5
    Leg=legendre(k,cos(Q));
    Legk=squeeze(Leg(1,:,:));
    Bes=sqrtR.*besselj(k,R);
    qiuk=Bes.*Legk;
    figure
    surfc(X,Z,qiuk)
    xlabel('x')
end
```

取上式两边的实部，在等式的右边，当 $l=2n$ 时得实部，

$$\cos(kr\cos\theta)=\sum_{n=0}^{\infty}(4n+1)\mathrm{j}^{2n}\mathrm{j}_{2n}(kr)\mathrm{P}_{2n}(\cos\theta)$$

等式的左边是向 z 传播的平面波，其图形如图 2.12 所示。

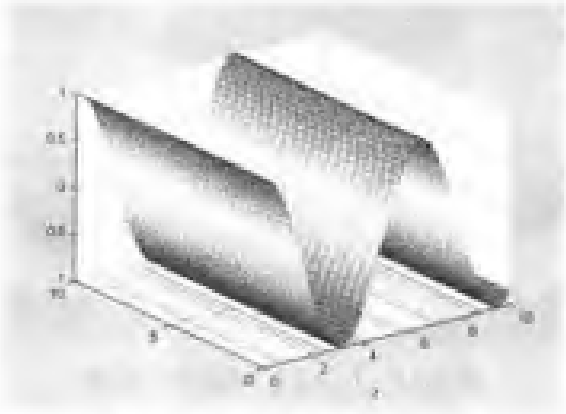


图 2.12 向 Z 方向传播的平面波

等式右边分别取 $k=1, n$ 最大分别取 10 和 50, 得到叠加后的图形如图 2.13所示.

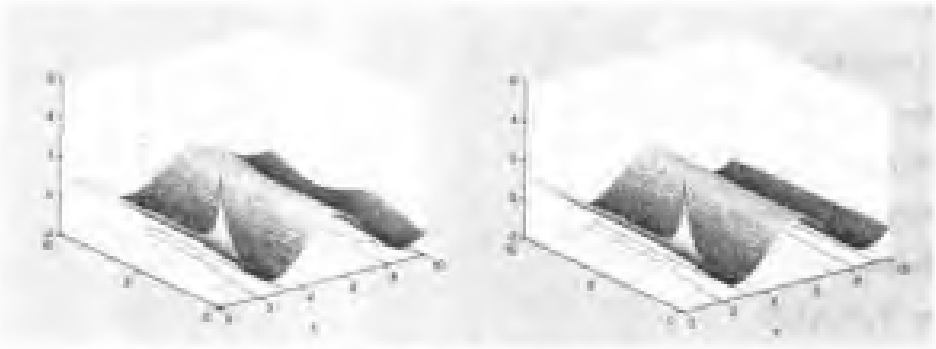


图 2.13 n 最大取分别 10,50 时得到球面波叠加的图形

可以看到, n 取到 50 的叠加效果比 n 取到 10 的效果要好得多. 另外在原点附近, 有奇异性, 所以球面波叠加的效果与平面波有一明显的差异. 画图的程序如下:

```
[X,Z]=meshgrid(0.05:0.1:10);
figure
g=cos(X);
contour(g)
meshc(X,Z,g)
xlabel('z')

[Q,R]=cart2pol(X,Z);
sqrtR=sqrt(pi/2./R);
```

```
Leg0=legendre(0,cos(Q));
Bes0=sqrtR.*besselj(0,R);
qiu=Bes0.*Leg0;
for k=2:2:50
    Leg=legendre(k,cos(Q));
    Legk=squeeze(Leg(1,:,:));
    Bes=sqrtR.*besselj(k,R);
    qiuk=(2*k+1)*i^k*Bes.*Legk;
    qiu=qiu+qiuk;
end
figure
meshc(X,Z,qiu)
xlabel('z')
```

2.3.4 平面波展开为柱面波的叠加

平面波展开为柱面波的叠加是另一个很有趣的广义傅里叶级数的例子。展开公式是*

$$e^{ik\rho\cos\varphi} = J_0(k\rho) + 2\sum_{m=1}^{\infty} \mathrm{i}^m J_m(k\rho)\cos m\varphi$$

等式左边的各个分量代表的是不同 m 的柱面波. 这时柱面波与 z 无关, 在水平面 XOY 内画出 $m = 0, 1, 2, \cdots, 5$ 的分量的图形如图 2.14, 图 2.15 以及图 2.16所示。

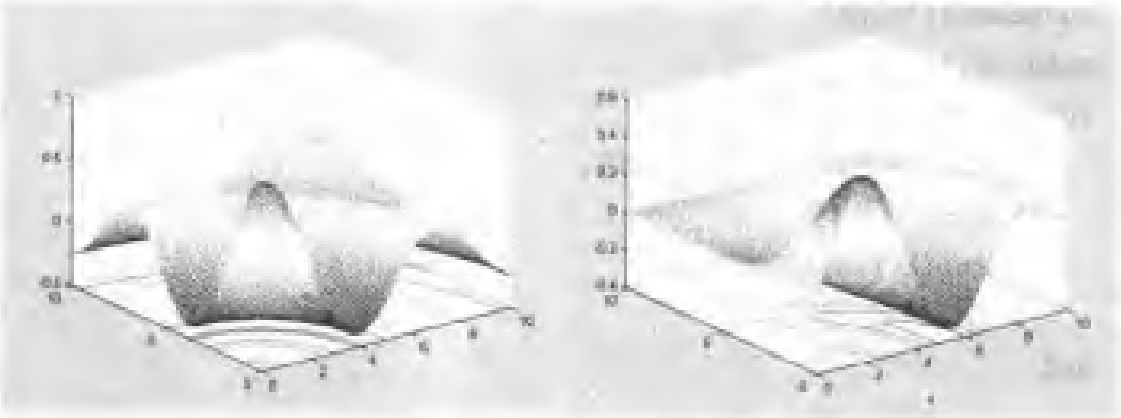


图 2.14 $m = 0, 1$ 柱面波分量图

*见参考文献 [4], 513 页。

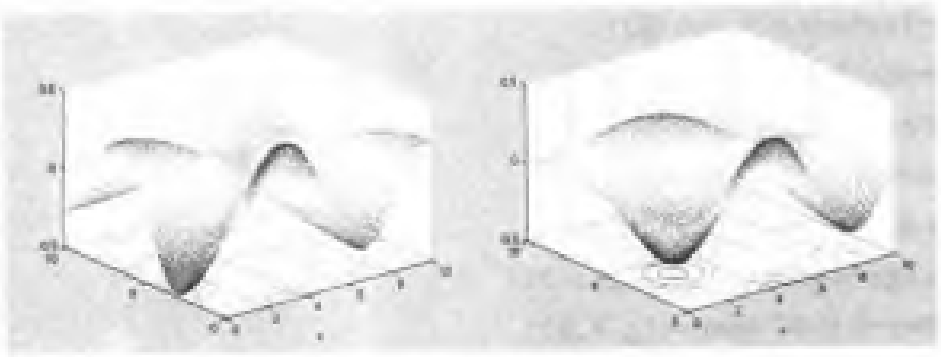


图 2.15 $m=2,3$ 柱面波分量图

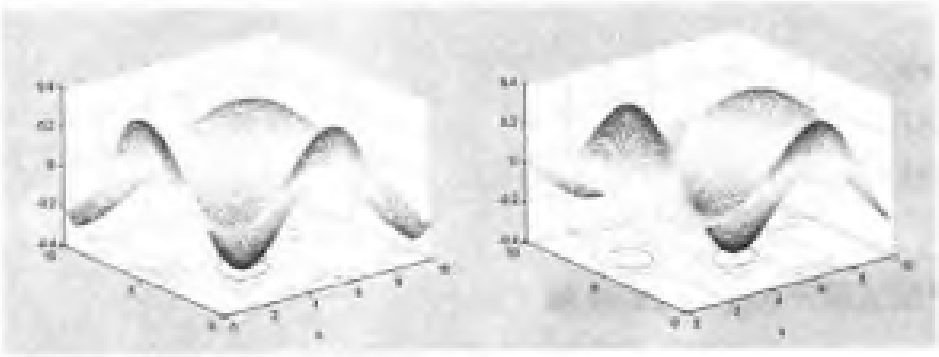


图 2.16 $m=4,5$ 柱面波分量图

所用的程序如下：

```
[X,Y]=meshgrid(0.05:0.1:10);
[Q,R]=cart2pol(X,Y);
figure
zu=besselj(0,R);
meshc(X,Y,zu)
for k=1:5
    zu=besselj(k,R).*cos(k*Q);
    figure
    meshc(X,Y,zu)
    xlabel('x')
end
```

取上式两边的实部，令 $x=\rho\cos\varphi$，得

$$\cos kx = \mathrm{J}_0(k\rho) + 2\sum_{n=1}^{\infty} \mathrm{i}^{2n}\mathrm{J}_{2n}(k\rho)\cos 2n\varphi$$

平面波的图形与前同，故不再画．等式右边分别取 $k=1$，n 最大分别取 10 和 20，得到叠加后的图形如图 2.17 所示．

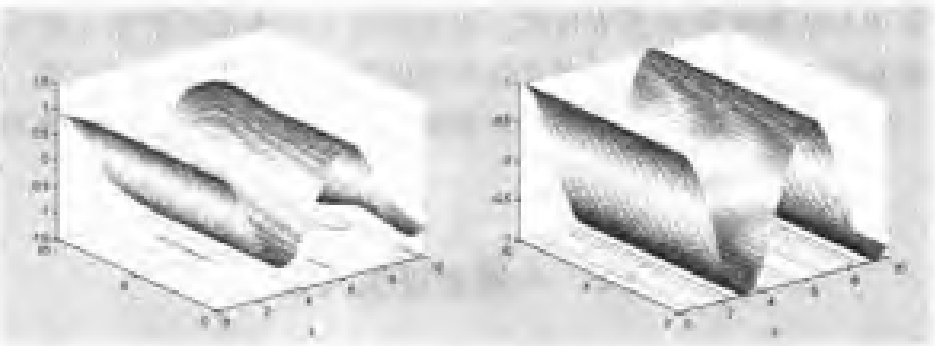


图 2.17 n 最大分别取 10,20 时得到柱面波叠加的图形

可以看到，n 取到 20 的叠加效果已经很不错了．所以柱面波叠加的效果球面波更好．画图的程序如下：

```
[X,Y]=meshgrid(0.05:0.1:10);    %k=1;
[Q,R]=cart2pol(X,Y);
zu=besselj(0,R);
for k=2:2:20
    zuk=2*i^k*besselj(k,R).*cos(k*Q);
    zu=zu+zuk;
end
figure
meshc(X,Y,zu)
xlabel('x')
```

第 3 章　本征值函数系与本征振动

在用分离变量法解数学物理方程时，最后都会归结到求解本征值问题．在用本征函数系展开法解数学物理方程时，也要对所用的本征函数系有较好的理解．所以，各种本征函数系在数学物理方法课程的学习有非常重要的地位，本章列举了常见的一些本征函数系及其图像，以帮助大家熟悉和更好地掌握这些本征函数系的性质．

3.1　一维本征值问题

一维的波动方程 (弦振动、杆振动) 是

$$u_{tt} = a^2 u_{xx} \quad (0 \leqslant x \leqslant l)$$

令 $u(x,t) = X(x)T(t)$, 分离变量得

$$X'' + \lambda X = 0; \quad T'' + \lambda a^2 T = 0$$

一维的热传导方程是

$$u_t = a^2 u_{xx} \quad (0 \leqslant x \leqslant l)$$

令 $u(x,t) = X(x)T(t)$, 分离变量得

$$X'' + \lambda X = 0; \quad T' + \lambda a^2 T = 0$$

二维的拉普拉斯方程是

$$u_{xx} + u_{yy} = 0 \quad (0 \leqslant x \leqslant a, \ \ 0 \leqslant x \leqslant b)$$

令 $u(x,t) = X(x)Y(t)$, 分离变量得

$$X'' + \lambda X = 0; \quad Y'' - \lambda Y = 0$$

在三类方程中分离变量都得到的方程 $X'' + \lambda X = 0$ 与边界条件可以组合成不同的本征值问题，下面介绍四种常见的．

3.1.1　四种常见的本征函数系

1. 本征函数系 $\sin \dfrac{n\pi}{l} x$

本征值问题是

$$\begin{cases} X''+\lambda X=0 \\ X(x=0)=0, \quad X(x=l)=0 \end{cases}$$

它在两个边界上都有第一类齐次边界条件，它的本征值和本征函数系分别是

$$\lambda=\frac{n^2\pi^2}{l^2}, \quad \sin\frac{n\pi}{l}x, \quad n=1,2,3,\cdots$$

2. 本征函数系 $\cos\dfrac{n\pi}{l}x$

本征值问题是

$$\begin{cases} X''+\lambda X=0 \\ X'(x=0)=0, \quad X'(x=l)=0 \end{cases}$$

它在两个边界上都有第二类齐次边界条件，它的本征值和本征函数是

$$\lambda=\frac{n^2\pi^2}{l^2}, \quad \cos\frac{n\pi}{l}x, \quad n=0,1,2,\cdots$$

3. 本征函数系 $\sin\dfrac{(n+1/2)\pi}{l}x$

本征值问题是

$$\begin{cases} X''+\lambda X=0 \\ X(x=0)=0, \quad X'(x=l)=0 \end{cases}$$

它在左边界上是第一类齐次边界条件，在右边界上是第二类齐次边界条件，它的本征值和本征函数系是

$$\lambda=\frac{(n+1/2)^2\pi^2}{l^2}, \quad \sin\frac{(n+1/2)\pi}{l}x, \quad n=0,1,2,\cdots$$

4. 本征函数系 $\cos\dfrac{(n+1/2)\pi}{l}x$

本征值问题是

$$\begin{cases} X''+\lambda X=0 \\ X'(x=0)=0, \quad X(x=l)=0 \end{cases}$$

它在左边界上是第二类齐次边界条件，在右边界上是第一类齐次边界条件，它的本征值和本征函数系是

$$\lambda = \frac{(n+1/2)^2\pi^2}{l^2}, \quad \cos\frac{(n+1/2)\pi}{l}x, \quad n = 0, 1, 2, \cdots$$

3.1.2 本征函数系的图像及其运动图像

在图 3.1 中从上到下依次是本征函数系 $\sin\frac{n\pi}{l}x$, $\cos\frac{n\pi}{l}x$, $\sin\frac{(n+1/2)\pi}{l}x$, $\cos\frac{(n+1/2)\pi}{l}x$ 中前四个本征函数的图像. 从图中可以看到，如果在边界是第一类齐次边界条件，则函数在边界点为零，曲线会在边界点与 x 轴相交，如果在边界是第二类齐次边界条件，则函数在边界点不为零，曲线不会在边界点与 x 轴相交，其实这就是不同的驻波。

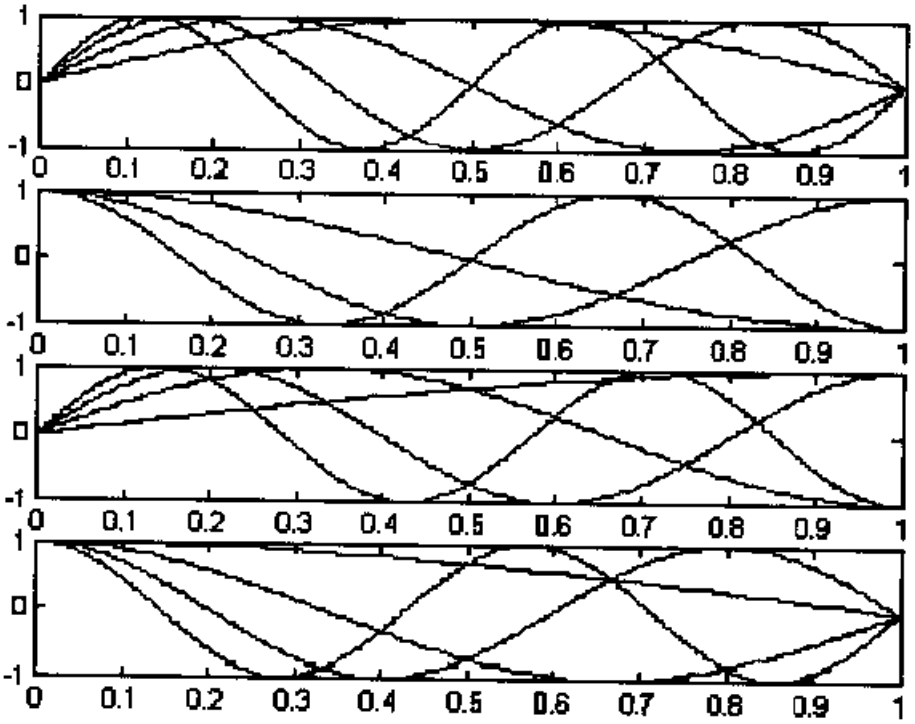


图 3.1 一维本征函数系的图形

下面的程序用以画出这四种本征函数系的图像. 这个程序没有用 for 循环来画不同的曲线，而是用矢量化编程来解决这个问题. 在构造矩阵时，又巧妙地运用了矩阵乘法，而不是采用直接输入各个矩阵元素的方法. 这些技巧都比较鲜明地体现了 MATLAB 的优点。

```
x=0:0.001:1;
A=sin(pi*[1:4]'*x);
B=cos(pi*(0:3)'*x);
C=sin(pi*(1/2:7/2)'*x);
D=cos(pi*(1/2:7/2)'*x);

subplot(4,1,1),
plot(x,A);
subplot(4,1,2),
plot(x,B);
subplot(4,1,3),
plot(x,C);
subplot(4,1,4),
plot(x,D);
```

为了看出各个驻波的演化，必须将本征函数与时间函数结合起来考查．在振动问题中，应该结合的时间因子是 $\sin\sqrt{\lambda}at$ 或 $\cos\sqrt{\lambda}at$ ．我们取第一种本征函数 $\sin\frac{n\pi}{l}x$, 而时间因子取 $\cos\frac{n\pi a}{l}t$ ．下面的程序画出了 $n=1,2,3,4$ 时驻波的运动．可以看到每一个驻波都是以两端为节点，这是两端固定弦的级数解的必然结果，但是每个驻波的运动频率是不同的．

```
function   zb
t=0:0.005:2.0;      x=0:0.001:1;
ww1=wfun(1,0);      ww2=wfun(2,0);
ww3=wfun(3,0);      ww4=wfun(4,0);
ymax1=max(abs(ww1));
figure
subplot(2,1,1)
h1= plot(x,ww1,'r','linewidth',5);
hold on;
h3= plot(x,ww3,'g','linewidth',5);
axis([ 0, 1, -ymax1, ymax1])
```

```
subplot(2,1,2)
ymax4=max(abs(ww4));
h2= plot(x,ww2,'r','linewidth',5);
hold on;
h4= plot(x,ww4,'g','linewidth',5);
axis([ 0, 1, -ymax4, ymax4])

for n=2:length(t)
    ww1=wfun(1,t(n));
    set(h1,'ydata',ww1);
    ww3=wfun(3,t(n));
    set(h3,'ydata',ww3);
    drawnow;
    ww2=wfun(2,t(n));
    set(h2,'ydata',ww2);
    ww4=wfun(4,t(n));
    set(h4,'ydata',ww4);
    drawnow;
end

function  wtx=wfun(k,t)
x=0:0.001:1;    a=1;
wtx=cos(k*pi*a*t).*sin(k*pi*x);
```

图 3.2 是按照时间顺序截取运动中的 4 幅画面.

在热传导问题中, 应该结合的时间因子是 $e^{-i\lambda a^2 t}$, 画 $n=1,2,3,4$ 前四个分波的图形, 可以看到, 它们都是逐渐衰减的, 程序如下:

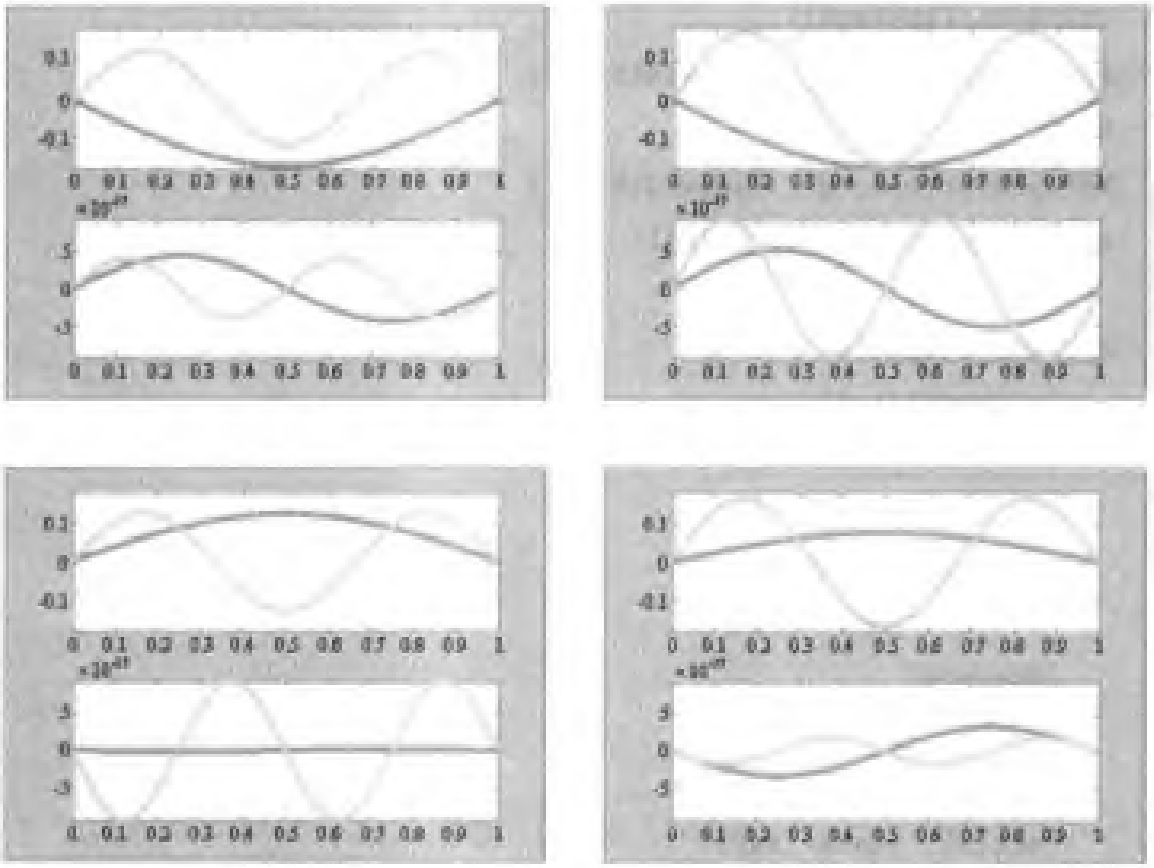


图 3.2 级数解中的前四个驻波分量的运动

```
function     sfb
x=0:0.01:1;              t=1e-5:0.0001:0.005;
ww1=sfbf(1,t(1));        ww2=sfbf(2,t(1));
ww3=sfbf(3,t(1));        ww4=sfbf(4,t(1));
ymax3=max(abs(ww3));
figure
subplot(2,1,1)
h1= plot(x,ww1,'r','linewidth',3);
hold on;
h3= plot(x,ww3,'g','linewidth',3);
axis([ 0, 1, -ymax3, ymax3])

subplot(2,1,2)
```

```
ymax4=max(abs(ww4));
h2= plot(x,ww2,'r','linewidth',3);
hold on;
h4= plot(x,ww4,'g','linewidth',3);
axis([ 0, 1, -ymax4, ymax4])

for n=2:length(t)
    ww1=sfbf(1,t(n));
    set(h1,'ydata',ww1);

    ww3=sfbf(3,t(n));
    set(h3,'ydata',ww3);
    drawnow;

    ww2=sfbf(2,t(n));
    set(h2,'ydata',ww2);

    ww4=sfbf(4,t(n));
    set(h4,'ydata',ww4);
    drawnow;
end
%-----------------

function cht=sfbf(k,t)
x=0:0.01:1;
cht=sin(k*pi*x);
cht=cht*exp(-(k^2*pi^2*2^2*t));
```

图 3.3 是按时间顺序截取的运动中的 4 幅画面.

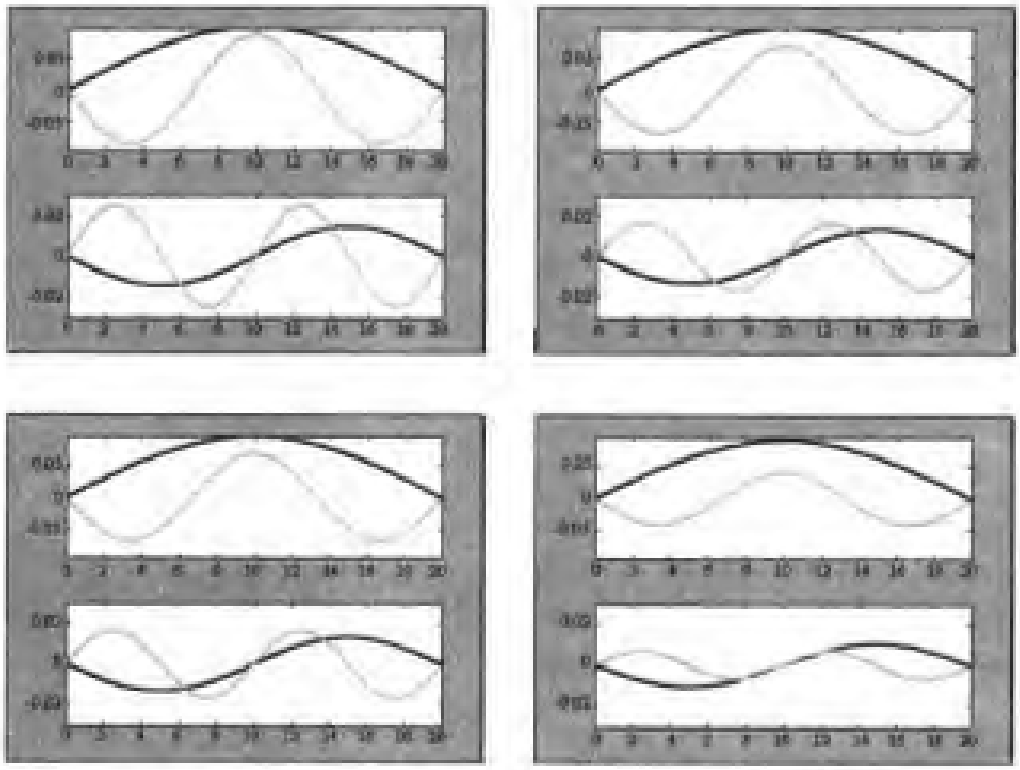


图 3.3 热传导方程前四个本征函数随时间的演化

3.2 二维本征值问题

在研究二维的振动方程与扩散方程时，必然会遇到二维的本征值问题，典型的是矩形区域与圆形区域的本征值问题．下面我们来讨论它们的本征函数与本征振动．

3.2.1 矩形区域的本征模与本征振动

边长分别为 b, c 的四周固定的矩形膜的本征值问题是

$$u(x,y)_{xx} + u(x,y)_{yy} = \lambda u(x,y)$$

由此得到的本征模是

$$X_m(x)Y_n(y) = \sin\frac{n\pi x}{c}\sin\frac{m\pi y}{b}$$

$$\lambda_{mn} = \left(\frac{m\pi}{b}\right)^2 + \left(\frac{n\pi}{c}\right)^2$$

关于它们的求解见 8.2 节．下面的程序是计算前 9 个本征值并画出前 4 个本征函数的图形，程序中是使用矩阵来计算本征值．

```
a=2;   b=1;
[m,n]=meshgrid(1:3);
L=((m*pi./b).^2+(n*pi./a).^2)
x=0:0.01:2; y=0:0.01:1;
[X,Y]=meshgrid(x, y);
w11=sin(pi*Y./b).*sin(pi*X./a);
w12=sin(2*pi*Y./b).*sin(pi*X./a);
w21=sin(pi*Y./b).*sin(2*pi*X./a);
w22=sin(pi*Y./b).*sin(3*pi*X./a);
figure
subplot(2,2,1);   mesh(X,Y,w11)
subplot(2,2,2);   mesh(X,Y,w12)
subplot(2,2,3);   mesh(X,Y,w21)
subplot(2,2,4);   mesh(X,Y,w22)
```

可得从小到大排列的前 9 个本征值为

```
12.3370   41.9458   91.2938
19.7392   49.3480   98.6960
32.0762   61.6850  111.0330
```

和前面 4 个最小的本征值对应的本征函数的图形如图 3.4 所示.

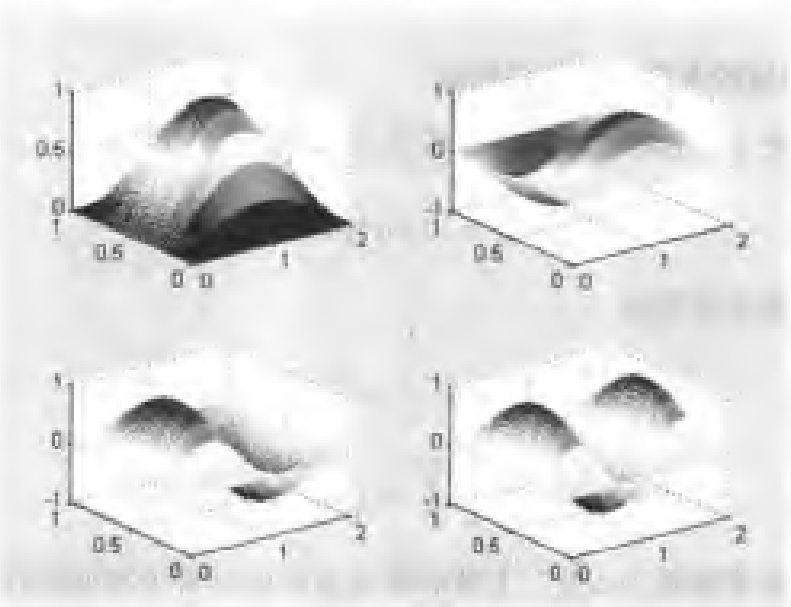


图 3.4　矩形膜的前 4 个本征函数图形

一般来说，在波动问题中时间因子的形式为

$$T_{mn}(t) = C_{mn}\cos\left(\sqrt{\lambda_{mn}}at\right) + D_{mn}\sin\left(\sqrt{\lambda_{mn}}at\right)$$

本征模与时间因子组合以后，就可以表示各个平面驻波分量的振动，下面的程序表示是时间因子取 $\sin\left(\sqrt{\lambda_{mn}}at\right)$ 时所对应的平面驻波运动.

```
b=2;    c=1;
x=0:0.02:b;y=0:0.02:c;
[X,Y]=meshgrid(x,y);
Z1=zeros(51,51);
for m=1:2
    for n=1:3
        for i=1:60
            a=sqrt((m*pi/c)^2+(n*pi/b)^2);
            Z=(sin(a*i*.02*pi))*sin(m*pi*Y./c).*sin(n*pi*X./b);
            surf(X,Y,Z);

            t=[' 本征振动:','m=',int2str(m),' n=',int2str(n)];
            title(t);
            axis([0,b,0,c,-1,1]);
            p(:,i)=getframe;
        end
        movie(p);
    end
end
```

图 3.5 是 $m=3, n=2$ 的本征模的几个运动画面.

偏微分方程工具箱可以直接解平面区域的本征值问题， 在指令窗口内键入 pdetool, 打开工具箱的界面窗口. 求解区域可取 $0 \leqslant x \leqslant 2$, $0 \leqslant y \leqslant 1$, 边界条件为四边固定，即正好是默认的设置，所以不必改变. 方程取 Eigenmodes(本征模式), 其形式为

$$-\nabla \cdot (c\nabla u) + au = \lambda \nabla du$$

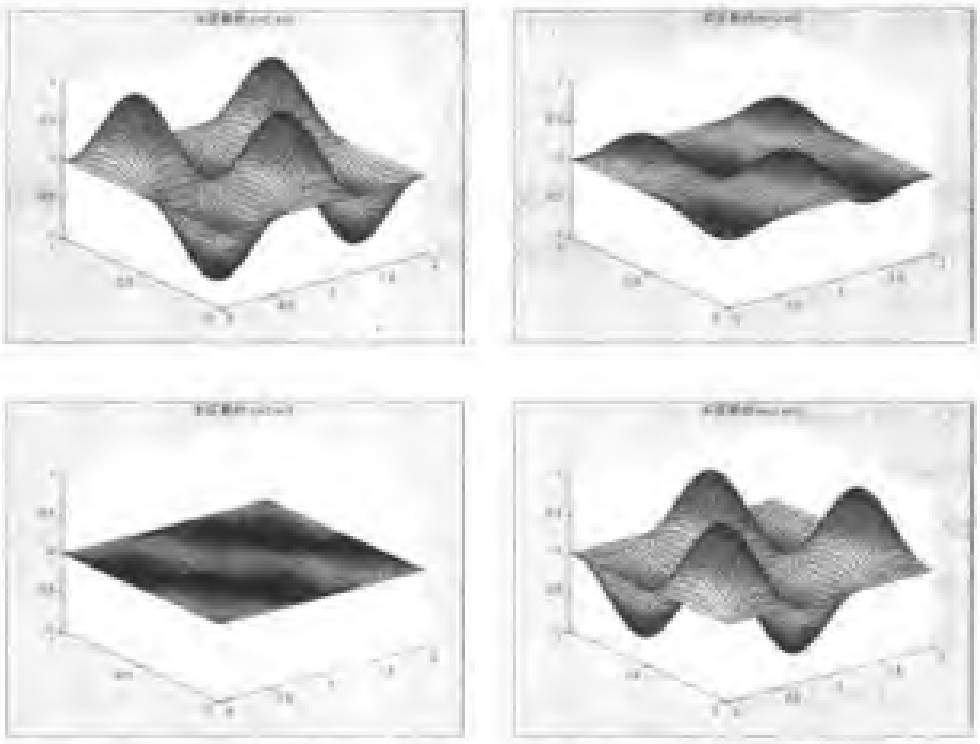


图 3.5 矩形区域本征模的运动

取 $c=1, a=0, d=1$ 。

在选择菜单 Solve/Parameter 时打开的对话框如图 3.6 所示。



图 3.6 设置本征值搜索范围

在对话框中可设置本征值搜索范围，默认的范围是 $0 \sim 100$ 。一般可保持不变。在作图对话框选择 Hight (3-D plot) 和 Plot in x−y grid 。注意在作图对话框中还有一个选择是 Eigenvalue(本征值)，即要求指定本征值以后再画本征函数的图形。图 3.7 和图 3.8 是前四个本征函数的图形，相应的本征值都显示在图形上。

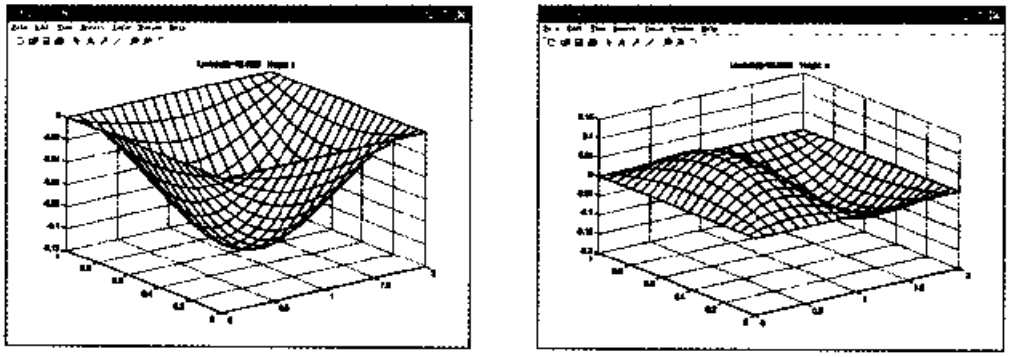


图 3.7 第一个本征函数和第二个本征函数

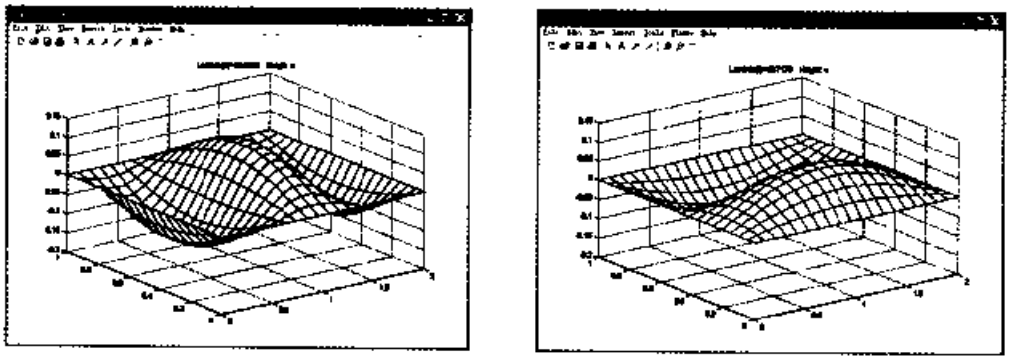


图 3.8 第三个本征函数和第四个本征函数

对于正方形区域，将会出现二重简并的情况，即相同的本征值可能对应两个不同的本征函数，不过这两个本征函数在图形上只是发生了一个转动。读者可以自己验证一下.

3.2.2 圆形区域的本征模与本征振动

半径为 ρ, 周边固定的圆形膜的本征问题是

$$\Delta u(\rho,\varphi) = \lambda u(\rho,\varphi)$$

本征模是

$$J_m(\sqrt{\mu}\rho)\sin m\varphi, \quad J_m(\sqrt{\mu}\rho)\cos m\varphi$$

其中 $\mu = \frac{x_n^m}{\rho_0}, m = 0,1,\cdots, n = 0,1,2,\cdots$, 这里 x_n^m 是 m 阶贝塞尔函数的第 n 个零点. 其中贝塞尔函数部分是区域 $[0,\rho_0]$ 之间 ρ 的本征函数. 作为一个例子，我们讨论关于 ρ 第一类齐次边界条件的本征值情形.

下面的程序画出 $m = 0, n = 0,1,2,3$ 的贝塞尔函数前 4 个本征函数图形如图 3.9 所示，其中取 $\rho_0 = 1$.

```
r=0:0.01:1;
```

```
y=inline('besselj(0,x)','x');
x(1)=fzero(y,1);
plot(r,y(x(1)*r))
hold on
for k=1:3
   x(k+1)=fzero(y,x(k)+3);
   plot(r,y(x(k+1)*r))
end
```

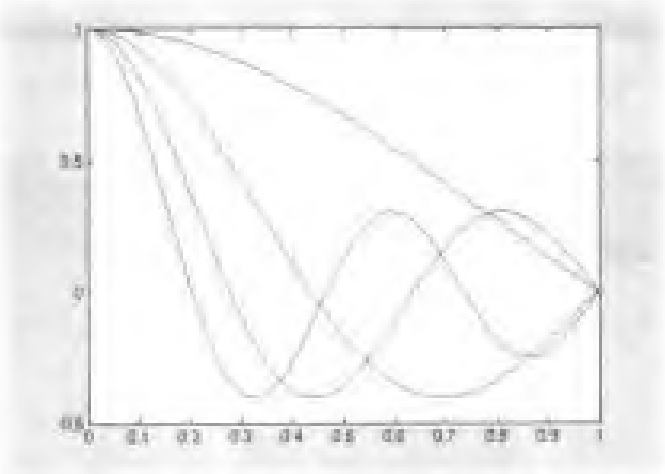


图 3.9 第一类齐次边界条件的贝塞尔函数 (m=0, n=0, 1, 2, 3) 的本征函数图形

φ 的本征函数有两种，即 $\sin m\varphi$、$\cos m\varphi$，φ 的本征函数和 ρ 的本征函数的组合形成了上述圆形区域的本征函数。

在圆形区域中 $m=0$ 的 4 个本征模，如图 3.10 和图 3.11 所示，这时 φ 本征函数是 1 。

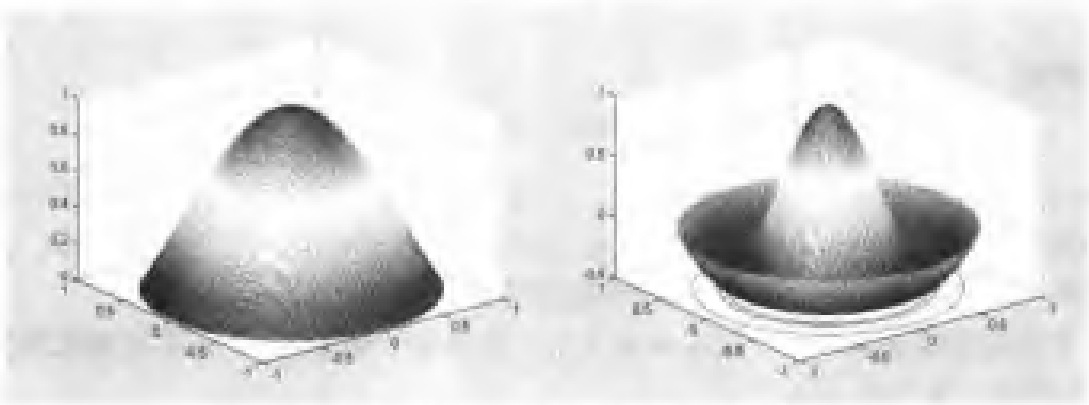


图 3.10 圆形区域 (m=0, n=0, 1) 的本征函数图形

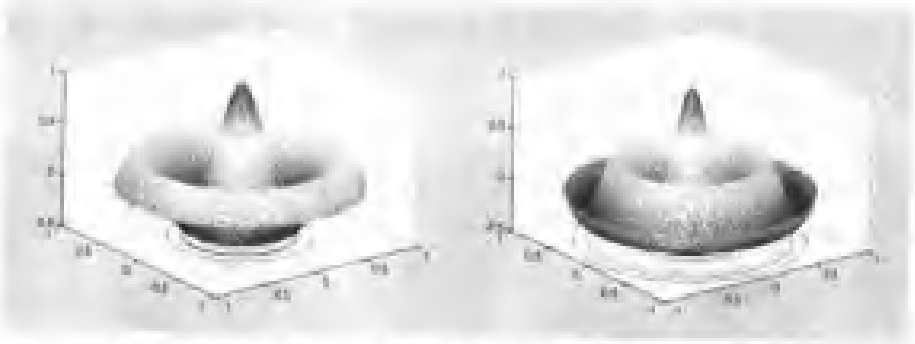


图 3.11 圆形区域内 (m=0, n=2, 3) 的本征函数图形

所用的程序是

```
r=0:0.01:1;
[XX,YY]=meshgrid(-1:0.01:1);
[Q, R]=cart2pol(XX,YY);
R(find(R>1))=NaN;
y=inline('besselj(0,x)','x');
x(1)=fzero(y,3)          %从x=3搜索第一个零点
meshc(XX,YY,y(x(1)*R))
for k=1:3
    x(k+1)=fzero(y,x(k)+2)
    figure
    meshc(XX,YY,y(x(k+1)*R))
end
```

下面再画出圆形区域 $m=1$ 时的前 4 个本征模式，作为例子我们取 φ 的本征函数 $\sin\varphi$ 来画图。

先画出第一类齐次边界条件的贝塞尔函数 (m=1,n=0,1,2,3) 的本征函数图形，如图 3.12 所示。

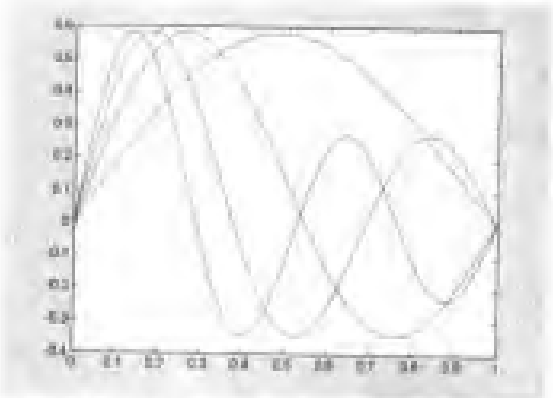


图 3.12 第一类齐次边界条件的贝塞尔函数 (m=1, n=0, 1, 2, 3) 的本征函数图形

再画出 $m=1, n=0,1,2,3$ 的圆形区域的前 4 个最小的本征模，如图 3.13 和图 3.14 所示.

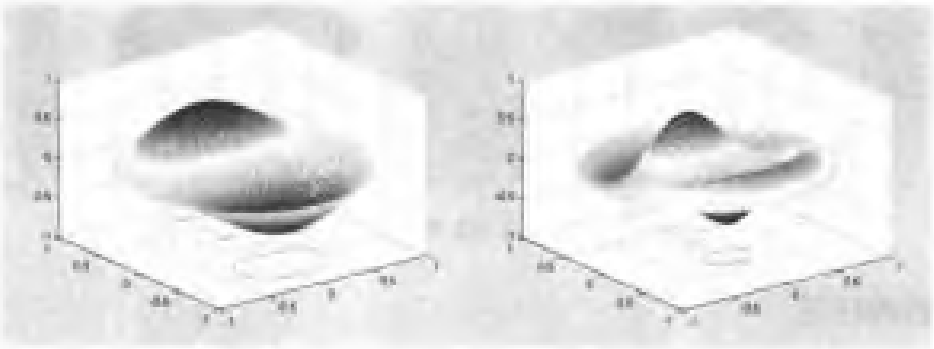


图 3.13 圆域内 (m=1, n=0, 1) 的本征函数图形

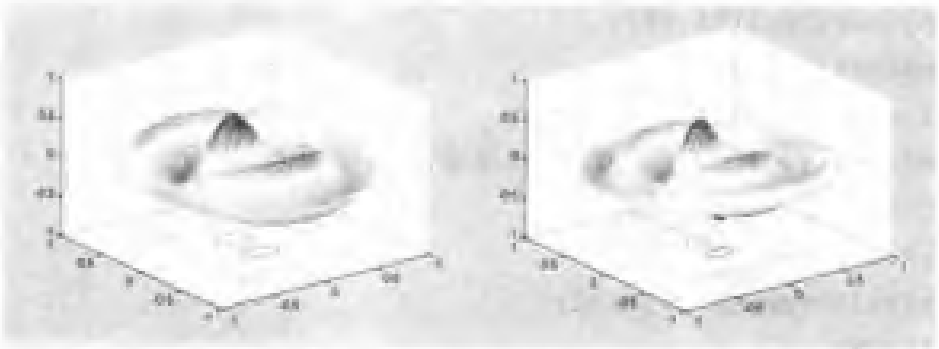


图 3.14 圆域内 (m=1, n=2, 3) 的本征函数图形

所用程序如下：

```
r=0:0.01:1;
[XX,YY]=meshgrid(-1:0.01:1);
[Q, R]=cart2pol(XX,YY);
R(find(R>1))=NaN;
y=inline('besselj(1,x)','x');
x(1)=fzero(y,3);
figure
meshc(XX,YY,y(x(1)*R).*sin(Q))
for k=1:3
    x(k+1)=fzero(y,x(k)+2)
```

```
    figure
     meshc(XX,YY,y(x(k+1)*R).*sin(Q))
end
```

可以看到，$m=0$ 的本征模是对称的，而 $m=1$ 的本征模是不对称的。不难想象，如果用 $\cos\varphi$ 作图，图形只是在 φ 的正方向旋转 $\pi/2$，它们的本征值相同，这种现象叫简并。为了与偏微分方程工具箱的数值解对比，这里再画出 $m=2, n=0,1,2,3$ 的图像，即取 $\sin 2\varphi$，程序与上面的相同，只需将程序中的 $\sin Q$ 改为 $\sin 2Q$ 即可，所得图像如图 3.15 和图 3.16 所示。

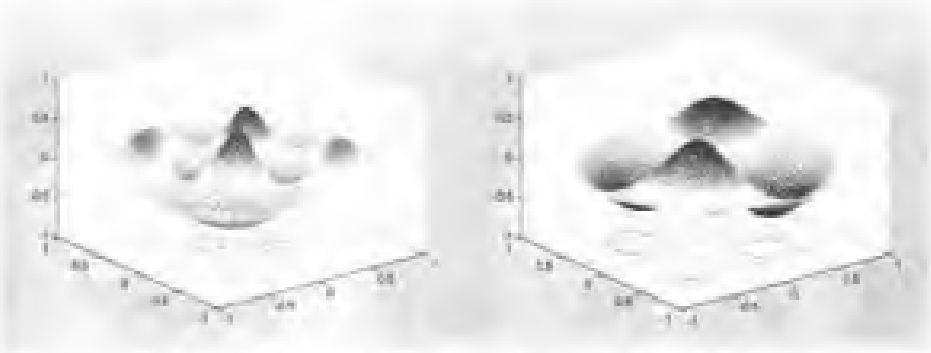


图 3.15 圆形区域 (m=2, n=0, 1) 的本征函数图形

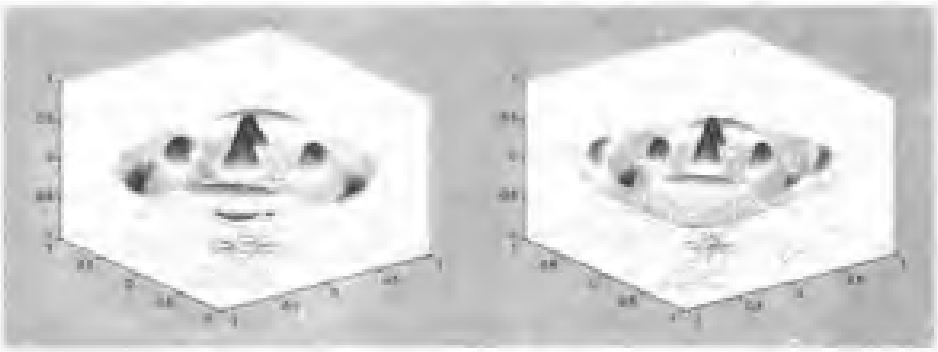


图 3.16 圆域内 (m=2, n=2, 3) 的本征函数图形

用偏微分方程工具箱来对这个问题作数值解并对照一下两个结果是很有趣的。步骤是：

在指令窗口内键入 pdetool，打开工具箱的界面窗口，以原点为圆心，画一个半径为 1 的圆，选择方程类型为 Eigenmodes(本征模式)，方程的形式为

$$-\nabla\cdot(c\nabla u)+au=\lambda\nabla du$$

其中系数取 $c=1, a=0, d=1$. 边界条件为周边固定, 即正好是默认的设置, 所以不必改变.

网格分割以后要细分一次, 以保证有足够的精度.

本征值搜索范围就用默认值是 $0 \sim 100$.

在作图对话框选择 Color,Hight (3-D plot) 和 Plot in x−y grid . Colormap 选 hsv 的效果更好. 在 Eigenvalue(本征值) 中选择前 4 个不同的本征值作图. 下面是前 4 个本征函数的图形, 相应的本征值都显示在图形上. 由于存在简并现象, 所以会出现非常相近的本征值. 由于这里的数值解是按本征值的大小排列本征函数的顺序, 所以这里有与 $m=0,1,2$ 对应的图形, 读者很容易从图 3.17 中找出与这些图形相对应的图形.

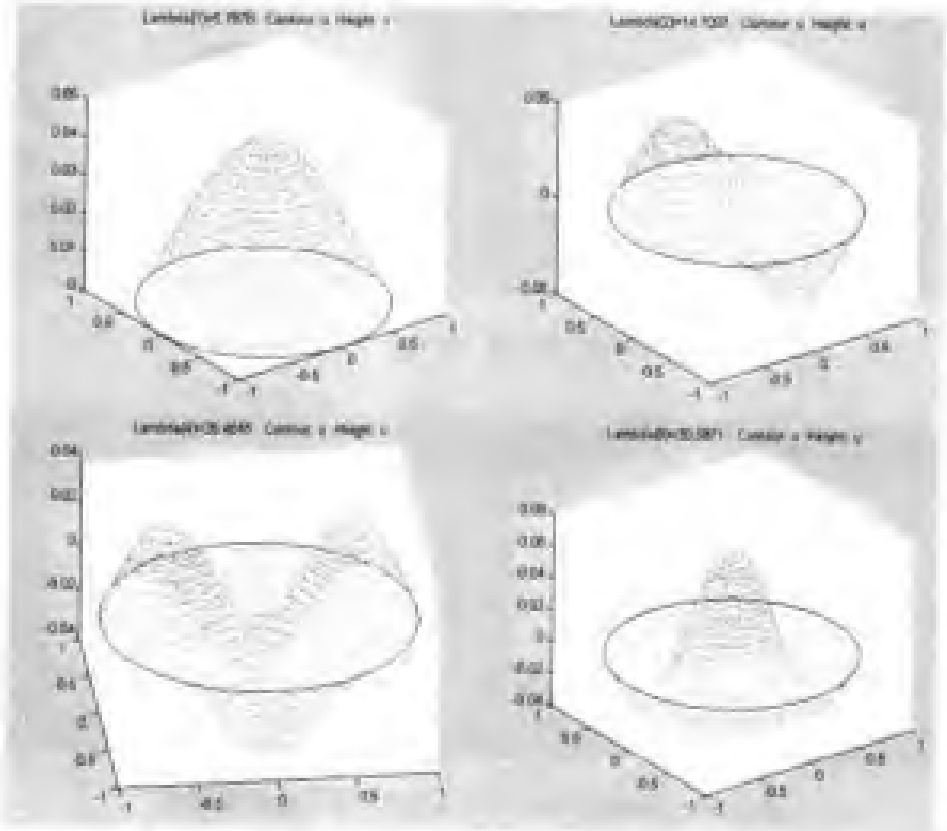


图 3.17 偏微分方程工具箱画出的圆域前 4 个本征函数

下面再画出环域 * 的前 4 个本征函数, 如图 3.18 所示.

*见参考文献 [1], 347 页.

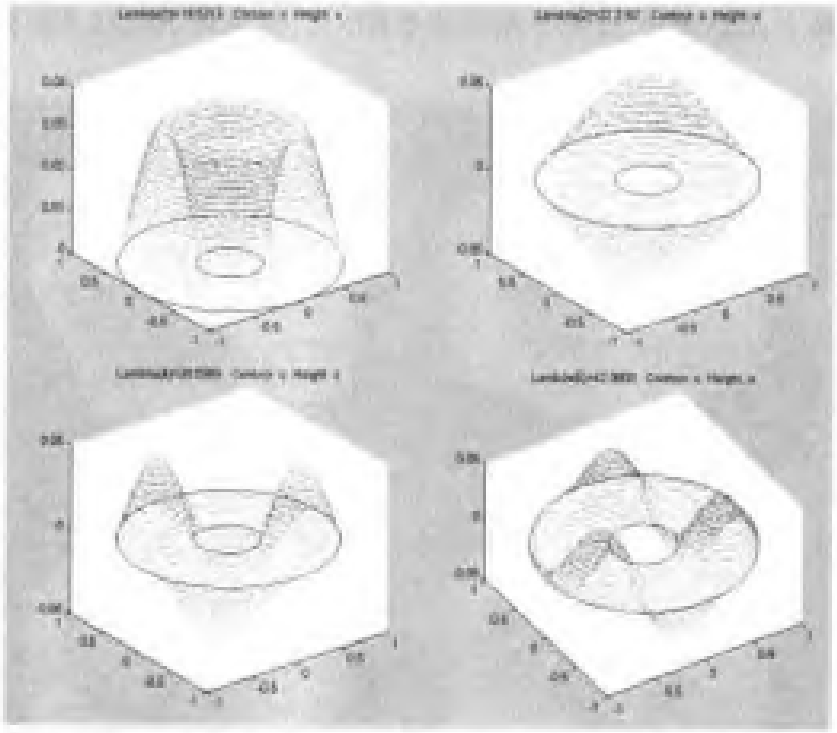


图 3.18 偏微分方程工具箱画出的环域前 4 个本征函数

图 3.19 是半圆域 * 的前 4 个本征函数图。

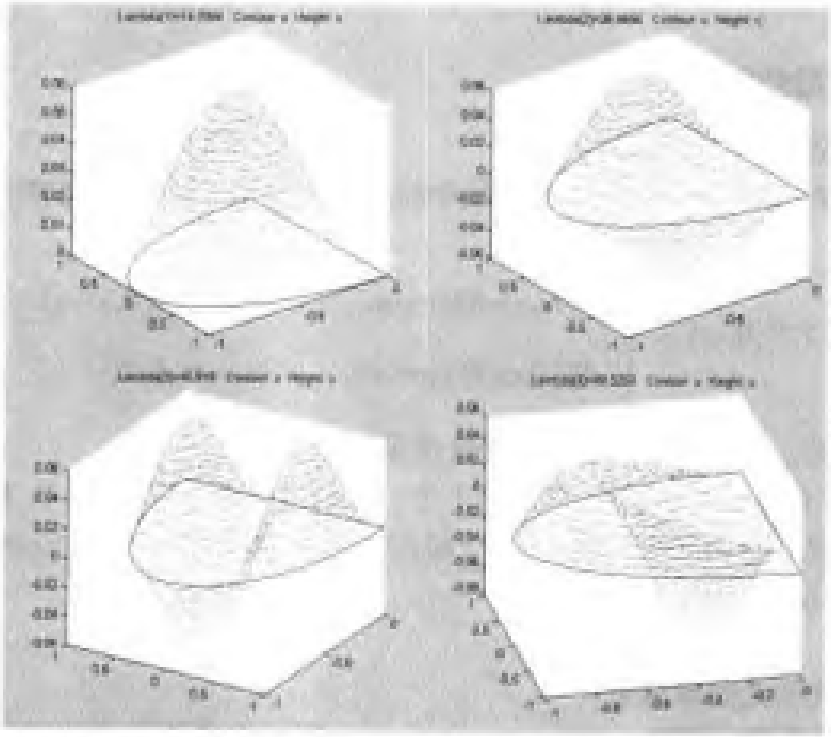


图 3.19 偏微分方程工具箱画出的半圆域前 4 个本征函数

*见参考文献 [1], 347 页。

图 3.20 是 L 形区域的前 4 个本征函数图，它是软件 MATLAB 公司的商标.

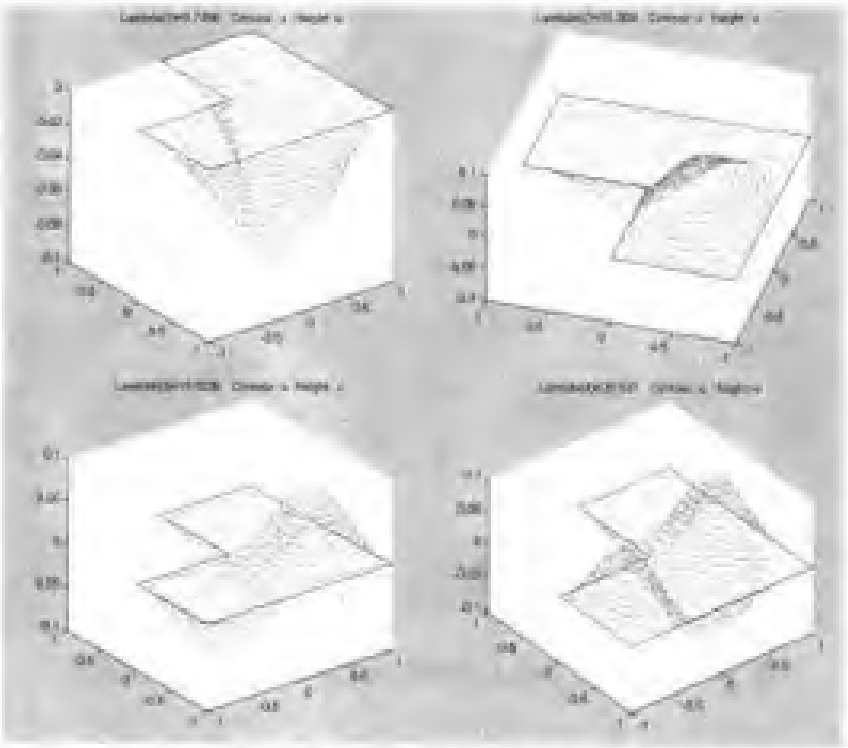


图 3.20 偏微分方程工具箱画出 L 形区域的前 4 个本征函数

3.2.3 球函数的图形

1. 实数形式的球函数

球谐函数 $Y_l^m(\theta,\varphi)$ 在不同的文献中的定义可能有所不同，通常实数形式的定义为

$$Y_l^m(\theta,\varphi)=\begin{cases} P_l^m(\cos\theta)\sin m\varphi, & m=1,2,3,\cdots,l \\ P_l^m(\cos\theta)\cos m\varphi, & m=0,1,2,\cdots,l \end{cases}$$

这里 $P_l^m(\cos\theta)$, $l=0,1,2,\cdots$ 是 l 阶连带勒让德函数.

线性独立的 l 阶球函数共有 $2l+1$ 个. 这是因为对应于 $m=0$, 有一个球函数 $P_l(\cos\theta)$, 而对应于 $m=1,2,3,\cdots,l$, 则各有两个球函数，即 $P_l^m(\cos\theta)\sin m\varphi$ 和 $P_l^m(\cos\theta)\cos m\varphi$.

2. 复数形式的球函数

根据欧拉公式 $\mathrm{e}^{\pm\mathrm{i}m\varphi}=\cos m\varphi\pm\mathrm{i}\sin m\varphi$, 可以将实数形式的球函数组合成如下的复数形式的球函数

$$Y_l^m(\theta,\varphi)=P_l^{|m|}(\cos\theta)\mathrm{e}^{\mathrm{i}m\varphi}$$

其中 $l=0,1,2,\cdots$, $m=-l,-l+1,\cdots,-1,0,1,\cdots,l$, 所以线性独立的 l 阶球函数还是有 $2l+1$ 个.

复球函数归一化的结果的是

$$Y_l^m(\theta,\varphi)=\sqrt{\frac{(l-|m|)!(2l+1)}{(l+|m|)!4\pi}}P_l^{|m|}(\cos\theta)\mathrm{e}^{\mathrm{i}m\varphi}$$

3. 球函数图形

球函数是在球面上的二元函数, 它是从球函数方程的本征值问题得到的本征函数系. 球函数的图形是空间的图形, 为了画出它的图形, 必须指定球的半径. 还有, 它是复数函数, 我们将它的实部与虚部分别作图. 还有一种画法是用角变量 θ,φ 作为平面上的 x,y 轴的变量, 也可以画球函数的图形.

为了画出球函数 $Y_l^m(\theta,\varphi)$ 的图形, 编写了以下的程序. 当 $l=0$ 时, $m=0$, $Y_0^0(\theta,\varphi)$ 是个常数, 不必作图. 作图可以从 $l=1$ 开始, 我们在程序中可任取 $l=3,m=2$, 画出图形依次为球函数实部和虚部的立体图形, 球函数实部、虚部以及绝对值的平面图形.

```
l = 3 ;    m = 2;
R= 4;    A= 3;
delta = pi/40;
theta0 = 0 :  delta :  pi;
phi = 0 :  2*delta :  2*pi;
[phi,theta] = meshgrid(phi,theta0);   %%构造 θ,φ 的数据网格

Ymn = legendre(l,cos(theta0));   %%计算勒让德函数的值
Ymn = Ymn(m+1,:)';
L = size(theta,1);
yy = repmat(Ymn, 1,L);
Reyy = yy.*cos(m*phi);   %%实球谐函数
Imyy = yy.*sin(m*phi);   %%虚球谐函数

  %%------------------ 以下语句画立体图形
ReM = max(max(abs(Reyy)));
Rerho = R + A*Reyy/ReM;
  %%计算实部在球坐标系下各点的值
Rer = Rerho.*sin(theta);
```

```
Rex = Rer.*cos(phi);
Rey = Rer.*sin(phi);
Rez = Rerho.*cos(theta);

subplot(1,2,1)   %%画实部的立体图形
surf(Rex,Rey,Rez)
light
lighting phong
axis('square')
axis([-5 5 -5 5 -5 5])
axis('off')
view(40,30)
title(' 实球谐函数 ')
ImM = max(max(abs(Imyy)));
Imrho = R + A*Imyy/(ImM+eps*(ImM==0));
Im = max(max(abs(Imyy)));
Imrho = R + A*Imyy/(ImM+eps*(ImM==0));

  %%计算虚部在球坐标系下各点的值
Imr = Imrho.*sin(theta);
Imx = Imr.*cos(phi);
Imy = Imr.*sin(phi);
Imz = Imrho.*cos(theta);

subplot(1,2,2)   %%画虚部的立体图形
surf(Imx,Imy,Imz)
light
lighting phong
axis('square')
axis([-5 5 -5 5 -5 5])
axis('off')
view(40,30)
title(' 虚球谐函数 ')

  %%---------------- 以下语句画平面图形
figure
```

```
subplot(3,1,1)    %%实部图形
surf(theta,phi,Reyy)
xlabel('\theta')
ylabel('\phi')

subplot(3,1,2)    %%虚部图形
surf(theta,phi,Imyy)
xlabel('\theta')
ylabel('\phi')

subplot(3,1,3)    %%球函数绝对值的图形
yy2=Reyy.^2+Imyy.^2;
surf(theta,phi,yy2)
xlabel('\theta')
ylabel('\phi')
```

下面就是用这些方法所画的球函数的图形，如图 3.21 ~图 3.25 所示。

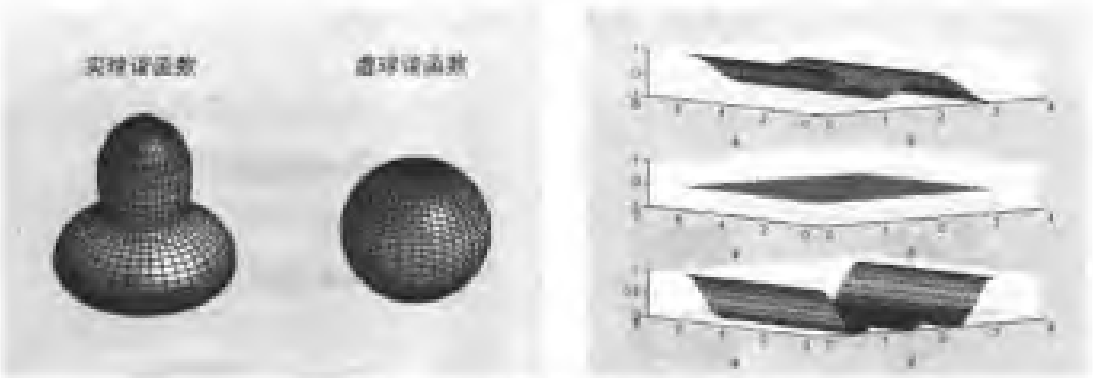


图 3.21 球函数 $Y_2^0(\theta,\varphi)$ 的图形

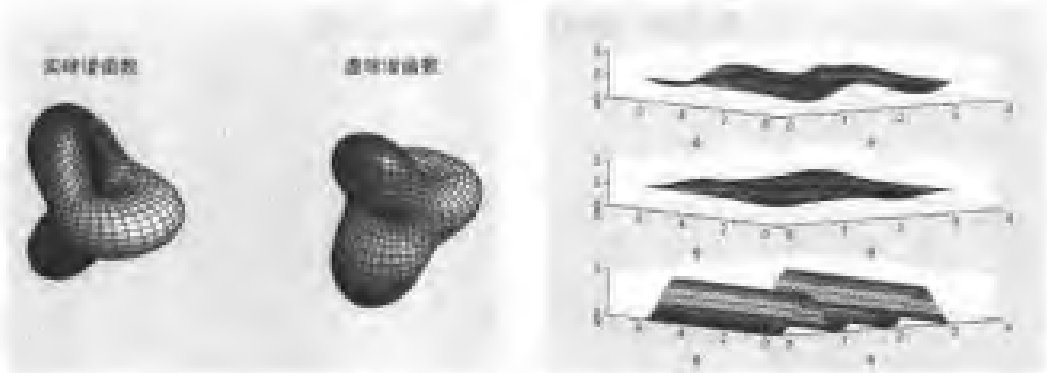


图 3.22 球函数 $Y_2^1(\theta,\varphi)$ 的图形

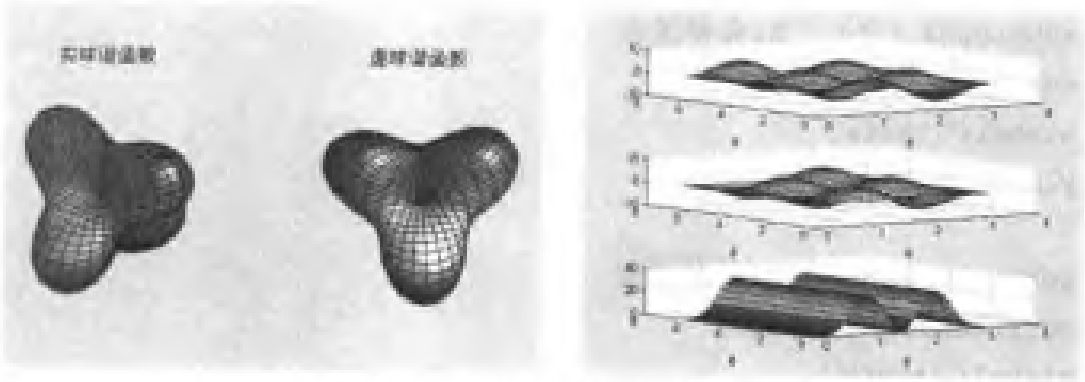


图 3.23 球函数 $Y_3^2(\theta,\varphi)$ 的图形

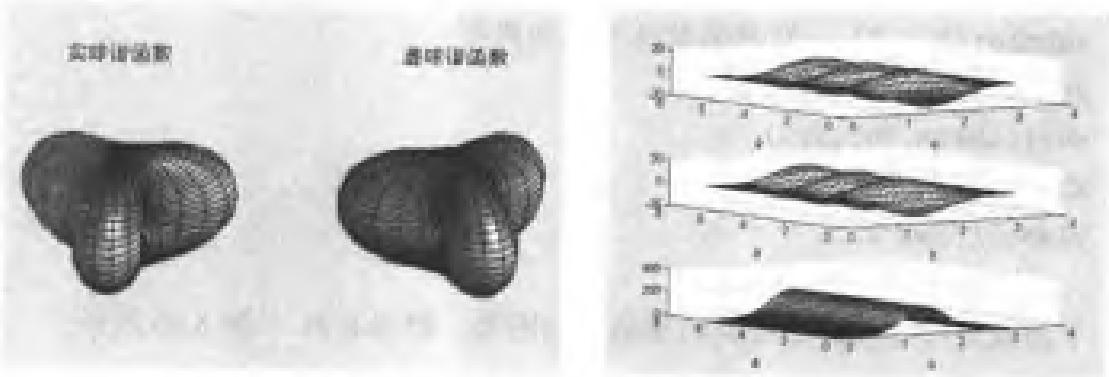


图 3.24 球函数 $Y_3^3(\theta,\varphi)$ 的图形

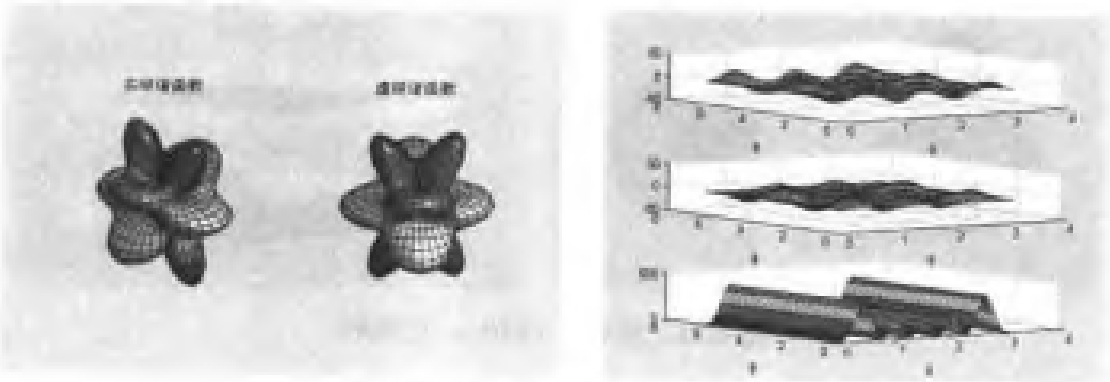


图 3.25 球函数 $Y_5^2(\theta,\varphi)$ 的图形

第 4 章　拉普拉斯方程与泊松方程

从本章起开始介绍偏微分方程的解法，首先介绍拉普拉斯方程与泊松方程的解法。方程中的变量都是空间变量，它只有二维与三维两种情况，对二维情况，用偏微分方程工具箱求解是十分方便，而且可以有不同的方法供选择。

这一章的例子较多，对各种不同的特殊函数的使用也较多，许多编程方法与技巧是有独到之处的。许多例子来自于电磁学，因为静电场电势可以用拉普拉斯方程与泊松方程描述，为了能更好表现电场的物理图像，我们往往会画出相应的电力线，这对学习矢量场可视化也是一个很好的机会。从后面的例子可以看出，这对于比较与正确理解问题的结果是很重要的。

4.1　二维拉普拉斯方程

4.1.1　矩形区域的拉普拉斯方程

求解定解问题 *

$$\begin{cases} u_{xx}+u_{yy}=0 & (0\leqslant x\leqslant a,\ 0\leqslant y\leqslant b) \\ u(x=0,y)=0, & u(x=a,y)=\mu\sin\dfrac{3\pi y}{b} \\ u(x,y=0)=0, & u(x,y=b)=\mu\sin\dfrac{3\pi x}{a}\cos\dfrac{\pi x}{a} \end{cases}$$

任意选取定解问题中参数的值，例如取 $\mu=1, a=1, b=1$，用偏微分方程工具箱来求解的步骤如下，因为是第一次使用，所以叙述较为详细。

1. 画求解区域

在指令窗口中，　输入 pdetool，打开偏微分方程工具箱的界面，　选择菜单 Options/ Axes Limits，打开对话框如图 4.1 所示。



图 4.1　设置坐标的对话框

* 见参考文献 [5]，125 页。

在 X-axis range 和 Y-axis range 栏中都输入 [−0.1 1.1]，单击按钮 Apply 确认，再关闭对话框。

单击按钮 □，画一个边长为 1 的正方形区域 $0 \leqslant x \leqslant 1$，$0 \leqslant y \leqslant 1$，这个正方形被自动命名为 R1，并显示在区域上方的公式栏 (Set Formula) 中。为了定位光标，可选择菜单 Options/Grid，在区域内画上网格，如图 4.2 所示。



图 4.2 画方程的正方形求解区域

如果要精确地确定正方形的位置，可以在正方形内部双击鼠标左键，打开图 4.3 所示的对话框。在对话框中输入正方形的左边界 (Left)、下边界 (Bottom)、宽度 (Width)、高度 (Height) 的数值，也可以在 Name 栏中改变正方形的名称。

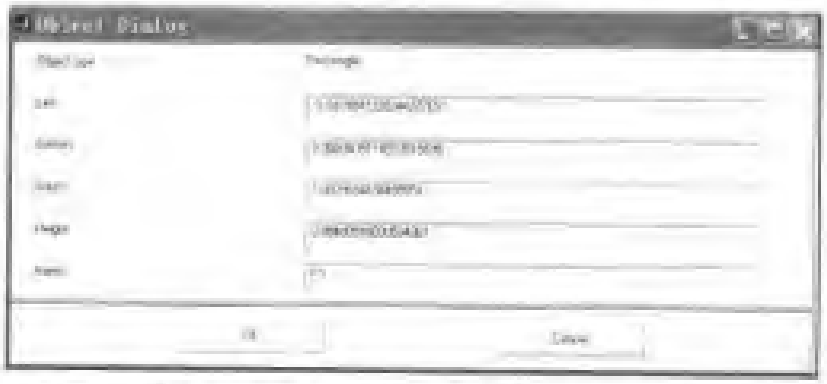


图 4.3 正方形区域的坐标和名称的对话框

如果要重新画区域，可选择菜单 File/New，

选择菜单 Options/Application/Heat Transfer，即求解热传导方程。

2. 设定边界条件

单击按钮 ，进入边界模式. 这时区域由灰色变成白色，而边界变成红色. 选择菜单 Boundary/Show Edge Labels，给四条边界标上序号 1, 2, 3, 4. 边界条件的默认设置是在所有的边界上 $u = 0$. 根据题意，在边界 3 和边界 4 有 $u = 0$, 所以这两条边界上的设置无需改变. 而在边界 1 上边界条件是 $u = \sin(3\pi x)\cos(\pi x)$, 在边界 2 上边界条件是 $u = \sin(3\pi y)$, 需要按下面的步骤设置. 双击边界 1，打开对话框如图 4.4 所示.

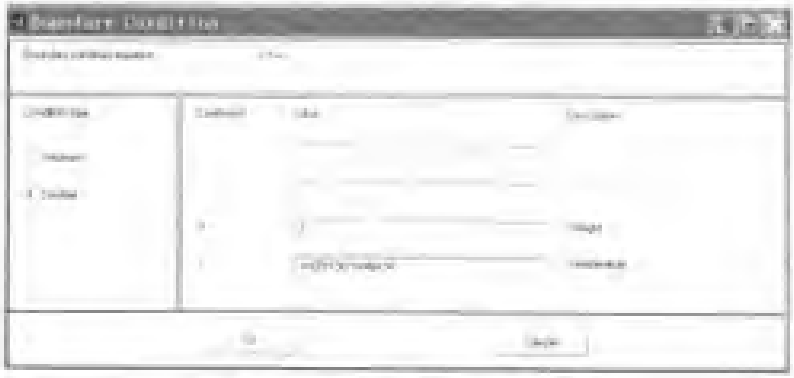


图 4.4 设置边界条件的对话框

在边界条件类型 (Condition Tyoep) 中选择 Dirichlet 也即第一类边界条件. 这时，描述边界条件的方程 (Boundary Condition equation) 是 $hT = r$, 这里的 T 即待求解的函数. 在 h 一栏中填入 1, 在 r 一栏中填入 sin(3 *pi .*x) .*cos (pi .*x), 注意不要填错. 单击按钮 OK, 然后关闭对话框. 用同样的方法设置边界 2 的边界条件，但在 r 一栏中填入的是 sin(3*pi.*y) .

3. 设定方程类型

单击按钮 ，打开对话框如图 4.5 所示.



图 4.5 设置方程类型的对话框

在方程类型 (Type of PDE) 中选择椭圆型 (Elliptic)，这时方程的形式为

$$-\nabla \cdot (k\nabla T) = Q + h(T_{ext} - T)$$

其中，k 是热扩散系数，Q 是热源，h 是对流传热系数，T_{ext} 是环境温度。按照题意，取 $k=1$，$Q=0$，$h=0$，$T_{ext}=0$。

补充说明一下，如果在菜单中选择的是 Option/Application/Generic scalar，这里出现的方程就是

$$-\nabla \cdot (c\nabla u) + au = f$$

只要取 $c=1$，$a=0$，$f=0$ 即可。

4. 建立网格

单击按钮 △，将区域划分为三角形网格。为了能得到更高的精度，再单击表示进一步细分的按钮 ⧋，将得到更小的三角形网格，如图 4.6 所示。

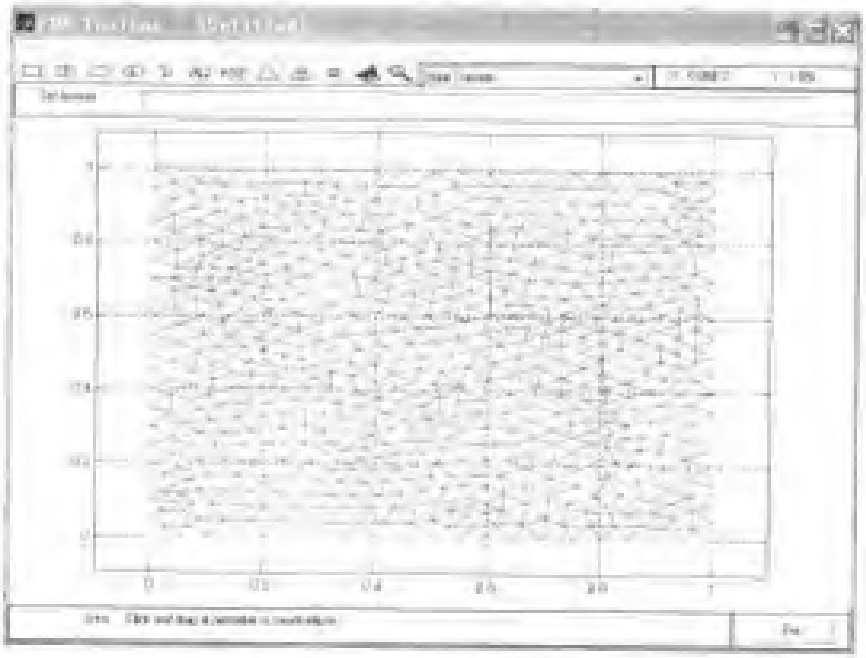


图 4.6 划分为三角形网格的区域

5. 输出图形

单击按钮 =，得到用颜色表示的解的分布图。为了得到更好的视觉效果，可单击按钮，打开对话框如图 4.7。



图 4.7 作图对话框

在其中选择 Hight (3-D plot) 和 Plot in x−y grid ，在这里 Plot in x−y grid 是用矩形网格画图，速度要快一些，而 Show mesh 则是用三角形网格作图，速度会慢一点。最后单击对话框中 Plot 按钮即得到结果如图 4.8 所示。

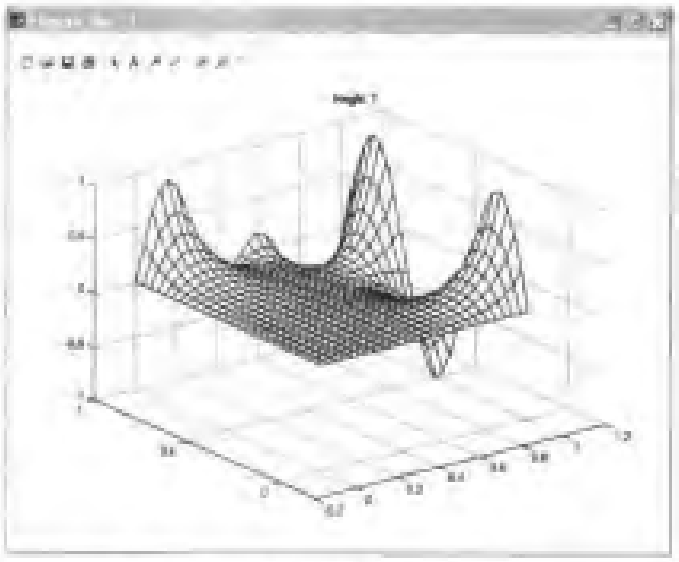


图 4.8 拉普拉斯方程的解的图形

事实上，这个问题是有解析解如下式：

$$u(x,y) = \frac{\mu}{2}\left[\frac{1}{\sinh\dfrac{2\pi b}{a}}\sinh\frac{2\pi y}{a}\sin\frac{2\pi x}{a} + \frac{1}{\sinh\dfrac{4\pi b}{a}}\sinh\frac{4\pi y}{a}\sin\frac{4\pi x}{a}\right]$$
$$+\mu\frac{1}{\sinh\dfrac{3\pi a}{b}}\sinh\frac{3\pi x}{b}\sin\frac{3\pi y}{b}$$

图 4.9 画出了这个问题的初始状态和解析解，以供对比.

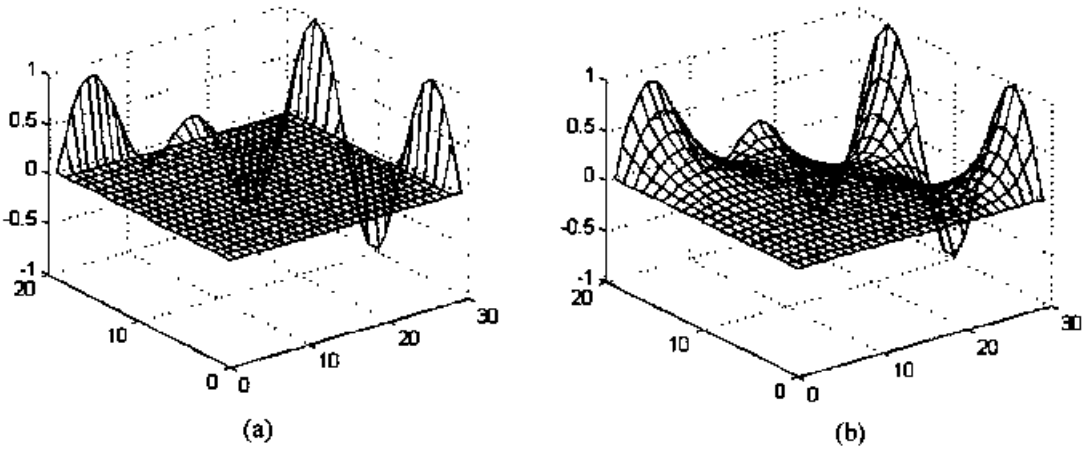


图 4.9 平面温度场问题图

(a) 初始状态; (b) 解析解

为了帮助读者练习偏微分方程工具箱的用法，下面给出两道简单的习题.

习题 1: 在圆域内求解 $\Delta u = 0$, 使满足边界条件为 $u(\rho = \rho_0) = A\cos\phi$.

习题 2: 圆环形区域的内外环半径为 ρ_1, ρ_2, 内环上保持温度为 $u_1\cos^2\phi$, 外环上温度为 $u_2\sin\phi$, 求此圆环区域内稳定的温度分布.

4.1.2 阳光照射的圆柱

半径为 a，表面熏黑的均匀长圆柱，在温度为零度的空气中受阳光的照射，阳光垂直于柱轴，热流强度为 q，试求柱内稳定的温度分布 * .

定解问题是

$$\begin{cases} \Delta u = 0 \\ (ku_\rho + Hu)|_{\rho=a} = f(\varphi) \end{cases}$$

其中

$$f(\varphi) = \begin{cases} q\sin\varphi & (0 < \varphi < \pi) \\ 0 & (\text{其余}) \end{cases}$$

这个问题的解是

$$u = \frac{q}{H\pi} + \frac{1}{k + Ha}\frac{q}{2}\rho\sin\varphi + \sum_{n=1}^{\infty}\frac{2q}{\pi a^{2n-1}(2nk + Ha)(1 - 4n^2)}\rho^{2n}\cos 2n\varphi$$

*见参考文献 [1], 202 页.

用下面的程序可以画出解析解的图形，程序中取 $a=1$; $H=1.5$; $k=0.3$; $h=0.2$; $q=5$;

```
a=1;   H=1.5;   k=0.3;   h=0.2;  q=5;   N=20;
p=0:0.05:1;       theta=0:pi/30:2*pi;
[Th,P]=meshgrid(theta,p);
[X,Y]=pol2cart(Th,P);
z=0;
u0=q/H/pi+1/(k+H*a)*q/2.*P.*sin(Th);
for n=1:N;
    zz=2*q/pi/a^(2*n-1)/(2*n*k+a*H)/(1-4*n^2)*P.^(2*n).*cos(2*n*
    Th);
    z=z+zz;
end
u=u0+z;
figure(1),    surfc(X,Y,u)
figure(2),    contour3(X,Y,u,20);
```

所得的图形曲面图和等值线如图 4.10 所示.

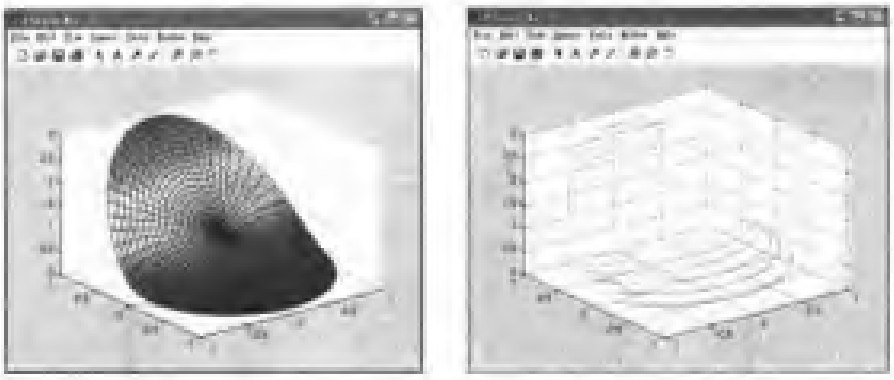


图 4.10 解析解的两个图形

再用偏微分方程工具箱求解这个问题. 以原点为心，画一个半径为 1 的圆.

按照题意，圆的下半周是齐次的狄里克利边界条件，可取 $h=1, r=0$. 圆的上半周是非齐次的狄里克利边界条件. 对话框中填入语句的左边是 h=1, r=5*sin(angle(x+i*y)), 右边是 h=1, r=5*sin(angle(x+i*y)).

方程的设置是 Elliptic 型，也就是 $c=0.3, a=0, f=0$.

为了有足够的精度，将初始化的网格作了两次细分，画两次图形，作图的选项分别为 Color 和 Contour，所得的图形如图 4.11 所示.

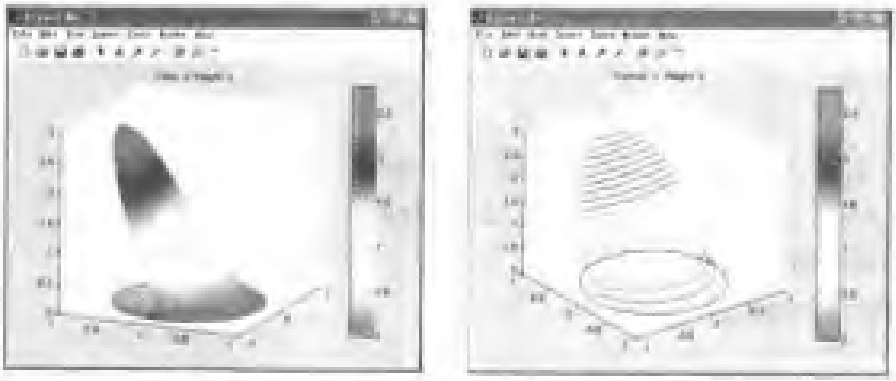


图 4.11 偏微分方程工具箱解得的结果

4.1.3 云与大地之间的电缆

带电的云与大地之间存在一个均匀的电场，平行于大地的电缆相当于一根无限长导体．在平行于电场的方向作垂直于电缆的截面，研究截面上电势分布*。

用偏微分方程工具箱求解这个问题．画一个中心在原点、大小为 2×2 的正方形 SQ1，在正方形中间再画一个圆 C1，圆心在 (0,0) 处，半径为 0.3．注意要将 Set formula 对话框中的公式写成 SQ1 - C1，表示求解区域是二者的之差．

矩形所有的边界条件是狄里克利边界条件，都取相同的 $h=1, r=y$，而圆的边界设为 $h=1, r=0$．

将方程的类型选为椭圆型 (Eliptic)，方程的系数设为 $c=1, a=0, f=0$．

网格初始化以后还要作一次细分．作图的选项为 Contour 和 Arrrows，所得的图形如图 4.12 所示．

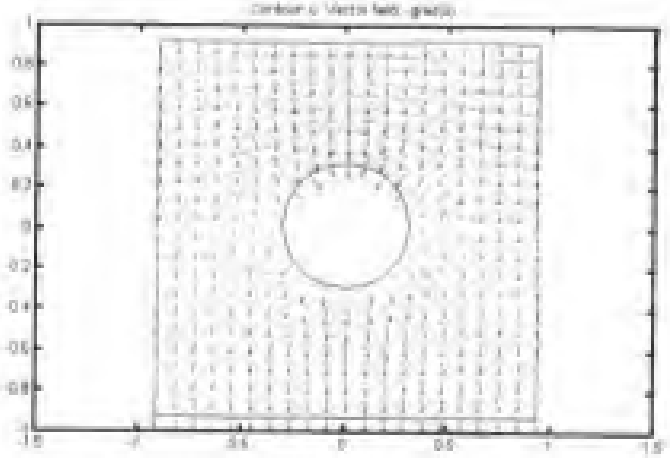


图 4.12 偏微分方程工具箱解得的结果

*见参考文献 [1]，194 页．

4.2 三维拉普拉斯方程

4.2.1 静电场中的介质球

在电场强度为 E 的静电场中放置均匀的介质球，球的半径是 r_0, 介电常数是 ε, 求介质球内外的电场强度 * .

用偏微分方程工具箱求解这个问题. 选坐标轴的原点在球心， z 轴平行于场的方向，则问题与 φ 无关，取过球心平行于场强方向的截面来作分析. 画一个中心在原点的 2×2 的正方形，在正方形中间再画一个圆，圆心在 $(0,0)$ 处，半径为 0.3 . 注意这时在 Set fomula 对话框中的公式是 SQ1 + C1 , 表示求解区域是二者的合并.

矩形所有的边界条件是狄里克利边界条件，都取相同的 $h=1, r=y$, 而圆的边界在矩形内部，自动由衔接条件决定，不必再设边界条件. 这是偏微分方程工具箱的特殊功能，应该掌握.

这是静电场问题，要用偏微分方程工具箱的专用格式. 在界面的工具栏中将方程的类型选为 Electrostatic . 在设置方程的参数时，先选菜单 PDE Mode, 再在圆内双击鼠标，打开对话栏以后，选 Eliptic 方程，方程系数 epsilon 取 0.2,rho 取 0, 关闭对话框. 然后再选菜单 PDE Mode, 在圆外双击鼠标，打开对话栏以后，选 Eliptic 方程，方程系数 epsilon 取 2,rho 取 0, 关闭对话框.

网格初始化以后还要作一次细分. 作图的选项为 Contour 和 Arrows, 所得的图形如图 4.14 所示.

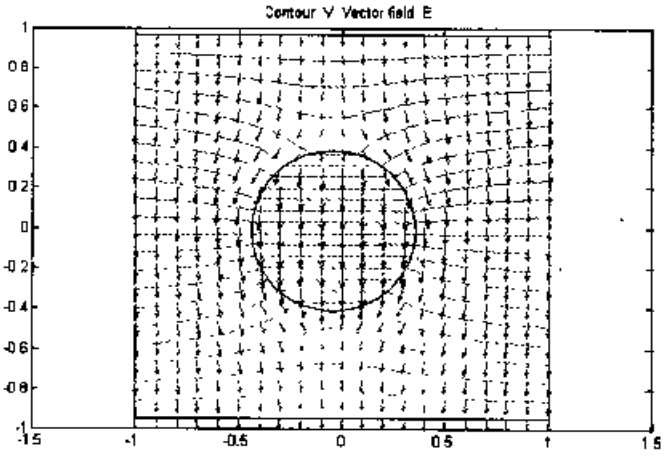


图 4.13 偏微分方程工具箱解得的结果

读者设想一下，如果在球内将方程系数 epsilon 设为 2, 在球外将方程系数

*见参考文献 [1], 287 页.

epsilon 设为 0.2, 那么等电势线将有什么变化?

4.2.2 带有电荷的细圆环的电场

均匀细圆环, 半径为 a, 环上带有 $4\pi\epsilon_0 q$ 单位的电荷, 求空间的电势分布 *.

本节用了三种方法来研究这道题目, 即解析解的可视化, 用偏微分方程工具箱求解, 直接用数值积分计算电势并作出电场的电力线.

1. 解析解的可视化

在圆环以外的区域, 电势满足拉普拉斯方程, 由电磁学的知识知, 在圆环的轴上, 距环心 r 处的电势为 $q/\sqrt{a^2+r^2}$. 可求得空间的电势分布为

$$u(r,\theta)=\begin{cases}\dfrac{q}{a}\displaystyle\sum_{l=0}^{\infty}(-1)^l\frac{(2l)!}{2^{2l}l!l!}\left(\frac{r}{a}\right)^{2l}P_{2l}(\cos\theta) & (r<a)\\ \dfrac{q}{a}\displaystyle\sum_{l=0}^{\infty}(-1)^l\frac{(2l)!}{2^{2l}l!l!}\left(\frac{a}{r}\right)^{2l+1}P_{2l}(\cos\theta) & (r>a)\end{cases}$$

这个解与 φ 无关, 用下面的程序可以画出它 xoz 面内的图形. 程序中先构造一个方形的 x,y 的数据网格, 并将它转换成用 θ,r 表示的数据网格. 以环的半径 a 为界, 分成圆内和圆外两部分分别计算电势. 最后将两部分的电势相加, 再画成图.

```
a=0.5; q=1;
[X,Y]=meshgrid([-1.5:0.02:1.5],[-1.5:0.02:1.5]);
[theta,r]=cart2pol(Y,X);
rout=r;      rout(find(rout<a))=NaN;
rin=r;       rin(find(rin>a))=NaN;
Uin=q/a;     Uout=q./rout;
rin=rin./a;     rout=a./rout;
for k=1:10
   fun=legendre(2*k,cos(theta));
   rfun=q/a*squeeze(fun(1,:,:));
   ck=(-1)^k*prod(1:2*k)/2^(2*k)/(prod(1:k))^2;
   ukin= ck.*rin.^(2*k);              %系数
   ukout= ck.*rout.^(2*k+1);
   Uin=Uin + ukin.*rfun;       %总和
   Uout=Uout+ukout.*rfun;
```

*见参考文献 [1], 297 页.

```
end
contour(Uout,14,'r')
hold on
contour(Uin,11,'b')
title(' 等势线')
```

所得到的图形如图 4.14 所示，可以看到，等势线很像两个等量同号的点电荷所产生的电势。在 $r=a$ 处，电势是不连续的，所以圆内与圆外的电场的等势线并没有连接上。

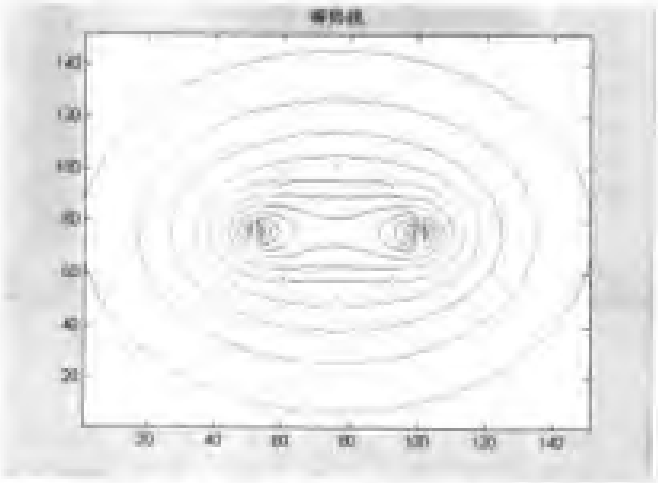


图 4.14 解析解所画出电势

2. 用偏微分方程工具箱求解

以原点为圆心画一个半径为 1 的圆 C1，在其内部再画两个半径为 0.05 的小圆 C2 和 C3，圆心位置分别是 (−0.4，0) 和 (0.4,0)。这里是以小圆代表圆环的两个截面，以大圆表示求解的区域。在 Set Formula 区域中设置的公式为 C1+C2+C3。

大圆的边界条件是狄里克利边界条件，取 $h=1, r=0$，小圆在大圆的内部，它的边界条件是自动由衔接条件决定，不必另设。

再来设置方程。在大圆与小圆之间的区域，没有电荷，满足拉普拉斯方程。在小圆的内部，它代表的是细圆环的截面，应该看作一个无限小的点，这在图中是无法表示的。作为一种近似，我们画一个尽量小的圆，并且认为小圆内的电荷是均匀分布的，也就是认为它满足泊松方程。按照这个分析，我们设置方程的步骤是，在菜单中选 PDE Mode，单击一个小圆内部，在打开的 PDE Specifacation 对话框中，选取方程为 Elliptic，系数为 c=1,a=0,f=0.2；在另一个小圆中也要用同样的步骤作同样的设置。在大圆与小圆之间的区域，则在菜单中选 PDE Mode 之后，将它设置为拉普拉斯方程，系数为 c=1, a=0, f=0。

为了有足够的精度，划分的网格要细分一次。

作图时选择 Color,Contour，在 Contour plot leves 填入 10，在 Colormap 一栏中选 cool，最后单击按钮 Plot 即可，画出的图形如图 4.15 所示。对比两个图形，可以看出它们的一致性。

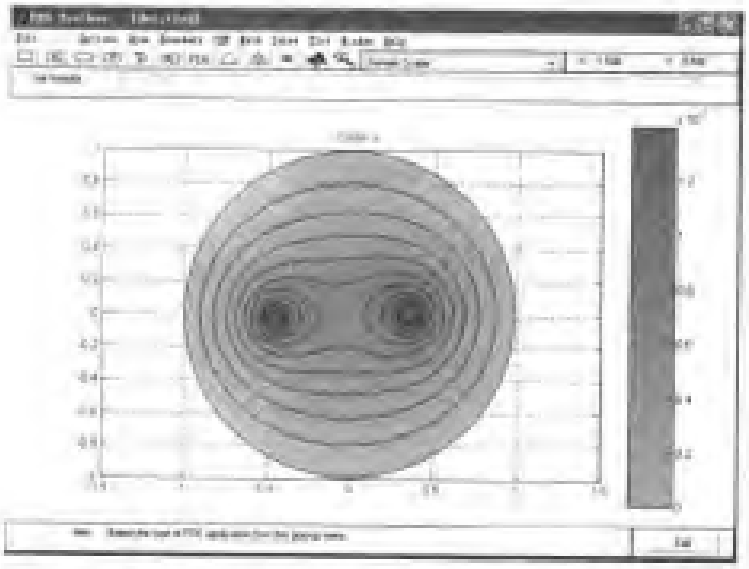


图 4.15 偏微分方程工具箱画出的电势

3. 数值积分法

作为对比，我们直接用库仑定律来计算均匀带电的细圆环在空间产生的电势，而且更进一步，还画出它的电力线，以便更清楚的显示它的物理图像。如图 4.16 所示建立坐标。

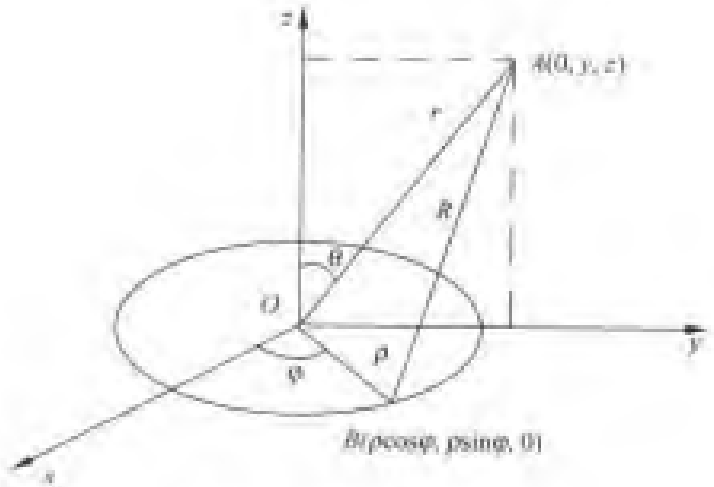


图 4.16 带有电荷的圆环

细圆环的电荷密度是 λ, 取无穷远的电势为零, 则带电圆环在在空间任意一点 $Q(x,y,z)$ 所产生的电势是

$$v(x,y,z)=\frac{1}{4\pi\varepsilon_0}\int_0^{2\pi}\frac{\lambda\rho\mathrm{d}\varphi}{\sqrt{(x-\rho\cos\varphi)^2+(x-\rho\sin\varphi)^2+z^2}}$$

取上式中的 $x=0$, 得到在带电圆环在 yOz 面上点 $A(0,y,z)$ 产生的电势. 这个数值积分可以直接用 MATLAB 计算结果, 并且还可以进一步由电势求出空间一点的电场为

$$\boldsymbol{E}=-\nabla\varphi$$

这是一个简单的微分运算, 可以直接用梯形积分指令 trapz 计算电势, 然后用计算梯度的指令 gradient 计算电场. 画电力线的指令是 streamline, 它是 MATLAB 画矢量场流线的指令. 画图所用的程序是:

```
clear;
R=1;   y=-4:0.11:4;     z=-4:0.11:4 ;
the=-0.5*pi : pi/400 : 0.5*pi;
[Y,Z,T]=meshgrid(y,z,the);
r=sqrt(cos(T).^2+Z.^2+(Y-sin(T)).^2);
dv = 1./r ;
v=trapz(dv,3);
[ey,ez]=gradient(-v,0.5);

figure(1)
subplot(2,2,1)
contour(Y(:,:,1),Z(:,:,1),v,10)

subplot(2,2,2)
[SY,SZ]=meshgrid(-4:0.2:4,[0.1,-0.1]);
streamline(Y(:,:,1),Z(:,:,1),ey,ez,SY,SZ);
[SY,SZ]=meshgrid([-4:0.2:4],[-0.005,0.005]);
streamline(Y(:,:,1),Z(:,:,1),ey,ez,SY,SZ);

R=1;  the= 0: pi/20 :2* pi;
x= -0.2:0.11:3;   y= -0.2:0.11:3;     z= -0.2:0.11:3 ;
[X,Y,Z,T]=ndgrid(y,x,z,the);
r=sqrt((X-cos(T)).^2+(Y-sin(T)).^2+Z.^2);
dv = 1./r ;
v=trapz(dv,4);
```

```
[ex,ey,ez]=gradient(-v,0.5);
[XX,YY,ZZ]=meshgrid(-0.2:0.11:3);
[SX,SY,SZ]=meshgrid(0:0.5:3,  0:0.5:3, [0.05]);

subplot(2,2,3)
Zmin=min(Z(:));    Zmax=max(Z(:));
streamslice(XX,YY,ZZ,ex,ey,ez,[],[],0.01)

subplot(2,2,4)
x=cos(the);y=sin(the);z=zeros(1,length(the));
plot3(x,y,z,'LineWidth',3,'Color','r');
hold on
h1=streamline(XX,YY,ZZ,ex,ey,ez,SX,SY,SZ) ;
h2=copyobj(h1,gca);  %用图形复制与旋转画其余的象限,代替下面三句
rotate(h2,[1,0,0],180,[0 0 0]);
h3=copyobj(allchild(gca),gca);
rotate(h3,[0,1,0],180,[0 0 0]);
view(-28,8)
```

图 4.17 中的 (a) 和 (b) 图分别是带电圆环的空间电势与圆环所在平面内第一象限的电力线分布。图像显示，电力线都是从环所在位置发出的，因为只有环上有电荷存在。电力线会聚在环的中央。

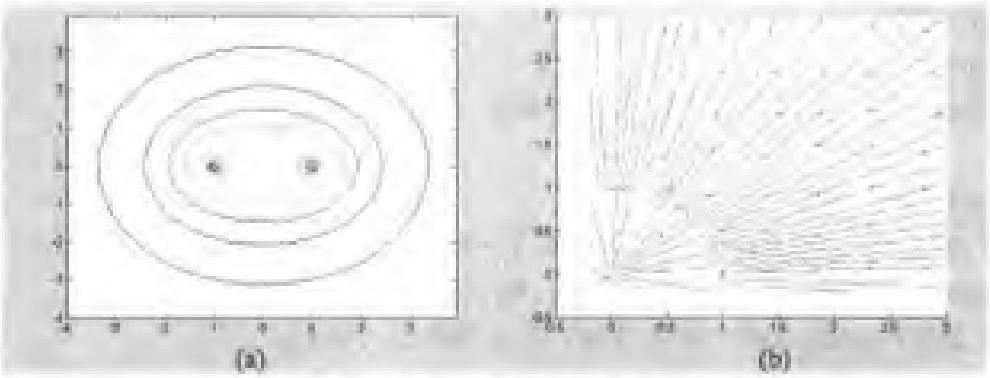


图 4.17 带电圆环的电场图之一

(a) 带电圆环的空间电势；(b) 圆环所在平面的电力线

图 4.18(a) 是二维的电力线图形，图 4.18(b) 是三维的电力线图形。从这些图上可以更清楚地看清电力线的走向。

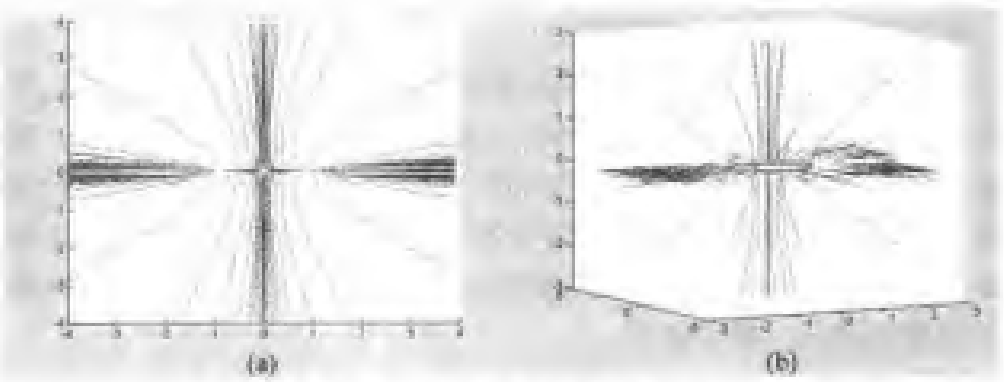


图 4.18 带电圆环的电场图之二

(a) 二维的电力线图形; (b) 三维的电力线图形

4.2.3 均匀圆盘的引力势

均匀圆盘的半径为 a, 质量为 M, 求它在空间的引力势 *。

在这道题目中, 我们分别用 MATLAB 和 MAPLE 的指令将解析解可视化, 这样做的目的是为了介绍 MAPLE 指令的应用。

定解问题是

$$\begin{cases} \Delta u = -4\pi G Q(r,\theta) \\ u(\theta=0)\text{有解}, \qquad u(\theta=\pi)\text{有解} \\ u(r=0)\text{有解}, \qquad u(r\to\infty)\to 0 \end{cases}$$

其中

$$Q(r,\theta) = \frac{M}{\pi a^2 r}\delta\left(\theta-\frac{\pi}{2}\right)H(a-r)$$

$$H(a-r) = \begin{cases} 1 & (a>r) \\ 0 & (a<r) \end{cases}$$

问题的解是

$$u_{\text{out}} = \frac{GM}{a}\sum_{l=0}^{\infty}\frac{1}{l+1}\left(\frac{a}{r}\right)^{2l+1}P_{2l}(0)P_{2l}(\cos\theta)$$

$$u_{\text{in}} = \frac{2GM}{a}\frac{r}{a}\sum_{l=0}^{\infty}\left(\frac{1}{2l-1}+\frac{1}{2l+2}\right)P_{2l}(0)P_{2l}(\cos\theta) - \frac{2GM}{a}\sum_{l=0}^{\infty}\frac{1}{2l-1}\left(\frac{r}{a}\right)^{2l}P_{2l}(0)P_{2l}(\cos\theta)$$

*见参考文献 [11], 382 页。

1. 用 MATLAB 指令作可视化

这个解画成图形可以清楚的表现出解的物理图像. 编程的思路是, 考虑到解是用球坐标表示, 但是与 φ 无关, 所以只要构造一个 r,θ 的数据网格, 圆盘内和圆盘外的区域用 r 的大小来区分. 然后用 MATLAB 的指令 legendre 计算勒让德函数的值, 进而用 for 循环求出级数解中各项的值并累加起来. 为了画出解的等值线, 还要将解再表示成直角坐标的数据网格, 最后作图. 在图中, 球内与球外的解分别用不同颜色表示. 程序如下:

```
a=0.35;      GM=1/4;      rin=0:0.005:a;      rout=a:0.005:1;
theta= 0:0.01:2*pi;      z=cos(theta);      uout=0;      uin=0;
for k=0:2:20
      fun=legendre(k,z);
      fun2=fun(1,:);
      fun20=legendre(k,0);
      fun20=fun20(1);

      Rin=2/a.^2*(1/(k-1)+1/(k+2)).*rin - 2/a/(k-1).*(rin./a).^k;
      [NRin, Nfunb2]=meshgrid(Rin, fun2);
      ukin = 1/4*fun20*NRin.*Nfunb2;
      uin=uin + ukin;

      Rout=1/a/(k/2+1)*(a./rout).^(k+1);
      [NRout, Nfuna2]=meshgrid(Rout, fun2 );
      ukout = 1/4*fun20*NRout.*Nfuna2;
      uout=uout + ukout;
end

[rout,thetaout] = meshgrid(rout, theta);
Yout=rout.*cos(thetaout);
Xout=rout.*sin(thetaout);

[rin,thetain] = meshgrid(rin, theta);
Yin=rin.*cos(thetain);
Xin=rin.*sin(thetain);
cla
```

```
contour(Xout,Yout,uout,'linespec','g')
hold on
contour(Xin,Yin,uin,'linespec','r');
axis equal
```

作出的图形如图 4.19 所示.

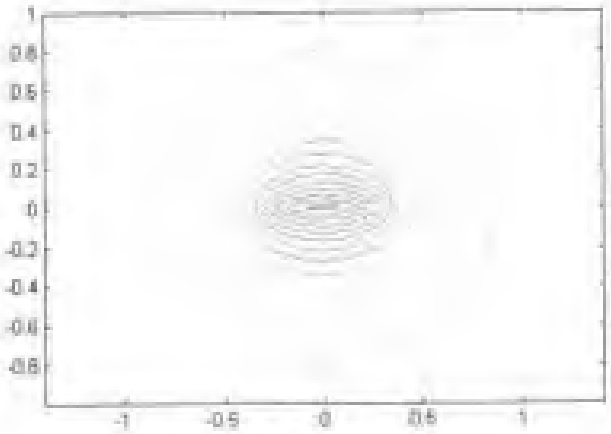


图 4.19 MATLAB 指令画的解析解的图形

2. 用 MAPLE 指令作可视化

为了使大家熟悉 MATLAB 中调用 MAPLE 函数的方法，下面的程序是使用 MAPLE 的函数来计算勒让德函数的值，编程的思路与前一个程序相同，只是使用的指令不同，关于 MAPLE 指令的用法，见前面的介绍. 从画出的图形可以看出，两个图形是相同的.

```
x1=-0.995 :0.01:1;    y1=-0.995:0.01:1;
[X1,Y1]=meshgrid(x1,y1);
theta1=atan(X1./Y1);
r1=sqrt(X1.^2+Y1.^2);       r2=r1;
z1=cos(theta1);             z2=z1;
a=0.35;    CM=1/4;

uout=0;     uin=0;
maple('with(orthopoly)');
for k=0:2:20
   d=['P(',int2str(k),',x)'];
```

```
    fun= maple(d);
    pfun=inline(fun,'x');
    pfun=vectorize(pfun);
    pfun20=(pfun(0));
    r2(find(r2<=0.35))=NaN;
    R2=1/a/(k+1)*(a./r2).^(2*k+1);
    pfun2=pfun(z2);

    ukout = 1/4*R2.*pfun2.*pfun20;
    uout=uout + ukout;

     r1(find(r1>0.35))=NaN;
     R1=2/a.^2*(1/(2*k-1)+1/(2*k+2)).*r1 - 2/a/(2*k-1).*(r1./a)
     .^(2*k);
     ukin = 1/4*R1.*pfun2.*pfun20;
     uin=uin + ukin;
end
cla
contour(X1,Y1,uout,'linespec','g')
hold on
contour(X1,Y1,uin,'linespec','r')
axis equal
```

所得的图形如图 4.20 所示.

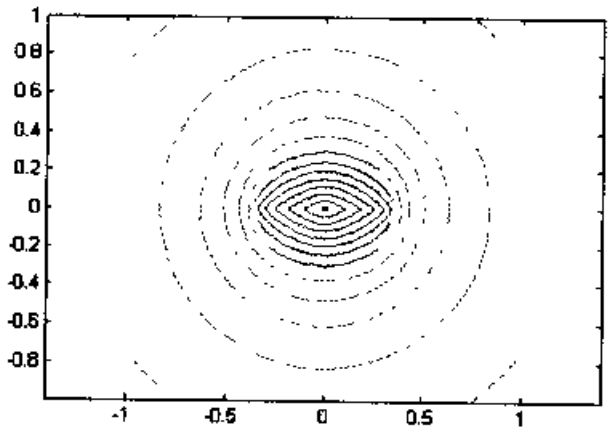


图 4.20 MAPLE 指令画的解析解的图形

3. 用偏微分方程工具箱求解

下面用偏微分方程工具箱画这个问题的数值解，方法与上一节相同，取 x-z 面作为画图的截面，画一个 $-1.5 \leqslant x \leqslant 1.5, -1 \leqslant y \leqslant 1$ 的矩形 R1，在中心再画一个小的矩形 R2，其值为 Left 取 -0.4;Bottom 取 -0.1;Width 取 0.8;Heigh 取 0.2，表示圆盘的水平截面，在 Set Fomula 区域中设置的公式为 R1+R2 。

矩形 R1 的边界条件是 Dirichlet 型，都取相同的 $h=1, r=0$ 。内部的小矩形 R2 的边界条件由衔接条件自动决定，不用设置。

在设置方程的参数时，先选菜单 PDE Mode，再在圆内双击鼠标，打开对话栏以后，选 Eliptic 方程，小矩形内部取 c=1，a=0，f=1./sqrt(x.^2+y.^2)./0.4^2 关闭对话框，然后再选菜单 PDE Mode，在小矩形外双击鼠标，打开对话栏以后，选 Eliptic 方程，方程系数取 $c=1, a=0, f=0$，关闭对话框。

在菜单中单击 Solve/Parameter，打开对话框以后，勾选 Adaptive mode 项，关闭对话框。

网格初始化以后还要作两次细分，作图的选项为 contour，所得的图形如图 4.21 所示，这个结果与解析解的结果是相吻合的。

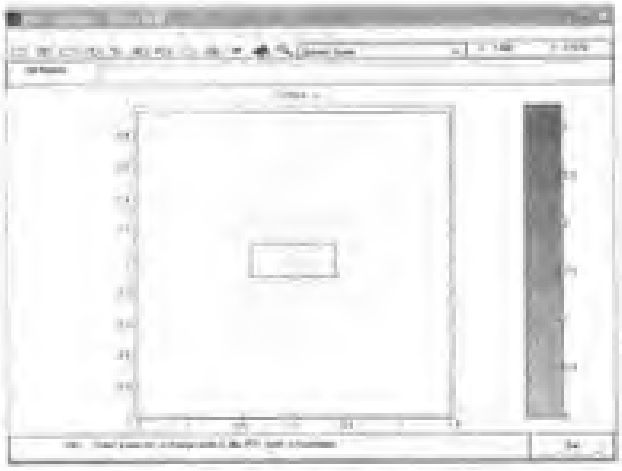


图 4.21 偏微分方程工具箱解得的结果

4.2.4 环形电流的磁感应强度

圆线圈的半径为 a，通有电流 I，求空间任意一点的磁感应强度 B 。

这是一个有趣的例子，我们用三种不同的方法求解，尽管所用的解法不同，所得结果的表达式也不同，但是所得到的磁场是一致的，当然，要从画出的磁力线才能看出这一点，而这需要用到 MATLAB 的画场线的专用指令，如果仅仅画出磁场的分量，由于所用的坐标不同，所得到的图形并不能看出它们的一致性。

1. 用连带勒让德函数表示的磁感应强度的解析解

首先，按照教材 * 的做法，先解矢势的方程，再求磁感应强度 B, 得到问题的解是

$$B_r(r,\theta)=-\frac{\mu_0 I}{2r}\sum_{l=0}^{\infty}\left(\frac{r}{a}\right)^{2l+1}P_{2l+1}^1(0)P_{2l+1}^1(\cos\theta), \qquad (r<a)$$

$$B_r(r,\theta)=-\frac{\mu_0 I}{2r}\sum_{l=0}^{\infty}\left(\frac{a}{r}\right)^{2l+2}P_{2l+1}^1(0)P_{2l+1}^1(\cos\theta), \qquad (r>a)$$

$$B_\theta(r,\theta)=-\frac{\mu_0 I}{2r}\sum_{l=0}^{\infty}\frac{1}{2l+1}\left(\frac{r}{a}\right)^{2l+1}P_{2l+1}^1(0)P_{2l+1}^1(\cos\theta), \qquad (r<a)$$

$$B_\theta(r,\theta)=\frac{\mu_0 I}{2r}\sum_{l=0}^{\infty}\frac{1}{2l+2}\left(\frac{a}{r}\right)^{2l+2}P_{2l+1}^1(0)P_{2l+1}^1(\cos\theta), \qquad (r>a)$$

下面的程序可以将解可视化，编程思路为，我们希望在 xOz 面上表现解的结果，首先建立一个以直角坐标表示的数据网格. 因为解是以球坐标表示的，所以要变换成对应的球坐标. 考虑到这里只有两个元素 r,θ, 所以没有使用球坐标变换的指令 cart2pol, 而是使用极坐标变换的指令 cart2pol, 但是根据对应关系改变了直角坐标变量的位置. 然后将 r 分成环内 rin 和环外 $rout$ 两部分.

在计算级数中的各项时，首先计算勒让德函数的值，然后分别计算每项中环内与环外的 B_r,B_θ 并用 for 循环累加起来. 级数的项数根据所要精度决定，这里是选取了 22 项. 最后还要将环内外的场合并成一个场. 为了画出场线即磁感应强度线，还要将结果再变换成直角坐标下的表示.

上面的计算只是对 xOz 而上第一象限作了计算，利用对称性，把其他的象限再画出来. 实际程序如下:

```
a=0.35;
[X,Y]=meshgrid(0:0.1:1);
[theta,r]=cart2pol(Y,X);   %%与极轴夹角，注意角的计算不是从 X 轴起
rout=r;  rin=r;
rin(find(rin>a))=NaN;   %%逻辑判断语句
rout(find(rout<a))=NaN;
Boutr=0;  Binr=0;  BoutQ=0;  BinQ=0;
for k=1:2:43
   fun10=legendre(k,0) ;
   fun0=fun10(2);
```

*见参考文献 [3], 246 页.

```
    fun=legendre(k,cos(theta));
    rfun=squeeze(fun(1,:,:));   %%将三维压缩成二维
    Qfun=squeeze(fun(2,:,:));
%-------------------------------------------------------
    Cr=1/a^k*(rin).^(k-1);   %%球内 r 分量
    Bkinr =- 1/2*fun0*Cr.*rfun;   %%第 K 项
    Binr=Binr + Bkinr;   %%总和
    CQ=1/k /a^k*rin.^(k-1);   %%球内 theta 分量
    BkinQ= -1/2*fun0*CQ.*Qfun;
    BinQ=BinQ + BkinQ;
%-------------------------------------------
    Dr=a^(k+1) ./rout.^(k+2);   %%球外 r 分量
    Bkoutr =- 1/2*fun0*Dr.*rfun;
    Boutr=Boutr + Bkoutr;
    DQ=1/(k+1)  *a^(k+1)./rout.^(k+2);   %%球外 theta 分量
    BkoutQ = 1/2*fun0*DQ.*Qfun;
    BoutQ=BoutQ + BkoutQ;
end

L1=isnan(Binr);   %%将非数元素找出并令其为零
Binr(find(L1==1))=0;    BinQ(find(L1==1))=0;
L2=isnan(Boutr);
Boutr(find(L2==1))=0;    BoutQ(find(L2==1))=0;
Br=Binr+Boutr;    BQ=BinQ+BoutQ;   %%合并为一个场 B
By=Br.*cos(theta)-BQ.*sin(theta);   %%换成直角坐标以作流线图
Bx=Br.*sin(theta)+BQ.*cos(theta);
vy=0;    vx=[0:0.01:0.35];
[Vx, Vy]=meshgrid(vx,vy);
cla
streamline(X,Y,Bx,By,Vx,Vy)
hold on
streamline(-X,Y,-Bx,By,-Vx,Vy)   %%利用对称性画其余的象限
streamline(-X,-Y,-Bx,-By,-Vx,-Vy)
streamline(X,-Y,Bx,-By,Vx,-Vy)
title('B 的场线 ')
```

```
figure
subplot(2,2,1)
surf(X,Y,Br)
title('Br 分量 ')
view(-130,20)
subplot(2,2,2)
contour(X,Y,Br,10)
title('Br 的等值线 ')
subplot(2,2,3)
surf(X,Y,BQ)
title('Bθ 分量 ')
subplot(2,2,4)
contour(X,Y,BQ,10)
title('Bthetа 的等值线 ')
```

所得的图形如图 4.22 所示。

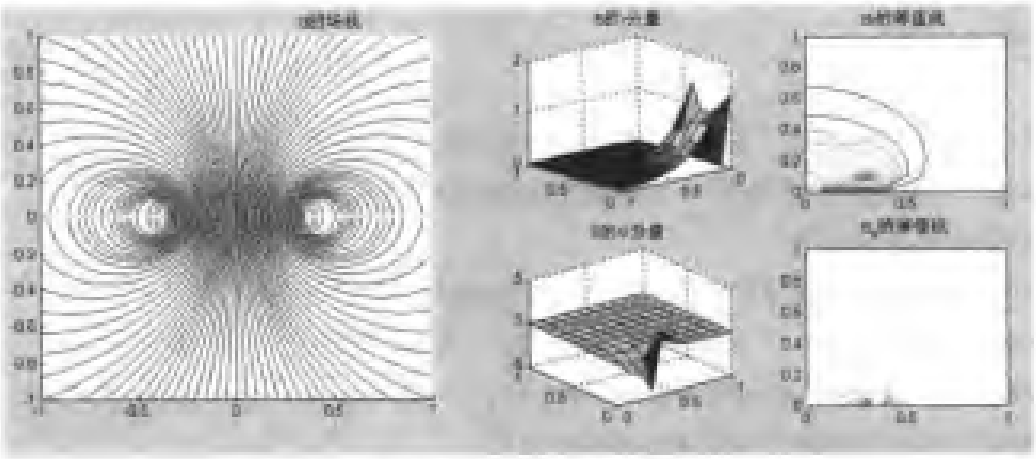


图 4.22 用勒让德函数表示的解析解的图形

2. 利用椭圆函数表示的磁感应强度解析表达式

利用毕奥 - 萨伐尔定律

$$\boldsymbol{B}=\frac{\mu_0}{4\pi}\oint\frac{I\mathrm{d}\boldsymbol{l}\times\boldsymbol{R}}{R^3}$$

求出圆环电流在空间的磁感应强度解析表达式，如图 4.23 建立坐标系，

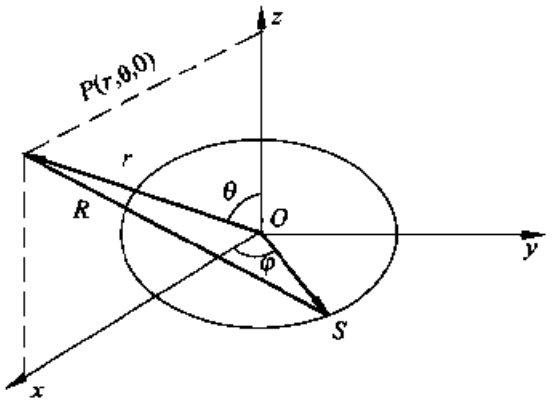


图 4.23 环形电流的示意图

积分计算得 *

$$B_x = \frac{\mu_0 I}{4\pi} \frac{2\cos\theta}{\sin\theta\sqrt{a^2+r^2+2ar\sin\theta}} \left[\frac{a^2+r^2}{a^2+r^2-2ar\sin\theta}E - K\right]$$

$$B_y = 0$$

$$B_z = \frac{\mu_0 I}{4\pi} \frac{2}{\sqrt{a^2+r^2+2ar\sin\theta}} \left[\frac{a^2-r^2}{a^2+r^2-2ar\sin\theta}E + K\right]$$

其中， E, K 椭圆函数，

$$K = \int_0^{\pi/2} \frac{\mathrm{d}x}{\sqrt{1-k^2\sin^2 x}} = \frac{\pi}{2}\left[1+\left(\frac{1}{2}\right)^2 k^2 + \left(\frac{1\cdot 3}{2\cdot 4}\right)^2 k^4 + \left(\frac{1\cdot 3\cdot 5}{2\cdot 4\cdot 6}\right)^2 k^6 + \cdots\right]$$

$$E = \int_0^{\pi/2} \sqrt{1-k^2\sin^2 x}\,\mathrm{d}x = \frac{\pi}{2}\left[1-\left(\frac{1}{2}\right)^2 k^2 + \left(\frac{1\cdot 3}{2\cdot 4}\right)^2 \frac{k^4}{3} - \left(\frac{1\cdot 3\cdot 5}{2\cdot 4\cdot 6}\right)^2 \frac{k^6}{5} + \cdots\right]$$

$$k^2 = \frac{4ar\sin\theta}{a^2+r^2+2ar\sin\theta}$$

用下面的程序可画出上述解的图形，在程序中我们使用了 MATLAB 计算椭圆函数的指令 Ellipke 。

```
a=0.35;
[X,Y]= meshgrid(0:0.1:1);
XY=X.^2+Y.^2;
k2=4*a*X./(a^2+XY+2*a*X);
[K,E]=ellipke(k2);
```

*曾令宏，张之翔. 圆环电流的磁场以及两共轴圆环电流之间的相互作用力. 大学物理，2002,21(9),14~16

```
aXY=a^2+XY-2*a*X;
aXY=aXY+eps*(aXY==0);
aE1=(a^2-XY)./aXY.*E;
aE2=(a^2+XY)./aXY.*E;
By=1/pi./sqrt(a^2+XY+2*a*X).*(aE1+K);
Bx=1/pi*Y./(X+eps*(X==0))./sqrt(a^2+XY+2*a*X).*(aE2-K);
vy=0;    vx=[0.0001:0.01:0.35];
[Vx, Vy]=meshgrid(vx,vy);

streamline(X,Y,Bx,By,Vx,Vy)
hold on
streamline(-X,Y,-Bx,By,-Vx,Vy)
streamline(-X,-Y,-Bx,-By,-Vx,-Vy)
streamline(X,-Y,Bx,-By,Vx,-Vy)
title('B 的场线 ')
figure
subplot(2,2,1)
surf(X,Y,Bx)
title('B 的 x 分量 ')
view(-130,20)
subplot(2,2,2)
contour(X,Y,Bx,10)
title('Bx 的等值线 ')
subplot(2,2,3)
surf(X,Y,By)
title('B 的 y 分量 ')
subplot(2,2,4)
contour(X,Y,By,10)
title('By 的等值线 ')
```

所得的图形如图 4.24 所示.

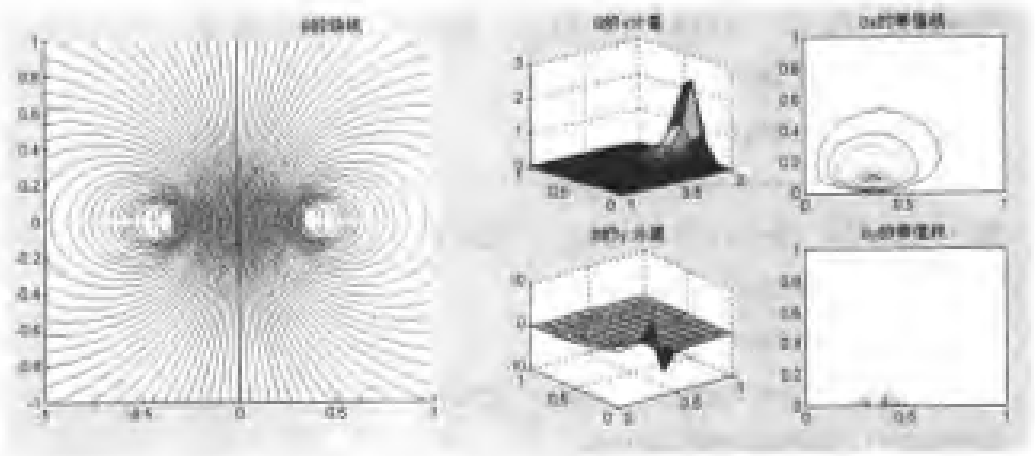


图 4.24 用椭圆函数表示的解析解的图形

还可以将结果做成三维动画，程序中先用类似于前面程序的方法，画出三维的磁感应线和相应的等值线，然后作动画，磁感应线的旋转是利用改变相机视角的方法，指令是 camorbit，而等值线的动画，则是用实时动画的方式做出的。

```
a=0.35;
[X,Y,Z]=meshgrid(-1:0.1:1, -1:0.1:1, 0:0.1:1);
rho2=X.^2+Y.^2; rho=sqrt(rho2);
theta=atan2(rho,Z);
r=sqrt(rho2+Z.^2);
rout=r;    rin=r;
rin(find(rin>a))=NaN;
rout(find(rout<a))=NaN;
Boutr=0;    Binr=0;    BoutQ=0;    BinQ=0;
for k=1:2:61
fun10=legendre(k,0) ;
   fun0=fun10(2);
   fun=legendre(k,cos(theta));
   rfun=squeeze(fun(1,:,:,:));
   Qfun=squeeze(fun(2,:,:,:));
%-------------------------------------------------------------
   Cr=1/a^k*(rin).^(k-1);   %%球内 r 分量
   Bkinr = - 1/2*fun0*Cr.*rfun;
   Binr=Binr + Bkinr;
```

```
    CQ=1/k  /a^k   *rin.^(k-1);   %%球内 theta 分量
    BkinQ=-1/2*fun0*CQ.*Qfun;
    BinQ=BinQ + BkinQ;
%-----------------------------------------
    Dr=a^(k+1) ./rout.^(k+2);   %%球外 r 分量
    Bkoutr=-1/2*fun0*Dr.*rfun;
    Boutr=Boutr+Bkoutr;
    DQ=1/(k+1)  *a^(k+1)./rout.^(k+2);   %%球外 theta 分量
    BkoutQ=1/2*fun0*DQ.*Qfun;
    BoutQ=BoutQ + BkoutQ;
end

L1=isnan(Binr);   %%合并为一个场 B
Binr(find(L1==1))=0;    BinQ(find(L1==1))=0;
L2=isnan(Boutr);
Boutr(find(L2==1))=0;    BoutQ(find(L2==1))=0;
Br=Binr+Boutr;    BQ=BinQ+BoutQ;
Bz=Br.*cos(theta)-BQ.*sin(theta);
Bxy=Br.*sin(theta)+BQ.*cos(theta);
phi=atan2(Y,X);
Bx=Bxy.*cos(phi);
By=Bxy.*sin(phi);
vy=0;    vx=[-0.3:0.1:0.3];
[Vx, Vy, Vz]=meshgrid(vx,vx,0);


cla
streamline(X,Y,Z,Bx,By,Bz,Vx,Vy,Vz)
view(-8,38)
hold on

B=sqrt(Bx.^2+By.^2+Bz.^2);
Zmin=min(Z(:));
hcont = contourslice(X,Y,Z,B,[ ],[ ],0.1);   %%等值线
```

```
set(hcont,'EdgeColor','red','LineWidth',.5);


axis vis3d
for i=1:20
   camorbit(10,0,'data',[0,0,1])   %%B 线的动画
   drawnow
   pause(0.5)
end

for k=1:20
   delete(hcont)
   hcont=contouralice(X,Y,Z,B,[ ],[ ],0.1+0.05*k);   %%等值线的动画
   pause(0.5)
end
```

图 4.25 是其中的一个画面。

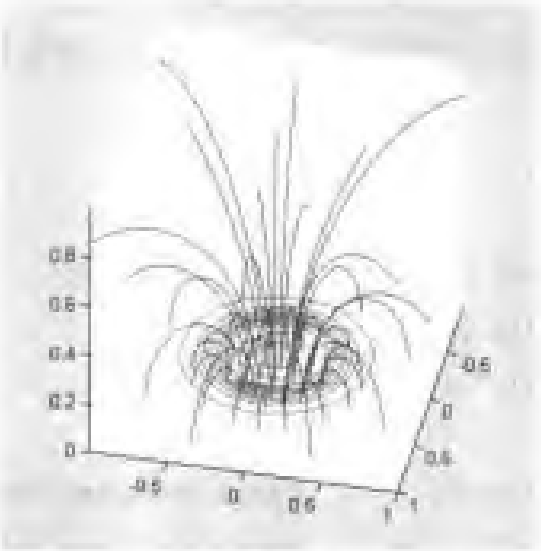


图 4.25 电流环磁感应线动画的一个画面

3. 用数值积分求磁感应强度

作为对比，我们直接用毕奥 —— 萨伐尔定律进行数值积分来计算细圆环电流在空间产生的磁场，如图 4.26 建立坐标。

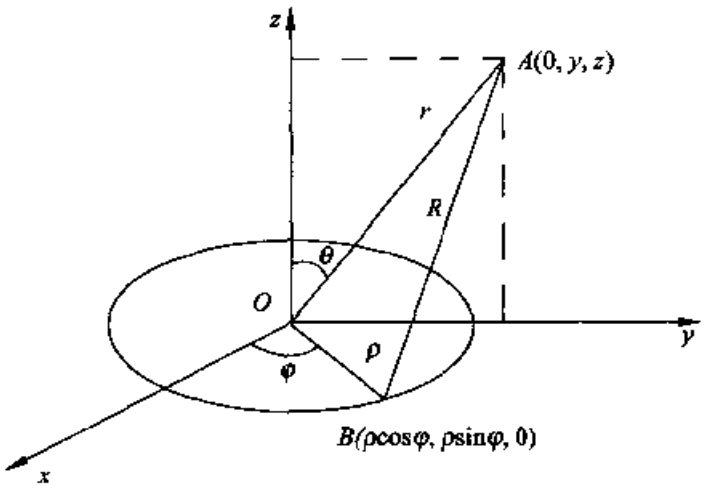


图 4.26 带有电流的细圆环

细圆环的电流密度是 λ, 则圆环电流在空间任意一点 $Q(x,y,z)$ 所产生的磁场是

$$v(x,y,z)=\frac{1}{4\pi\varepsilon_0}\int_0^{2\pi}\frac{\lambda\rho\mathrm{d}\varphi}{\sqrt{(x-\rho\cos\varphi)^2+(y-\rho\sin\varphi)^2+z^2}}$$

取上式中的 $x=0$, 得到在圆环电流在 yOz 面上点 $A(0,y,z)$ 产生的磁场. 利用这个式子, 可以编出下面的程序, 程序中省略了一些常数.

```
clear;
figure(1)
a=0.35;    y= -1:0.04:1;    the=0:pi/20:2*pi;
[Y,Z,T]=meshgrid(y,y,the);
r=sqrt((a*cos(T)).^2+Z.^2+(Y-a*sin(T)).^2);
r3=r.^3;
dby =a*Z.*sin(T)./r3;          by=pi/40*trapz(dby,3);
dbz =a*(a-Y.*sin(T))./r3;      bz=pi/40*trapz(dbz,3);

subplot(1,2,1)   %%二维图形
[bSY,bSZ]=meshgrid([0:0.05:0.2],0);
h1=streamline(Y(:,:,1),Z(:,:,1),by,bz,bSY,bSZ,[0.1,1000]);
h2=copyobj(h1,gca);   %%用图形复制与旋转画其余的象限, 代替下面三句
rotate(h2,[1,0,0],180,[0 0 0]);
```

```
h3=copyobj(allchild(gca),gca);
rotate(h3,[0,1,0],180,[0 0 0]);
title(' 磁场的二维图 ','fontsize',15);

for kk=1:4   %%曲率较大的场线要取不同的步长
   [bSY,bSZ]=meshgrid(0.2+kk*0.02,0);
   streamline(Y(:,:,1),Z(:,:,1),by,bz,bSY,bSZ,[0.02/(kk+1),4500]);
   streamline(-Y(:,:,1),Z(:,:,1),-by,bz,-bSY,bSZ,[0.02/(kk+1),4500]);
end


[X,Y,Z]=meshgrid(-0.5:0.04:0.5);   %%画三维图形
r2=X.^2+Y.^2+Z.^2;
for k=1:81
   phi=pi/40*(k-1);    costh=cos(phi);    sinth=sin(phi);
   R3=(r2+a^2-2*a*(X*costh+Y*sinth)).^(3/2);
   Bx0(:,:,:,k)=a*Z*costh./R3;
   By0(:,:,:,k)=a*Z*sinth./R3;
   Bz0(:,:,:,k)=a*(a-X*costh-Y*sinth)./R3;
end
Bx=pi/40*trapz(Bx0,4);   %%梯形积分
By=pi/40*trapz(By0,4);
Bz=pi/40*trapz(Bz0,4);

subplot(1,2,2)
v=[-0.2,-0.1,0,0.1,0.2];
[Vx, Vy, Vz]=meshgrid(v,v,0);
plot3(Vx(:),Vy(:),Vz(:),'r*')
streamline(X,Y,Z,Bx,By,Bz,Vx,Vy,Vz,[0.01,2000]);
hold on;
axis([-0.5,0.5,-0.5,0.5,-0.5,0.5]);
view(-23,26);
```

```
box on;
title(' 磁场的三维图 ','fontsize',15);
t=0:pi/100:2*pi;
plot(a*exp(i*t),'r-','LineWidth',3);
hold off;
pause(1)   %%从不同角度观察
view(-44,-8)
pause(1)
view(-44,90)
```

所得到的二维与三维的磁力线图形分别如图 4.27 所示，注意对照图形与程序，学习如何画出闭合的磁力线。程序中使用的指令 streamline 的作图精度与所选的步长有关，在语句中是最后两个参数，一个表示步长，另一个表示步数。适当选择这两个参数的值，才能保证所画的场线闭合，否则可能产生场线不闭合的现象。

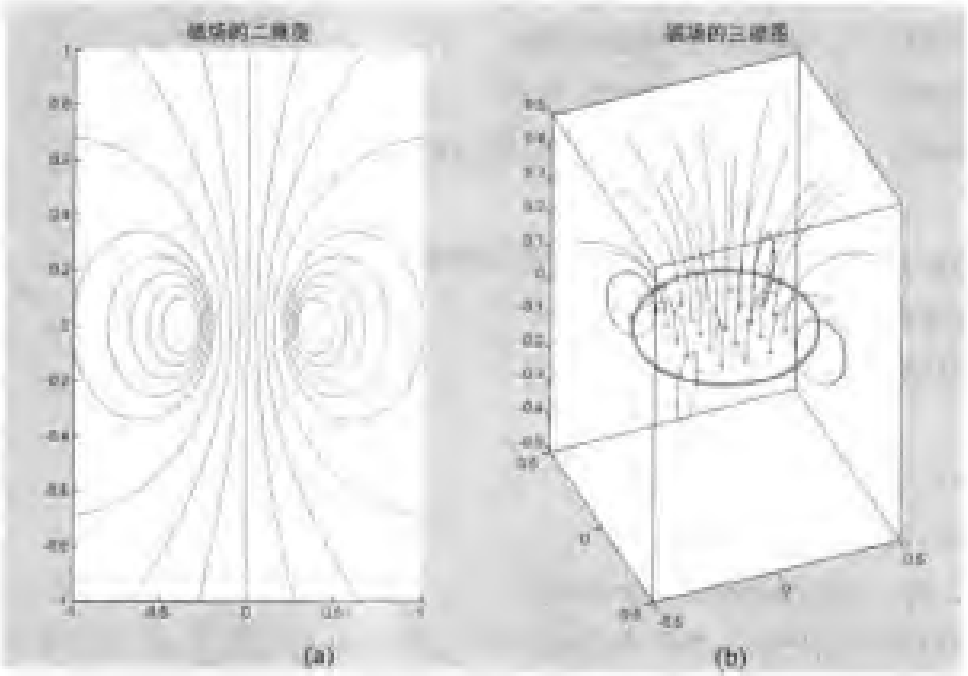


图 4.27　二维及三维磁力线图形

(a) 二维的磁力线图形；(b) 三维的磁力线图形

4.2.5 柱体内温度场分布之一 (J_ν 的应用)

半径为 b 高为 h 的圆柱体，其侧面绝热，下底温度保持零度，上底温度分布为 $u_0\rho^2$，求圆柱体内稳定的温度分布 * 。

定解问题是

$$\begin{cases} \Delta u = 0 \\ u_\rho(\rho = b, z) = 0 \\ u(\rho, z = 0) = 0, \quad u(\rho, z = h) = u_0\rho^2 \end{cases}$$

这个问题的解是

$$u(\rho, z) = \frac{u_0 b^2}{2h} z + \sum_{n=1}^{\infty} \frac{-4u_0 J_2(x_n^1)}{\left(\frac{x_n^1}{b}\right)^2} \frac{\sinh(\frac{x_n^1}{b} z)}{\sinh(\frac{x_n^1}{b} h)} \frac{J_0\left(\frac{x_n^1}{b}\rho\right)}{\left[J_0\left(x_n^1\right)\right]^2}$$

取 $b = 1.5, u0 = 1, h = 0.5$ 用以下的程序画出图形，如图 4.28 所示。注意在这里要计算贝塞尔函数的零点，计算的方法已经在 1.2.3 节介绍过。

```
LD=□;   N=50;  j=1;
yy=inline('besselj(1,xx)','xx');
for k=1:N
    while(besselj(1,j)*besselj(1,j+1)>0),    j=j+1;    end
    q=fzero(yy,j);
    j=j+1;
    LD=[LD,q] ;
    k=k+1;
end
r0=1.5;    u0=1;  L=0.5;
 [X,Y]=meshgrid(0:0.05:0.5,0:0.05:1.5);
A=0;
for n=1:N
    a=-4*u0*besselj(2,LD(n));
```

*见参考文献 [5], 304 页.

```
    a=a*besselj(0,LD(n)/r0.*Y)./LD(n)^2./besselj(0,LD(n))^2;
    c=sinh(LD(n).*X./r0)./sinh(LD(n)*L/r0).*a;
    A=A+c;
end
C=X./2./L*(u0*r0^2)+A;
figure
contour(C,20);
```

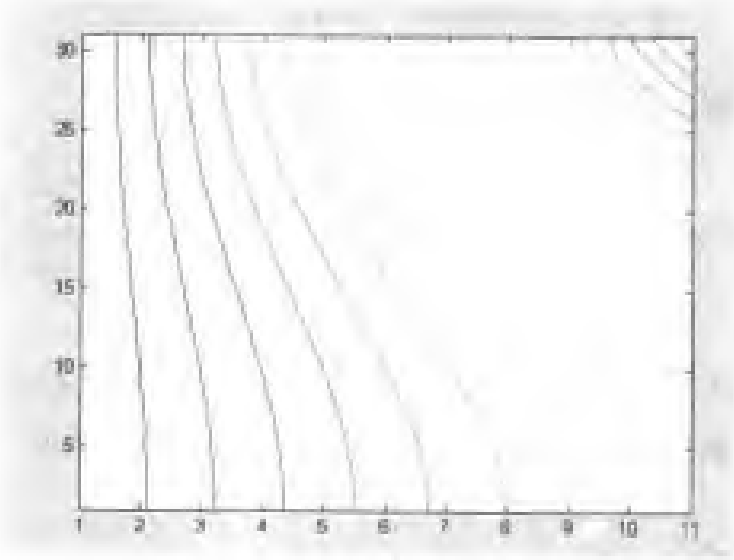


图 4.28 解析解的图形

直接用偏微分方程工具箱求解这个问题。在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 0.5$, y 轴范围为 $-1 \sim 1$。这个问题是轴对称的，与 φ 无关，以 x 代表 z, 以 y 代表 ρ, 画一个矩形表示高 0.8 半径 0.4 柱体的过柱轴的纵切面，矩形的顶点为 $(0,-0.4),(0.8,-0.4),(0.8,0.4),(0,0.4)$。

按照题意，柱体下底是齐次的狄里克利边界条件，在矩形中，代表柱体下底的边是 $x=0$, 可取 $h=1, r=0$, 代表柱体侧面的边是 $y=0.4$ 和 $y=-0.4$, 是齐次的诺伊曼边界条件，可取 $g=0, q=0$ 柱体的上底是非齐次的狄里克利边界条件，代表柱体上底的边是 $x=0.8$, 框图中可取 `h=1`，`r=y.^2`。

方程的设置是 Elliptic 型，也就是 $c=1, a=0, f=0$。

为了有足够的精度，将初始化的网格作了两次细分。画作图的选项为 Contour，所得的图形如图 4.29 所示。注意两个图形中 ρ 的取值范围不同，所以前一个图形只是下面图形的一半。

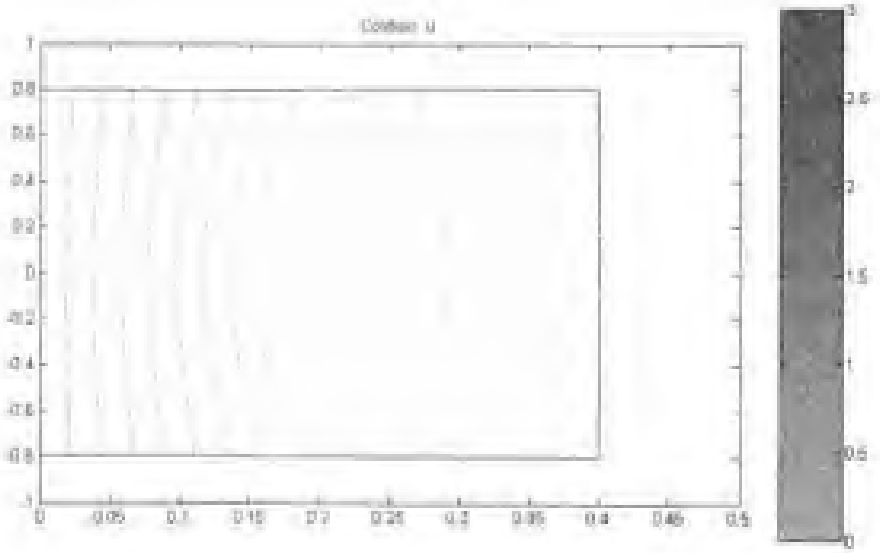


图 4.29 偏微分方程工具箱解得的结果

4.2.6 柱体内温度场分布之二 (J_ν 的应用)

匀质圆柱，半径 ρ_0，高 L，下底和侧面保持零度，上底温度分布为 $f(\rho)=\rho^2$，求解柱内稳定的温度分布*。

定解问题是

$$\begin{cases} \Delta u = 0 \\ u(\rho=\rho_0)=0, \quad u(\rho=0)\text{有限} \\ u(z=0)=0, \quad u(z=L)=\rho^2 \end{cases}$$

问题的解为

$$u = 2\rho_0^2 \sum_{n=1}^{\infty} \frac{1}{x_n^{(0)} J_1\left(x_n^{(0)}\right)} \left(1-\frac{4}{\left(x_n^{(0)}\right)^2}\right) \left(\frac{\sinh \frac{x_n^{(0)} z}{\rho_0}}{\sinh \frac{x_n^{(0)} L}{\rho_0}}\right) J_0\left(\frac{x_n^{(0)}}{\rho_0}\rho\right)$$

用下面的程序可以画出解析解的图形，程序中取 $\rho=0.2$，$L=0.8$，程序的第一段是计算零阶贝塞尔函数的前 N 个零点，计算中取 $N=50$，程序的第二段则是以等值线表示温度场和分布，为了使等值线分布均匀，指定了画等值线 Z 的值。

```
j=1;        a=[ ];    N=50;
```

*见参考文献 [1]，346 页。

```
yy=inline('besselj(0,xx)','xx');
for k=1:N
    while(besselj(0,j)*besselj(0,j+1)>0),    j=j+1;    end
    q=fzero(yy,j);
    j=j+1;
    a=[a,q]
    k=k+1;
end

r0=0.2;  L=0.8
x=0:0.01:L;
y=-r0:0.01:r0;
[X,Y]=meshgrid(x,y);
Z=0;
for n=1:N
    ZZ=2*r0^2/a(n)/besselj(1,a(n))*(1-4./a(n).^2)*sinh(a(n).*X./...
    r0)./sinh(a(n)*L/r0).*besselj(0,a(n)/r0.*Y);
    Z=Z+ZZ;
end
contour(x,y,Z,[0.00001:0.0003:0.001,0.0015,0.002:0.001:0.04]);
```

所得的图形如图 4.30 所示。

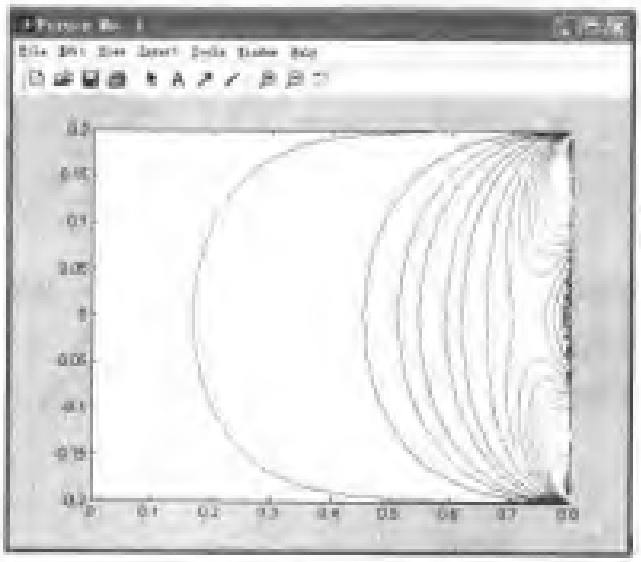


图 4.30　解析解的图形

再用偏微分方程工具箱求解这个问题。在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 1$, y 轴范围为 $-1 \sim 1$。这个问题是轴对称的，与 φ 无关。以 x 代表 z, 以 y 代表 ρ, 画一个矩形表示高 0.8 半径 0.4 柱体的纵切面，矩形的顶点为 $(0,-0.4),(0.8,-0.4),(0.8,0.4),(0,0.4)$。

按照题意，柱体下底侧面是齐次的狄里克利边界条件，在矩形中，代表柱体下底的边是 $x=0$, 代表柱体侧面的边是 $y=0.4$ 和 $y=-0.4$ 可取 $h=1, r=0$, 柱体的上底是齐次的狄里克利边界条件，代表柱体上底的边是 $x=0.8$, 可取 $h=1, r=\rho^2$。

方程的设置是 Elliptic 型，也就是 $c=1, a=0, f=0$。

为了有足够的精度，将初始化的网格作了两次细分。而作图的选项为 Color 和 contour, 同时在 colormap 中选择 prism。所得的图形如图 4.31 所示。

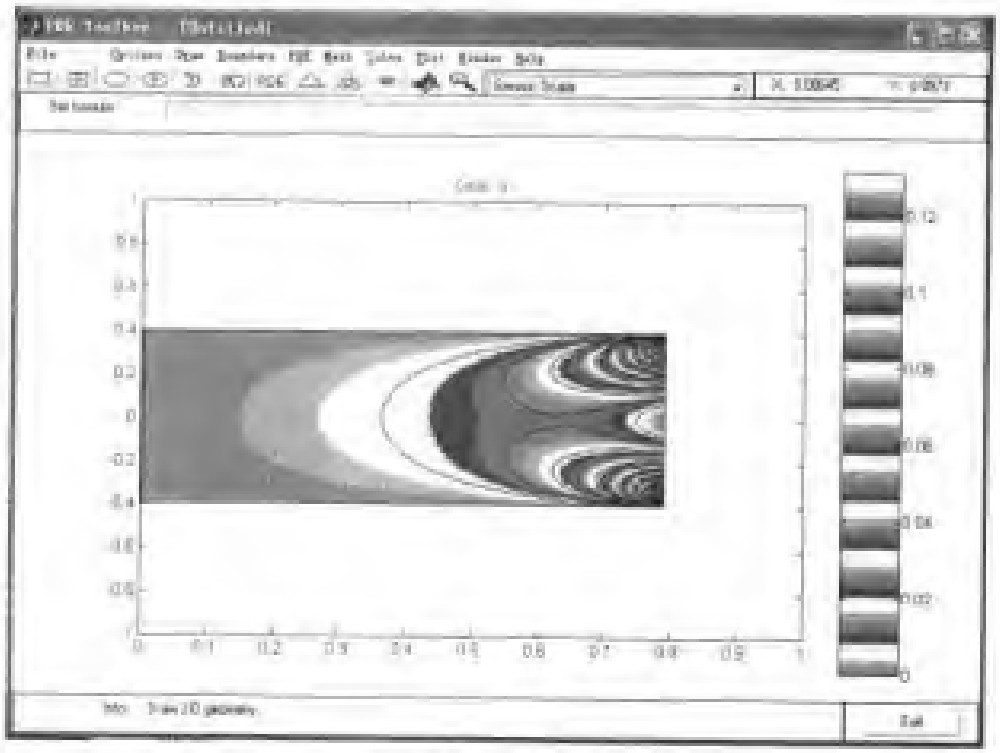


图 4.31 偏微分方程工具箱解得的结果

4.2.7 柱体内温度场分布之三 (I_0 的应用)

匀质圆柱，半径 ρ_0, 高 L, 柱侧有均匀分布的热流进入，其强度为 q_0。圆柱上下两底保持为恒定的温度 u_0。求解柱内稳定的温度分布*。

*见参考文献 [1], 357 页。

令 $u=u_0+v$, 则 v 满足上下底面齐次边界条件的定解问题

$$\begin{cases} \Delta v=0 \\ kv_\rho(\rho=\rho_0)=q_0, \quad v(\rho=0)\text{有限} \\ v(z=0)=0, \qquad v(z=L)=0 \end{cases}$$

的解为

$$u=u_0+\frac{4Lq_0}{k\pi^2}\sum_{l=0}^{\infty}\frac{1}{(2l+1)^2}\frac{I_0\left(\dfrac{(2l+1)\pi\rho}{L}\right)}{I_0'\left(\dfrac{(2l+1)\pi\rho_0}{L}\right)}\sin\frac{(2l+1)\pi z}{L}$$

用下面的程序可以画出解析解的图形, 程序中取 $\rho=0.6$, $L=2.5$, $q_0/k=1$。

```
L=2.5;    y0=0.6;
N=7;      q0k=1;
x=0:.01:2.5;    y=-0.6:.05:0.6;
[XX,YY]=meshgrid(x,y);
uu=0;
for p=1:2:N
    a=4*L/p^3/pi^2;
    b=p*pi/L;
    I1=besseli(0,b*(y0-0.05))-besseli(0,b*y0);
    I1=I1/0.01;
    II=besseli(0,b*YY)./I1;
    vv=a*II.*sin(b.*XX);
    uu=vv+uu;
end
contourf(x,y,uu,30)
colormap(prism)
```

在程序中, 侧面边界的导数是用差分作近似计算, 改变 N 值可以得到不同的求和项数. 所得的图形如图 4.32 所示。

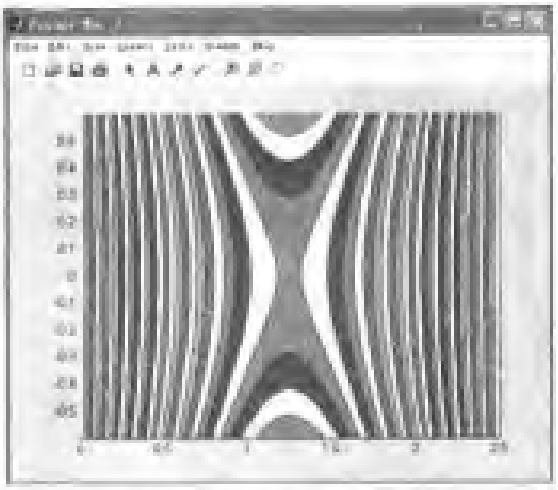


图 4.32 解析解的图形

再用偏微分方程工具箱求解这个问题. 在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 2.5$, y 轴范围为 $-1 \sim 1$. 这个问题是轴对称的, 与 φ 无关, 以 x 代表 z, 以 y 代表 ρ, 画一个矩形表示柱体的纵切面. 矩形的顶点为 $(0, -0.6)$, $(2.5, -0.6)$, $(2.5, 0.6)$, $(0, 0.6)$.

按照题意, 柱体上下底是狄里克利边界条件, 在矩形中, 代表柱体上下底的边是 $x = 0$, $x = 2.5$, 可取 $h = 1, r = 0$, 柱体的侧面是诺伊曼边界条件, 代表柱体侧面的边是 $y = -0.6, y = 0.6$, 可取 $g = 1$, $q = 0$.

方程的设置是 Elliptic 型, 也就是 $c = 1, a = 0, f = 0$.

为了有足够的精度, 将初始化的网格作了两次细分. 而作图的选项为 Color 和 contour, 同时在 colormap 中选择 prism . 所得的图形如图 4.33 所示.

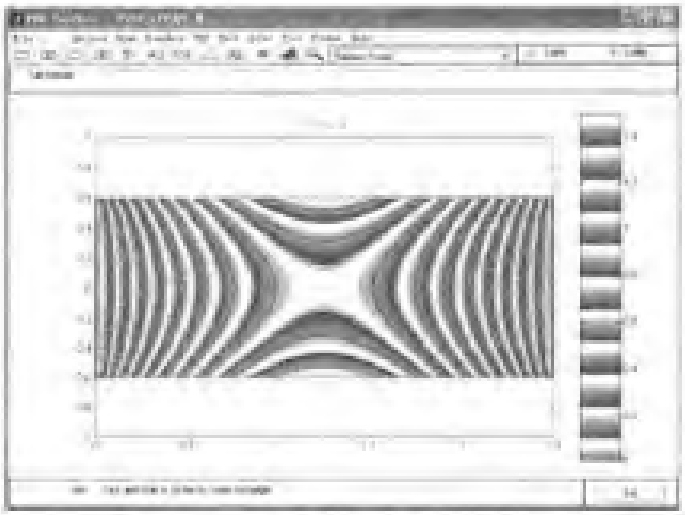


图 4.33 偏微分方程工具箱解得的结果

从图形可以看出，等温线是柱体侧面正交的，表面热流是从柱体侧面流入。

4.2.8 圆柱内的温度场 (I_0 的应用)

匀质圆柱半径 ρ_0 高 L . 下底和上底的温度分别为 u_1 和 u_2, 侧面的温度为 $f(z)=\frac{2u_2}{L^2}\left(z-\frac{L}{2}\right)+\frac{u_1}{L}(L-z)$ 求解柱体内稳恒的温度场 * .

定解问题是

$$\begin{cases} \Delta u=0 \\ u(\rho=\rho_0)=f(z), \quad u(\rho=0)\text{有限} \\ u(z=0)=u_1, \qquad u(z=L)=u_2 \end{cases}$$

其中

$$f(z)=\frac{2u_2 z}{L^2}\left(z+\frac{L}{2}\right)+\frac{u_1}{L}(L-z)$$

令 $u=v+\psi$, 其中 $\psi=\frac{u_2-u_1}{L}z+u_1$, 可将上下底的边界条件齐次化，得到 u 的解是

$$u=u_1+\frac{u_2-u_1}{L}z-\sum_{p=0}^{\infty}\frac{16u_2}{(2p+1)^3\pi^3}\frac{I_0\left(\frac{(2p+1)\pi}{L}\rho\right)}{I_0\left(\frac{(2p+1)\pi}{L}\rho_0\right)}\sin\frac{(2p+1)\pi}{L}z$$

用下面的程序可以画出 v 的图像，程序中取 $\rho=0.6$, $L=2.5$, $u_1=-5$, $u_2=30$

```
u1=-5; u2=30; L=2.5; y0=0.6;
N=11;
x=0:.005:2.5;  y=-0.6:.01:0.6;
[XX,YY]=meshgrid(x,y);
u0=u1+(u2-u1)/L.*XX;
uu=0;
for p=1:2:N
    a=16*u2/p^3/pi^3;
    b=p*pi/L;
    II=besseli(0,b*YY)./besseli(0,b*y0);
    vv=a*II.*sin(b.*XX);
    uu=vv+uu;
```

*见参考文献 [1], 361 页.

```
end
UU=u0-uu;
contourf(x,y,UU,40)
colormap(prism)
```

所得的图形如图 4.34 所示.

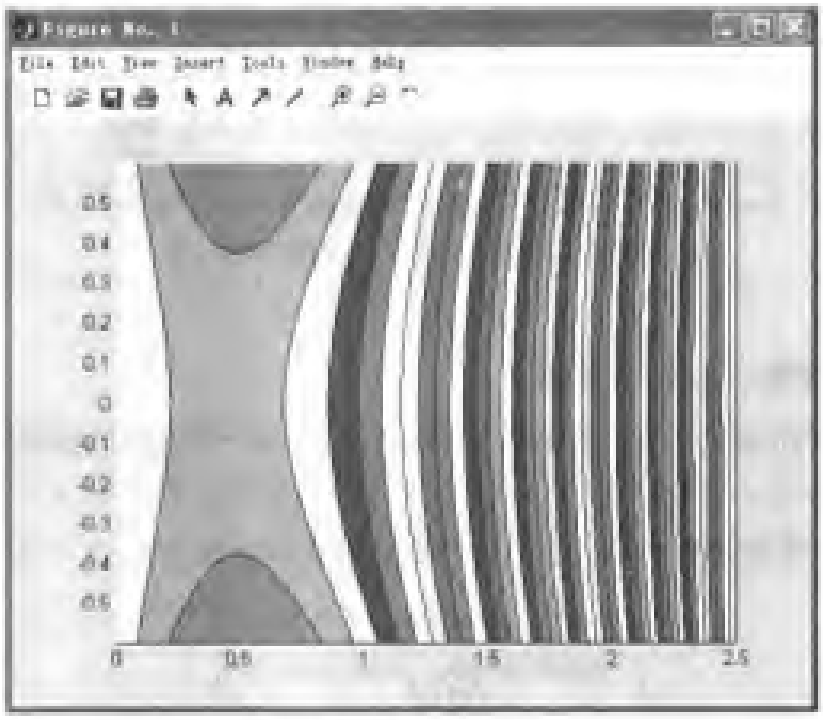


图 4.34 解析解的图形

再用偏微分方程工具箱求解这个问题. 在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 2.5$, y 轴范围为 $-1 \sim 1$. 这个问题是轴对称的, 与 φ 无关, 以 x 代表 z, 以 y 代表 ρ, 画一个矩形表示柱体的纵切面, 矩形的顶点为 $(0, -0.6)$, $(2.5, -0.6)$, $(2.5, 0.6)$, $(0, 0.6)$.

按照题意, 柱体 4 个边界都是狄里克利边界条件, 在矩形中, 代表柱体上下底的边是 $x = 0$, $x = 2.5$, 是齐次的边界条件, 对于下底, 可取 $h = 1, r = -5$, 对于上底, 可取 $h = 1, r = 30$, 代表柱体侧面的边是 $y = -0.6, y = 0.6$, 可取 $h = 1$, $r = 2 * 30/2.5^2 * x. * (x - 2.5/2) - 5/2.5 * (2.5 - x)$.

方程的设置是 Elliptic 型, 也就是 $c = 1, a = 0, f = 0$.

为了有足够的精度, 将初始化的网格作了两次细分. 而作图的选项为 Color 和 contour, 同时在 colormap 中选择 prism. 所得的图形如图 4.35 所示.

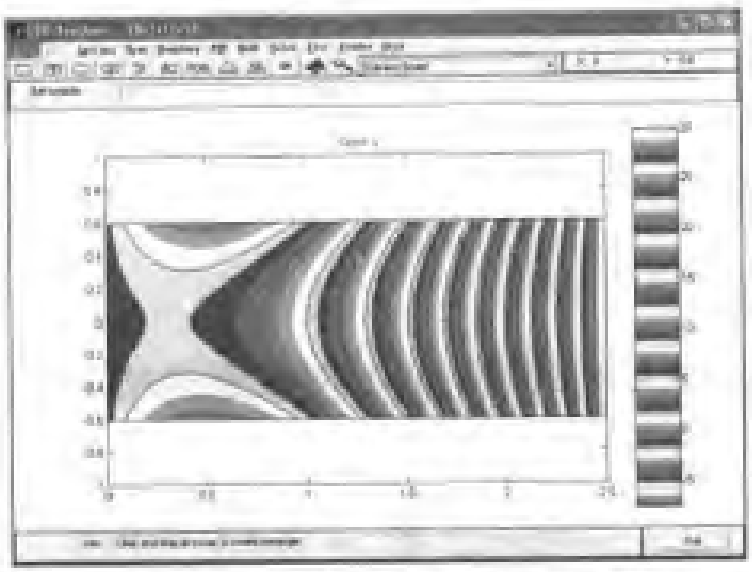


图 4.35 偏微分方程工具箱解得的结果

4.2.9 电子透镜 (J_0, I_0 的应用)

如图 4.36 所示，电子光学透镜的某一部件由两个中空圆柱筒组成，其电势分别为 $+Q_0$ 和 $-Q_0$，在圆柱中间缝隙的边缘处电势可近似表为 $u = Q_0 \sin \dfrac{\pi z}{2\delta}$，求圆柱筒内的电势分布. 圆柱两端边界条件可近似表为 $u|_{z=\pm l} = \pm Q_0$. 圆柱筒的半径为 ρ_0 *.

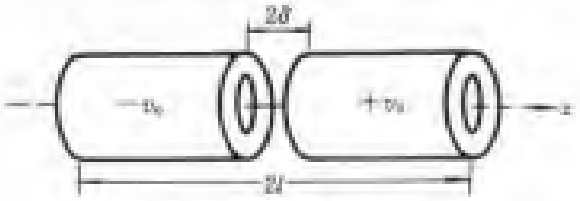


图 4.36 圆柱筒内的电势

取柱坐标系，由于电势为 z 的奇函数，可取通过缝隙中间的平面为柱坐标系底面，这样可在 $[0, l]$ 上求解. 定解问题是

$$\begin{cases} \Delta u = 0 \\ u(z=0) = Q_0 \sin \dfrac{\pi z}{2\delta}\Big|_{z=0} = 0, \quad u(z=l) = Q_0 \\ u(\rho = R) = \begin{cases} Q_0 \sin \dfrac{\pi z}{2\delta} & (0 \leqslant z \leqslant \delta) \\ Q_0 & (\delta \leqslant z \leqslant l) \end{cases} \end{cases}$$

*见参考文献 [1], 361 页.

令 $u=v+w$ 将它分为两个定解问题.

$$\begin{cases} \Delta v=0 \\ v(z=0)=0, \quad v(z=l)=0 \\ v(\rho=R)=\begin{cases} Q_0\sin\dfrac{\pi z}{2\delta} & (0\leqslant z\leqslant\delta) \\ Q_0 & (\delta\leqslant z\leqslant l) \end{cases} \end{cases} \qquad \begin{cases} \Delta w=0 \\ w(z=0)=0, \quad w(z=l)=Q_0 \\ w(\rho=R)=0 \end{cases}$$

它们的解分别是

$$v=\frac{2v_0}{\pi}\sum_{p=1}^{\infty}\frac{1}{p}\left[(-1)^{p+1}+\frac{l^2}{l^2-4p^2\delta^2}\cos\frac{p\pi\delta}{l}\right]\frac{I_0(p\pi\rho/l)}{I_0(p\pi\rho_0/l)}\sin\frac{p\pi}{l}z$$

和

$$w=2v_0\sum_{n=1}^{\infty}\frac{1}{x_n^{(0)}J_1(x_n^{(0)})}\frac{\sinh(x_n^{(0)}z/\rho_0)}{\sinh(x_n^{(0)}l/\rho_0)}J_0\left(x_n^{(0)}\rho\right)$$

下面的程序画出了解析解的图形.

```
j=1;      LD=[ ];   N=50;  v0=100;
yy=inline('besselj(0,xx)','xx');
for k=1:N
    while(besselj(0,j)*besselj(0,j+1)>0),    j=j+1;    end
    q=fzero(yy,j);
    j=j+1;
    LD=[LD,q] ;
    k=k+1;
end

 [X,Y]=meshgrid(-1.5:0.05:1.5,0:0.05:0.5);
A=0;    B=0;
for n=1:N
    a=besselj(0,LD(n)/0.5.*Y)/LD(n)/besselj(1,LD(n));
    c=sinh(LD(n).*X./0.5)./sinh(LD(n)*1.5/0.5).*a;
    A=A+c;
end
for m=1:N
    d=besseli(0,m.*pi.*Y/1.5)./besseli(0,m.*pi.*0.5/1.5);
    f=1/m*((-1).^(m+1)+(1.5)^2*cos(m*pi*0.05/1.5)./...
```

```
    (1.5^2-4*m^2*0.05^2)).*d.*sin(m*pi.*X./1.5);
    B=B+f;
end
C=2*v0*A+(2*v0/pi)*B;
surfc(X,Y,C);
figure(2)
contour(C,20);
```

所得的图形如图 4.37 所示.

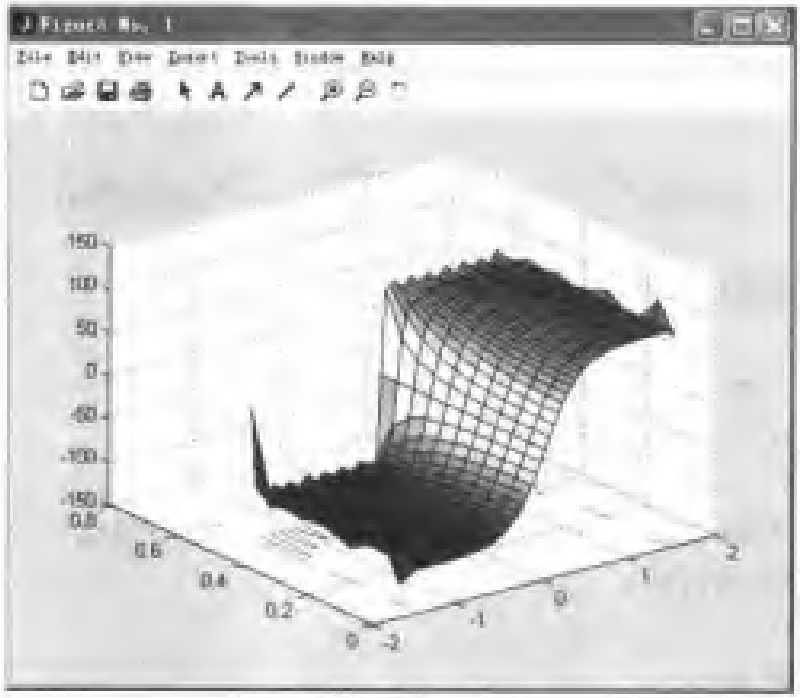


图 4.37 解析解的图形

再用偏微分方程工具箱求解这个问题. 在 Options/Axes Limits 下选择 x 轴范围为 $-2\sim2$, y 轴范围为 $-1\sim1$. 选用柱坐标, 则这个问题是轴对称的, 与 φ 无关, 以 x 代表 z, 以 y 代表 ρ, 画两个矩形如下图. 大的矩形 R1 表示电子透镜的区域, 小的矩形 R2 表示缝隙. 为了使矩形的位置比较准确, 可双击矩形, 打开矩形位置的对话框, 分别输入矩形的位置数据, 对 R1 是: Left 取 -0.1, Bottom 取 -0.5, Width 取 0.2, Height 取 1, 对 R2 是: Left 取 -1.5, Bottom 取 -0.5, Width 取 3, Height 取 1. 在 Set Formula 区域设置的公式是 R1+R2.

选择菜单 Boundary/Show Edge Labels, 给边界写上标号如图 4.38 所示.

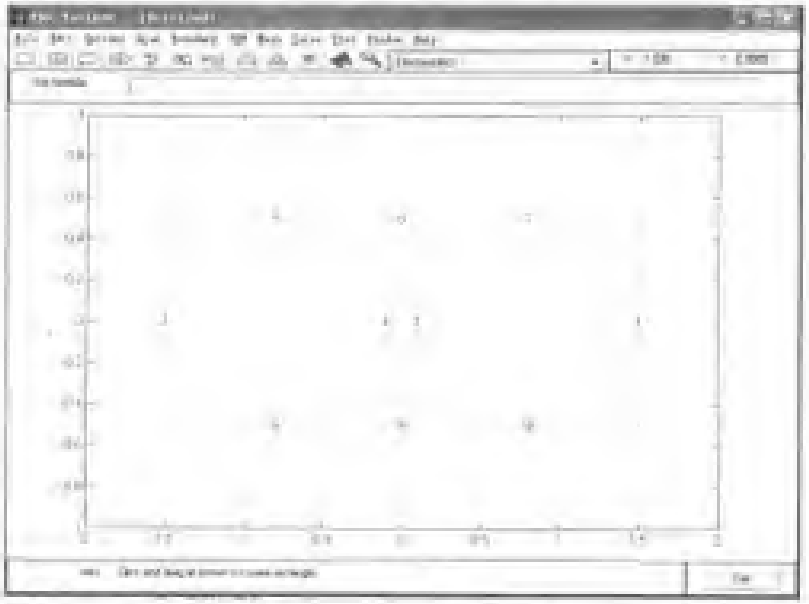


图 4.38 方程的求解区域

按照题意，边界 2,5,8 都是齐次狄里克利边界条件，取 $h=0, r=-100$, 边界 7,1,10 也是齐次狄里克利边界条件，取 $h=0, r=100$ 边界 6,9 都是非齐次狄里克利边界条件，取 b=0, r=100*sin(pi*x./0.2) ，边界 3,4 在求解区域的内部，不必设边界条件。

方程的设置是 Elliptic 型，也就是 $c=1, a=0, f=0$ 。

为了有足够的精度，将初始化的网格作了两次细分。而作图的选项为 Color 和 Contour ，所得的图形如图 4.39 所示。

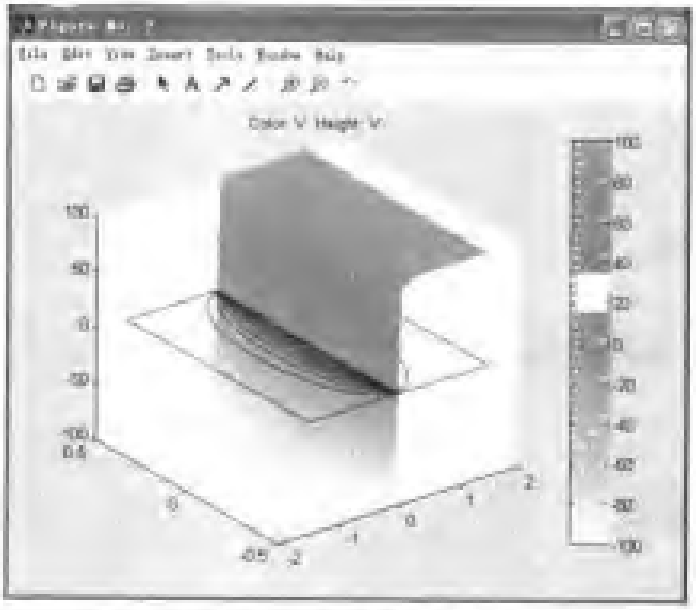


图 4.39 偏微分方程工具箱解得的结果

4.2.10 柱体外的电势 (K_0 的应用)

半径 ρ_0 高为 L 的导体圆柱壳，上下底和侧面被绝缘物质隔离开来，柱壳侧面的电势为 z，上下底接地。求柱体外的静电场的电势分布 *。

定解问题是

$$\begin{cases} \Delta u = 0 & (\rho > \rho_0) \\ u(\rho = \rho_0) = z \\ u(z = 0) = 0, \quad u(z = L) = 0 \end{cases}$$

方程的解析解如下：

$$u = \sum_{n=1}^{\infty} (-1)^n \frac{2L}{n\pi} \frac{K_0\left(n\pi\frac{\rho}{L}\right)}{K_0\left(n\pi\frac{\rho_0}{L}\right)} \sin\left(n\pi\frac{z}{L}\right)$$

以下是解析解的作图程序：

```
r0=0.5;       L=3;       A=0;
 [X,Y]=meshgrid(0:0.05:3,0.5:0.05:3);
for n=1:50
    a=(-1)^n*2*L/n/pi./besselk(0,n*pi*r0/L);
    a=a*besselk(0,n*pi*Y/L).*sin(n*pi*X./L);
    A=A+a;
end
A=L*A;
figure
contour(A,20);
```

解析解图如图 4.40 所示。

*见参考文献 [1], 360 页.

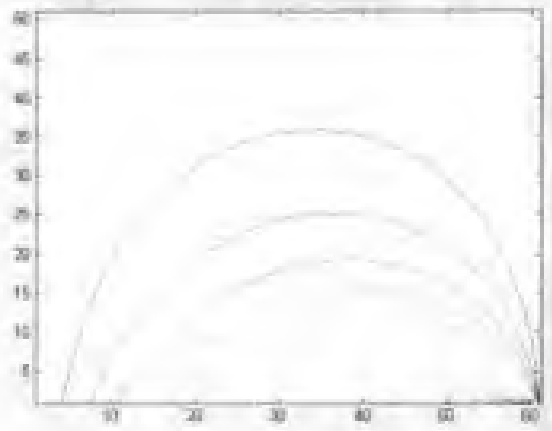


图 4.40 解析解的图形

用偏微分方程工具箱的解法步骤如下：

先设定坐标轴的范围为 $0 \leqslant x \leqslant 3, -1 \leqslant y \leqslant 1$，画两个矩形 R1 和 R2，在描述 R1 坐标的对话框中填入数值为，Left 取 0, Bottom 取 -1, Width 取 3, Heigh 取 2, 在描述 R2 的坐标的对话框中填入的数值为，Left 取 0, Bottom 取 -0.1, Width 取 2.5, Heigh 取 0.2, 在 Set fomula 对话框中，填入 R1−R2。在这里，R2 代表柱体，R1A#−R2 表示柱体外的区域。

在 R2 的两条长边上设置 Dirichlet 边界条件，系数为 $h = 1, r = x$。这是柱体侧面的边界条件。其余的边界保持原来默认的设置。这种做法是模拟远处的电势为 0，由于柱体外的区域不够大，所以这会造成一定的误差。

方程选为 Elliptic, 系数为 $c = 1, a = 0, f = 0$。划分网格后要细分 2 次。

画作图的选项为 Color 和 contour，所得的图形如图 4.41 所示。可以看出，定性地说，这个结果与解析解是一致的。这里所画的区域稍微大一些，包含了柱体的上下底之外的区域。而在解析解可视化的程序中，我们只画了上半部区域。

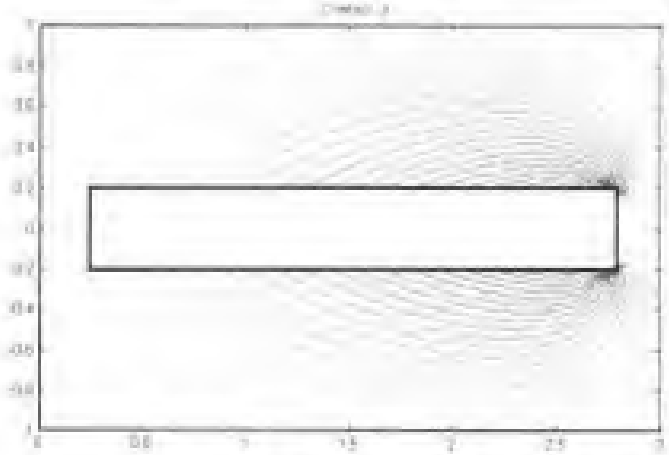


图 4.41 偏微分方程工具箱解得的结果

4.3 泊松方程与格林函数

4.3.1 矩形区泊松方程

在矩形区 $0<x<a,\ -b/2<y<b/2$ 上求解 $\Delta u=-x^2y$, 且 u 在边界上的值为零 * .

定解问题是

$$\begin{cases} \Delta u=0 \\ u(x=0)=0, \qquad u(x=a)=0 \\ u(y=-b/2)=0, \quad u(y=b/2)=0 \end{cases}$$

它的解是

$$u=\frac{xy}{12}(a^3-x^3)+\sum_{n=1}^{\infty}\frac{a^4b[(-1)^n n^2\pi^2+2-2(-1)^n]}{n^5\pi^5\sinh(n\pi b/2a)}\sinh\left(\frac{n\pi y}{a}\right)\sin\left(\frac{n\pi x}{a}\right)$$

用下面的程序可以画出 u 的图像, 程序中取 $a=5,\ b=5$.

```
[X,Y]=meshgrid(0:0.1:5,-5/2:0.1:5/2);
Z1=0;
for n=1:1:10
    Z2=5^4*5*((-1)^n*n^2*pi^2+2-2*(-1)^n)*sinh(n*pi.*Y/5).*...
    sin(n*pi.*X/5)/(n^5*pi^5*sinh(n*pi*5/10));
    Z1=Z1+Z2;
end
Z=Z1+X.*Y.*(5^3-X.^3)/12;
colormap(hot);
mesh(X,Y,Z)
view(119,7)
```

所得的图形如图 4.42 所示.

再用偏微分方程工具箱求解这个问题. 在 Options/Axes Limits 下选择 x 轴范围为 $0\sim5$, y 轴范围为 $-2.5\sim2.5$. 画一个矩形, 矩形的顶点为 (0, −2.5), (5,−2.5), (5, 2.5), (0, 2.5) .

*见参考文献 [1], 223 页.

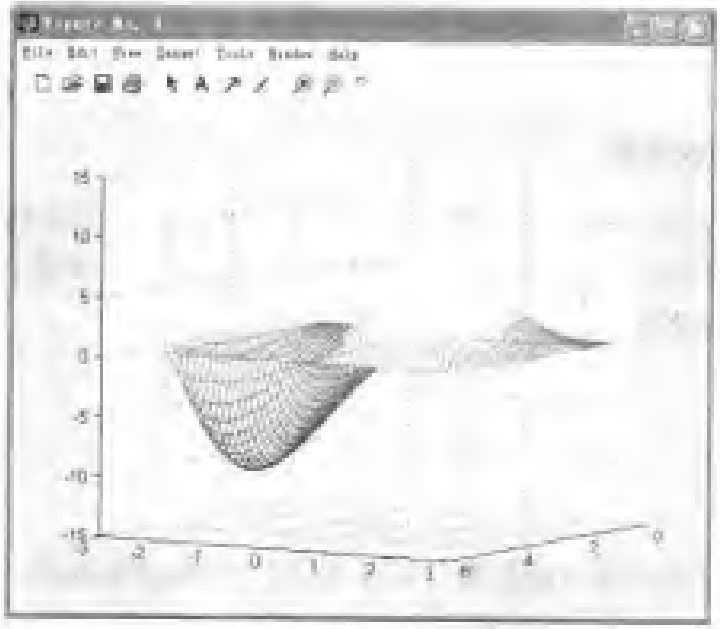


图 4.42 解析解的图形

按照题意，柱体 4 个边界都是狄里克利边界条件，而且是齐次的边界条件，可取 $h=1, r=0$ 。

方程的设置是 Elliptic 型，也就是 $c=1, a=0, f=0$ 。

为了有足够的精度，将初始化的网格作了两次细分，而作图的选项为 Color 和 Hight(3-D plot)，同时在 colormap 中选择 hot ，所得的图形如图 4.43 所示。

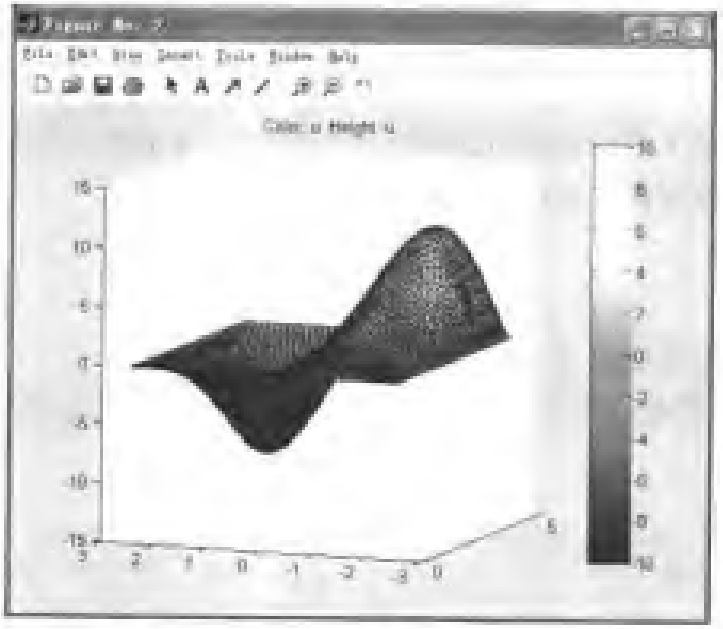


图 4.43 偏微分方程工具箱解得的结果

习题：在圆域上 $\rho < a$ 偏微分方程工具箱求解 $\Delta u = -xy$, 边界条件是 $u(\rho = 0) = 0$ 。

4.3.2 球域的格林函数

格林函数可以视为点源产生的场，下面研究球域内外的格林函数。

在半径为 a 的导体球内 (或者在导体球外)、距球心 r_0 处放置电量为 $4\pi\varepsilon_0 q$ 的点电荷，求它形成的静电场 * .

定解问题是

$$\begin{cases} \Delta G = -\delta(x - x_0, y - y_0, z - z_0) \\ G(r = a) = 0 \end{cases}$$

这里以 r_0 表示点电荷的位置，以 r 表示所计算的电场点. 当点电荷在球外时，要求 $r > a, r_0 > a$, 当点电荷在球内时，要求 $r < a, r_0 < a$.

这个问题的解可以用电像法求得，对球外与球内，所得解的形式相同.

$$u(r,\theta) = \frac{q}{\sqrt{r_0^2 - 2r_0 r\cos\theta + r^2}} - \frac{q(a/r_0)}{\sqrt{\left(\frac{a^2}{r_0}\right)^2 - 2\left(\frac{a^2}{r_0}\right) r\cos\theta + r^2}}$$

其中第一项是原来的点电荷在空间产生的电场，第二项可以看成是一个想像的点电荷产生的场，这个假想的点电荷的电量为 $-4\pi\varepsilon_0 q(a/r_0)$, 位置在球心与原电荷的联线上，距球心的距离为 a^2/r_0 . 这个想像的电荷通常叫电像. 它就是我们要求的球域外 (内) 的格林函数.

首先画点电荷在球外的情况，即要求 $r > a, r_0 > a$. 取 r_0 为极轴，以 θ 表示 r 与 r_0 的夹角，用下面的程序可以画出这个解在 r_0, θ 面的等值线图像. 为了能更清楚地表现其物理图像，我们利用公式

$$\boldsymbol{E} = -\nabla u$$

画出了电力线，结果如图 4.44 所示.

将图 4.44 中源电荷的场与像电荷的场单独画出得到图 4.45 .

*见参考文献 [4], 396 页.

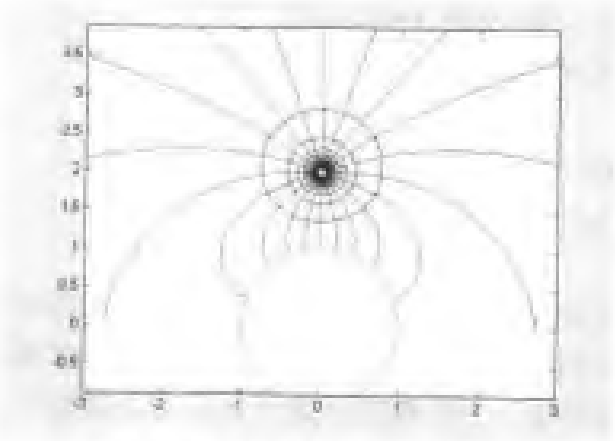


图 4.44 球域外点电荷的电场

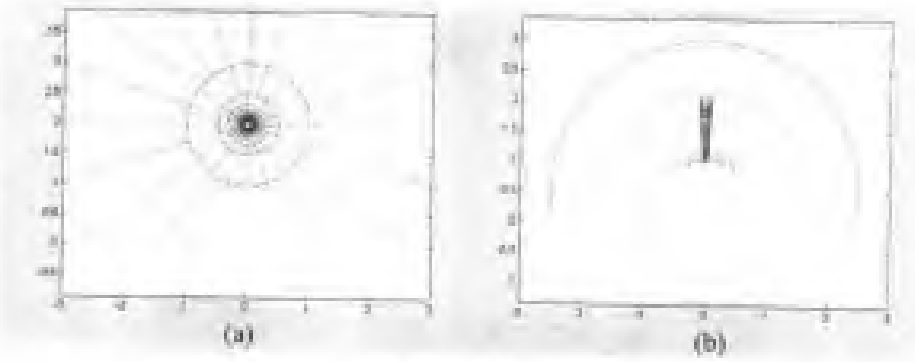


图 4.45

(a) 球域外源电荷的电场; (b) 球域外像电荷的电场

```
a=1;  r1=2; q=1;
[X,Y]=meshgrid(-3:0.01:3,0:0.01:4);
[Q,R]=cart2pol(Y,X);
R(R<=1)=NaN;
%R(R==2)=NaN;
u1=1./sqrt(r1^2-2*r1*R.*cos(Q)+R.^2);
ar=a/r1;
u2= -ar./sqrt((a*ar)^2-2*a*ar.*R.*cos(Q)+R.^2);
figure
u=u1+u2;
contour(X,Y,u,[-1:20,3000])
hold on
[ex,ey]=gradient(-u);
t=0:pi/10:2*pi;
```

```
sx=0.1*cos(t);    sy=2+0.1*sin(t);
streamline(X,Y,ex,ey,sx,sy)
axis equal
tt=0:pi/30:2*pi;
plot(exp(i*tt),'r')
```

用与上面相同的方法，可以画出球域内解的图像，这时要求 $r<a, r_0<a$。同样，也画出了电力线，结果如图 4.46 所示。所用的程序某些语句要作改变，读者可以对比下面的程序与上一个程序的差别。在两个图形中，电力线都是垂直于导体球的表面。

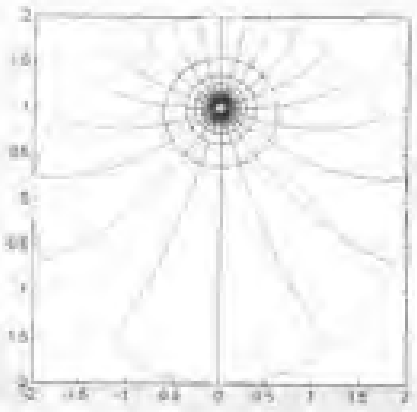


图 4.46　球域内点电荷的电场

将上面图中源电荷的场与像电荷的场单独画出得到图 4.47。

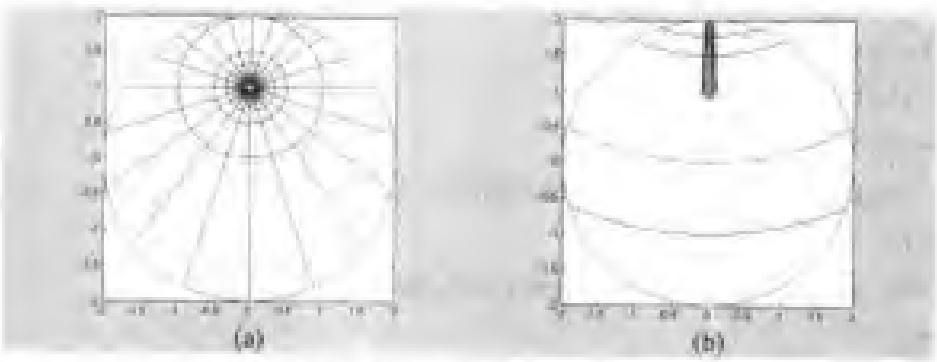


图 4.47　源电荷和像电荷的电场图

(a) 球域内源电荷的电场；(b) 球域内像电荷的电场

```
a=2;  r1=1; q=1;
```

```
[X,Y]=meshgrid(-2:0.01:2,-2:0.01:2);
[Q,R]=cart2pol(Y,X);
R(R>=2)=NaN;
u1=1./sqrt(r1^2-2*r1*R.*cos(Q)+R.^2);
ar=a/r1;
u2= -ar./sqrt((a*ar)^2-2*a*ar.*R.*cos(Q)+R.^2);
figure
u=u1+u2;
contour(X,Y,u,[-1:20,3000])
hold on
[ex,ey]=gradient(-u);
t=0:pi/10:2*pi;
sx=0.1*cos(t);    sy=1+0.1*sin(t);
streamline(X,Y,ex,ey,sx,sy)
axis square
tt=0:pi/30:2*pi;
plot(2*exp(i*tt),'r')
```

4.3.3 圆域的格林函数

在半径为 a 的圆外、距圆心 r_0 处放置电量为 $4\pi\varepsilon_0 q$ 的点电荷，求它形成的静电场 *。

定解问题为

$$\begin{cases} \Delta G = -\delta(x-x_0, y-y_0) \\ G(\rho = a) = 0 \end{cases}$$

这里以 r_0 表示点电荷的位置，以 r 表示所计算的电场点。当点电荷在球外时，要求 $r > a, r_0 > a$，当点电荷在球内时，要求 $r < a, r_0 < a$。这个问题的解可以用电像法求得，结果是用平面极坐标表示。

$$G = \frac{1}{2\pi}\ln\frac{1}{a\sqrt{r_0^2 - 2r_0 r\cos\theta + r^2}} + \frac{1}{2\pi}\ln\rho_0\sqrt{\left(\frac{a^2}{r_0}\right)^2 - 2\left(\frac{a^2}{r_0}\right)r\cos\theta + r^2}$$

其中第一项是原来的点电荷在空间产生的电场，第二项是电像的场。

*见参考文献 [4], 396 页。

首先画点电荷在圆外的情况，这时要求 $r>a, r_0>a$. 用下面的程序可以画出这个解的图像. 为了能更清楚地表现其物理图像，我们利用公式

$$\boldsymbol{E}=-\nabla u$$

画出了电力线，结果如图 4.48 所示.

```
[X,Y]=meshgrid(-10:0.1:10);
[Q,R]=cart2pol(X,Y);
R(R<=1)=NaN;
r0=2;  a=1;
ar=a/r0;
g1=sqrt((a*ar)^2-2*a*ar.*R.*cos(Q)+R.^2)
g2=sqrt(r0^2-2*r0*R.*cos(Q)+R.^2);
G=1/2/pi*log(r0/a*g1./g2);
contour(X,Y,G,7,'g')
hold on
axis equal
tt=0:pi/10:2*pi;
plot(exp(i*tt),'r:')
[ex,ey]=gradient(-G);
sx=2+0.3*cos(tt);   sy=0.3*sin(tt);
streamline(X,Y,ex,ey,sx,sy)
```

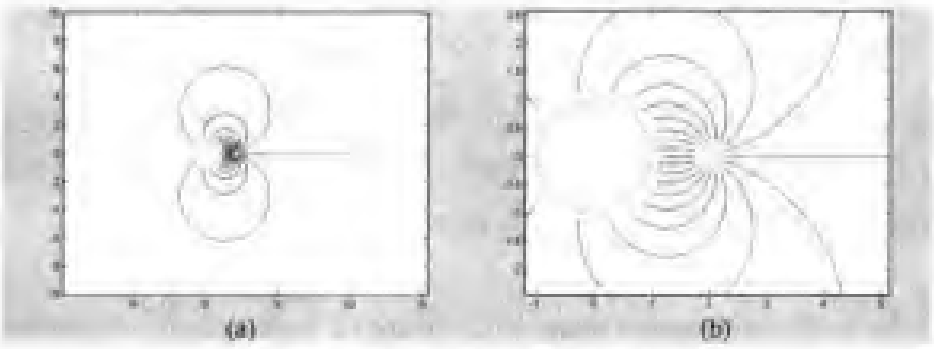


图 4.48 圆域内外点电荷电场图

(a) 圆域外点电荷的电场整体图；(b) 圆域外点电荷的电场局部放大图

将图 4.48 中源电荷的场与像电荷的场单独画出得到图 4.49 .

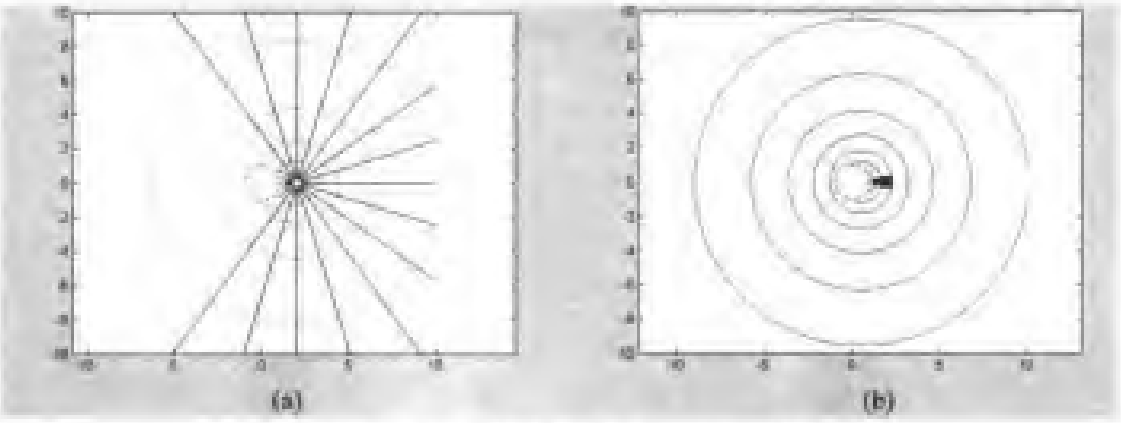


图 4.49 圆域内源电荷和像电荷的电场图

(a) 圆域外源电荷的电场; (b) 圆域外像电荷的电场

同样，如下所示，只要将程序稍加改动，可以画出圆域内点电荷的电场，如图 4.50 所示，这时有 $r<a, r_0<a$.

```
[X,Y]=meshgrid(-3:0.1:3);
[Q,R]=cart2pol(X,Y);
R(R>=2)=NaN;
r0=1;  a=2;
ar=a/r0;
g1=sqrt((a*ar)^2-2*a*ar.*R.*cos(Q)+R.^2);
g2=sqrt(r0^2-2*r0*R.*cos(Q)+R.^2);
G=1/2/pi*log(r0/a*g1./g2);
contour(X,Y,G,7,'g')
hold on
axis equal
tt=0:pi/10:2*pi;
plot(2*exp(i*tt),'r:')
[ex,ey]=gradient(-G);
sx=1+0.25*cos(tt);   sy=0.25*sin(tt);
streamline(X,Y,ex,ey,sx,sy)
```

将图 4.50 中源电荷的场与像电荷的场单独画出得到图 4.51 .

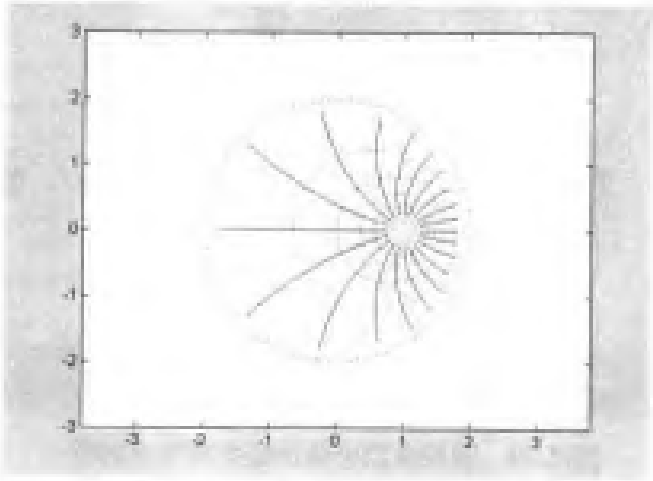


图 4.50 圆域内点电荷的电场

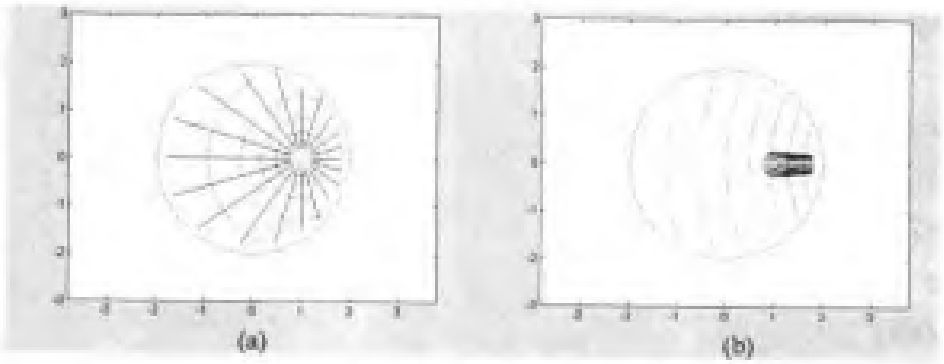


图 4.51 圆域内源电荷与像电荷电场图

(a) 圆域内源电荷的电场; (b) 圆域内像电荷的电场

第 5 章　热传导方程

热传导方程除含有空间变量之外还含有时间变量，所以解法要比拉普拉斯方程相对复杂一些．与上一章的分类方法相同，我们还是将本章的内容按空间变量的数目分为一维、二维和三维．如果用差分方程作数值解，要注意参数的选择与时间的长短对结果的影响很大．

5.1　一维热传导问题

5.1.1　无限长细杆的热传导

无限长细杆的热传导的定解问题是 *

$$\begin{cases} u_t = a^2 u_{xx} \\ u(x, t=0) = \varphi(x) \end{cases}$$

利用傅里叶变换求得问题的解是

$$u(x,t) = \int_{-\infty}^{+\infty} \varphi(\xi) \left[\frac{1}{2a\sqrt{\pi t}} \mathrm{e}^{-\frac{(x-\xi)^2}{4a^2 t}} \right] \mathrm{d}\xi$$

其中取初始温度分布如下：

$$\varphi(x) = \begin{cases} 1 & (0 \leqslant x \leqslant 1) \\ 0 & (x \leqslant 0, x \geqslant 1) \end{cases}$$

这是在区间 $0 \sim 1$ 之间高度为 1 的一个矩形脉冲，于是得

$$u(x,t) = \int_0^1 \frac{1}{2a\sqrt{\pi t}} \mathrm{e}^{-\frac{(x-\xi)^2}{4a^2 t}} \mathrm{d}\xi$$

可以用图 5.1 所示的瀑布图来表示温度随时间与空间的变化．图中 $-10 \leqslant x \leqslant 10, 0 \leqslant t \leqslant 1, 0 \leqslant u \leqslant 100$.

从图中可以看到，在开始时，温度分布是原点附近的一个脉冲状的分布，随着时间增加，热量向两边传播，形成一个平缓的波包，不难想像，如果时间足够长，最终杆上的温度会全部为零．

---

*见参考文献 [1], 407 页．

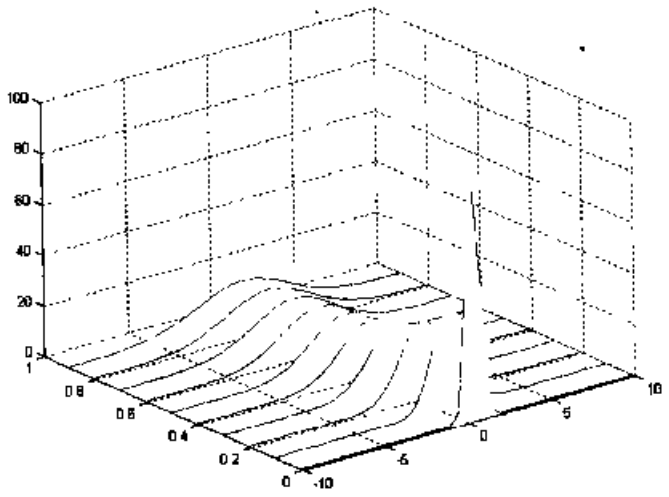


图 5.1 一维热传导的积分解

所用的程序如下：

```
xx=-10:.5:10;
tt=0.01:0.1:1;
tau=0:0.01:1;
a=2;
[X,T,TAU]=meshgrid(xx,  tt,   tau);
F=1/2/2./sqrt(pi*T).*exp(-(X-TAU).^2/4/2^2./T);
js=trapz(F,3);
waterfall(X(:,:,1),  T(:,:,1),   js)
```

也可以用动画来表示杆上的温度随时间的变化，在上面的程序后面加上如下语句：

```
figure
h=plot(xx',js(1,:))
set(h,'erasemode','xor');
for j=2:10
    set(h,'ydata',js(j,:));
    drawnow;
    pause(0.1)
end
```

图 5.2 是其中的一个画面，所有的画面按时间排列起来，就形成上一幅瀑布

图. 图中 $-10 \leqslant x \leqslant 10, 0 \leqslant u \leqslant 45$.

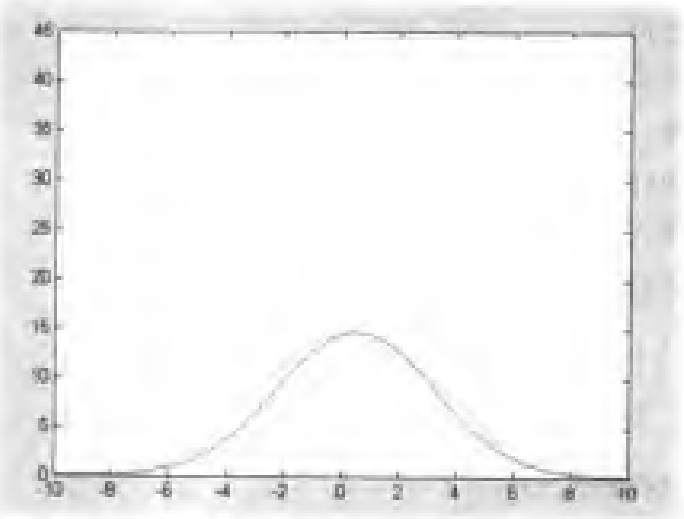


图 5.2 无限长的杆上的热传导

5.1.2 有限长细杆的热传导

下面讨论有限长的细杆在第一类边界条件下的热传导问题，可以将这个结果与无限长细杆的热传导问题对比.

有限长细杆的热传导的定解问题是

$$\begin{cases} u_t = a^2 u_{xx} \\ u(0,t) = 0, \qquad u(l,t) = 0 \\ u(x, t=0) = \varphi(x) \end{cases}$$

其中 $0 \leqslant x \leqslant 20$. 即 $l = 20$, 取 $a = 10$ 且

$$\varphi(x) = \begin{cases} 1 & (10 \leqslant x \leqslant 11) \\ 0 & (x < 10, x > 11) \end{cases}$$

所得的解为

$$u(x,t) = \sum_{n=1}^{\infty} \frac{2}{n\pi} \left[\cos \frac{10n\pi}{20} - \cos \frac{11n\pi}{20} \right] \mathrm{e}^{-\frac{a^2\pi^2}{400}t} \sin \frac{n\pi}{20} x$$

再画出前 50 个分波的合成图形，在程序中用子函数来计算各个分波，然后在主函数中调用. 所用的程序是

```
function    jxj
N=50;  t=1e-5:0.00001:0.005;      x=0:0.21:20;
w=rcdf(N,t(1));
h= plot(x,w,'linewidth',5);
axis([ 0, 20, 0, 1.5])
for n=2:length(t)
    w=rcdf(N,t(n));
    set(h,'ydata',w);
    drawnow;
end

function u=rcdf(N,t)
x=0:0.21:20;   u=0;
for k=1:2*N
    cht=2/k/pi*(cos(k*pi*10/20)-cos(k*pi*11/20))*sin(k*pi*x./20);
    u=u+cht*exp(-(k^2*pi^2*10^2/400*t));      %a=10,l=20
end
```

图 5.3 是其中的一个画面，图中 $0 \leqslant x \leqslant 20, 0 \leqslant u \leqslant 1.5$.

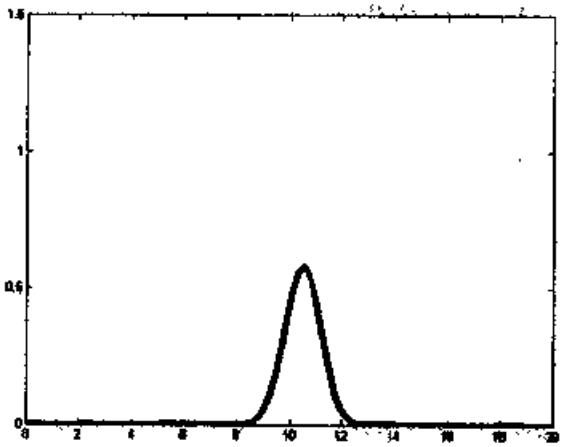


图 5.3 有限长杆热传导的解析解的动画

从画面可以看到，当热量没有传到边界上时，也就是边界的影响没有起作用时，由于初始条件相同，有限长的杆的温度与无限长的杆的温度分布是一样的. 所以，所谓的无限长的杆的热传导现象实际上是指边界的影响还没有产生作用时，有限长的杆上的热传导现象的一种近似.

再用 PDETOOL 来解这个问题。

求解区域为 $-10 \leqslant x \leqslant 10, -0.2 \leqslant y \leqslant 0.2$, 矩形的上下边界为诺伊曼边界条件，取 $q = 0, g = 0$. 左右边界为 Dirichlet 边界条件，取 $h = 1, r = 0$。

方程类型应该选抛物型 (Parabolic)。

求解参数 time 取 0 : 0.005 : 0.2, 初始条件 u(t0) 为 $10 * \tanh(10000 * x) - 10 * \tanh(10000(x - 0.05))$, 这个初始温度分布近似于定解问题中初始条件。这个设置是为了用函数而不是用语句来描述初始温度。

作图选择的参数为 Color,Heigh(3-D plot) 和 Animation，图 5.4 是其中的一个画面。

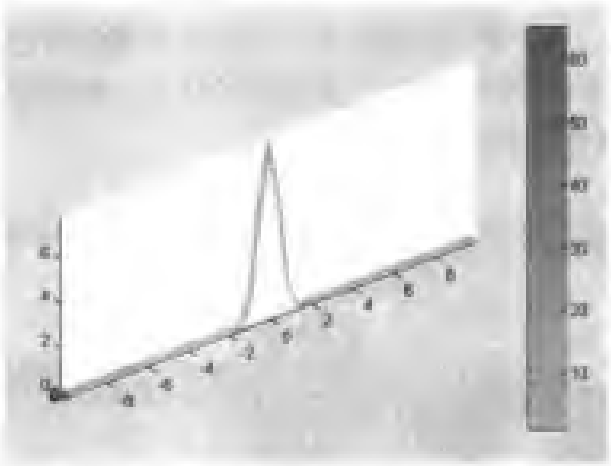


图 5.4 有限长杆热传导的偏微分方程工具箱解

用同样的方法，读者很容易用偏微分方程工具箱对如下的定解问题求解。

$$\begin{cases} u_t = a^2 u_{xx} \\ u(0,t) = 0, \qquad u_x(l,t) = 0 \\ u(x,0) = u_0 x/l \end{cases}$$

5.1.3 输运问题 (非齐次方程)

定解问题是 *

$$\begin{cases} u_t - a^2 u_{xx} = -bu_x, & (0 \leqslant x \leqslant l, t \geqslant 0) \\ u(x=0) = 0, & u(x=l) = 0 \\ u(t=0) = (x - l/2)^2 \end{cases}$$

*见参考文献 [1],215 页。

解析解是

$$u(x,t)=\sum_{n=1}^{\infty}A_n e^{-\left(\frac{n^2\pi^2a^2}{l^2}+\frac{b^2}{4a^2}\right)t}e^{\frac{bx}{2a^2}}\sin\frac{n\pi}{l}x$$

其中

$$A_n=\frac{2}{l}\int_0^l\left(x-\frac{l}{2}\right)^2 e^{-\frac{bx}{2a^2}}\sin\frac{n\pi}{l}x\mathrm{d}x$$

我们分别用解析解与差分方程的数值解画图. 读者不妨自己用偏微分方程工具箱解这个问题.

计算中取 $0\leqslant t\leqslant 0.0005$, 这个时间很短, 因为这个过程的时间很短. 解析解的图形如图 5.5 所示. 其中图 (a) 是以 t,x 为变量所画的表面图, 从图中可以看出变化的全貌图 (b) 是初始状态; 图 (c) 是最后的状态. 解析解在初始状态所画出的图形与差分方程的解有一定的偏差.

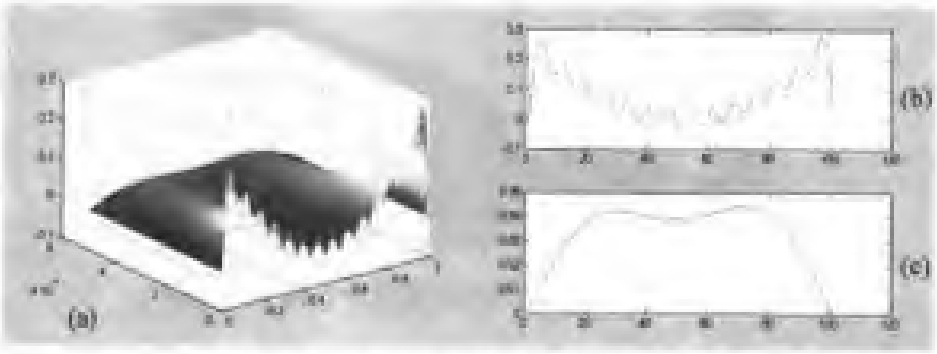


图 5.5 非齐次方程的输运问题的解析解

(a) 整体图; (b) 初始状态; (c) 最后状态

下面所用的程序中 $a^2=50, b=5, l=1$, 为了方便比较, 两个计算采用了相同的数据.

```
a2=50; b=5;
[x, t] = meshgrid(0:0.01:1,0:0.000001:0.0005);
Anfun=inline('2*(x-0.5).^2.*exp(5*x./2./50).*sin(n*pi*x)','x','n');
u=0
for n=1:30
    An=quad(Anfun,0,1,[ ],[ ],n);
    un=An*exp(-(n*n*pi*pi*50+25/4/2500).*t).*...
       exp(5/2/50.*x).*sin(n*pi*x);
    u=u+un;
    size(u)
```

```
end
mesh(x,t,u)

figure
subplot(2,1,1)
plot(u(1,:))
subplot(2,1,2)
plot(u(end,:))
```

差分方程所得的数值解的图形如图 5.6 所示，其中图 (a) 是开始的状态，图 (b) 是最后一幅图.

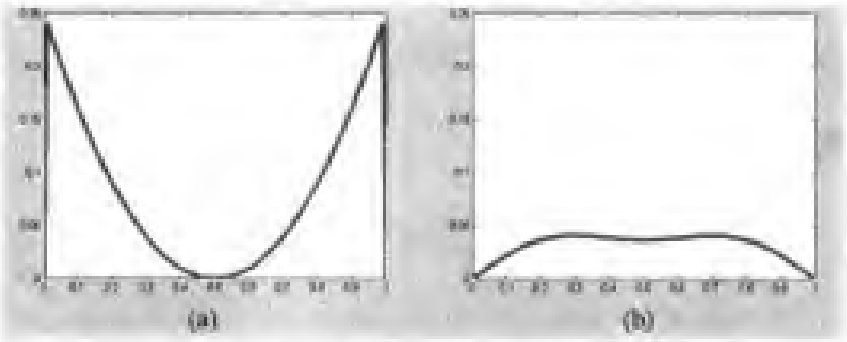


图 5.6 非齐次方程的输运问题的数值解

(a) 初始状态; (b) 最后状态

所用的程序是

```
N=500;  dx=0.01;  dt=0.000001;
c=50*dt/dx/dx;
A=500; b=5;
x=linspace(0,1,100)';
uu(1:100,1)=(x-0.5).^2;          %初温度为零
figure
h=plot(x,uu(:,1),'linewidth',5);
set(h,'EraseMode','xor')
axis([0,1,0,0.25]);
for k=2:200
   uu(2:99,2)=(1-2*c)*uu(2:99,1)+c*(uu(3:100,1)+uu(1:98,1))-...
              b*dt/dx*(uu(3:100,1)-uu(2:99,1));
   uu(1,2)=0;  uu(100,2)=0;   %加上边界条件
```

```
    uu(:,1)=uu(:,2);
    set(h,'YData',uu(:,1)) ;
    drawnow;
    pause(0.01)
end
```

5.1.4 第三类边界条件下的细杆导热问题

定解问题是

$$\begin{cases} u_t(x,t)=a^2u_{xx}(x,t) & (0\leqslant x\leqslant l,\ t>0) \\ u(x=0,t)=0, & \dfrac{\partial}{\partial x}u(x=l,t)+hu(x=l)=0 \quad (t>0) \\ u(x,t=0)=\sin(x) & (0<x<l) \end{cases}$$

这个问题的解析解是*

$$u(x,t)=\sum_{n=1}^{\infty}B_n\mathrm{e}^{-a^2\mu_n^2t}\sin\mu_nx$$

其中

$$B_n=2\int_0^l\frac{f(\xi)\sin\mu_n\xi\mathrm{d}\xi}{l+\dfrac{\cos^2\mu_nl}{h}}$$

$$\mu_n^2=\left(\frac{\eta_n}{l}\right)^2 \quad (n=1,2,3,\cdots)$$

而 η 是方程

$$\tan\eta=-\beta\eta$$

的解. 这里

$$\eta=\mu l,\quad \beta=\frac{1}{hl}$$

下面用偏微分方程工具箱求它的数值解.

将坐标轴的范围设为 $0\leqslant x\leqslant 2,-0.2\leqslant y\leqslant 0.2$, 求解区域为 $0\leqslant x\leqslant 2,-0.1\leqslant y\leqslant 0.1$.

矩形的上下边界为 Neuman 边界条件, 取 $q=0,g=0$, 左边界为 Dirichlet 边界条件, 取 $h=1,r=0$, 右边界还是 Neuman 边界条件, 取 $q=1,g=0$.

*见参考文献 [2], 207 页.

方程类型应该选抛物型 (Parabolic)，求解参数设置是 $c=1, a=0, f=0$，时间范围取 0 : 0.005 : 0.2, 初条件 u(t0) 为 sin(πx)。

作图选择的参数为 Color,Contour,Heigh(3-D plot) 和 Animation，图 5.7 是按时间顺序截取的几个画面，从图形上所标出高度值可以明显看出它们的变化。

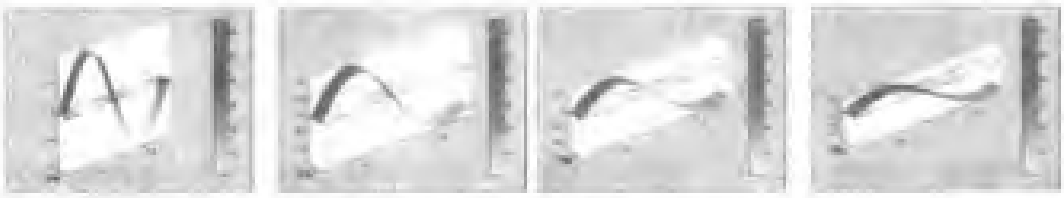


图 5.7 第三类齐次边界条件的有限长杆热传导

为了对比，我们对相同初始条件的第一类齐次边界条件下的问题求解，即定解问题是

$$\begin{cases} u_t(x,t) = a^2 u_{xx}(x,t) & (0<x<l,\ t>0) \\ u(x=0,t)=0, & u(x=l,t)=0 \quad (t>0) \\ u(x,t=0)=\sin(x) & (0<x<l) \end{cases}$$

在用偏微分方程工具箱求解时，只要将右边界条件改为 Dirichlet 边界条件，即取 $h=1, r=0$ 就行，其余不变。按时间顺序截取所得图形如图 5.8，与图 5.7 对比，明显的差别是，这里两端的温度始终为零。

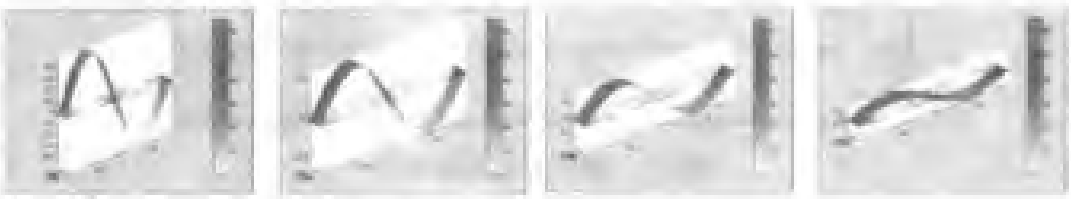


图 5.8 第一类齐次边界条件的有限长杆热传导

5.2 二维热传导问题

为简单起见，我们直接将 4.1.1 节标题中的拉普拉斯方程用热传导方程替代。从结果可以看到，无源的热传导问题，当时间足够长以后，初始条件的影响实际上已经不存在了。

现在的定解问题是

$$\begin{cases} u_t = k^2(u_{xx}+u_{yy}) & (0 \leqslant x \leqslant a,\ 0 \leqslant y \leqslant b) \\ u(x=0,y,t)=0, & u(x=a,y,t)=\mu \sin\dfrac{3\pi y}{b} \\ u(x,y=0,t)=0, & u(x,y=b,t)=\mu \sin\dfrac{3\pi x}{a}\cos\dfrac{\pi x}{a} \\ u(x,y,t=0)=0 \end{cases}$$

为了简单，可取 $k=1$ 。这个问题除了方程不同之外，其余条件与 4.1.1 节中的相同，所以在求解过程的前几步是相同的，读者可以参看 4.1.1 节中问题的解法。

方程类型应该选抛物型 (Parabolic)，系数取 $c=1, a=0, f=0, d=0$ 。

在用与 4.1.1 节中问题相同的方法设置边界条件和三角形网格之后，还要设定初始条件，选择菜单 Solve/Parameters，打开对话框如图 5.9 。

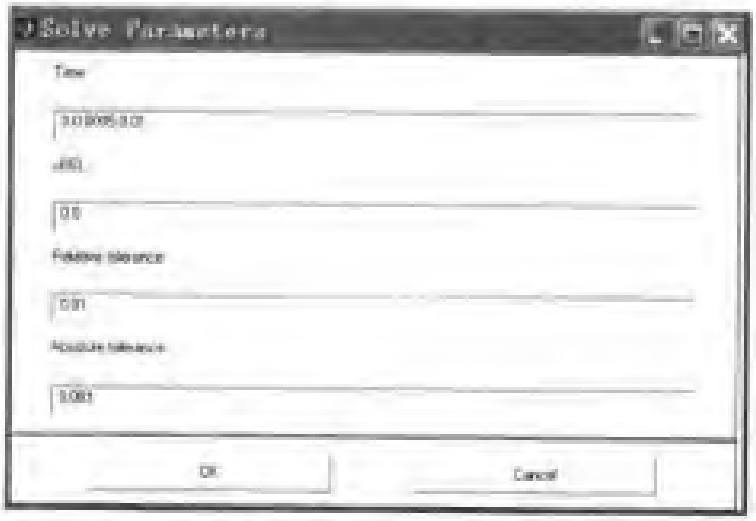


图 5.9 解方程的初始条件

在时间 (Time) 栏设定求解方程的时间范围为 0 : 0.0005 : 0.01，将时间步长取小些可以使下面的动画过程进行得慢一些。由题意 $u(x,y,t=0)=0$，在 u(t0) 栏中填入 0 。相对误差栏 (Relative tolerance) 和绝对误差 (Absolute tolerance) 栏可以保持默认的设定值。

再单击按钮 ，在对话框除了选择 Hight (3-D plot) 和 Plot in x - y grid 之外，还要选择 Animation 来激活动画。为了放慢动画演示的速度，可以单击按钮 Options，打开图 5.10 所示的对话框。

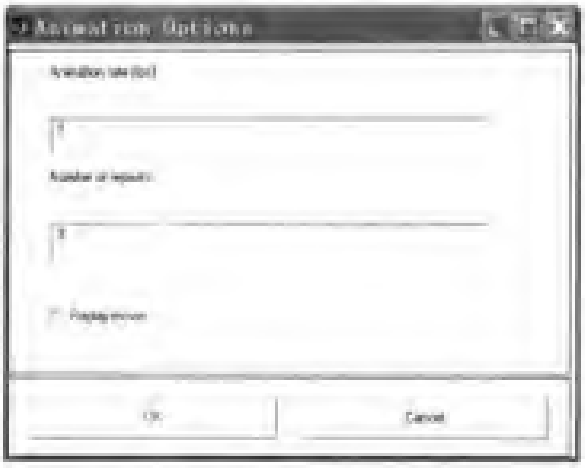


图 5.10 设置动画放映参数

在 Animation rate (fps)(每秒放映的画面数) 栏中填入 1, 在 Number of repeats(重放次数) 栏中填入 4, 单击 OK 按钮确认, 关闭对话框, 再单击对话框中 Plot 按钮即得到动画演示的结果。

在 MATLAB 中有一个矩形区域的有源的热传导的演示例子。在指令窗口键入 pdedemo5 即可运行这个程序。它求解的定解问题是, 在矩形的四边温度为零, 在区域的中央, 半径为 0.4 的圆内, 有个恒定的热源, 其值为 1 。在附录 A.1.5 节有对它的介绍以及程序所得的图形。

5.3 三维热传导问题

5.3.1 球体内的热传导问题之一 (j_0 的应用)

半径为 r_0 的匀质球放在温度为 U_0 的烘箱中, 初始时刻球体的温度为 u_0, 求此后球内各点的温度变化情况 * 。

令 $u = w + U_0$, 则 w 的定解问题是

$$\begin{cases} w_t = \Delta w \\ w(r = r_0) = 0 \\ w(t = t_0) = u_0 - U_0 \end{cases}$$

这个问题的解是

$$w = \frac{2(U_0 - u_0)}{\pi r} \sum_{n=1}^{\infty} \frac{(-1)^n}{n} \mathrm{e}^{-n^2\pi^2 a^2 t/r_0^2} \sin \frac{n\pi r}{r_0}$$

*见参考文献 [1], 367 页。

在解中只有一个空间变量 r, 为了与偏微分方程工具箱的结果作对比, 我们将 r 表示成 x, y 的函数, 画成一个随时间变化的二维图形, 这就是下面程序中第一个图形的结果. 不难想像, 这时等温线其实是一些同心圆. 也可以用瀑布图来表示解析解的结果, 在瀑布图中, 是以 r, t 作为变量, 这就是下面程序中第二部分的做法. 它表示的是一条半径上的温度随 t 的变化.

为了作图, 取 $r_0 = 1$, $a^2 = 2$, $U_0 = 2$, $u_0 = 0$, 在级数求和中只取了前面 10 项.

```
U0=2;     u0=0;       a2=2;   N=10;
r=eps:0.05:1;        theta=linspace(0,2*pi,100) ;
t=0.1:0.001:0.2;
[RR,TT]=meshgrid(r,t);
figure(1)
[R,TH]=meshgrid(theta,r);
[X,Y]=pol2cart(R,TH);
for  tt=1:100
    un=0;
    for  k=1:N
        unn =2*(U0-u0)*(-1)^k.*sin(k.*pi.*(X.^2+Y.^2).^0.5).*...
              exp(-k^2*pi^2*a2*t(tt))./(pi *k*((X.^2+Y.^2).^0.5));
        un=unn+un;
    end
    mesh(X,Y,un);
    axis([-1  1  -1  1    -0.4   0]);
    pause(0.1)
end
figure(2)
wn=0;
for  k=1:N
    wnn =2*(U0-u0)*(-1)^k.*sin(k.*pi.*RR).*...
          exp(-k^2*pi^2*a2*TT)./(pi *k.*RR);
    wn=wnn+wn;
end
waterfall(RR,TT,wn)
xlabel('r')
ylabel('t')
```

图 5.11 是动画中的一幅图形, 所得的瀑布图形如图 5.12 .

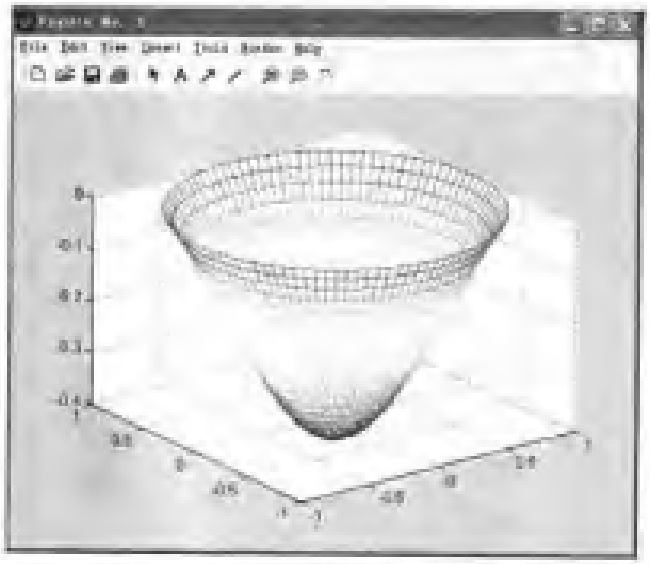


图 5.11 解析解的动画图形

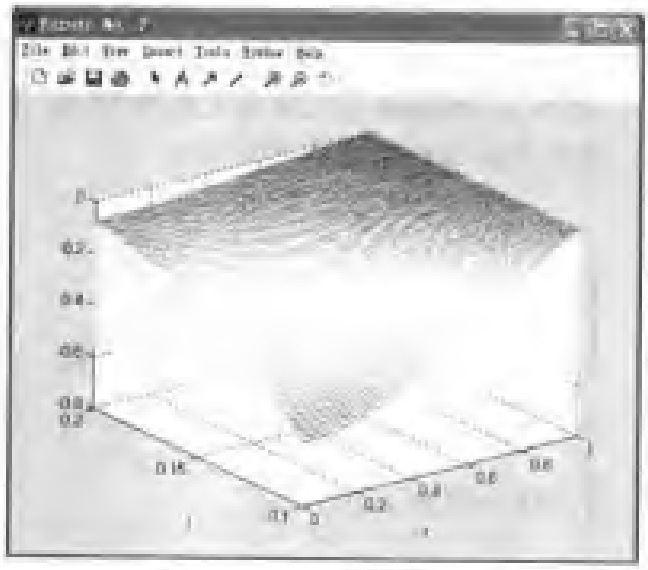


图 5.12 解析解的瀑布图形

再用偏微分方程工具箱求解这个问题。画一个中心在原点半径为 1 的圆，按照题意，圆的边界都是狄里克利边界条件，可取 $h=1, r=0$ 。

方程的设置是 parobolic 型，系数取为 $c=1, a=0, f=0, d=1$ 。

解题的时间范围为 $t=0:0.1:1$，初始条件是 u(t0)=2 。

为了有足够的精度，将初始化的网格作了两次细分。而作图的选项为 Color, Height (3 - D plot), Animation 和 Show mesh, 同时在 colormap 中选择 hot 。图 5.13 是动画中的一幅图形。

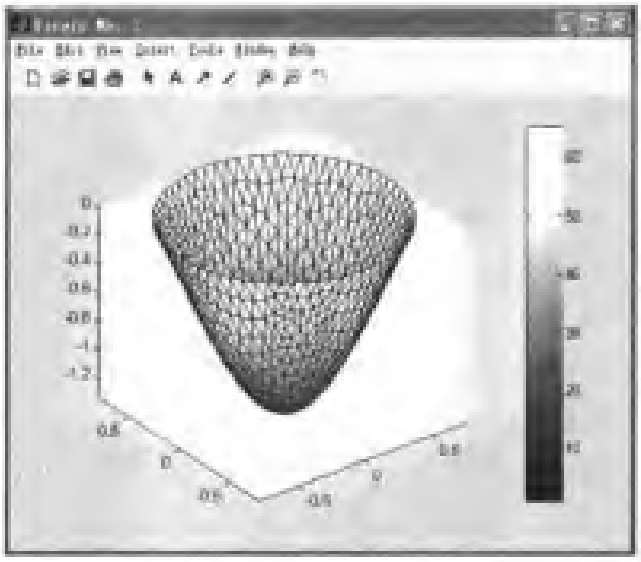


图 5.13 数值解的图形

5.3.2 柱体内的热传导

圆柱体半径为 ρ_0, 高为 L, 侧面和下底面温度保持为 u_0, 上底面绝热, 初始温度为 $u_0 + f_1(\rho)f_2(z)$, 求圆柱体内各处 u 的变化情况 * .

定解问题是

$$\begin{cases} u_t = a^2 \Delta u \\ u(\rho = \rho_0) = 0, \qquad u(\rho = 0) = \text{有限} \\ u(z = 0) = 0, \qquad u_z(z = l) = 0 \\ u(t = 0) = f_1(\rho) f_2(z) \end{cases}$$

其解为

$$u = \sum_{n=1}^{\infty} \sum_{\rho=0}^{\infty} A_{n\rho} \mathrm{e}^{-\left[\left(\frac{x_n^{(0)}}{\rho_0}\right)^2 + \frac{\left(\rho+\frac{1}{2}\right)^2 \pi^2}{L^2}\right] a^2 t} \mathrm{J}_0\left(\frac{x_n^{(0)}}{\rho_0}\rho\right) \sin\frac{\left(\rho+\frac{1}{2}\right)\pi}{L} z$$

其中

$$A_{n\rho} = \frac{2}{\rho_0^2 \left[\mathrm{J}_0'\left(x_n^{(0)}\right)\right]^2} \int_0^{\rho_0} f_1(\rho) \mathrm{J}_0\left(\frac{x_n^{(0)}}{\rho_0}\rho\right) \rho \mathrm{d}\rho \cdot \frac{2}{L} \int_0^L f_2(z) \sin\frac{\left(\rho+\frac{1}{2}\right)\pi}{L} z \mathrm{d}z$$

这个问题解析解很复杂, 我们直接用偏微分方程工具箱求解这个问题, 初始条件取 $u(t = 0) = xy^2$.

*见参考文献 [1], 339 页.

坐标轴的范围设置成 $0 \leqslant x \leqslant 2, -0.5 \leqslant y \leqslant 0.5$, 以原点为柱体底部的中心, 画一个高为 2 宽为 0.4 的柱体. 在这个图上, 坐标 x 对应 z, 坐标 y 对应 ρ.

按照题意, 在柱体的边界都是狄里克利边界条件, 可取 $h = 1, r = 0$.

方程的设置是 parobolic 型, 系数取为 $c = 1, a = 0, f = 0, d = 1$.

解题的时间范围为 t=0:0.001:0.1, 初始条件是 x.*y.^2.

为了有足够的精度, 将初始化的网格作了两次细分, 而作图的选项为 Contour 和 Animation. 图 5.14 是动画中的几幅图形.

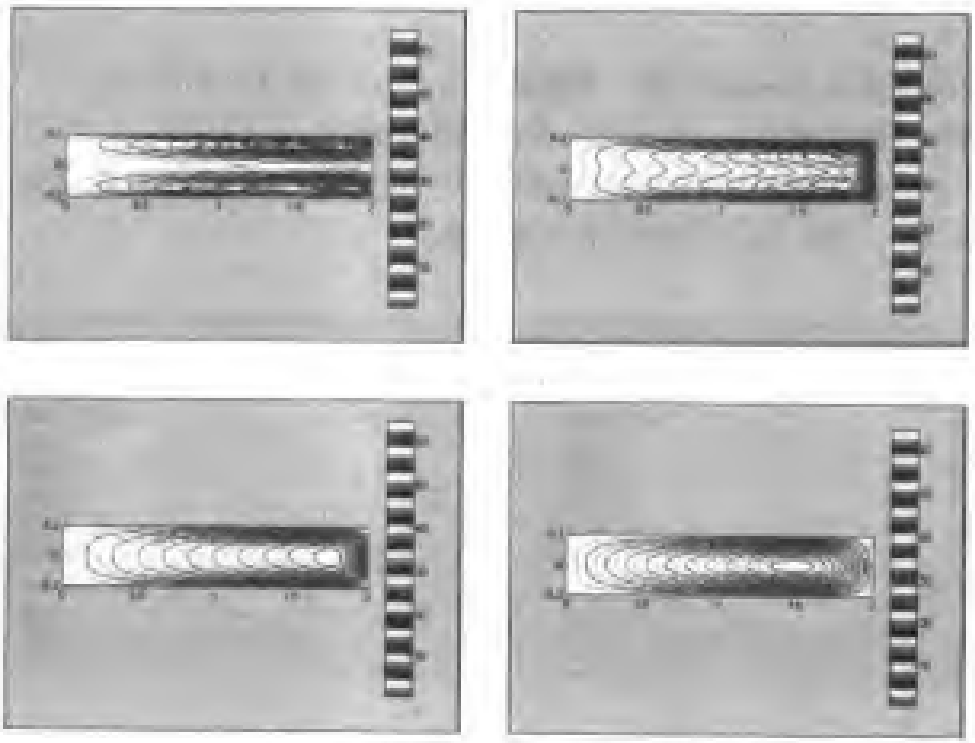


图 5.14 初始条件为 xy^2 的数值解的图形

5.3.3 球体内的热传导问题之二 (J_1 的应用)

半径为 r_0 的均匀球, 初始温度分布为 $f(r)\cos\theta$, 保持球面温度为零度让它冷却, 求球内各点的温度变化*.

定解问题是

$$\begin{cases} u_t = a^2 u \\ u(r = r_0) = 0 \\ u(t = 0) = f(r)\cos\theta \end{cases}$$

*见参考文献 [1], 372 页.

解析解是

$$u(r,\theta)=\frac{2}{r_0^3}\sum_{n=1}^{\infty}\frac{\mathrm{j}_1(k_n r)}{[\mathrm{j}_0(k_n r_0)]^2}\mathrm{P}_1(\cos\theta)\mathrm{e}^{-a^2k_n^2t}\int_0^{r_0}\mathrm{j}_1(k_n r)f(r)r^2\mathrm{d}r$$

其中 $k_n=x_n/r_0$，而 x_n 是方程 $x=\tan x$ 的第 n 个根。

这个问题解析解很复杂，我们直接用偏微分方程工具箱求解这个问题。初始条件是 $u(t=0)=r\cos\theta$。

画一个中心在原点半径为 1 的圆，按照题意，圆的边界都是狄里克利边界条件，可取 $h=1,r=0$。

方程的设置是 parobolic 型，系数取为 $c=1,a=0,f=0,d=1$。

解题的时间范围为 t= 0 : 0.001 : 0.1，初始条件是 $\mathrm{r}\cos\theta=\mathrm{x}$。

为了有足够的精度，将初始化的网格作了两次细分。而作图的选项为 Contour 和 Animation。图 5.15 是动画中的几幅图形。

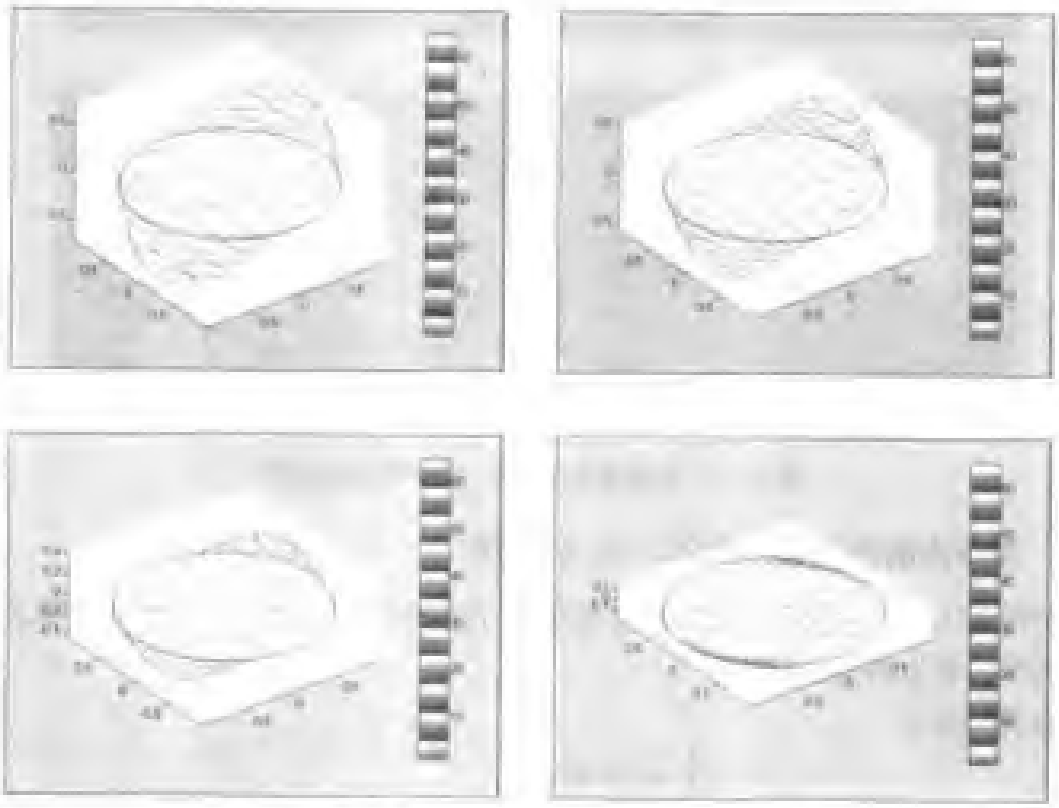


图 5.15 初始条件为 $r\cos\theta=x$ 的数值解的图形

作为对比，我们还设置了一个初始条件是 x.*y，所得的结果如图 5.16 所示。

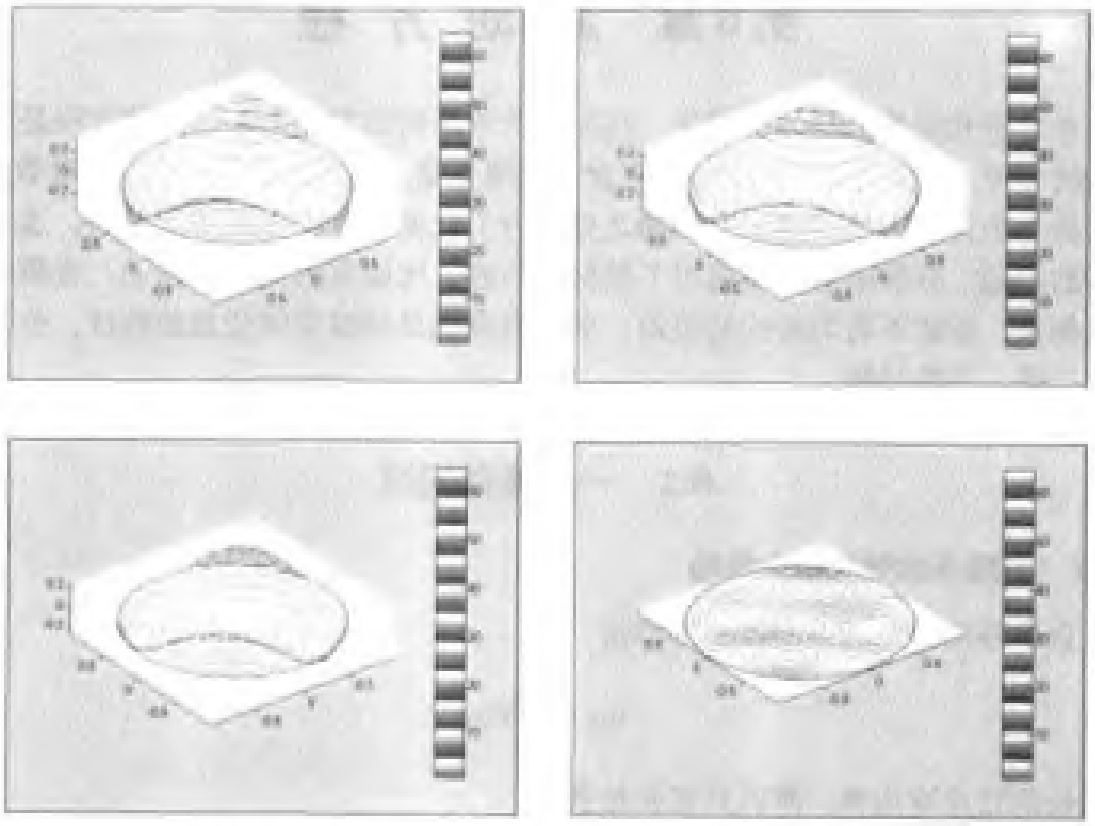


图 5.16 初始条件为 $r^3\cos\theta = x*(x^2+y^2)$ 的数值解的图形

只要时间足够长，这两种不同的初始条件经过不同的过程，最终都趋于相同的温度分布，这是无源的热传导问题的必然结果. 这时的等温线将是一些同心圆。读者可以将前面的程序增加时间以后运行，就会看到这个结果。

第6章 波 动 方 程

本章研究波动方程的可视化. 在本书所涉及到的方程之中, 波动方程是相对复杂的一种, 和热传导方程一样, 它含有时间变量, 而且是对时间的二阶导数. 它的解法与热传导方程的解法有相通之处. 为了让读者体会各种解题技巧, 选择了较多的习题, 有些题目略微超出了教材的内容, 比如有阻尼的弦振动, 有驱动力的弦振动, 质量不均匀的弦的振动. 本章内容也是按照空间变量的数目, 分为一维、二维、三维问题.

6.1 一维波动问题

6.1.1 无限长的弦的自由振动

自由振动的弦满足的振动方程是

$$u_{tt} = a^2 u_{xx}$$

无限长的弦没有边界, 所以只有初始条件

$$u(x, t=0) = \varphi(x), \quad u_t(x, t=0) = \psi(x)$$

问题的解是达朗贝尔公式

$$u(x,t) = \frac{1}{2}\left[\varphi(x+at) + \varphi(x-at)\right] + \frac{1}{2a}\int_{x-at}^{x+at} \psi(\xi)\mathrm{d}\xi$$

为了考察公式所表达的物理内容, 下面考虑两种典型的情况.

1. 初位移 $\varphi(x)$ 不为零, 初速 $\psi(x)$ 为零

设初位移为

$$\varphi(x) = \begin{cases} \sin\dfrac{7\pi}{l}x & \left(\dfrac{3l}{7} \leqslant x \leqslant \dfrac{4l}{7}\right) \\ 0 & (\text{其余}) \end{cases}$$

这个初始条件表示在弦中央的一段驻波, 设置这样的初始条件是为了与后面的有限长的弦的振动问题作对比, 因为这个初始条件很容易进行积分计算.

由达朗贝尔公式得

$$u(x,t) = \frac{1}{2}\left[\varphi(x+at) + \varphi(x-at)\right]$$

作图时，先画出 $\varphi(x)$ 的图形，然后再用 $x+at$ 或 $x-at$ 替代其中的 x，改变 at 的值，就画出了不同时刻 $\varphi(x+at)$, $\varphi(x-at)$ 的图形．程序中给定 $l=1$，在上面取 140 个点，再取 $0 \leqslant at \leqslant 60$．这样做可以保证波形的运动只在画出的弦的范围．

```
u(1:140)=0;        x=linspace(0,1,140);
u(61:80)=0.05*sin(pi*x(61:80)*7);
uu=u;
h=plot(x,u,'linewidth',3);
axis([0,1,-0.05,0.05]);
set(h,'EraseMode','xor')
for at=2:60
    lu(1:140)=0;          ru(1:140)=0;
    lx=[61:80]-at;        rx=[61:80]+at;
    lu(lx)=0.5*uu(61:80);     ru(rx)=0.5*uu(61:80);
    u=lu+ru;
    set(h,'XData',x,'YData',u) ;
    drawnow;
    pause(0.1)
end
```

图 6.1 是动画中几个画面．

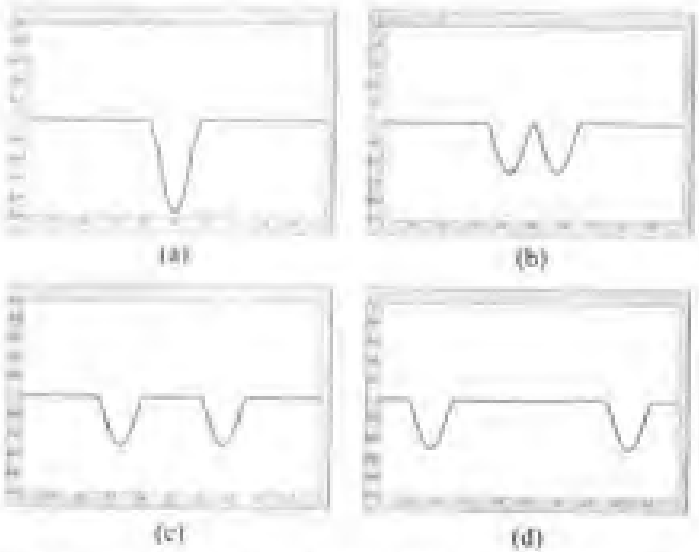


图 6.1 初位移不为零初速为零的达朗贝尔公式的图形

2. 初位移 $\varphi(x)$ 为零，初速 $\psi(x)$ 不为零

设初速度为

$$\psi(x)=\begin{cases}1 & (0\leqslant x\leqslant 1)\\ 0 & (\text{其余})\end{cases}$$

这表示初速度其值为 1，只分布在 $0\sim 1$ 。这也是一个简单的可以积分的函数，设置这个初速度也是为了与后面的有限长的弦的振动作对比.

由达朗贝尔公式得

$$u(x,t)=\frac{1}{2a}\int_{x-at}^{x+at}\psi(\xi)\mathrm{d}\xi=\frac{1}{2a}\int_{-\infty}^{x+at}\psi(\xi)\mathrm{d}\xi-\frac{1}{2a}\int_{-\infty}^{x-at}\psi(\xi)\mathrm{d}\xi$$

由已知的初始条件得

$$\frac{1}{2a}\int_{-\infty}^{x+at}\psi(\xi)\mathrm{d}\xi=\begin{cases}0 & (x+at\leqslant 0)\\ \dfrac{1}{2a}(x+at), & 0\leqslant(x+at)\leqslant 1\\ \dfrac{1}{2a}, & 1\leqslant(x+at)\end{cases}$$

$$\frac{1}{2a}\int_{-\infty}^{x-at}\psi(\xi)\mathrm{d}\xi=\begin{cases}0 & (x-at\leqslant 0)\\ \dfrac{1}{2a}(x-at), & 0\leqslant(x-at)\leqslant 1\\ \dfrac{1}{2a}, & 1\leqslant(x-at)\end{cases}$$

这两个式子分别代表将函数

$$\Psi(x)=\frac{1}{2a}\int_{-\infty}^{x}\psi(\xi)\mathrm{d}\xi$$

表示的波形向左和向右以 a 的速度移动.

利用上式，并给定 $-10\leqslant l\leqslant 10, a=1$，用以下的程序可以作出解的模拟动画. 在程序中首先定义函数 $xpat=x+at$，它在 $0\leqslant x+at\leqslant 1$ 的范围成立，再定义函数 $xmat=x-at$，它在 $0\leqslant x-at\leqslant 1$ 范围成立. 超出这些区域之外的函数值按照上面函数表达式由程序中接下来的 4 个语句来规定. 当然，这几个语句也可以用 if 形式的分支语句来表述. 不过，采用前者编程显得更简洁.

```
t=0:0.005:8;    x=-10:0.1:10;    a=1;
```

```
[X,T]=meshgrid(x,t);

xpat=X+a*T;
xpat(find(xpat<=0))=0;
xpat(find(xpat>=1))=1;

xmat=X-a*T;
xmat(find(xmat<=0))=0;
xmat(find(xmat>=1))=1;

jf=1/2/a*(xpat-xmat);
h=plot(x,jf(1,:),'linewidth',3) ;  %画动画
set(h,'erasemode','xor');
axis([-10  10  -1  1])
hold on
for j=2:length(t)
    pause(0.01)
    set(h,'ydata',jf(j,:));
    drawnow;
end
```

图 6.2 是动画中的几个画面.

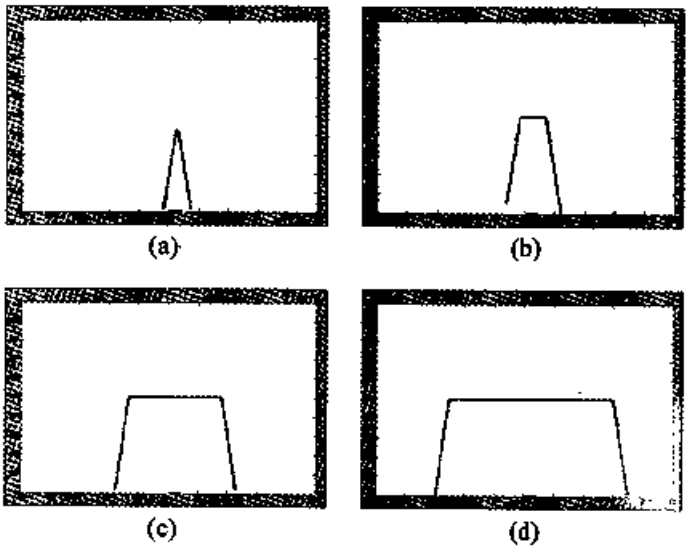


图 6.2 初位移为零初速不为零的达朗贝尔公式的图形

在程序中，如果将语句

```
jf=1/2/a*(xpat-xmat);
```

改为

```
jf=1/2/a*xpat;  或  jf=-1/2/a*xmat;
```

就可以看到如图 6.3 所示的 $\psi(x+at)$ 的波形在向左移动或 $-\psi(x-at)$ 的波形向右移动。

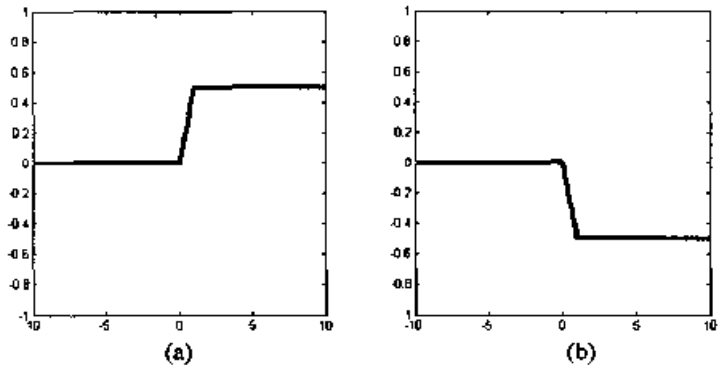


图 6.3 一维波动图形

(a) 开始时 $\psi(x+at)$ 的波形； (b) 开始时 $-\psi(x-at)$ 的波形

6.1.2 两端固定的弦振动问题之一 (初位移不为零初速为零)

两端固定的均匀弦的自由振动的定解问题是

$$\begin{cases} u_{tt}=a^2u_{xx} \\ u(x=0,t)=0, \qquad u(x=l,t)=0 \\ u(x,t=0)=\varphi(x), \quad u_t(x,t=0)=\psi(x) \end{cases}$$

它的解是

$$u(x,t)=\sum_{n=1}^{\infty}\left(A_n\cos\frac{n\pi at}{l}+B_n\sin\frac{n\pi at}{l}\right)\sin\frac{n\pi x}{l}$$

其中的系数是

$$A_n=\frac{2}{l}\int_l^0\varphi(\xi)\sin\frac{n\pi\xi}{l}\mathrm{d}\xi$$

$$B_n = \frac{2}{n\pi\alpha}\int_l^0 \psi(\xi)\sin\frac{n\pi\xi}{l}\mathrm{d}\xi$$

对有限长的弦，如果在讨论的时间范围内，边界的影响还没有到达，则产生的现象与无限长的弦是一样的．在这里，我们可以与达朗贝尔公式的解作对比．为了讨论方便，我们在本节与 6.1.3 节也分别讨论同样的两种典型的情况，即初位移不为零初速度为零以及初位移为零初速度不为零两种情况，并取相同的初始条件．

1. 初位移不为零，初速为零时的解析解

设初位移为

$$\varphi(x) = \begin{cases} \sin\dfrac{7\pi}{l}x & \left(\dfrac{3l}{7} \leqslant x \leqslant \dfrac{4l}{7}\right) \\ 0 & (\text{其余}) \end{cases}$$

这个初始条件与 6.1.1 节达朗贝尔公式中的第一种初始条件的情况相同．取 $l = 1, a = 1$, 得

$$\begin{aligned} A_n = & \frac{1}{(7-n)\pi}\left[\sin(7-n)\pi\frac{4}{7} - \sin(7-n)\pi\frac{3}{7}\right] \\ & -\frac{1}{(7+n)\pi}\left[\sin(7+n)\pi\frac{4}{7} - \sin(7+n)\pi\frac{3}{7}\right] \quad (n \neq 7) \end{aligned}$$

$$A_7 = \frac{1}{7}, \qquad B_n = 0$$

将系数代入解的表达式中，级数中的每一项都是一个驻波．这些驻波的运动在 3.1 节已经讨论过．

级数解有无穷项，为了画出级数解的图形，需要给定 n 值．为了比较傅里叶级数中求和项数的多少对解的影响，我们画出了 $n = 10, 50$ 的两种动画，所用的程序如下．程序中使用了子程序 wfun.m 来计算级数中不同 k 的求和各项，然后在主程序 jxj 中将它们加起来．

```
function  jxj
N=50
t=0:0.005:2.0;
x=0:0.001:1;
ww=wfun(N,0);
```

```
ymax=max(abs(ww));
h= plot(x,ww,'linewidth',3);
axis([ 0, 1, -ymax, ymax])
sy=[ ];
for n=2:length(t)
   ww=wfun(N,t(n));
   set(h,'ydata',ww);
   drawnow;
   sy=[sy,sum(ww)];
end

function  wtx=wfun(N,t)
x=0:0.001:1;    a=1;  wtx=0;
for I=1:N
    if I~=7
      wtx=wtx+0.05*(   (sin(pi*(7-I)*4/7)-sin(pi*(7-I)*3/7))...
          /(7-I)/pi-(sin(pi*(7+I)*4/7)-sin(pi*...
          (7+I)*3/7))/(7+I)/pi  )*cos(I*pi*a*t).*sin(I*pi*x);
     else
        wtx=wtx+0.05/7*cos(I*pi*a*t).*sin(I*pi*x);
     end
end
```

其中 N 是级数求和的项数，图 6.4 与图 6.5 分别是取前 10 项和前 50 项时所得的动画图形。

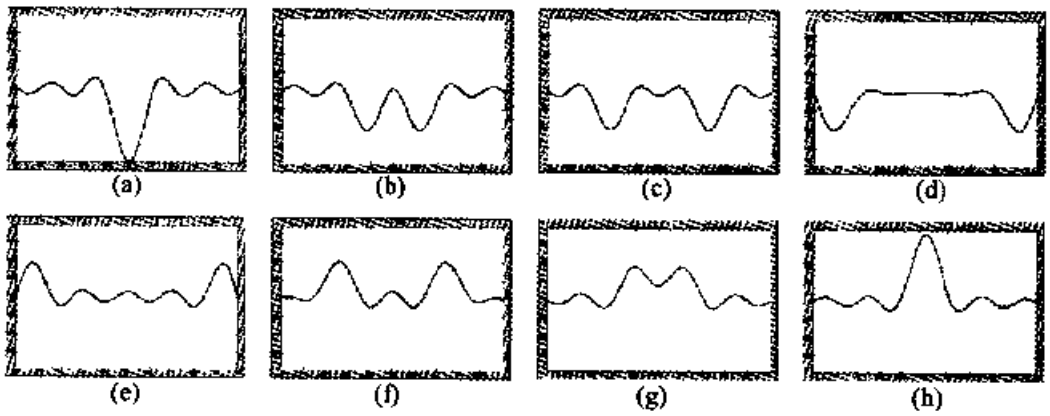


图 6.4 级数解的前 10 项的图形

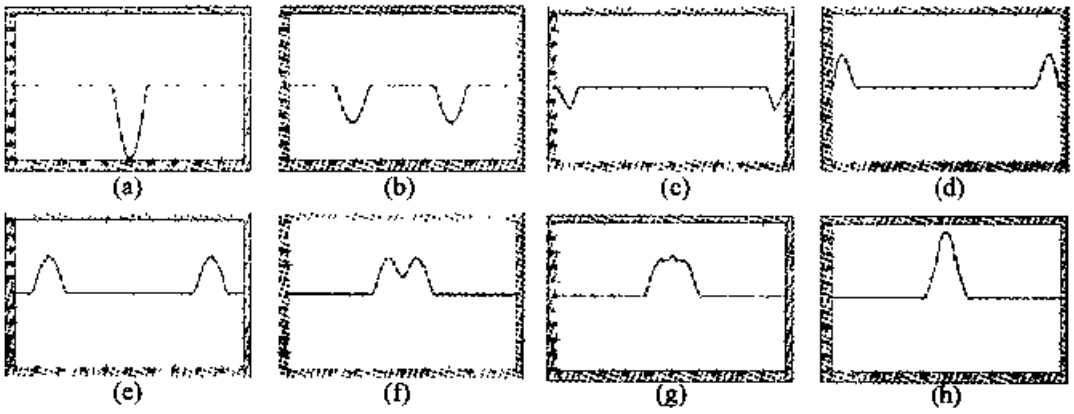


图 6.5 级数解前 50 项的图形

将它们与图 6.1 达朗贝尔公式的图形对比，可以看到：

(1) 在运动开始时，初始位移平均分为两半向正、负方向传播，在到达边界之前，相当于波在无限长弦上的传播，这一点符合达朗贝尔行波公式的描述。

(2) 到达边界以后，发生反射，产生半波损失。而达朗贝尔公式描述的是无界弦的振动，所以没有波在边界的反射现象。

(3) 从边界反射回来的两个波，相向运动。相遇以后，产生叠加，然后继续传播。

(4) 级数解要取 50 项以后才能有较高的精度。

(5) 不同频率的驻波叠加成行波。

因此加深对傅里叶级数的理解。

2. 用差分方程求出的数值解

利用差分方程同样可以求出问题的解，关于差分方程的解法可以参看 8.3. 令 $x = i\Delta x, t = j\Delta t$, 将微分方程改写成差分方程，即有

$$u_{i,j+1} = c(u_{i+1,j} + u_{i-1,j}) + 2(1-c)u_{i,j} - u_{i,j-1}$$

其中

$$c = a^2\frac{(\Delta t)^2}{(\Delta x)^2}$$

初始条件可以表示成

$$u_{i,1} = \varphi$$

$$u_{i,2} = \frac{1}{2}[c(u_{i+1,1} + u_{i-1,1}) + 2(1-c)u_{i,1}]$$

所用的程序是

```
clear
N=4010;   dx=0.0024;
dt=0.0005;    c=dt*dt/dx/dx;
x=linspace(0,1,420)';
u(1:420,1)=0;
u(181:240,1)=0.05*sin(pi*x(181:240)*7);

u(2:419,2)=u(2:419,1)+c/2*(u(3:420,1)-2*u(2:419,1)+u(1:418,1));
h=plot(x,u(:,1),'linewidth',3);
axis([0,1,-0.05,0.05]);
set(h,'EraseMode','xor','MarkerSize',18)
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
   u(2:419,3)=2*u(2:419,2)-u(2:419,1)+c*(u(3:420,2)...
       -2*u(2:419,2)+u(1:418,2));
   u(2:419,1)=u(2:419,2);
   u(2:419,2)=u(2:419,3);
end
```

图 6.6 是其中的几个动画画面。显然，它与解析解的图像是一致的。

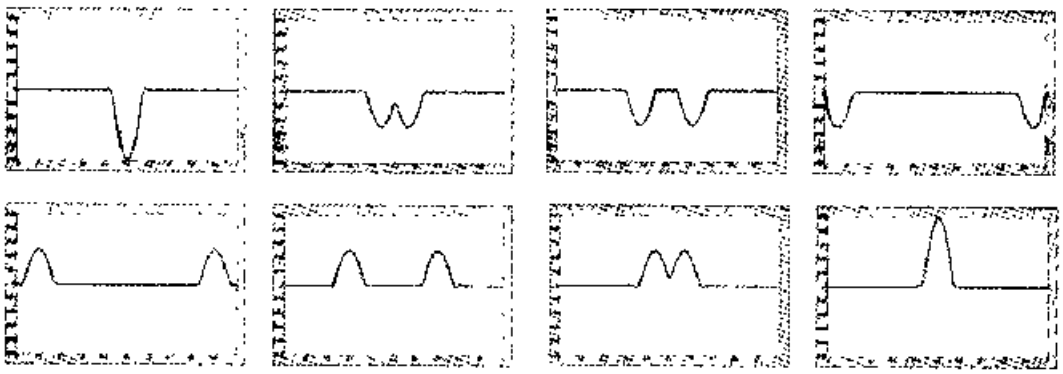


图 6.6 差分方程求得的数值解

3. 用 PDETOOL 求得的数值解

使用 PDETOOL, 可以在用户界面上操作，省去编程的过程，关键是如何利用解二维问题的工具箱来解一维问题。我们是用二维的窄带来表示一维的弦，带的侧面不受力，也就是自由的，对于两端固定的弦而言，带的两端是固定的.

在菜单 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 2$, y 轴范围为 $-0.2 \sim 0.2$，以原点为顶点画一个长为 2 宽为 0.3 的矩形，矩形的顶点为 (0, −0.15), (2, −0.15), (2, 0.15), (0, 0.15)。

按照题意，矩形的左右边界是齐次的狄里克利边界条件，可取 $h = 1, r = 0$，上下边界则取齐次的诺伊曼边界条件，即 $g = 0, q = 0$。

方程的设置是 hyperbolic 型，系数是 $c = 1, a = 0, f = 0, d = 1$。

为了有足够的精度，将初始化的网格作了两次细分。在解方程的参数设置对话框 Solve Parameters 中，将 Time 设为 0 : 0.05 : 3.6, u(t0) 设为 `exp(-(x-1).^2.*log10(2)/0.1^2)`，而 u'(t0) 设为 0，其余不变。这里由于输入方式的限制，初始条件的设置与前面有所不同，它只是一个很窄的高斯波包，而不是前面例题中设置的一段正弦驻波。

作图选项对话框 Plot Parameters 中，在 Plot type 下，选 Color, Height(3-D Plot), Animation，图 6.7 是所得的图形。

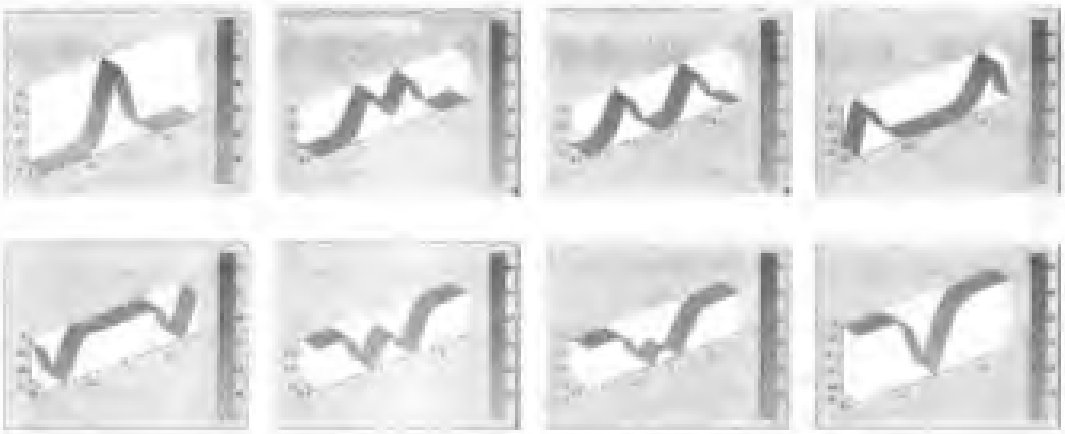


图 6.7 用偏微分方程工具箱求得的有半波损失的数值解

在上述图形中，可以很清楚地看到在固定边界上所产生的半波损失。如果在边界上是第二类齐次边界条件，那么看到的反射是没有半波损失的。如果将矩形的左右边界设为齐次的诺伊曼条件，即 $g = 0, q = 0$ 而保持其他不变，就可以得到没有半波损失情形的图形，如图 6.8 所示。

4. 任意初位移下运动

对于任意初位移下的弦的振动，一般来说，用差分方法讨论更方便。下面举一个例子，使弦在离起点 2/3 处偏离平衡位置一个小位移 h，释放它，研究其后弦的运动。这时的初位移是

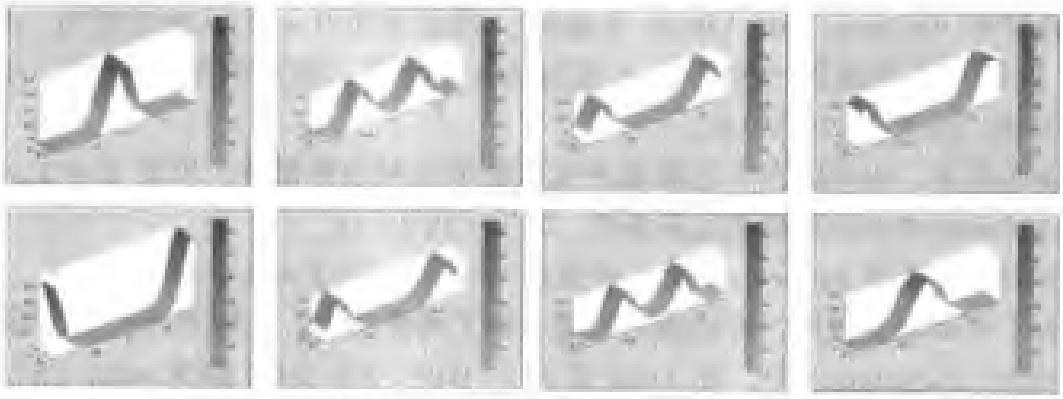


图 6.8 用偏微分方程工具箱求得的没有半波损失的数值解

$$u(x,0)=\begin{cases}\dfrac{h}{2/3}x & (0\leqslant x\leqslant 2/3)\\ h\dfrac{1-x}{1-2/3} & (2/3<x\leqslant 1)\end{cases}$$

初速仍然为零。对这种情形，我们用差分法求出它的运动图像如图 6.9 所示。

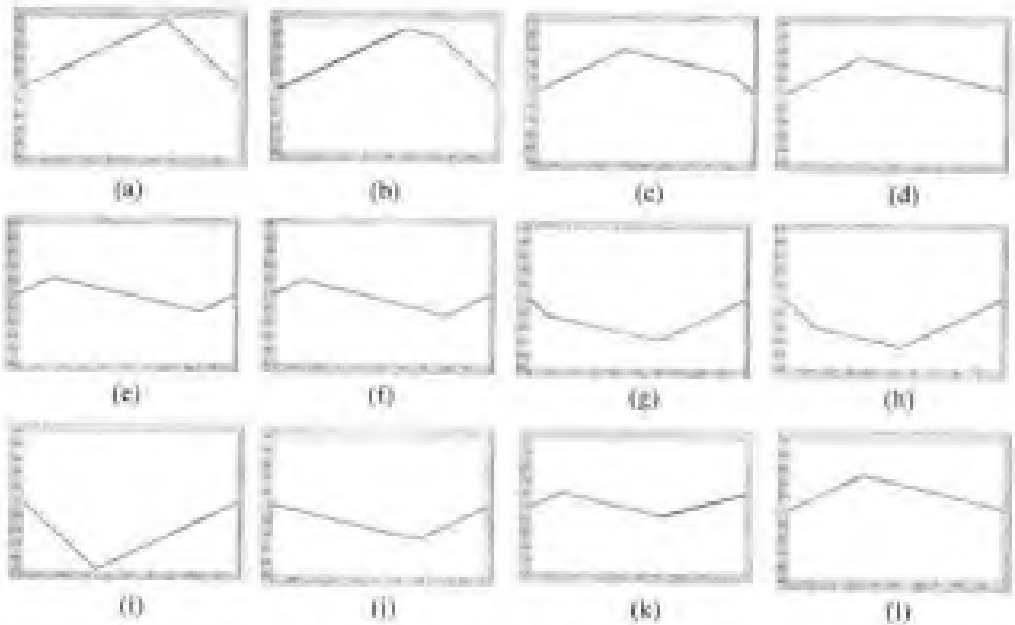


图 6.9 任意初位移形成的运动

作图所用程序如下，其中取 $a=1, l=1, h=0.05$，这里使用的方程与初始条件表示方法与 6.1.1 节相同。

```
N=4010;    dx=0.0024;
dt=0.0005;     c=dt*dt/dx/dx;
```

```
u(1:420,1)=0;      x=linspace(0,1,420)';
u(2:280,1)=0.05/280*(2:280)';
u(281:419,1)=0.05/(419-280)*(419-(281:419)');
u(2:419,2)=u(2:419,1)+c/2*(u(3:420,1)-2*u(2:419,1)+u(1:418,1));
h=plot(x,u(:,1),'linewidth',3);
axis([0,1,-0.05,0.05]);
set(h,'EraseMode','xor','MarkerSize',18)
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
   u(2:419,3)=2*u(2:419,2)-u(2:419,1)+c*(u(3:420,2)...
      -2*u(2:419,2)+u(1:418,2));
   u(2:419,1)=u(2:419,2);
   u(2:419,2)=u(2:419,3);
end
```

6.1.3 两端固定的弦振动问题之二 (初位移为零初速不为零)

设初速度为

$$\psi(x)=\begin{cases}1 & \left(\dfrac{3l}{7}\leqslant x\leqslant\dfrac{4l}{7}\right)\\ 0 & (\text{其余})\end{cases}$$

这个初始条件与 6.1.1 节达朗贝尔公式中第二种情况的初始条件相同. 它的解析解是

$$u=\sum_{n=1}^{\infty}B_n\sin\frac{n\pi a}{l}t\sin\frac{n\pi}{l}x$$

它的系数是

$$B_n=\frac{2l}{n^2\pi^2a}\left(\cos\frac{3}{7}na-\cos\frac{4}{7}na\right)$$

取 $a=1,l=1$, 用下面的程序可以作出解析解的动画. 这里也是用子函数来计算级数中的各项, 然后在主函数中调用.

```
function  psi
N=50;
t=0:0.005:2.0;          x=0:0.001:1;
ww=psi1fun1(N,0);
```

```
h= plot(x,ww,'linewidth',3);
axis([ 0, 1, -0.08, 0.08])
sy=[ ];
for n=2:length(t)
     ww=psi1fun1(N,t(n));
     set(h,'ydata',ww);
     drawnow;
     sy=[sy,sum(ww)];
end

function  wtx=psi1fun1(N,t)
x=0:0.001:1;    a=1;  wtx=0;
for k=1:N
         Bk=2/(k*k*pi*pi)*(cos(3*k*pi/7)-cos(4*k*pi/7));
         wtx=wtx+Bk*sin(k*pi*t)*sin(k*pi*x);
end
```

当求和项数为 50 时，所得的按时间顺序的动画如图 6.10 .

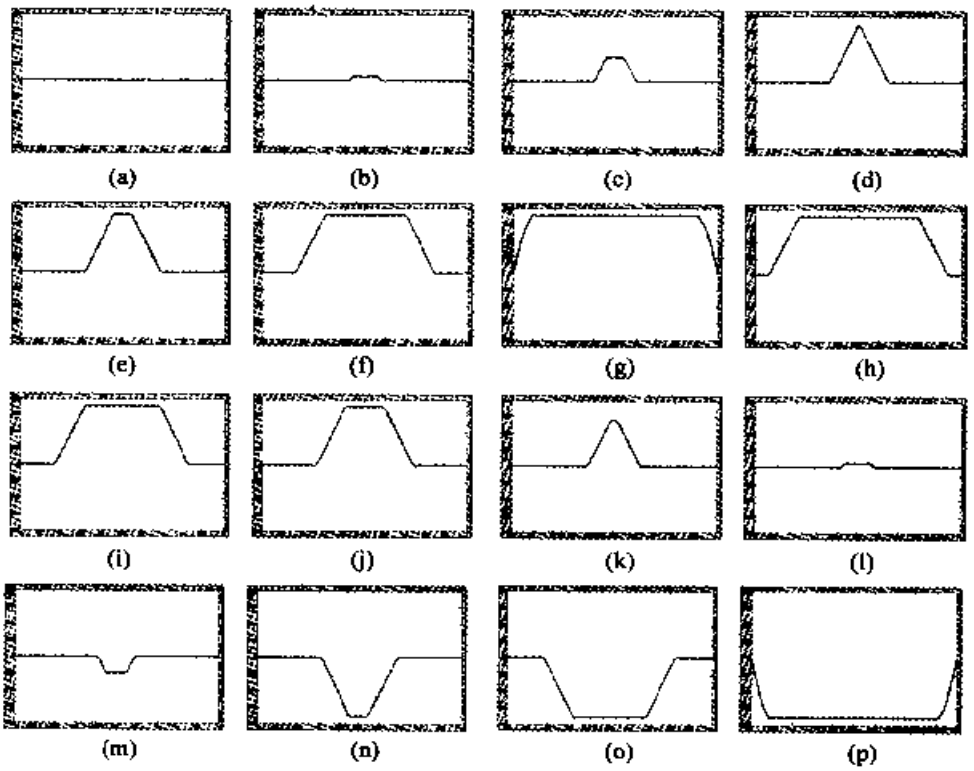


图 6.10 初位移为零初速不为零的解析解的图形

同样，我们也可以用差分方程对它求数值解，所得结果与此相同，所用的差分方程与 6.1.2 节相同，但其中初始条件应该表示成

$$u_{i,2} = \psi \Delta t$$

为了节约篇幅，我们不再给出图形，仅仅给出所用程序如下：

```
clear
N=4025; dx=0.0024;
dt=0.0005;  c=dt*dt/dx/dx;
u(1:420,1)=0;       x=linspace(0,1,420)';

u(1:420,1)=0;
u(180:240,2)=dt*0.5;
h=plot(x,u(:,1),'linewidth',5);
axis([0,1,-0.05,0.05]);
set(h,'EraseMode','xor','MarkerSize',18)
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
   %pause
   u(2:419,3)=2*u(2:419,2)-u(2:419,1)+c*(u(3:420,2)...
      -2*u(2:419,2)+u(1:418,2));
   u(2:419,1)=u(2:419,2);
   u(2:419,2)=u(2:419,3);
end
```

使用 PDETOOL 可以省去编程的麻烦，图 6.11 是用 PDETOOL 解得的结果.

使用 PDETOOL 解这个问题的方法与前面相似，也是用二维的窄带来表示一维的弦，带的侧面不受力，也就是自由的，带的两端则是固定的.

在 Options/Axes Limits 下选择 x 轴范围为 $-0.1 \sim 1$, y 轴范围为 $-0.1 \sim 0.1$. 以原点为顶点画一个长为 1 宽为 0.08 的矩形，矩形的顶点为 $(0, -0.04)$, $(1, -0.04)$, $(1, 0.04)$, $(0, 0.04)$.

按照题意，矩形的左右边界是齐次的狄里克利边界条件，可取 $h = 1, r = 0$, 上下边界则取齐次的诺伊曼边界条件，即 $g = 0, q = 0$.

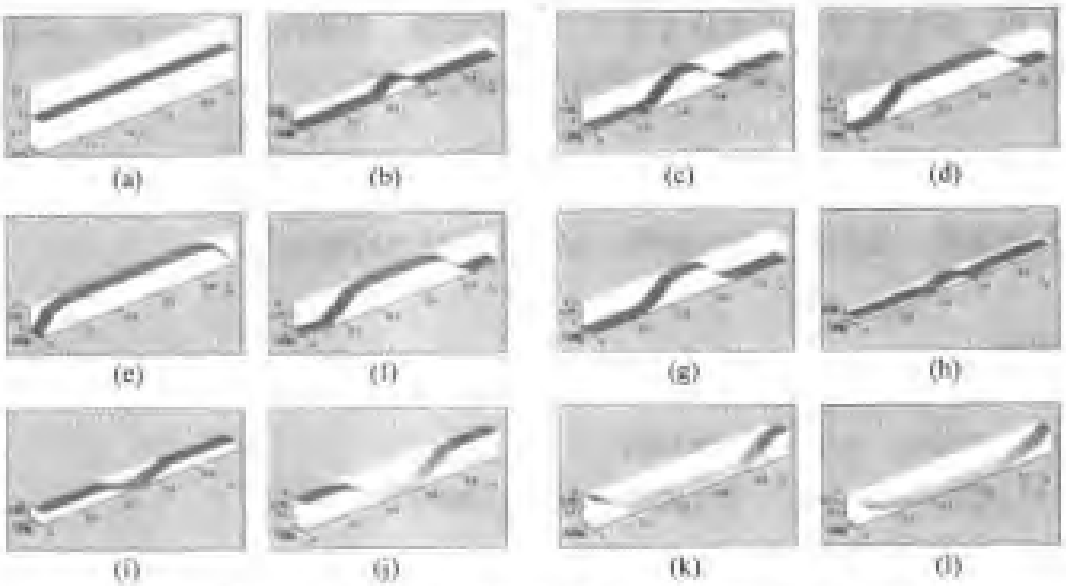


图 6.11 用工具箱解的初位移为零初速不为零的图形

方程的设置是 hyperbolic 型，系数是 $c=1, a=0, f=0, d=1$ 。

为了有足够的精度，将初始化的网格作了两次细分。在解方程的参数设置对话框 Solve Parameters 中，将 Time 设为 0 : 0.01 : 2, u(t0) 设为 0, 而 u'(t0) 设为 tanh(10000*(x−0.45))−tanh(10000*(x−0.55)), 其余不变。这里由于输入方式的限制，初始条件的设置与前面有所不同，其形状只是近似于一个矩形脉冲。

在作图选项对话框 Plot Parameters 的 Plot type 栏目下，选 Color, Height(3-D Plot), Animation, 就得到图 6.11 所示的图形。

6.1.4 两端固定的弦振动问题之三 (有阻尼)

弦振动如果遇到阻尼，振幅会不断减小，假定阻尼与速度成正比，定解问题是

$$\begin{cases} u_{tt}=c^2u_{xx}-2\lambda u_t \\ u(0,t)=0, \quad u(l,t)=0 \\ u(x,0)=\begin{cases} \dfrac{0.05}{2/3}x & (0\leqslant x\leqslant 2/3) \\ 0.05\dfrac{1-x}{1-2/3} & (2/3\leqslant x\leqslant 1) \end{cases} \\ u_t(x,0)=0 \end{cases}$$

首先将方程写成差分形式

$$\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{(\mathrm{d}t)^2}=c^2\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{(\mathrm{d}t)^2}-2\lambda\frac{u_{i,j+1}-u_{i,j-1}}{2\mathrm{d}t}$$

整理以后得

$$u_{i,j}=\frac{1}{1+\lambda \mathrm{d}t}[c(u_{i+1,j-1}+u_{i-1,j-1})+2(1-c)u_{i,j-1}-u_{i,j-2}+\lambda \mathrm{d}t u_{i,j-1}]$$

在使用上式表示初始条件时，初速为零可以表示为

$$\frac{u_{i,2}-u_{i,0}}{2\mathrm{d}t}=0$$

在程序中，初位移由 $u(i,1)$ 表示，由初速为零和初位移的表达式可以求出 $u(i,2)$，然后利用上式可以求出全部数值解．实际的程序如下，其中 $\lambda=5$ 。

```
clear
N=800; dx=0.01; dt=0.0005;
c=400*dt*dt/dx/dx;  g=5;
u(1:100,1)=0;
x=linspace(0,1,100)';
u(2:66,1)=0.05/66*(2:66)';
u(66:99,1)=0.05/34.*(100-(66:99)');
u(2:99,2)=2*u(2:99,1)+c*(u(3:100,1)-2*u(2:99,1)+u(1:98,1));
u(2:99,2)= 1/ (2+g*dt)*(u(2:99,2) + g*dt.*u(2:99,1) );
h=plot(x,u(:,1),'linewidth',5);
set(h,'EraseMode','xor')
axis([0,1,-0.05,0.05]);
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
   u(2:99,3)=2*u(2:99,2)-u(2:99,1)+c*(u(3:100,2)...
       -2*u(2:99,2)+u(1:98,2));
   u(2:99,3)= 1/ (1+g*dt)*(u(2:99,3)  + g*dt.*u(2:99,1) );
   u(:,1)=u(:,2);
   u(:,2)=u(:,3);
   pause(0.01)
end
```

下面的图 6.12 中的图 (a)、(b) 展示了运动开始时的上下最大的振幅，图 (c)、(d) 表示运动结束时上下最大的振幅．可以明显地看到，振幅已经减小了．

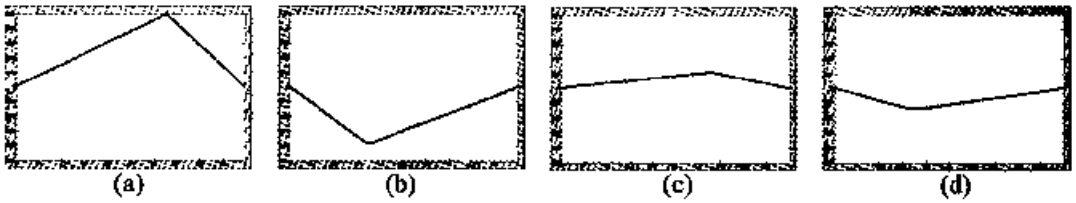


图 6.12 有阻尼的解析解的图形

6.1.5 两端固定的弦振动问题之四 (有驱动力)

如果有驱动力，即使初始条件全部为零，弦也可以发生振动．在没有阻力的情形下，随着时间的增加，振动还会越来越大．下面就是一个典型的这类定解问题．

$$\begin{cases} u_{tt} = c^2 u_{xx} + A\cos\dfrac{\pi x}{l}\sin\omega t \\ u_x(0,t) = 0, \quad u_x(l,t) = 0 \\ u(x,0) = 0, \quad u_t(x,0) = 0 \end{cases}$$

这个问题可以用傅里叶级数法、冲量定理法或格林函数法来求解，所得到的解析形式的解为

$$u(x,t) = \frac{Al}{\pi a}\frac{1}{\omega^2 - \dfrac{\pi^2 a^2}{l^2}}\left(\omega\sin\frac{\pi a t}{l} - \frac{\pi a}{l}\sin\omega t\right)\cos\frac{\pi x}{l}$$

下面的程序画出了这个弦振动的图形如图 6.13 所示．

```
l=1;    a=1;
w=2;      al=pi*a/l; A=1;
x=0:0.05:1;
m=moviein(301);
for k=0:300
    u=1/al/(w^2 - al^2)*(w*sin(al*k*0.1)...
    -al*sin(w*k*0.1))*cos(pi*x./l);
    plot(x,u);
    axis([0 1 -0.5 0.5])
    m(:,k+1)=getframe;
end
movie(m,1)
```

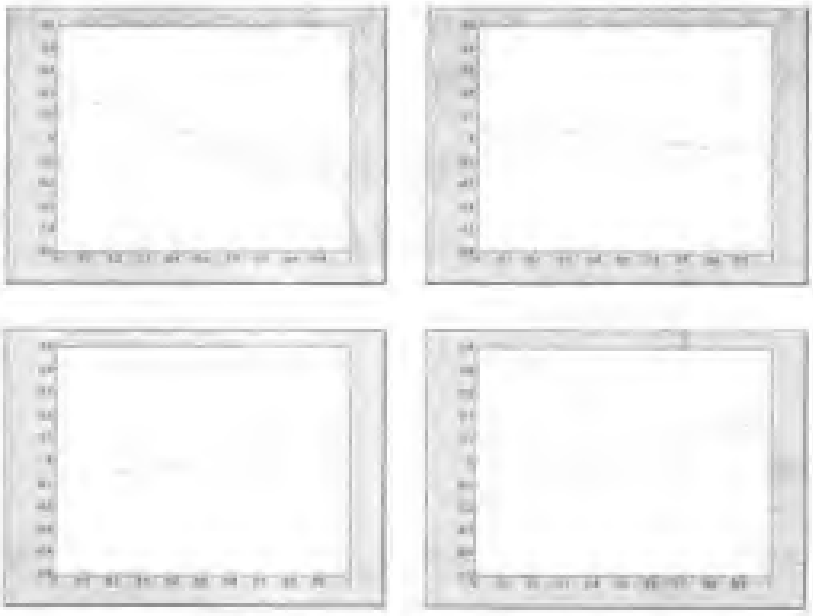


图 6.13 有驱动力的解析解的图形

用 PDETOOL 很容易画这个解的图形，画一个狭长的带状区域，四个边界都设成诺伊曼边界条件，$q=0, g=0$。方程选 hyperbolic，设成非齐次的，即 f 选项中填入 cos(pi*x/1.5).*sin(2*t)，其中 ω 取 2，其余系数为 $c=1, a=0, d=1$。

初始化的网格作了两次细分。在解方程的初始位移与初始速度都设为零，时间可以取 0:0.1:5。

作图选项是 Color,Contour,Height(3-D Plot),Animation，所得图形如图 6.14所示。

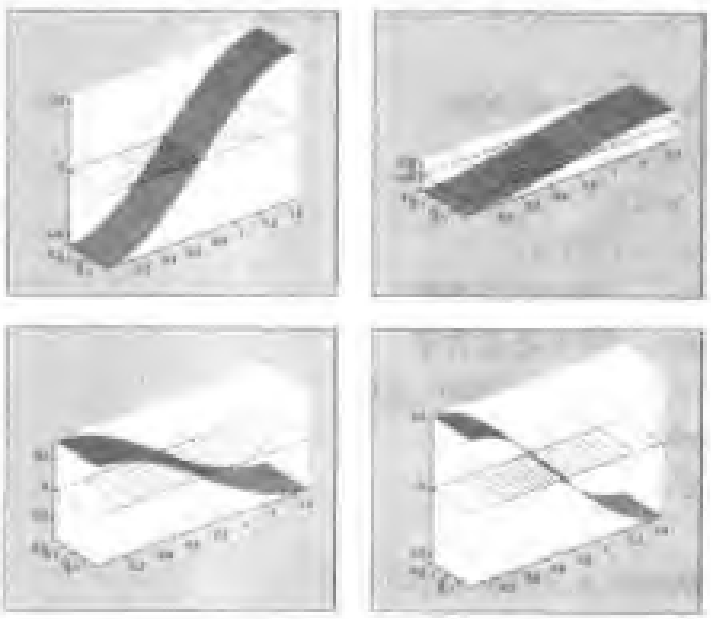


图 6.14 偏微分方程工具箱所得的图形

弦振动如果不仅有驱动力，而且有阻力，随着时间的增加，初始条件的影响会越来越小，直至形成一种稳定的振动．下面是定解问题：

$$\begin{cases} u_{tt} = c^2 u_{xx} - 2\lambda u_t + A\cos\dfrac{\pi x}{l}\sin\omega t \\ u_x(0,t) = 0, \quad u_x(l,t) = 0 \\ u(x,0) = 0 \\ u_t(x,0) = 0 \end{cases}$$

写出差分方程为

$$u_{i,j} = \frac{1}{1+\lambda \mathrm{d}t}[c(u_{i+1,j-1} + u_{i-1,j-1}) + 2(1-c)u_{i,j-1} - u_{i,j-2}$$

$$+\lambda \mathrm{d}t u_{i,j-1} + A(\mathrm{d}t)^2 \cos\frac{\pi x_i}{l}\sin\omega t_j]$$

初位移为零是 $u_{i,1} = 0$; 初速度为零表示为

$$\frac{\partial u_{i,1}}{\partial t} = \frac{u_{i,2} - u_{i,0}}{2\mathrm{d}t} = 0$$

由此得

$$u_{i,2} = \frac{c}{2+\lambda \mathrm{d}t}[u_{i+1,1} + u_{i-1,1} - 2u_{i,1}] + u_{i,1} = 0$$

编好的程序如下，这里取 $\lambda = 4, A = 200, \omega = 12$.

```
N=4000; dx=0.01; dt=0.0005;
c=400*dt*dt/dx/dx;
g=4; A=200; w=12;
x=linspace(0,1,100)';
u(1:100,1)=0;   %%初位移为零
u(1:100,2)=0;   %%初速度为零
h=plot(x,u(:,1),'linewidth',5);
set(h,'EraseMode','xor')
axis([0,1,-0.07,0.07]);
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
   u(2:99,3)=2*u(2:99,2)-u(2:99,1)+c*(u(3:100,2)...
      -2*u(2:99,2)+u(1:98,2));
```

```
    u(2:99,3)= 1/ (1+g*dt)*(u(2:99,3) + g*dt.*u(2:99,1) );
    u(2:99,3)= u(2:99,3)+1/ (1+g*dt)*(A*dt^2.*...
       cos(pi*x(2:99))*sin(w*(k+1)*dt));
    u(1,3)=u(2,3); u(100,3)=u(99,3);   %%加上边界条件
    u(:,1)=u(:,2);   %%变量轮换
    u(:,2)=u(:,3);
    pause(0.001)
end
```

图 6.15 是其中的几个画面。

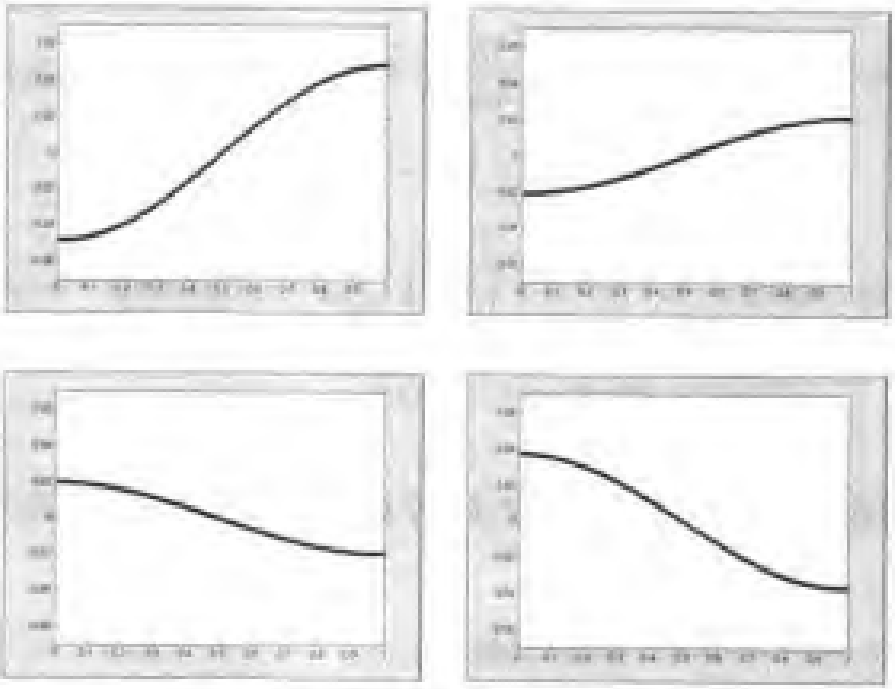


图 6.15 有驱动力的解析解的图形

6.1.6 两端固定的弦振动问题之五 (质量不均匀的弦)

如果弦的密度可变，则弦的张力也不会是常数。这时波方程的形式为

$$\frac{\partial}{\partial x}\left[T(x)\frac{\partial y(x,t)}{\partial x}\right]=\rho(x)\frac{\partial^2 y(x,t)}{\partial t^2}$$

将等式左边展开得

$$\frac{\partial T(x)}{\partial x}\frac{\partial y(x,t)}{\partial x}+T(x)\frac{\partial^2 y(x,t)}{\partial x^2}=\rho(x)\frac{\partial^2 y(x,t)}{\partial t^2}$$

当密度和张力为常数时，波在弦上的传播速度为 $c=\sqrt{T}/\rho$，对于变质量的

弦，这条件不再成立，在高密度的区域，波的运动可能更快或更慢．为了说明这一点，下面假定密度与张力都是指数型的．

$$\rho(x)=\rho_0 e^{\alpha x},\quad T(x)=T_0 e^{\alpha x}$$

将它代入方程，得到

$$\frac{\partial^2 y(x,t)}{\partial x^2}+\alpha\frac{T_0}{\rho_0}\frac{\partial y(x,t)}{\partial x}=\frac{T_0}{\rho_0}\frac{\partial^2 y(x,t)}{\partial t^2}$$

相应的差分方程是

$$\begin{aligned}y(i,2)=\ & y(i,1)+\left(\frac{\Delta t}{\Delta x}\right)^2\frac{T_0}{2\rho_0}[y(i+1,1)+y(i-1,1)-2y(i,1)]\\ &+\frac{1}{2}\frac{\alpha(\Delta t)^2T_0}{\rho_0\Delta x}[y(i+1,1)-y(i,1)]\end{aligned}$$

$$\begin{aligned}y(i,j+1)=\ & 2y(i,j)-y(i,j-1)+\left(\frac{\Delta t}{\Delta x}\right)^2\frac{T_0}{\rho_0}[y(i+1,j)+y(i-1,j)-2y(i,j)]\\ &+\frac{\alpha(\Delta t)^2T_0}{\rho_0\Delta x}[y(i+1,j)-y(i,j)]\end{aligned}$$

所用的程序如下，这里初位移是弦的中点有一个小扰动，而初速为零．

```
N=320;  dx=0.01;   a=4;
dt=0.0005;  c=300*dt*dt/dx/dx;
u(1:100,1)=0;       x=linspace(0,1,100)';
u(2:50,1)=0.06/50*(2:50)';
u(50:99,1)=0.06/50.*(100-(50:99)');
u(2:99,2)=  u(2:99,1)+c/2*(u(3:100,1)-2*u(2:99,1)+u(1:98,1)) +...
   c/2*a*dx*(u(3:100,1)-u(2:99,1));
plot([0,1],[0,0],'r')
hold on
h=plot(x,u(:,1),'linewidth',3);
set(h,'EraseMode','xor')
axis([0,1,-0.1,0.1]);
for k=2:N
   set(h,'XData',x,'YData',u(:,2)) ;
   drawnow;
```

```
    u(2:99,3)=2*u(2:99,2)-u(2:99,1)+c*(u(3:100,2)...
              -2*u(2:99,2)+u(1:98,2));
    u(2:99,3)= u(2:99,3)  + c*a*dx*(u(3:100,2)-u(2:99,2)) ;
    u(:,1)=u(:,2);
    u(:,2)=u(:,3);
    pause(0.1)
end
```

图 6.16 是其中的几个画面. 由于各处的质量密度不同, 所以运动的状态差异很大, 可以看到, 开始时, 波很快的向右边的密度大的区域运动, 但是它的振幅却是衰减的, 因为这个区域的弦较重.

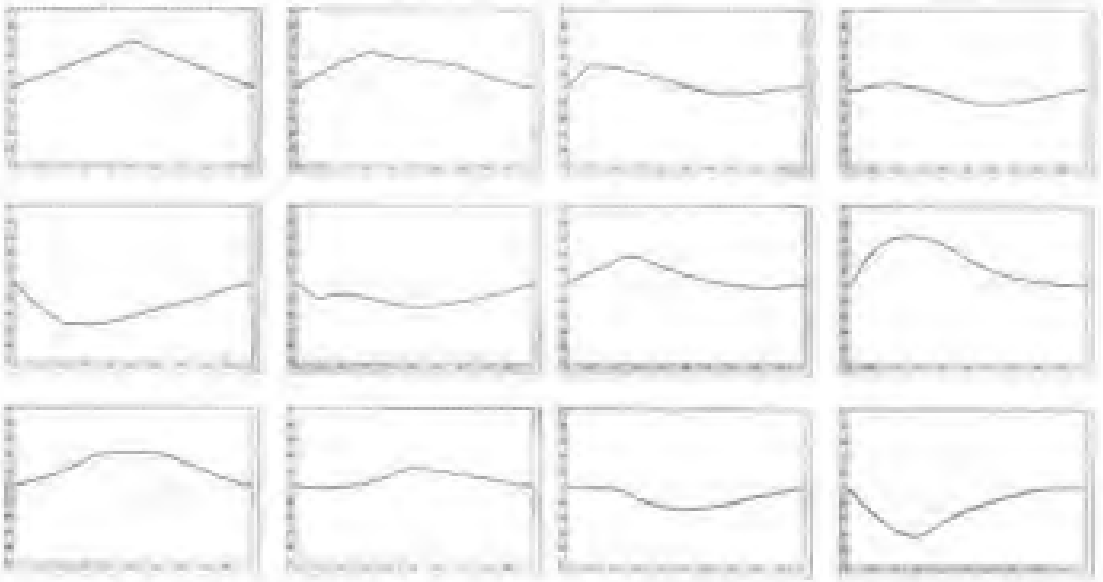


图 6.16 变质量的弦的运动

6.1.7 非齐次边界条件下弦的振动

弦的 $x=0$ 端固定, $x=l$ 端受迫作谐振动 $A\sin\omega t$, 弦的初位移和初始速度都是零, 求弦的振动 * .

定解问题是

$$\begin{cases} u_{tt}-a^2u_{xx}=0 \\ u(x=0)=0, \qquad u(x=l)=A\sin\omega t \\ u(t=0)=0, \qquad u_t(t=0)=0 \end{cases}$$

*见参考文献 [1], 217 页.

这个问题的解是

$$u = A\frac{\sin(\omega x/a)}{\sin(\omega l/a)}\sin\omega t + \frac{2A\omega}{al}\sum_{n=1}^{\infty}\frac{(-1)^{n+1}}{\omega^2/a^2 - n^2\pi^2/l^2}\sin\frac{n\pi at}{l}\sin\frac{n\pi x}{l}$$

可以用动画来表现这个解. 图 6.17 是动画中的一幅画面. 可以看出, 固定端一直保持不动. 而作谐振动的端使弦也产生了一种在传播的振动. 这相当于将一根绳的一端固定, 而周期性的摆动另一端时, 在绳上所产生的运动.

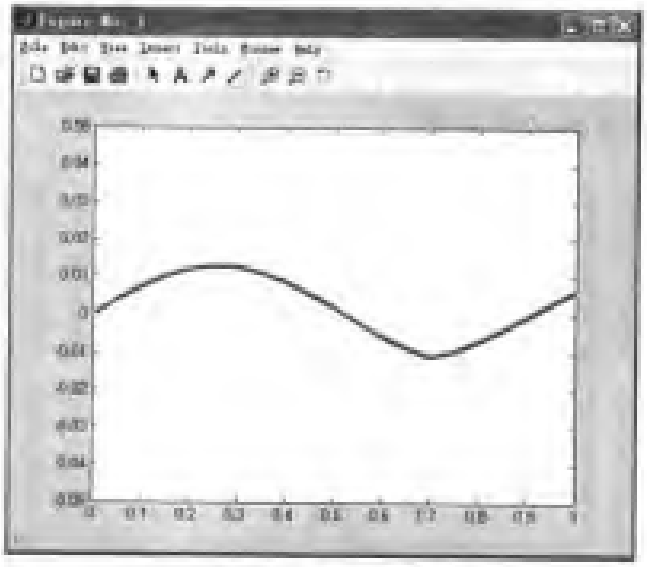


图 6.17 解析解的动画图形

也可以用瀑布图形来表示这个过程, 如图 6.18 所示. 由于数据过多, 程序是挑选一部分数据来作图.

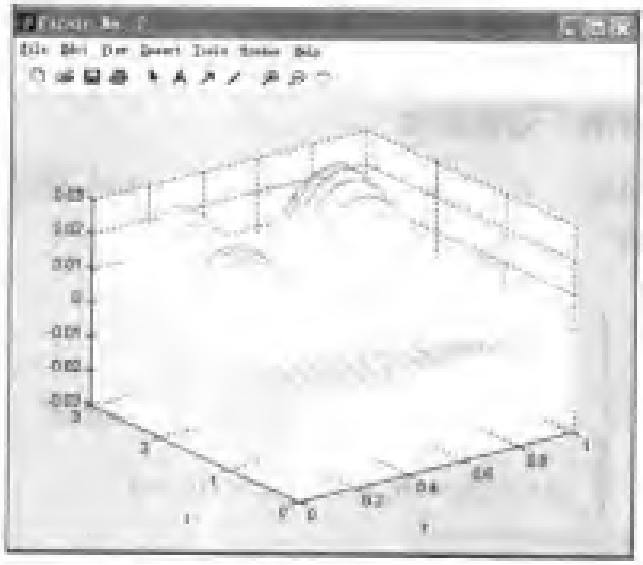


图 6.18 解析解的瀑布图形

实际的程序如下，程序中取 $a=1, l=1, A=0.01, w=6$. 这样选择的结果是为了使振动在弦中传播速度较慢，而振动源的振动速度较快但振动幅度不大. 所以能看到较好的演示效果.

```
clear
a=1;     l=1;
A=0.01;     w=6;
x=0:0.05:1;
t=0:0.001:4.3;
[X,T]=meshgrid(x,t);
u0=A*sin(w*X./a).*sin(w.*T)/sin(w*l/a);
u=0;
for n=1:100;
   uu=(-1)^(n+1)*sin(n*pi*X/l).*sin(n*pi*a*T/l)...
      /(w*w/a/a-n*n*pi*pi/l/l);
   u=u+uu;
end
u=u0+2*A*w/a/l.*u;
figure(1)
axis([0 1 -0.05  0.05])
h=plot(x,u(1,:),'linewidth',3);
set(h,'erasemode','xor');
for j=2:length(t);
    set(h,'ydata',u(j,:));
    axis([0 1 -0.05 0.05])
    drawnow
end
figure(2)
waterfall(X(1:50:3000,:),T(1:50:3000,:),u(1:50:3000,:))
xlabel('x')
ylabel('t')
```

再用偏微分方程工具箱求解这个问题. 我们用二维的图形来表示一维的弦，让弦的侧面不受力，左端固定，右端作受迫振动.

在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 1$, y 轴范围为 $0 \sim 1$. 以原

点为顶点画一个长为 1 宽为 0.4 的矩形，矩形的顶点为 (0, 0), (1, 0), (1, 0.4), (0, 0.4)。

按照题意，矩形的左边界是齐次的狄里克利边界条件，可取 $h=1, r=0$，右边界是非齐次的狄里克利边界条件，可取 h=1, r=0.01*sin(6*t)，上下边界则取齐次的诺伊曼边界条件，即 $g=0, q=0$。

方程的设置是 hyperbolic 型，系数是 $c=1, a=0, f=0, d=1$。

为了有足够的精度，初始化的网格要再作两次细分。

在解方程的参数设置对话框 Solve Parameters 中，将 Time 设为 0 : 0.1 : 0.5, u(t0) 和 u'(t0) 都设为 0，其余不变。

而作图选项对话框 Plot Parameters 中，各项选择如下：在 Plot type 下，选 Color 和 Height(3-D Plot)，在 Property 下，对应的位置中都选 User entry，在 User entry 下，再在相应的位置都输入 10 * u，这样做的目的是，弦的振幅太小，为了达到更好的显示效果，所以将振幅放大 10 倍来画图。所得的图形如图 6.19 所示。

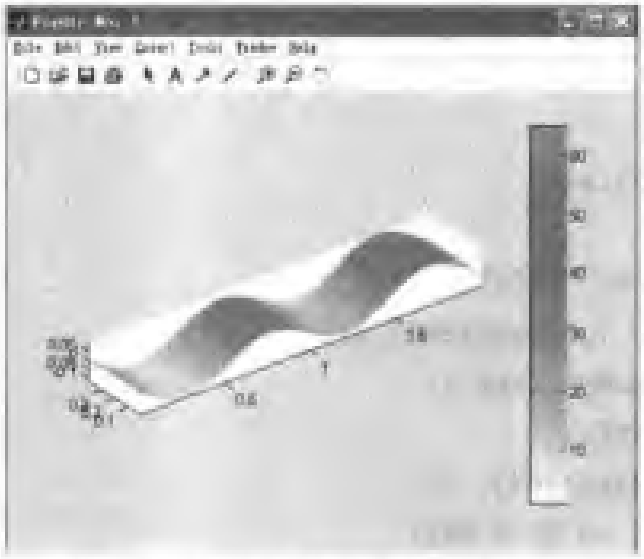


图 6.19 用工具箱所得的数值解的图形

6.1.8 杆的纵振动

长为 l 的杆，两端左右，初始位移为 $c_0 x/l$，初始速度为零，研究它的运动。定解问题是

$$\begin{cases} u_{tt} - a^2 u_{xx} = 0 \\ u_x(x=0) = 0, \qquad u_x(x=l) = 0 \\ u(t=0) = c_0 x/l, \quad u_t(t=0) = 0 \end{cases}$$

问题的解是

$$u(x,t)=A_0+\sum_{n=1}^{\infty}A_n\cos\frac{n\pi a}{l}t\cos\frac{n\pi}{l}x$$

其中

$$A_0=\frac{C_0}{2}$$

$$A_n=\frac{-4C_0}{\pi^2(2n+1)^2}\quad(n=0,1,2,\cdots)$$

取 $C_0=0.05, a=1, l=1$, 可以用下面的程序画出它的动画. 程序中使用了子程序 gzdfun.m 来计算级数中求和的项。

```
function  gzd   %杆振动的解析解
figure(2)
N=50;
t=0:0.005:2.0;          x=0:0.002:1;
ww=gzdfun(N,0);
subplot(2,1,1)
h1= plot(x,ww,'linewidth',3);
set(h1,'erasemode','xor');
axis([ 0, 1, 0, 0.05])
xx=1:10:length(x);        yy=0*xx;
subplot(2,1,2)
h2=plot(x(xx), yy, 'r.','marker','.','markersize',25);
set(h2,'erasemode','xor');
axis([ 0, 1.05, -0.1, 0.1])
for n=2:length(t)
     ww=gzdfun(N,t(n));
     set(h1,'ydata',ww);
     uu=ww(xx)+x(xx);
     set(h2,'xdata',uu);
     drawnow;
     pause(0.02)
end

%-------------------------------------------------
```

```
function  wtx=gzdfun(N,t)
x=0:0.002:1;     a=1;  wtx=1/2*0.05;
for k=1:2:N
        Bk=-4/(k*k*pi*pi)*cos(k*pi*t)*cos(k*pi*x)*0.05;
        wtx=wtx+Bk;
end
```

所得到的动画如下，每幅动画的上半部用 y 轴表示位移，下半部则是用质点在水平方向的移动来表示位移. 在初始时刻，杆的左端在原点，右端有一个小位移，杆的各点的初速度为零. 这就是图 6.20 左上角的图所表现的图形. 然后杆上各质点依次在原来的平衡位置振动.

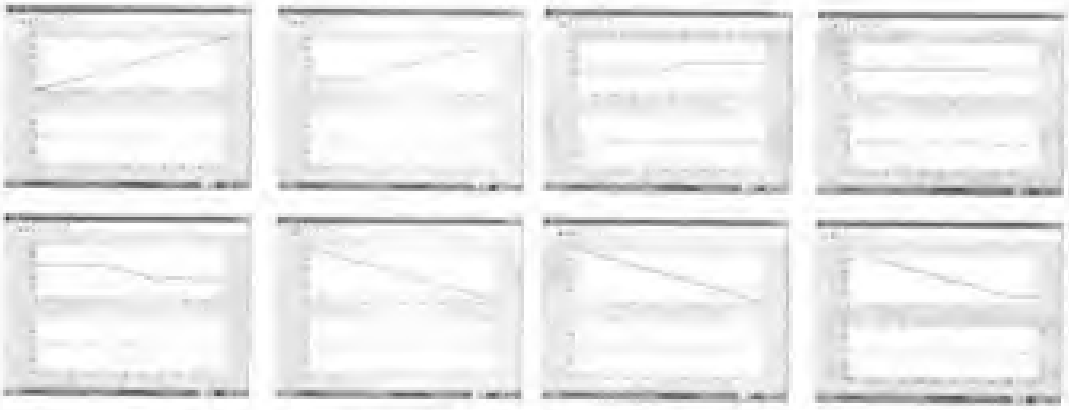


图 6.20 杆的振动的解析解

再用偏微分方程工具箱求解这个问题. 我们用二维的图形来表示一维的杆，杆的侧面不受力，左端固定，右端作受迫振动.

在 Options/Axes Limits 下选择 x 轴范围为 $0 \sim 2$, y 轴范围为 $0 \sim 0.1$ ，以原点为顶点画一个长为 2 宽为 0.05 的矩形，矩形的顶点为 (0, 0.01), (2, 0.01), (2, 0.06), (0, 0.06) .

按照题意，矩形的四个边界都取齐次的诺伊曼边界条件，即 $g = 0, q = 0$.

方程的设置是 hyperbolic 型，系数是 $c = 1, a = 0, f = 0, d = 1$.

为了有足够的精度，将初始化的网格作了两次细分. 在解方程的参数设置对话框 Solve Parameters 中，将 Time 设为 0 : 0.01 : 2, u(t0) 为 0.025. $* x$, u'(t0) 设为 0, 其余不变.

画作图选项对话框 Plot Parameters 中，各项选择如下：在 Plot type 下，选 Color,Height(3-D Plot),Animation . 在 Property 下，对应的位置中都选 User entry, 在 User entry 下，再在相应的位置都输入 10 * u, 这样做的目的是，弦的振幅太小，为了达到更好的显示效果，所以将振幅放大 10 倍来画图. 图 6.21 是其中的一

幅画面.

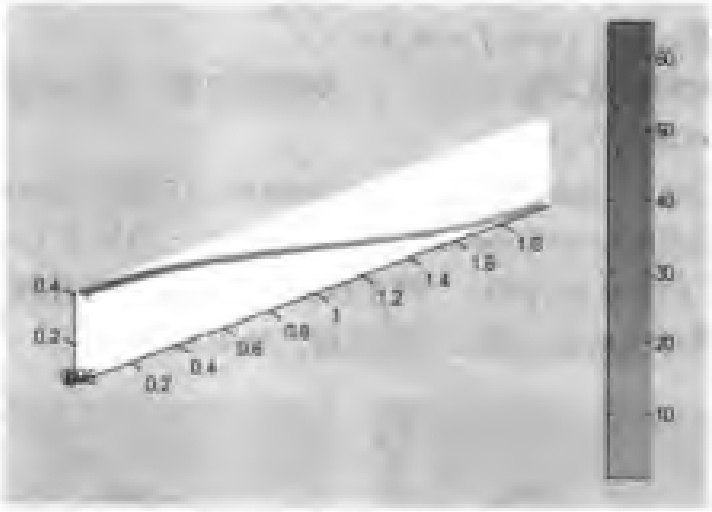


图 6.21 用工具箱所得的数值解的图形

6.2 二维波动问题

二维问题用偏微分方程工具箱求解十分方便，所有操作基本相同，可以很快完成，使用差分方程编程则比较麻烦，本书的目的是作计算机模拟，所以二维问题用偏微分方程工具箱求解也就够了. MATLAB 带有两个二维波动方程的例子，在指令窗中输入 pdedemo6 演示的是矩形膜的振动，在指令窗中输入 vibes 演示的是 L 形膜的振动. 下面再举一个矩形膜振动的例子，圆形膜和环形膜的振动可以用类似的操作完成.

6.2.1 矩形膜的振动

求矩形薄膜的横向振动*，定解问题为

$$\begin{cases} u_{tt} = a^2(u_{xx} + u_{yy}) & (0 \leqslant x \leqslant c, 0 \leqslant y \leqslant b, 0 \leqslant t) \\ u(x=0,y,t) = 0, & u(x=a,y,t) = 0 \\ u(x,y=0,t) = 0, & u(x,y=b,t) = 0 \\ u(x,y,t=0) = Ax(x-c)\sin\dfrac{\pi y}{b}, & u_t(x,y,t=0) = 0 \end{cases}$$

为了求数值解，任取 $a=1, b=1, c=2, A=1$，求解区域可取 $0 \leqslant x \leqslant 2$，$0 \leqslant y \leqslant 1$，边界条件为四边固定，所以不必改变，方程取 Hyperbolic(双曲型)，方程的

*见参考文献 [5], 113 页.

形式为

$$d \cdot u_{tt} - \nabla \cdot (c\nabla u) + au = f$$

在系数对话框中取 $c = 1, a = 0, f = 0, d = 1$.

方程求解的时间范围可取 0 : 0.1 : 4 , 初速则取零. 而在初始位移一栏中输入 `0.1* x.*(x-1).*sin(pi*y)` .

作图时选择 Color,Height (3-D plot), Animation, 在选项 Options 中, Animation rate 一栏中填入 5,Number of repeats 一栏中填入 2 . 另外在 Colormap 一栏中选 pink, 在动画时间 (Time for plot) 一栏中填入 2 . 最后单击按钮 Plot 即可. 可以看到, 是初始位移形成的波在膜上来回传播, 如图 6.22 所示.

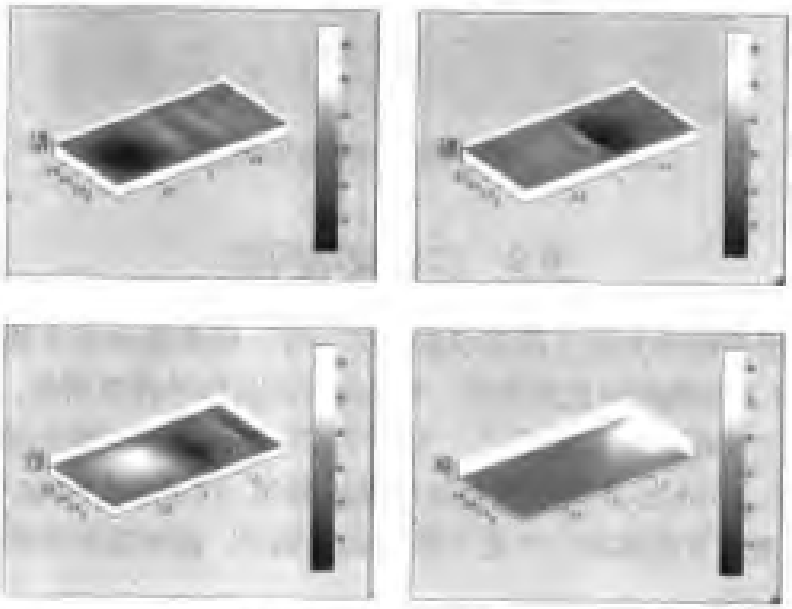


图 6.22 矩形膜的振动

这个问题是有解析解的, 解法如下. 将定解问题分离变量得三个方程为

$$\begin{cases} X''(x) + (\lambda - \mu)X(x) = 0 \\ X(x=0) = 0, \qquad\qquad X(x=c) = 0 \end{cases}$$

$$\begin{cases} Y''(y) + \mu Y(y) = 0 \\ Y(y=0) = 0, \qquad Y(y=b) = 0 \end{cases}$$

$$T''(t) + \lambda a^2 T = 0$$

分别求解得

$$Y_m(y) = \sin\frac{m\pi y}{b}, \quad \mu_m = \left(\frac{m\pi}{b}\right)^2, \quad m = 1, 2, 3, \cdots$$

$$X_m(x)=\sin\frac{n\pi x}{c},\quad \lambda_{mn}-\mu_m=\left(\frac{n\pi}{c}\right)^2,\quad n=1,2,3,\cdots$$

$$T_{mn}(t)=C_{mn}\cos\left(\sqrt{\lambda_{mn}}at\right)+D_{mn}\sin\left(\sqrt{\lambda_{mn}}at\right)$$

定解问题的解的形式为

$$\begin{aligned}u(x,y,t)&=\sum_{m,n=1}^{\infty}T_{mn}(t)X_m(x)Y_m(y)\\&=\sum_{m,n=1}^{\infty}\left(C_{mn}\cos\left(\sqrt{\lambda_{mn}}at\right)+D_{mn}\sin\left(\sqrt{\lambda_{mn}}at\right)\right)\sin\frac{n\pi x}{c}\sin\frac{m\pi y}{b}\end{aligned}$$

由初始条件定出

$$D_{mn}=0\quad(m,n=1,2,3,\cdots)$$

$$C_{mn}=0\quad(m=2,3,\cdots)$$

$$C_{1n}=\begin{cases}\dfrac{-8Ac^2}{(n\pi)^3}, & n=2k+1,k=0,1,2,\cdots\\ 0, & n=2k,k=1,2,3,\cdots\end{cases}$$

最终的解为

$$\begin{aligned}u(x,y,t)=&\sum_{k=0}^{\infty}\frac{-8Ac^2}{(2k+1)^3\pi^3}\cos\left(\sqrt{\left(\frac{\pi}{b}\right)^2+\left(\frac{(2k+1)\pi}{c}\right)^2}at\right)\cdot\\&\sin\left(\frac{(2k+1)\pi}{c}x\right)\sin\frac{\pi y}{b}\end{aligned}$$

从这个解中可以看出，矩形膜的本征函数和本征值分别是

$$X_m(x)Y_m(y)=\sin\frac{n\pi x}{c}\sin\frac{m\pi y}{b}$$

$$\lambda_{mn}=\left(\frac{m\pi}{b}\right)^2+\left(\frac{n\pi}{c}\right)^2$$

这些本征模的运动在 3.2.1 已经讨论过.

6.2.2 圆膜的振动

边缘固定，半径为 ρ_0 的圆形膜，初始形状是旋转抛物面 $u(t=0)=u_0(1-\rho^2/\rho_0^2)$，初始速度为零，求膜的振动情况*。

定解问题是

$$\begin{cases} u_{tt}=a^2\Delta u \\ u(\rho=\rho_0)=0 \\ u(t=0)=u_0(1-\rho^2/\rho_0^2) \end{cases}$$

它的解是

$$u(x,t)=8u_0\sum_{n=1}^{\infty}\frac{1}{[x_n^{(0)}]^3J_1(x_n^{(0)})}J_0\left(\frac{x_n^{(0)}}{\rho_0}\rho\right)\cos\frac{x_n^{(0)}}{\rho_0}at$$

下面直接用偏微分方程工具箱来解这个问题。

计算中取 $a=1,\rho_0=1,u_0=0.4$。先画一个半径为 1 圆心在原点的圆，边界条件为周边固定，所以不必改变默认设置。为了有足够的计算精度，网格初始化以后，至少要细分一次。

方程取 Hyperbolic(双曲型)，方程的系数取 $c=1,a=0,f=0,d=1$。

方程求解的时间范围可取 0 : 0.1 : 6，初速取零而在初始位移一栏中输入 `0.4*(1-x.^2-y.^2)`。

作图时选择 Contour,Height (3–D plot), Animation, 最后单击按钮 Plot 即可。所得图形如图 6.23 所示。

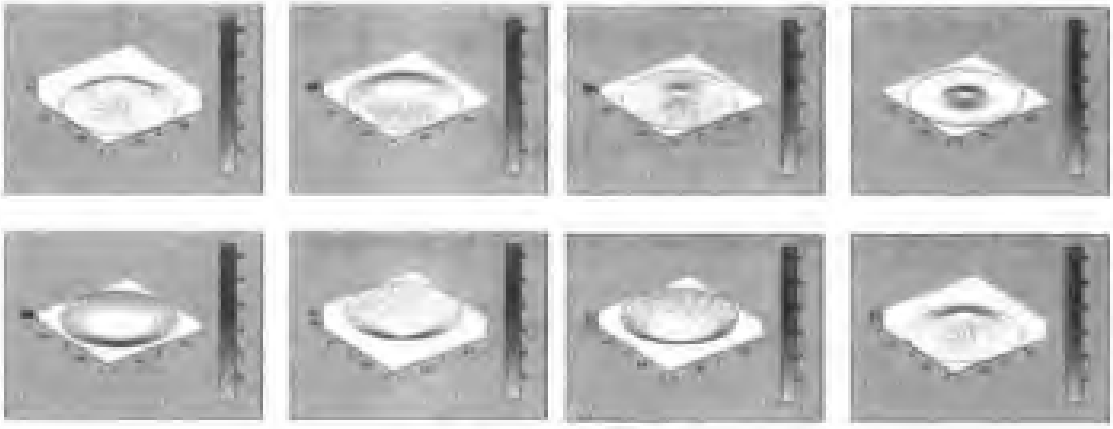


图 6.23 圆形膜的振动

仿照本题的做法，读者很容易用 PDETOOL 解出下面这题：

*见参考文献 [1], 346 页。

习题：边缘固定，半径为 ρ_0 的圆形膜，初始位移与初始速度为零，每单位质量上的作用力是 $f = A\sin\omega t$, 求膜的振动情况。

6.3 三维振动问题

6.3.1 柱体内的振动

匀质圆柱，半径为 ρ_0, 高 L, 上下底面固定，侧面自由，初始位移为零，初始速度为 $u_0\rho^2 z$, 求柱体内各处的振动情况 * 。

定解问题是

$$\begin{cases} u_{tt} = a^2\Delta u \\ u_\rho(\rho = \rho_0) = 0 \\ u(z=0) = 0, \quad u(z=L) = 0 \\ u(t=0) = 0, \quad u_t(t=0) = u_0\rho^2 z \end{cases}$$

问题的解析解是

$$u = \sum_{n=0}^{\infty}\sum_{p=1}^{\infty} \frac{(-1)^{p+1}8u_0 L\rho_0^2}{ap\pi(x_n^{(1)})^2 J_0(x_n^{(1)})} \frac{1}{\sqrt{\left(\frac{x_n^{(1)}}{\rho_0}\right)^2 + \left(\frac{p\pi}{L}\right)^2}} \cdot \sin\sqrt{\left(\frac{x_n^{(1)}}{\rho_0}\right)^2 + \left(\frac{p\pi}{L}\right)^2} at \cdot \sin\frac{p\pi}{L}z \cdot J_0\left(\frac{x_n^{(1)}}{\rho_0}\rho\right)$$

下面用 PDETOOL 求数值解. 以 x 表示柱体高度， y 表示柱体宽度，画一个柱体的纵切面. 为了较好的演示效果，求解的区域是 $0 \leqslant x \leqslant 1.5, -0.5 \leqslant y \leqslant 0.5$.

在柱体的上下底，取狄里克利边界条件即在对话框中取 $h = 1, r = 0$, 在侧边界上取诺伊曼边界条件，即在对话框中取为 $g = 0, q = 0$. 方程取 Hyperbolic(双曲型), 在对话框中取 $c = 1, a = 0, f = 0, d = 1$.

方程求解的时间范围可取 $0 : 0.01 : 5$, 在初始位移取零，初速取 $y.*y.*x$, 为了获得更高的精度，在将区域划分网格以后，要作二次细分.

在作图对话框中，选择 Color,Hight(3-D plot) 和 Animation, 此外为了看得更清，我们把函数值适当放大，做法是在 Property 栏目下与 High(3-D plot) 对齐的

*见参考文献 [1], 347 页.

位置，选择 user entry，再在 User entry 栏目下，与它对齐的位置的空白栏中填入 3.4u，最后单击按钮 Plot 即可。

这里画的过柱轴的截面的运动，不难想像，面上的每个点都应该围绕平衡位置来回振动，这正是动画图 6.24 所表现的图像。

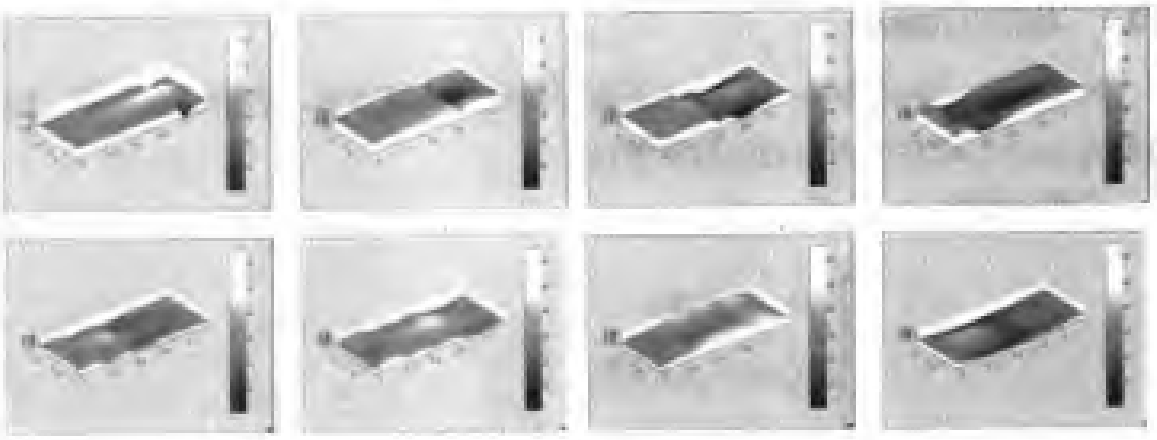


图 6.24 柱体内的振动

6.3.2 柱体外的振动问题之一

半径为 ρ_0 的长圆柱面，其径向速度分布为 $v = v_0 \cos \omega t$，试求解这个长圆柱面在空气中辐射出去的声场的中的速度势。设 ρ_0 远小于声波的波长*。

所求的速度势满足二维波动方程，取平面极坐标系，极点在柱轴上，则定解问题是

$$\begin{cases} u_{tt} = a^2 \Delta u \\ u_\rho(\rho = \rho_0, t) = v_0 \cos \omega t \end{cases}$$

问题的解析解是下式的实部：

$$u = -\mathrm{i} \frac{\pi v_0 \rho_0}{2} \mathrm{H}_0^{(1)} \left(\frac{\omega}{a} \rho \right) \mathrm{e}^{-\mathrm{i}\omega t}$$

在远场区即 ρ 大的区域，渐近解为

$$u = v_0 \rho_0 \sqrt{\frac{\pi a}{2\omega \rho}} \sin \left(\frac{\omega}{a} \rho - \omega t - \frac{\pi}{4} \right)$$

这是振幅按 $1/\sqrt{\rho}$ 减小的柱面波。

*见参考文献 [1]，345 页。

下面用 PDETOOL 求数值解. 画两个同心圆 C_1 和 C_2, 圆心都在原点, 半径分别为 1 和 0.1 两个值. C_2 代表柱体的横切面, C_1 表示求解区域的外边界. 问题的求解区域是由这两个同心圆组成的环形区域 $C_1 - C_2$.

在外边界 C_1 上, 取狄里克利边界条件 $u = 0$ 表示无穷远函数为零, 即 $h = 1, r = 0$, 在内边界 C_2 上取纽曼边界条件, 为 $g = \cos 6t, q = 0$. 方程取 Hyperbolic (双曲型), 在对话框中取 $c = 1, a = 0, f = 0, d = 1$.

方程求解的时间范围可取 $0:5$, 在初始位移和初速都取零.

作图时选择 contour 和 Animation, 最后单击按钮 Plot 即可. 图 6.25 是画出的函数在某个时刻的等值线, 在动画中, 它像水波一样一圈一圈往外传播.

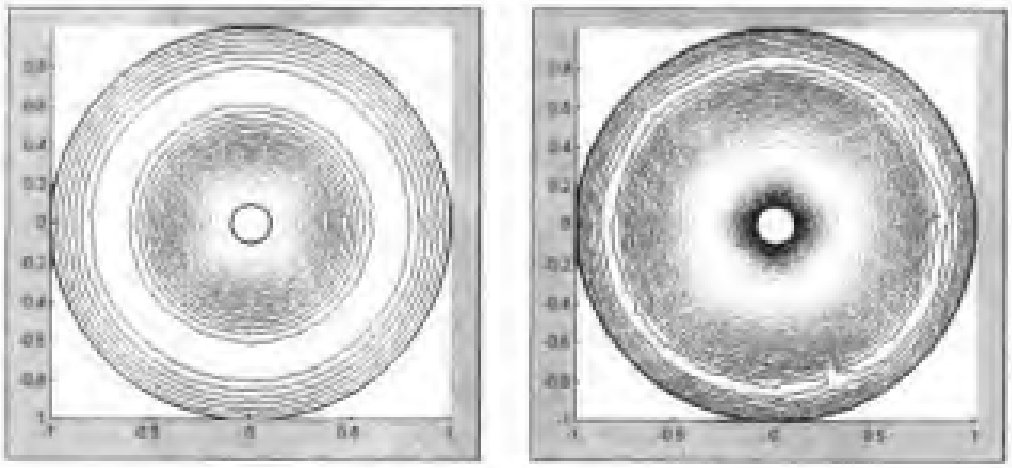


图 6.25 向外传播的柱面波

6.3.3 柱体外的振动问题之二

半径为 ρ_0 的长圆柱面, 其径向速度分布为 $v = v_0 \cos\varphi\cos\omega t$, 试求解这个长圆柱面在空气中辐射出去的声场的中的速度势. 设 ρ_0 远小于声波的波长*.

所求的速度势满足二维波动方程. 取平面极坐标系, 极点在柱轴上, 则定解问题是

$$\begin{cases} u_{tt} = a^2 \Delta u \\ u_\rho(\rho = \rho_0, t) = v_0 \cos\varphi \cos\omega t \end{cases}$$

问题的解析解是下式的实部

$$u = -\mathrm{i}\frac{\pi v_0 \omega \rho_0^2}{2a} \mathrm{H}_0^{(1)}\left(\frac{\omega}{a}\rho\right) \cos\varphi \mathrm{e}^{-\mathrm{i}\omega t}$$

*见参考文献 [1], 347 页.

在远场区即 ρ 大的区域，渐近解为

$$u = v_0\rho_0^2\sqrt{\frac{\pi\omega}{2a\rho}}\cos\varphi\cos\left(\frac{\omega}{a}\rho - \omega t + \frac{3\pi}{4}\right)$$

使用 PDETOOL 求解的过程与上式相同，只是在内边界 C_2 上取诺伊曼边界条件为 $g = \cos\varphi\cos 6t, q = 0$。图 6.26 画出的是函数在某个时刻的等值线，在动画中，它们表示逐渐向外传播的柱面波。

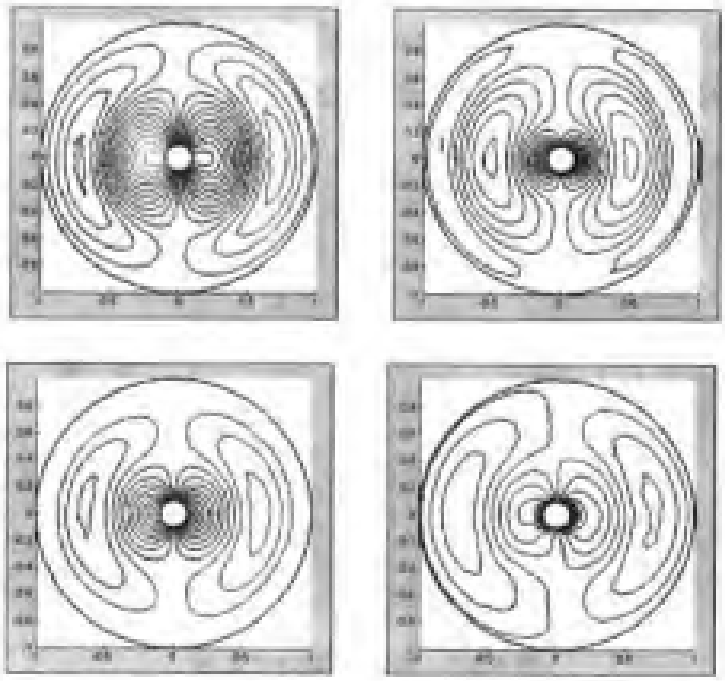


图 6.26 向外传播的柱面波

6.3.4 偶极声源

半径为 r_0 的球面径向速度分布为 $v = v_0\cos\theta\cos\omega t$，试求解这球面所发射的稳恒声振动的速度势 u，设 r_0 远小于声波的波长 λ*。

用球坐标，极点取在球心，定解问题是

$$\begin{cases} u_{tt} - a^2\Delta u = 0 \\ u_r(r = r_0) = v_0 P_1(\cos\theta)\mathrm{e}^{-\mathrm{i}\omega t} \end{cases}$$

在球面的边界条件是 $\cos\omega t$ 即为 $\mathrm{Re}(\mathrm{e}^{-\mathrm{i}\omega t})$，在上面写成了 $\mathrm{e}^{-\mathrm{i}\omega t}$，这要求在计算结果中也只取实部。

*见参考文献 [1]，369 页。

问题的解析解是

$$u=\frac{v_0r_0^3}{2}\left(-\frac{1}{r^2}+\mathrm{i}\frac{\omega}{ar}\right)p_1(\cos\theta)\mathrm{e}^{\mathrm{i}\frac{\omega}{a}(r-at)}$$

上式的实部就是所要求的解.在远场取渐近公式近似,并取实部,得到的解为

$$u=-\frac{v_0\omega r_0^3}{2ar}p_1(\cos\theta)\sin\frac{\omega}{a}(r-at)$$

解析解可以用以下程序作动画，程序中取 $r_0=0.2$, $v_0=2$, $k=60$, $a=2$;

```
clear
r0=0.2;  v0=2;
k=60;  a=2;
theta=linspace(0,2*pi,50);
rho=0.2:0.1:4;
[Th,Rh]=meshgrid(theta,rho);
[X,Y]=pol2cart(Th,Rh);
rh=sqrt(X.^2+Y.^2);
th=atan(Y./X);
for t=0:0.001:0.03
u=real(v0/2*r0^3*(-1./rh.^2+i*k./rh).*...
   cos(th).*exp(k*(rh+2*t)*i));
surf(X,Y,u)
view(-32,28)
pause(0.5)
end;
```

图 6.27 是动画中的几幅画面，这是由球面向外传播的球面波. 虽然从远场近似解的表达式可以看出远场的极化现象，也就是声波传播具有很明显的两极的方向性，但是从近场的表达式是看不出这一点来，将近场的解析解画出图形以后，它的极化现象其实更显著.

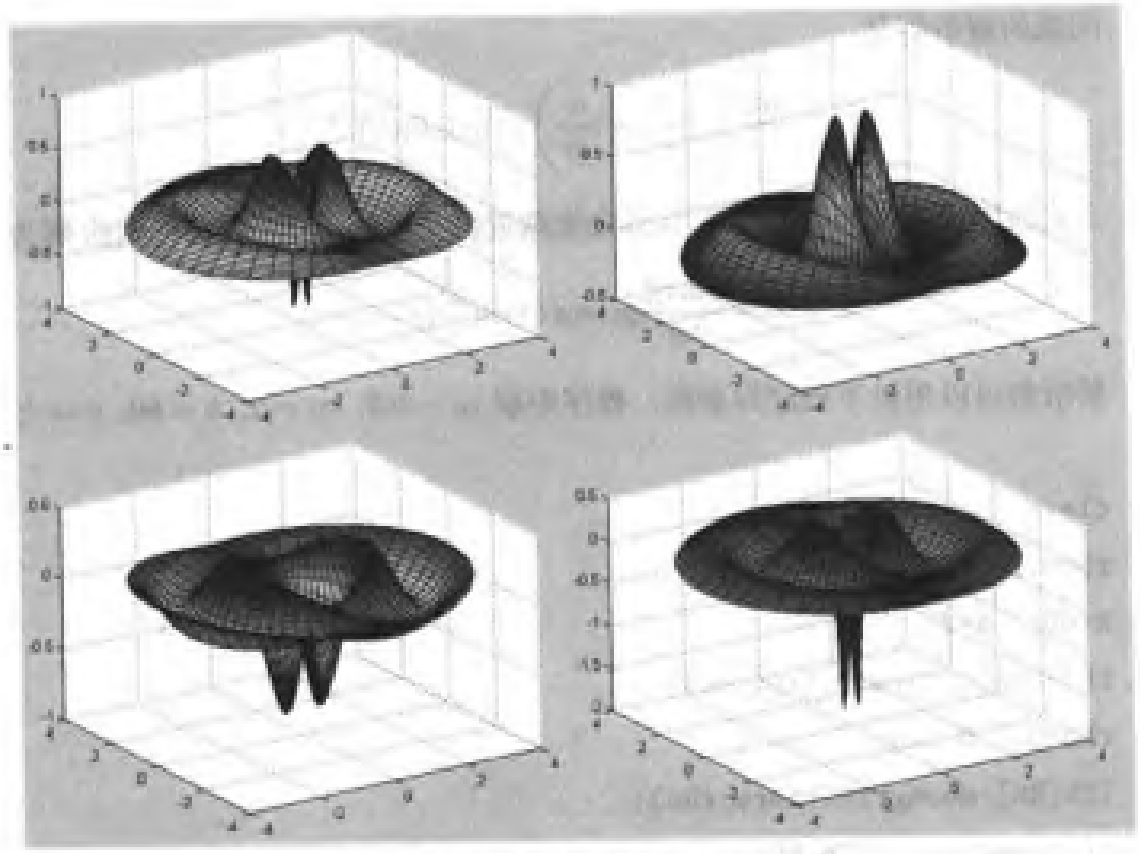


图 6.27 解析解的表面图

图 6.27 也可以用等值线来作动画演示。只要将程序中下列两句

```
surf(X,Y,u)
view(-32,28)
```

换成

```
contour(X,Y,u)
axis([-4 4 -4 4])
axis square
```

即可。图 6.28 就是动画中的几幅画面。

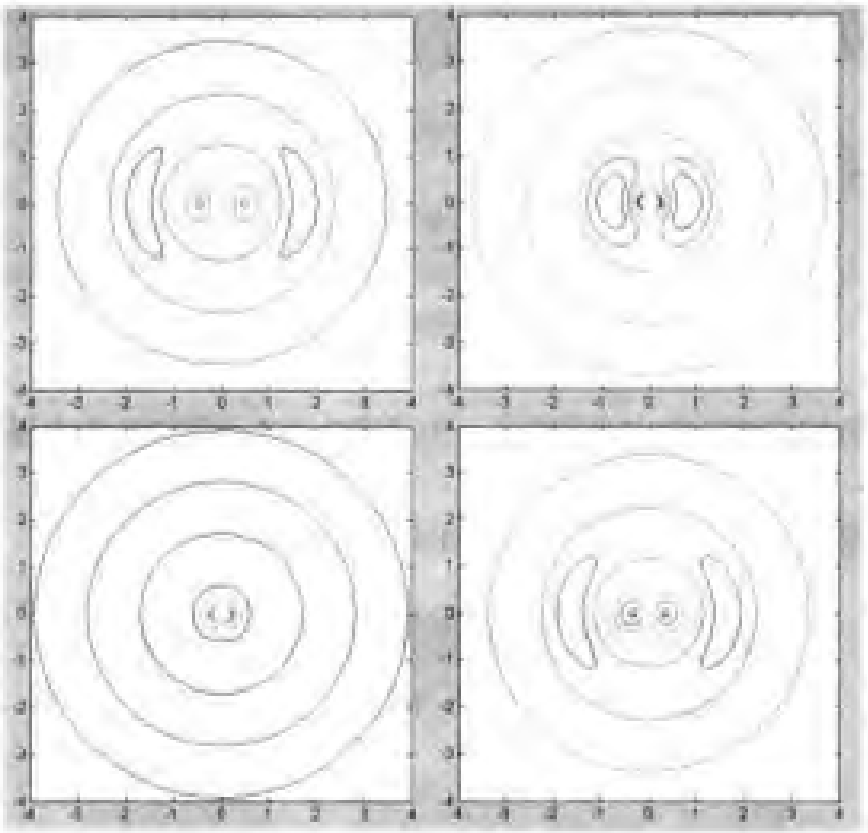


图 6.28 解析解等值线图

再用偏微分方程工具箱求解这个问题. 将时间变量分离以后, 得到亥姆霍兹方程为

$$\Delta u + k^2 u = 0$$

以原点为圆心画两个圆 C1 和 C2, 一个半径为 0.2, 另一个为 1.C1 为柱体的横截面, C2 为求解区域的外边界. 问题的求解区域是 C2-C1 . 按照题意, 内圆边界 C1 是狄里克利边界条件, 可取 h=1, r=cos(atan(y./x)), 外圆边界 C2 取诺伊曼边界条件, 取 g=0,q=−60i . 这个边界条件的推导可参看 7.8.1 节.

方程的设置是 Elliptic 型, 系数是 $c = 1, a = -3600, f = 0$.

为了有足够的精度, 将初始化的网格作了两次细分.

画作图选项对话框 Plot Parameters 中, 在 Plot type 下, 选 Color 和 Height (3-D Plot), 其余不变. 所得的图形如图 6.29 所示, 为了与解析解的结果对比, 将图形作了适当的旋转. 注意方程所得的解为复数, 对于复数解, 只有解的实部会画出来同时给出警告提示框如下:

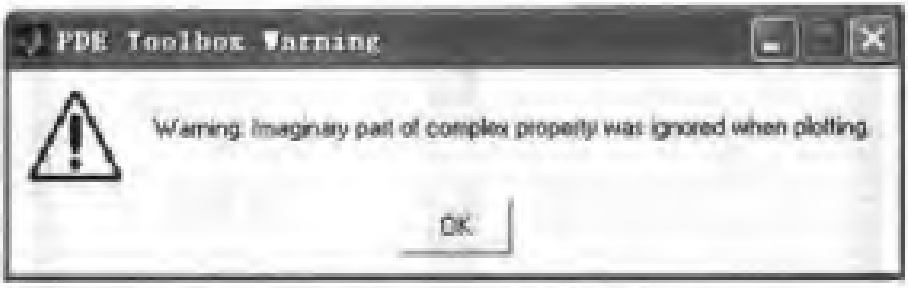


图 6.29 画图时的警告框

只要单击按钮 OK 将它关闭即可. 为了用动画表示原来波动方程的解, 可采取如下操作:

先选择菜单 Mesh/Export Mesh 打开对话框如图 6.30 所示.

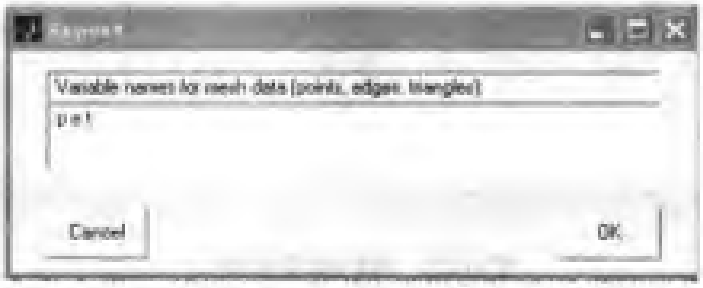


图 6.30 输出网格数据的警告框

单击 OK 按钮将网格数据 p,e,t 输出到指令窗口, 关闭对话框. 再选择菜单 Solve/Export Solution 将打开对话框如图 6.31 所示.

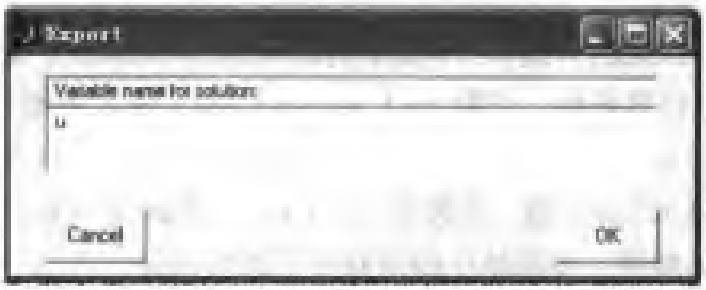


图 6.31 输出解的对话框

单击 OK 按钮将解 u 输出到指令窗口, 关闭对话框. 然后运行下面的程序文件 sswt.m, 就可以看到动画演示的解如图 6.32 所示. 这时, 偶极声源形成的声波由声源向外传播.

```
% program sswt.m
h=newplot;
hf=get(h,'Parent');
set(hf,'Renderer','zbuffer')
axis tight
set(gca,'DataAspectRatio',[1 1 1]);
axis off
M=moviein(10,hf);
maxu=max(abs(u));
colormap(cool)
for j=1:10,
   ur=real(exp(-j*2*pi/10*sqrt(-1))*u);
   pdeplot(p,e,t,'xydata',ur,'colorbar','off','mesh','off');
   caxis([-maxu maxu]);
   axis tight,
   set(gca,'DataAspectRatio',[1 1 1]);
   axis off
   M(:,j)=getframe;
end
movie(hf,M,10);
```

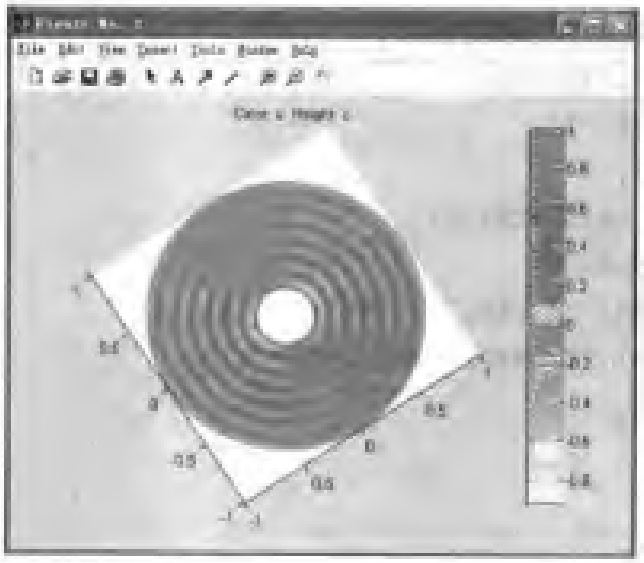


图 6.32 用工具箱所得的数值解的图形

6.3.5 四极声源

半径为 r_0 的球面径向速度分布为 $v=\frac{v_0}{4}(3\cos 2\theta+1)\cos\omega t$, 试求解这球面所发射的稳恒声振动的速度势 u, 设 r_0 远小于声波的波长 λ *.

用球坐标, 极点取在球心, 定解问题是

$$\begin{cases} u_{tt}-a^2\Delta u=0 \\ u_r(r=r_0)=v_0P_2(\cos\theta)\mathrm{e}^{-\mathrm{i}\omega t} \end{cases}$$

在球面的边界条件是 $\cos\omega t$ 即为 $\mathrm{Re}(\mathrm{e}^{-\mathrm{i}\omega t})$, 在上面写成了 $\mathrm{e}^{-\mathrm{i}\omega t}$, 这要求在计算结果中也只取实部.

问题的解析解是

$$u=\mathrm{i}\frac{v_0r_0^4}{3}\left(-\frac{1}{r^3}+\mathrm{i}\frac{\omega}{ar^2}+\frac{\omega^2}{3a^2r}\right)p_2(\cos\theta)\mathrm{e}^{\mathrm{i}\frac{\omega}{a}(r-at)}$$

取其实部即为所求. 在远场可以取渐近公式后再取实部, 得

$$u=\frac{v_0k^2r_0^4}{9r}p_2(\cos\theta)\cos kr(r-at)$$

解析解可以用以下程序作动画, 程序中取 $r0=0.2$; $v0=3$; $k=60$; $a=2$;

```
r0=0.2;  v0=3;  k=60;  a=2;
theta=linspace(0,2*pi,50);
rho=0.2:0.1:4;
[Th,Rh]=meshgrid(theta,rho);
[X,Y]=pol2cart(Th,Rh);
rh=sqrt(X.^2+Y.^2);
th=atan(Y./X);
P2=1/4.*(3.*cos(2.*th)+1);
for t=0:0.002:0.05
u=real(i*r0^4*( -1./rh.^3+i*60./rh.^2+...
   3600/3./rh).*P2.*exp(i*k*(rh+2*t)));
surf(X,Y,u)
view(-12,44)
pause(0.5)
end
```

*见参考文献 [1], 372 页.

图 6.33 是动画中的几幅画面. 这是由球面向外传播的球面波. 这种球面波具有很明显的 4 个方向性. 虽然从远场近似解的表达式可以看出远场的极化现象, 也就是声波传播具有很明显的 4 极的方向性. 但是从近场的表达式是看不出这一点来, 将近场的解析解画出图形以后, 这种特性就一目了然了.

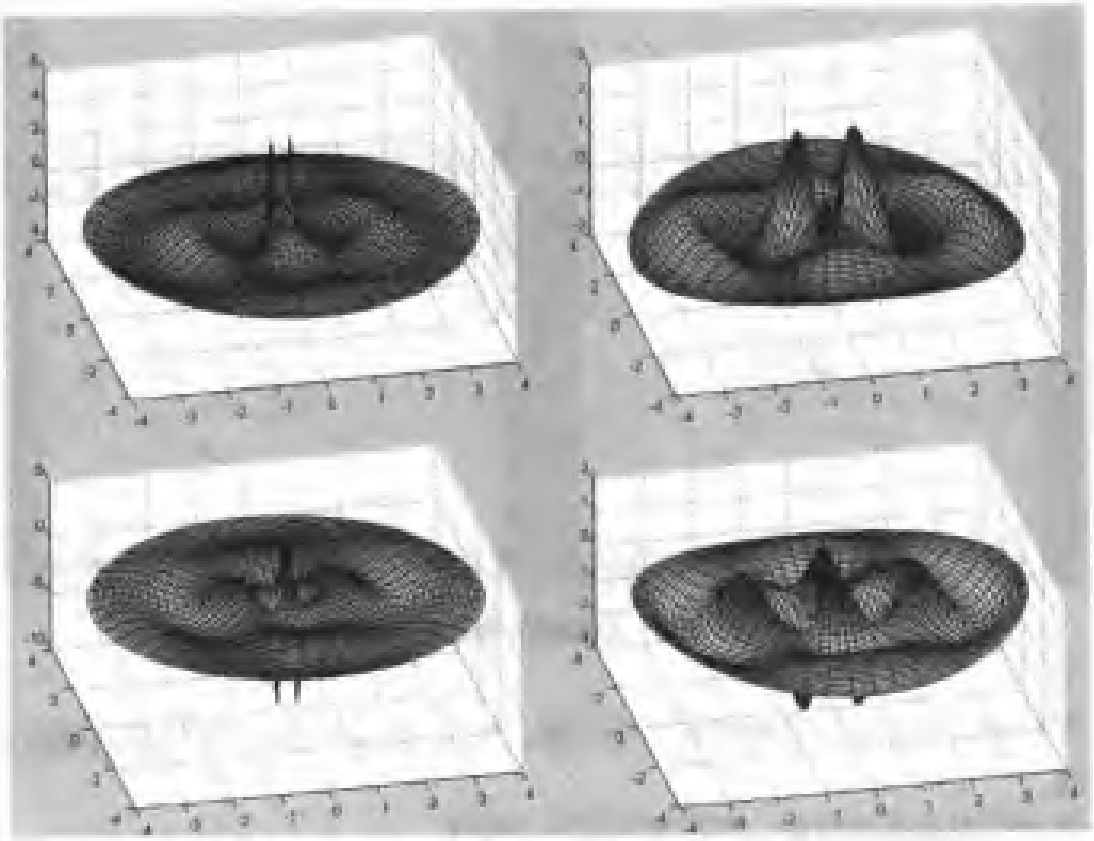


图 6.33 解析解的表面图动画

图 6.33 是用表面图画的动画, 也可以用等值线画动画. 方法与画偶极声源的方法相同, 也是将下列语句

```
surf(X,Y,u)
view(-12,44)
```

置换为

```
contour(X,Y,u)
```

即可, 图 6.34 是等值线动画中的几幅画面.

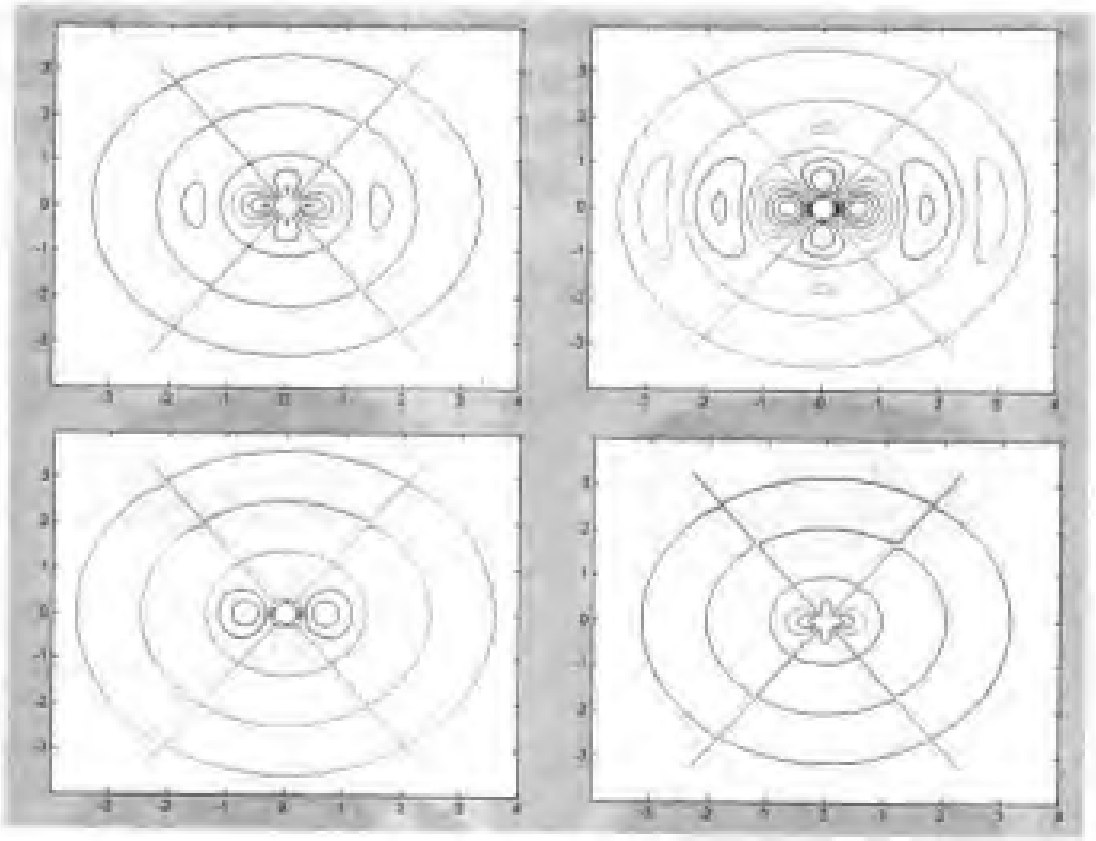


图 6.34 解析解的等值线动画

再用偏微分方程工具箱求解这个问题，做法与上题相同，仅仅是内圆边界条件不同。

将时间变量分离以后，得到亥姆霍兹方程为

$$\Delta u + k^2 u = 0$$

以原点为圆心画两个圆，一个半径为 0.2，另一个为 1，按照题意，内圆边界是齐次的狄里克利边界条件，可取 $h = 1, r = \frac{1}{4}(3 * \cos(2 * \mathtt{atan}(y./x)) + 1)$，外圆边界诺伊曼边界条件，取 $g = 0, q = -60i$。这个边界条件的推导可参看 7.8.1 节。

方程的设置是 Elliptic 型，系数是 $c = 1, a = -3600, f = 0$。

为了有足够的精度，将初始化的网格作了两次细分。

画作图选项对话框 Plot Parameters 中，在 Plot type 下，选 Color 和 Height (3-D Plot)，其余不变。所得的图形如图 6.35 所示。为了与解析解的结果对比，将图形作了适当的旋转，改变视角，可以得到与解析解相似的图形。

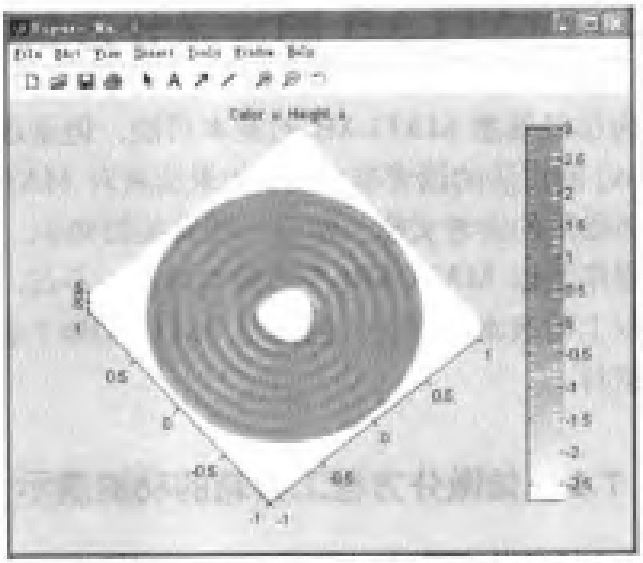


图 6.35 用工具箱所得的数值解的图形

和前面的例子一样，只要将网格数据 p,e,t 与方程的解 u 输出到工作空间，再运行与前相同的程序 sswt.m 文件就可以看到动画。

第7章　MATLAB 的偏微分方程工具箱

本章的内容是为已经熟悉 MATLAB 的基本用法，但是还不了解偏微分方程工具箱 (PDE toolbox) 的用法的读者准备的。如果读者对 MATLAB 的用法还不熟悉，则可以参考本书提供的参考文献，学习一些有关的知识。

这里所介绍的程序均以 MATLAB6.5 的版本为准，不过，这些程序基本上都能在 MATLAB6.0 以上的版本中运行，但是用 MATLAB6.5 编写的用户界面程序不能在更低的版本运行。

7.1　偏微分方程工具箱的功能演示

偏微分方程是广泛应用于科学与工程技术的各个领域的一种数学工具，例如应用椭圆型方程与抛物型方程的领域有，固体中稳恒的与非稳恒的热传导现象，可渗透介质中流动与扩散问题，绝缘与导电材料中的静电场问题，流场的势。应用双曲型方程的领域有，声波和电磁波的传播问题，薄膜的横振动。应用本征值方程的领域有，薄膜和结构力学中的固有振动问题。

偏微分方程工具箱 (PDE Toolbox) 就是 MATLAB 中求解二维偏微分方程 (含时或不含时) 的工具，查看偏微分方程工具箱的演示程序可以简捷地了解偏微分方程工具箱的基本功能，操作如下：

在指令窗口键入 pdedemos, 打开一个 Command Line Demos 窗口如图 7.1所示。

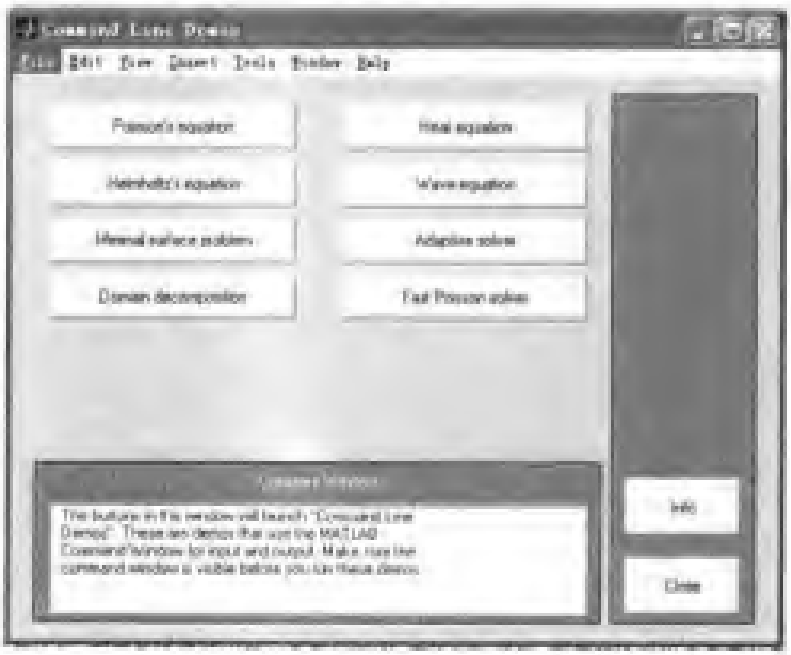


图 7.1　Command Line Demos 窗口的演示例子

窗口的左边是 8 个演示例子的按钮，下面是注解信息框 (Comment Wimdow)，右边是信息提示按钮 (Info) 和退出按钮 (Close)。

从信息框中的说明可以知道，这 8 个按钮可以直接运行指令形式的 8 个偏微分方程的演示程序。演示程序的输入与输出都显示在指令窗口，所以在运行演示程序时，一定要保持指令窗口显示在前面。程序在运行时，会把程序的内容显示在窗口中，每显示一段程序都会暂停一下，并在最后一行出现如下的提示文字：

```
pause % Strike any key to end.
```

只要敲击任何一个键都会使程序继续运行，如此进行下去，直到程序运行结束。

单击信息提示按钮可以打开帮助系统中有关这个演示窗口的说明文字。这 8 个演示程序的源程序依次是 pdedemo1 ~ pdedemo8，可以单独在命令窗口中运行，只需键入

```
>>pdedemo1
```

就是运行第一个程序，而键入

```
>>edit pdedemo1
```

则是查看第一个程序的源程序。对其他程序的运行方法与此相似。

偏微分方程的 8 个演示例子介绍如下。

7.1.1 泊松方程

所解的问题是，在单位圆内求解泊松方程

$$-\Delta u = 1$$

在单位圆的边界上有 $u = 0$。这个问题的精确解是

$$u = (-x^2 - y^2)/4$$

这个演示例子求出了数值解，画出了数值解的图形如图 7.2 所示，然后又画出了数值解与精确解的误差的图形如图 7.3 所示。

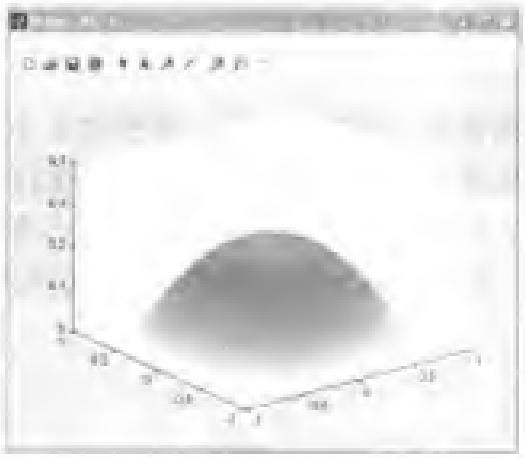


图 7.2 数值解的图形

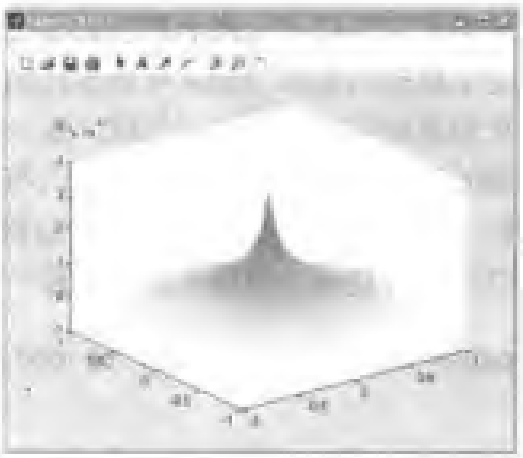


图 7.3 数值解与精确解的误差

7.1.2 亥姆霍兹方程

这个演示例子是研究平面上的方形物体对波的反射. 亥姆霍兹方程是波动方程分离时间变量以后得到的, 形式为

$$-\Delta u - k^2 u = 0$$

边界条件是, 在物体的边界 (正方形) 上, 反射波满足狄里克利边界条件, 在外边界即圆周 (计算区域取一个圆) 上, 反射波满足诺伊曼边界条件.

平面电磁波是用复指数函数表示, 所以所得的反射波也是复指数函数. 在将数值计算结果可视化时, 使用的是它的实部. 首先画的是一个三维视图如图 7.4 所示, 相应的平面视图如图 7.5 所示.

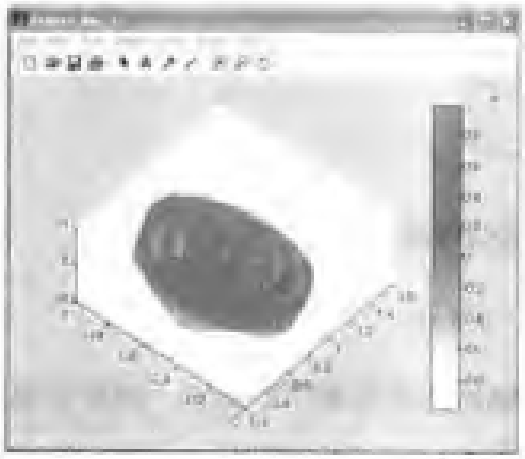


图 7.4 数值解三维视图



图 7.5 数值解的二维视图

程序演示的最后结果是动画, 动画演示就是用平面视图做成的. 在 7.8.1 节中对这个问题作了详细的分析与求解.

7.1.3 最小表面问题

最小表面问题就是，给定一条闭合的空间曲线，求张在这一曲线上的曲面，要求使曲面的面积为最小。在本节中，问题等价于求解方程

$$-\nabla\cdot\left(\frac{\nabla u}{\sqrt{1+|\nabla u|^2}}\right)=0$$

求解的区域是单位圆，在单位圆的边界上有 $u=x^2$ 。

所得到的数值解的图形如图 7.6 所示。

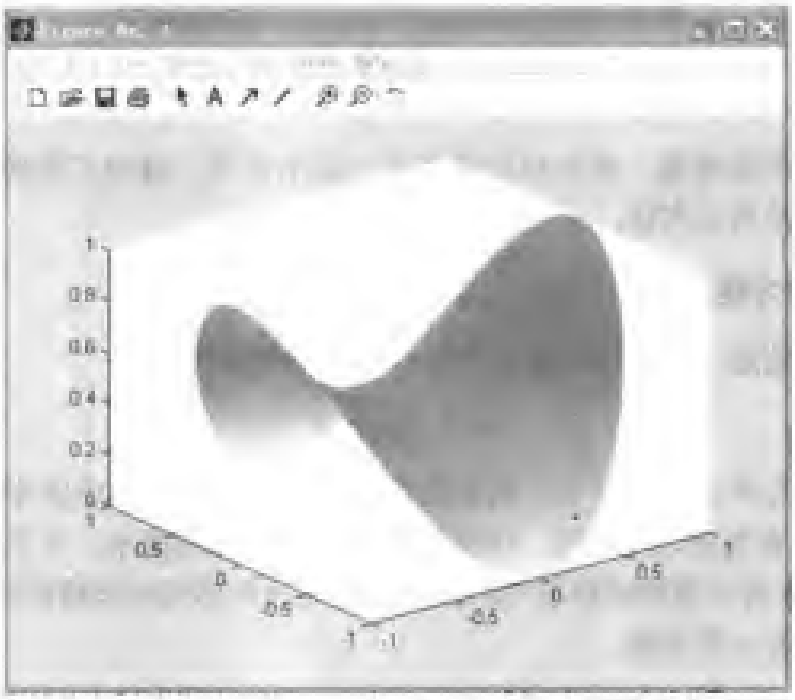


图 7.6 数值解的图形

7.1.4 区域分解方法

所解的问题是 L 形区域内的泊松方程

$$-\Delta u=1$$

在区域的边界上，有 $u=0$ 。与第一个演示例子不同的是，本例使用的解法是子区域分解法解有限元问题。所得的结果如图 7.7 所示。

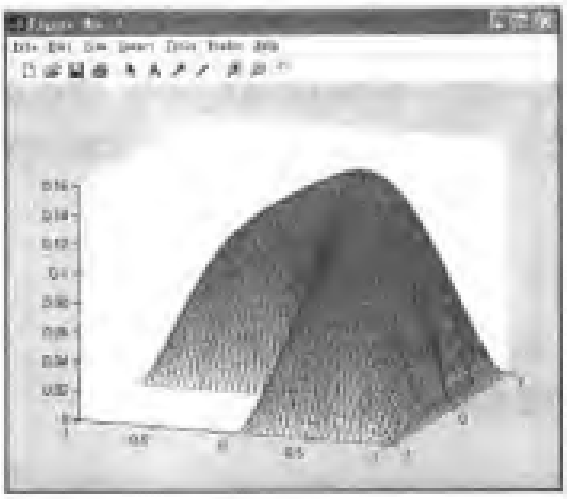


图 7.7 数值解的图形

为了使图形更清楚，我们对程序作了一点小改动，增加了画网格的语句，所以在图形中显示出了网格.

7.1.5 热传导方程

所解的问题是，在方形区域上求解有热源的热传导方程

$$u_t - \Delta u = 1$$

在方形区域的边界上有 $u = 0$. 初始温度分布是在方形区域中间半径为 0.4 的圆内，温度为 1，在其余区域为零. 所得的结果可以用动画演示，为了加快动画绘制的速度，将原来用三角形网格求得的数据改变成用矩形网格作插值计算. 图 7.8 是某一时刻动画的一个画面.

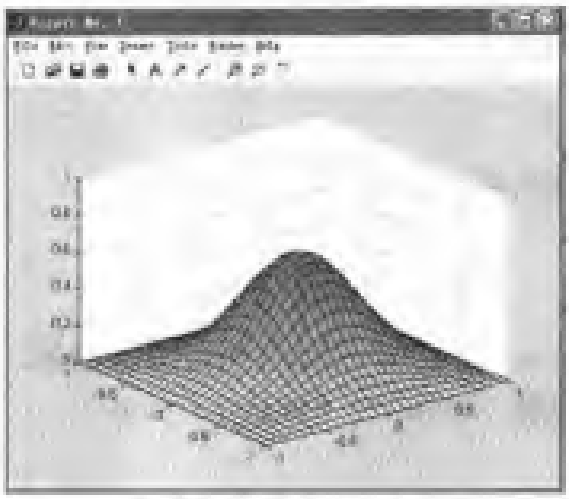


图 7.8 数值解的图形

7.1.6 波动方程

所解的问题是方形薄膜的振动，即在方形区域解波动方程

$$u_{tt} - \Delta u = 0$$

在区域的左右边界上有狄里克利边界条件，在上下边界上有诺伊曼边界条件，薄膜的初始位移和初始速度分别是

$$u(t=0) = \arctan(\cos(\pi x/2))$$

$$u_t(t=0) = 3\sin(\pi x)e^{\sin \pi y/2}$$

所得的结果可以用动画演示，并使用了矩形网格插值来加速动画的绘制。图 7.9 是动画的一个画面。

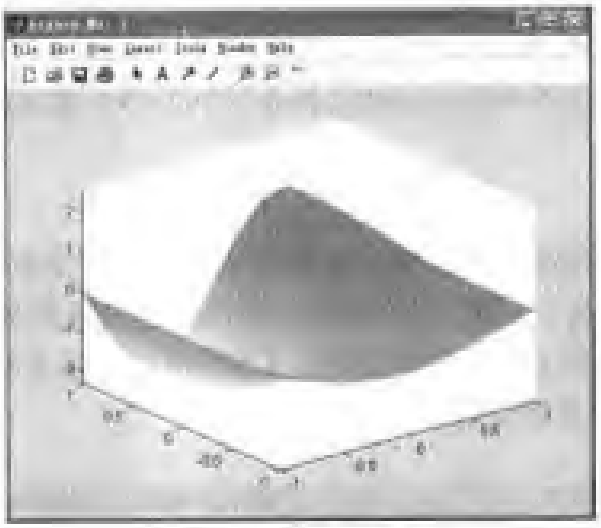


图 7.9 数值解的图形

本例在后面 7.9.3 节用偏微分方程工具箱的用户界面窗口重做了一遍，读者可以将两者进行对比。

7.1.7 椭圆型方程自适应解法

所解的问题是单位圆内带点源的泊松方程

$$-\Delta u = \delta(x, y)$$

在区域的边界上有 $u = 0$。

这个问题的精确解是 $-\ln r/(2\pi)$，显然原点是奇异点，用自适应解法可以求出除原点以外各点的解。所得的数值解的图形如图 7.10 中的图 (a)，数值解与精确解的误差的图形如图 (b)。

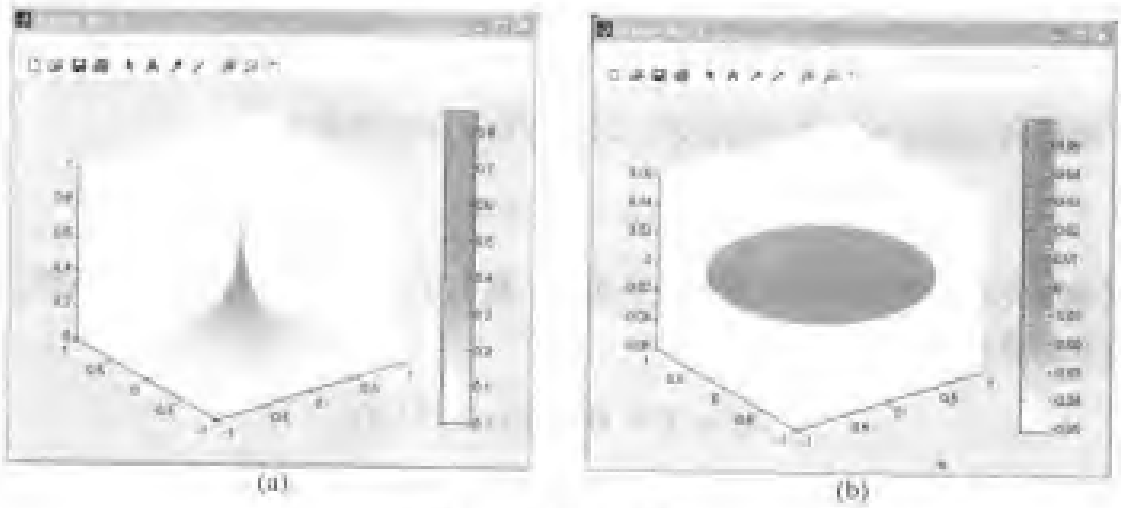


图 7.10 数值解图形

(a) 数值解的图形。(b) 数值解与精确解的误差

7.1.8 泊松方程快速解法

所解的问题是，在带有狄里克利边界条件的方形区域求解泊松方程

$$-\Delta u = 3x^2$$

所用解法是快速泊松方程解法，即使用矩形网格来求解方程。为了对比，又用三角形网格重解了这个方程。可以看到，前一种方法解题所花费的 CPU 的时间为 0.8100 秒，而后一种方法解题所花费的 CPU 的时间是 8.3500 秒，是前者的 10 倍多。图 7.11 是泊松方程快速解法图。

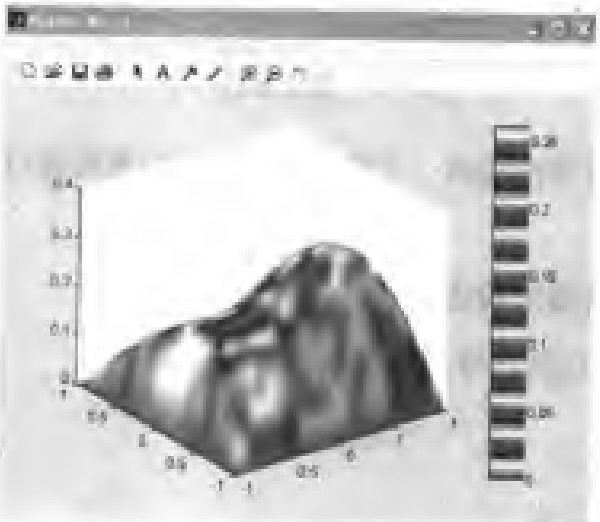


图 7.11 泊松方程快速解法的结果

7.2 偏微分方程工具箱的功能

偏微分方程工具箱的功能是指它能解的偏微分方程的类型和边界条件. 由上一节的演示可以看到, 对于一般常见的二维偏微分方程, 它都能解, 是一个很实用的数值计算工具. 尤其是在边界条件比较简单的情况下, 使用者几乎可以完全不必学习编程仅仅在图形用户界面窗口进行操作, 就能得到方程的数值解. 这对许多需要使用偏微分方程而不擅长编程的物理学工作者, 是一个不可多得的工具.

7.2.1 可解方程的类型

定义在二维有界区域 Ω 上的下列形式的方程可以用偏微分方程工具箱求解

$$-\nabla\cdot(c\nabla u)+au=f \qquad \text{椭圆型}$$

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f \qquad \text{抛物型}$$

$$d\frac{\partial^2 u}{\partial t^2}-\nabla\cdot(c\nabla u)+au=f \qquad \text{双曲型}$$

$$-\nabla\cdot(c\nabla u)+au=\lambda du \qquad \text{本征值方程}$$

在这里, u 是偏微分方程的解, 在本书的叙述中, 都是采用这个符号, c,a,f,d 是标量复函数形式的系数, 在抛物型和双曲型方程中, 它们也可以是 t 的函数, λ 是待求的本征值.

当系数 c,a,f 是 u 的函数时, 我们称之为非线性方程, 形式为

$$-\nabla\cdot(c(u)\nabla u)+a(u)u=f(u)$$

的非线性方程也可以用偏微分方程工具箱求解.

偏微分方程工具箱还可以解下列形式的偏微分方程组, 在解方程组之前必须先选择菜单 Options/Application/Generic System, 以选择专用于此问题的特定解法,

$$\begin{cases}-\nabla\cdot(c_{11}\nabla u_1)-\nabla\cdot(c_{12}\nabla u_2)+a_{11}u_1+a_{12}u_2=f_1\\ -\nabla\cdot(c_{21}\nabla u_1)-\nabla\cdot(c_{22}\nabla u_2)+a_{21}u_1+a_{22}u_2=f_2\end{cases}$$

此时 c 是 2×2 的矩阵形式的函数.

对于高阶的方程组系统, 可以用指令求解. 对于椭圆型方程, 可以用自适应网格算法求解, 这种方法也能与非线性解法联合使用. 此外, 对于泊松方程有一种利用矩形网格的快速算法. 这些方法既可以通过在菜单 Solve/Parameters 下的选项来实现, 也可以用指令 adaptmesh,pdenonlin,poisolv 来实现.

7.2.2 边界条件

解方程所需要的边界条件可以是以下两种之一：

$hu = r$ 狄利克里 (Diriclet) 边界条件

$\boldsymbol{n} \cdot (c\nabla u) + qu = g$ 广义诺伊曼 (Generalized Neumann) 边界条件

这里，$\boldsymbol{n}$ 为边界外法向单位向量，g, q, h, r 是在边界上定义的复函数. 对于本征值问题，仅限于齐次条件即 $g = 0, r = 0$. 对于非线性问题，系数 g, q, h, r 可以与 u 有关. 对于抛物型方程和双曲型方程，系数可以与时间 t 有关.

在数学物理方法教材中，狄利克里 (Diriclet) 边界条件也叫第一类边界条件，而广义诺伊曼 (Generalized Neumann) 边界条件则称之为第三类边界条件，如果 $q = 0$ 则叫第二类边界条件.

对于方程组而言，狄利克里边界条件是

$$h_{11}u_1 + h_{12}u_2 = r_1$$

$$h_{21}u_1 + h_{22}u_2 = r_2$$

而广义诺伊曼边界条件是

$$\boldsymbol{n} \cdot (c_{11}\nabla u_1) + \boldsymbol{n} \cdot (c_{12}\nabla u_2) + q_{11}u_1 + q_{12}u_2 = g_1$$

$$\boldsymbol{n} \cdot (c_{21}\nabla u_1) + \boldsymbol{n} \cdot (c_{22}\nabla u_2) + q_{21}u_1 + q_{22}u_2 = g_1$$

混合边界条件则是

$$h_{11}u_1 + h_{12}u_2 = r_1$$

$$\boldsymbol{n} \cdot (c_{11}\nabla u_1) + \boldsymbol{n} \cdot (c_{12}\nabla u_2) + q_{11}u_1 + q_{12}u_2 = g_1 + h_{11}\mu$$

$$\boldsymbol{n} \cdot (c_{21}\nabla u_1) + \boldsymbol{n} \cdot (c_{22}\nabla u_2) + q_{21}u_1 + q_{22}u_2 = g_2 + h_{12}\mu$$

其中求得的 μ 要使得狄利克里边界条件得到满足. 在有限元法中，狄利克里边界条件也叫基本边界条件，诺伊曼边界条件也叫自然边界条件.

7.3 工具箱的用户界面窗口

在指令窗键入 pdetool 就打开了它的图形用户界面窗口如图 7.12 所示.

窗口中第一行是菜单，包括 File, Edit, Options, Draw, Boundary, PDE, Mesh, Solve, Plot, Window, Help . 在后面 7.6 节会详细介绍它们.

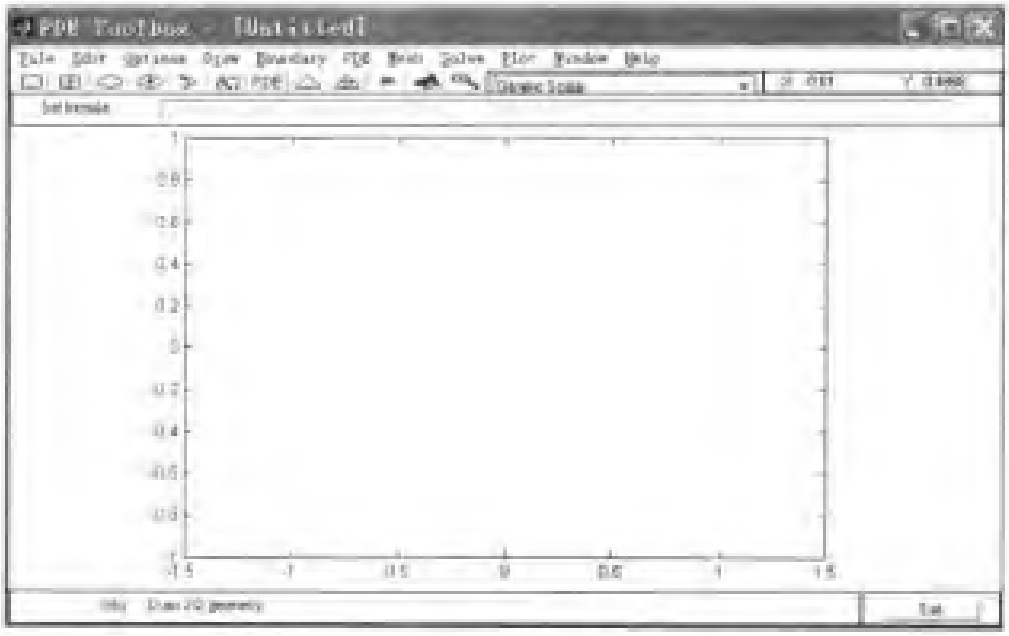


图 7.12 偏微分方程工具箱的界面

第二行是工具栏，工具栏中有 12 个图标按钮，各个图标按钮的用途如下：

从顶点画矩形或正方形；

从中心画矩形或正方形；

从周边上一点画椭圆或圆；

从中心画椭圆或圆；

画多边形；

定义边界条件；

定义偏微分方程；

将区域划分为三角形网格；

将三角形网格细分为更小的三角形；

求解偏微分方程；

将结果可视化；

将图形放大。

工具栏的图标可以提供偏微分方程工具箱的主要功能，但是不包括一些辅助功能。利用它们可以方便快捷地完成解方程的基本操作。当然，它们的所有功能都可以通过菜单上的操作来完成。

在 MATLAB 中用于解偏微分方程操作称之为一种模式 (Mode)，工具栏左边的 5 个图标是用于画方程的求解区域，统称为画区域图 (Draw) 模式。其他图标代表的模式有边界 (Boundary) 模式、方程 (PDE) 模式、网格 (Mesh) 模式、解方程

(Solve) 模式和画图形解 (Plot) 的模式。

接着的长方形框是方程应用的 10 种类型的问题（见 7.4.4 节），这些问题使用的对话框界面有所不同，用户可根据问题的物理内容来选择，通常可用一般标量场格式 (General Scalar)，而静电场问题则是选 electrostatics。最后是光标的坐标，供作图时选择位置用。

第三行 Set formula 区域是将画好的解方程的区域用公式表示。

窗口的中间区域是用作图方式来确定方程的求解区域， x,y 轴的范围用菜单 Options 下的 Axes Limits 对话框来选择。

窗口最下面是信息栏，提示当前操作的内容，右下角是退出按钮 (Exit)。

7.4 用工具箱解偏微分方程的步骤

用偏微分方程工具箱解偏微分方程有两种方法：

一种是在它的图形用户界面中进行操作。这种用法较简单，使用者无须了解解题的具体过程，也不必编程，只要学习图形界面上按钮或菜单的简单操作即能完成解题。

另一种是利用偏微分方程工具箱提供的指令编程计算，它可以解决更复杂的问题，但这种方法对使用者有较高的要求。使用者必须掌握偏微分方程工具箱的专用指令，对解题的过程也应该有所了解才能正确地编程。有时候可以将两种方法结合起来灵活使用，即某些步骤可以使用图形界面来完成，然后将结果输出，再将这些结果用到用指令编写的程序中去，最终完成解题的全过程。

用偏微分方程工具箱解偏微分方程的步骤大致如下：

7.4.1 设置定解问题

定义一个偏微分方程的定解问题，最简单的方法是使用偏微分方程工具箱的用户界面。在用户界面中设置定解问题要用到其中的三个模式。

Draw 画出对方程求解的区域。它们可以是矩形，正方形，椭圆，圆和多边形以及它们的组合 (参见 7.5.1 的详细解释)，称之为组合的实心的几何图形 CSG(constructive solid geometry)。

Boundary 定义边界条件。

PDE 定义一个偏微分方程，即给定方程的类型及其系数 c, a, f 和 d。可以在不同的子区域给定不同的系数，这意味着它们是不同的材料。

7.4.2 解方程

解方程要用到下面两个模式：

Mesh 将区域分割为三角形网格，网格的参数可以改变。

Solve 求解方程。对于椭圆型方程，可以调用非线性解法与自适应解法。对于抛物型方程和双曲型方程，在给定初值以后，能求出指定时刻的解。对于本征值问题，可以设定搜索本征值的范围。

在问题求解以后，还可以返回到 Mesh 模式，将网格细分以后重新求解方程。如果使用自适应解法，则它会自动寻找最适合于该问题的网格。

对非标准化问题，就要由用户来将边界条件的描述传输到工作空间，然后再用工具箱的指令来处理这些尚未分解成网格的数据。用户可以自由地访问工具箱中各种指令，如网格生成法，有限元离散方法，边界条件，插值计算等。用户也可以设计自己的解法，或用有限元法去解一个更复杂的问题中的一个子问题。详细内容可以参看 7.7 节。

7.4.3 将结果可视化

结果的可视化是在 Plot 模式下实现的。

Plot 在这个模式下，有广泛的可视化手段可供使用。图像可以显示在偏微分方程工具箱的用户界面内，也可以是一个新开的图形窗口。可以同时用彩图、高度图和矢量场图来表现解的三个不同的特性。曲面图、网线图、等值线图和箭头图都可以使用。对于曲面图，既可以用插值法也可用平坦法来渲染色彩。每种图形中的网格线既可以显现也可以隐藏。对抛物型方程和双曲型方程，还可以产生一个时间序列的动画。所有这些可视化的指令都是可以在 MATLAB 的指令窗中访问的。

7.4.4 有特定解法的问题类型

偏微分方程工具箱的用户界面的使用十分方便，除了一般的标量情况 (generic scalar) 和方程组 (systems) 情况，还为 8 种类型的问题设置了特定解法，这些问题有：

Generic Scalar	一般标量场
Generic system	一般的偏微分方程组
Structual Mechnics,Plane Stress	结构力学 – 平面应力场
Structual Mechnics,Plane Strain	结构力学 – 平面应变场
Electrostatics	静电场
Magnetostatics	静磁场
Ac Power Electromagnetics	交流电磁场
Conductive Media DC	直流导电介质问题
Heat Transfer	热传导问题

Diffusion 扩散问题

每种特定解法在调用以后呈现的对话框都有所不同，在对话框中对方程系数、边界条件和方程的解用物理概念都作了说明。在这些对话框中，用户可以设置一些特殊的参数如结构力学中的杨氏模量，而一些相关的物理量也能实现可视化。

本书提供了一些例子，有些例子是用用户界面和工具箱指令两种方法求解。工具箱本身也带有许多供演示的 M 文件，从中可以学到如何去编写自己的程序。

偏微分方程工具箱是一个完全开放的系统，其中没有黑箱函数，尽管某些指令并不是一下子就能看懂。数据结构和公式都是写成文件的。通过研究已有的函数，用户就能建立自己的函数。

7.4.5 如何解三维问题

偏微分方程工具箱是专门设计用来解二维空间的问题，是否可以用它解决三维空间的问题呢？

一种可能是，如果三维空间的变化可以用二维方程来解释，那么问题就可以简化成二维空间的。例如，平面应力分析就可以通过修改材料的参数使其降维。

在某些问题中，如果第三维 (z) 的变化可以忽略，那么所有的 z 的导数都可以去掉，所得的二维方程实际上和三维方程有完全一样的单位和系数。

还有另一种可能就是，对有厚度的板可以处理成沿厚度的积分。所得到的是关于 z 方向的平均解 $D(x,y)$ 的二维方程，方程中的所有系数 c,d,a,f 都要乘以厚度。例如想从焊接在一起的不同厚度的一堆薄板中计算其中某一片板上的应力，就可以用 $D(x,y)$ 乘以杨氏模量 E 、体积力和表面张力。这种做法也适用于其他的关子厚板的例子和应用模式。

7.5 用图形用户界面窗口的工具栏解方程

下面介绍用图形用户界面解方程的方法，同时提供了一个简单的实例。由于图形界面按钮的功能与菜单命令的功能相同，所以有些在 7.6 节会详细介绍的内容在本节只作简单的介绍，在了解这些功能时，读者可以同时参看 7.6 节的介绍。

7.5.1 解方程的步骤

1. 画方程求解的区域

用鼠标单击按钮 ，光标变成十字形状，操作界面进入画图模式 (Draw mode)。注意，只有在这个模式下才能画出求解的区域或修改已经画好的区域。将光标移到指定位置，其位置可以从工具栏中光标的 x,y 坐标来确定，按住鼠标左键，移

动光标，就会画出一个矩形，如果按住右键，就会画出一个正方形。而光标的起始位置则成为矩形或正方形的一个顶点。

按钮 □ 的用法相似，是从中心开始画矩形或正方形。

画好的图形可以用鼠标拖动，还可以精确地定位，方法是双击画好的矩形或正方形，打开一个对话框如图 7.13 所示。



图 7.13 矩形与正方形的位置对话框

在对话框中，第一行是矩形或正方形的左边线的 x 坐标，第二行是下边线的 y 坐标，第三行是矩形或正方形的宽度，第四行是矩形或正方形的高度，第五行是图形的名称，是图形画好以后，由 MATLAB 自动命名并显示在表示区域的公式栏中，矩形为 R1,R2,⋯ 正方形为 SQ1,SQ2,⋯，可以根据需要来修改对话框中的图形的位置坐标或重新命名图形，改好以后，单击 OK 按钮是保存改动的结果，单击 Cancel 是取消改动。

按钮 ⬭ 是从边上开始画椭圆或圆，画图时按住鼠标左键画出的是椭圆，按住鼠标右键画出的是圆。按钮 ⊕ 则是从中心开始画椭圆或圆。

画多边形用按钮 ⬠ ，用鼠标单击按钮以后，将鼠标移到指定位置，按下鼠标左键不要松开，移动光标到适当的位置，然后松开左键，就会画出多边形的一条边，再按下左键，移动鼠标就会画出第二条边，依次进行直至把多边形所有边全部画好。当终点与起点重合时，只要松开鼠标按钮，就会生成一个多边形图形。

这些画图功能都可以用菜单 draw 下的命令来完成。先选择命令 draw mode，再选择要画的图形命令，然后就可以在窗口中进行上面所描述的画图操作。

双击画好的椭圆，打开一个对话框如图 7.14 。

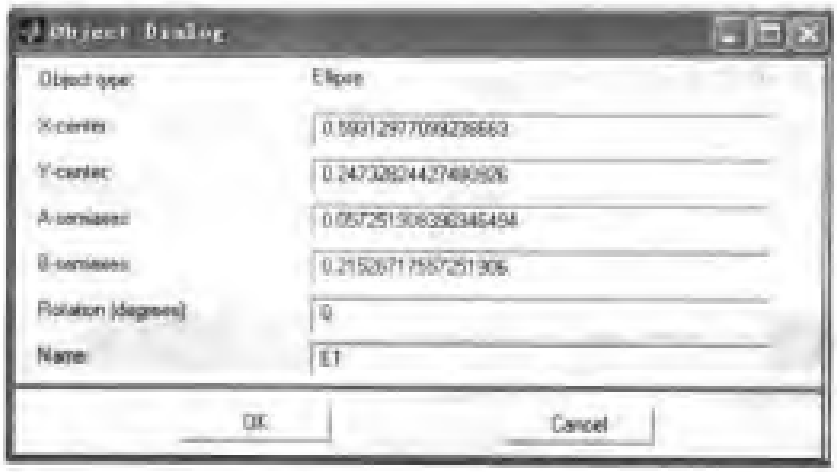


图 7.14 椭圆的位置对话框

在对话框中，第一，第二行是椭圆中心的 x, y 坐标，第三、四行是椭圆的半 A 轴、半 B 轴的长度，第五行是图形旋转的角度，第六行是图形的名称，是图形画好以后，由 MATLAB 自动命名椭圆为 E1,E2,E3, ... ，并显示在表示区域的公式栏中，可以根据需要来修改对话框中的图形的位置坐标或重新命名图形。改好以后，单击 OK 按钮是保存改动的结果，单击 Cancel 是取消改动。

双击画好的圆，打开一个对话框如图 7.15 所示。



图 7.15 圆的位置对话框

在对话框中，第一、第二行是圆心 x, y 坐标，第三行是圆的半径，第四行是图形的名称，是图形画好以后，由 MATLAB 自动命名圆为 C1,C2,C3, ..., 并显示在表示区域的公式栏中，可以根据需要来修改对话框中的图形的位置坐标或重新命名图形。改好以后，单击 OK 按钮是保存改动的结果，单击 Cancel 是取消改动。

双击画好的多边形，打开一个对话框如图 7.16 所示。



图 7.16 多边形的位置对话框

在对话框中，第一行是一个弹出式的列表，其中按顺序列出了所有多边形顶点的 x, y 坐标，选定其中的一个顶点，在第三行和第四行编辑框中会显示它的 x, y 坐标，在编辑框中，就可以修改这个顶点的坐标。第五行是图形的名称，是图形画好以后，由 MATLAB 自动命名多边形为 P1, P2, P3, ... 并显示在表示区域的公式栏中，可以根据需要来修改对话框中的图形的位置坐标或重新命名图形。改好以后，单击 OK 按钮是保存改动的结果，单击 Cancel 是取消改动。

为了画图方便，可以在菜单 Options 下选择 Axes Limits 和 Axes Equal 来确定坐标范围与比例，选择 Grid 和 Grid Spacing 来画网格和指定网格间距，选择 Snap 可以使画的图形边缘与网格线对齐，选择 Rotate 来旋转图形，并根据问题的物理性质来选 Application 下的各个选项。

在表示区域的公式栏中，运算符号 +, −, * 可以将多个图形组成一个集合，+ 表示集合的并，* 表示集合的交，这两种运算的优先级别相同，− 表示集合的差，它具有更高的运算优先级别。括号可以改变运算的先后次序。

这个模式中的 5 个图标的操作有共同之处。如每次只能使用一个操作而不能同时使用。用鼠标左键单击图标一次，可以画出一个所选定的图形。如果用鼠标左键连击图标二次，则“锁定”了这个图形，可以连续多次地重复画相同的图形，直到再次用鼠标左键单击这个图标，才能解除“锁定”。

即使不在 Draw 模式下，只要单击其中的一个图标，就自动地进入了 Draw 模式。而其余模式的图标则具有“即点即用”的功能，就是每当你单击图标，就会启动相应的功能。

在这个模式中，可以用菜单 Edit 下的各个指令对图形进行剪切、复制、粘贴和清除，还可以用鼠标左键拖动图形到新的位置。

2. 设定边界条件

用鼠标单击按钮 $|\infty|$，进入边界模式 (Boundary mode)，然后把光标移到选定的边界上，双击鼠标左键，就打开了一个对话框，在对话框中有两类边界条件，

狄里克利条件和诺依曼边界条件，选择一个并给定有关的系数，就设定了边界条件。如果是对所有的边界设定相同的边界条件，可以在按住 shift 键的同时再选取边界。在输入边界条件时，必须使用矢量化公式，例如输入 $4xy$ 的格式是 `4*x.*y`，注意表达式中表示数组运算的小圆点。

在菜单 Boundary 下，选择 Remove Subdomain Border 或 Remove All Subdomain Borders 可以将不需要的子区域的边界去掉，选择 Show Edge Labels 可以给边界编号，而 Show Subdomain Labels 可给出各个子区域的编号。

如果此时又想改变画好的几何区域或对表示区域的公式进行编辑，则应先回到画图模式才行，即要重新单击菜单 Draw /Draw mode，才能对几何区域或对表示区域的公式进行编辑。

3. 定义方程

用鼠标单击按钮 PDE，进入方程模式（ PDE mode ），并打开一个对话框，对话框中显示四种方程类型：椭圆型 (elliptic)、抛物型 (parabolic)、双曲型 (hyperbolic) 和本征值问题 (eigenmodes)。每种方程的形式和系数各不相同，它们都是用矢量运算符号书写的。可按照问题的需要在对话框中指定求解的方程类型及其系数。

其实，定义方程也可以放在解方程前的任何一个步骤之前。

4. 将网格初始化和细化

用鼠标单击按钮 △，进入网格模式 (Mesh mode)，就将求解区域分解成三角形网格。这是用有限元方法求解的所需要的，偏微分方程工具箱用的就是有限元方法。关于有限元方法的简单介绍，请看 7.4 节。如果输出的解没有达到需要的精度，就可以将网格分解为更细的网格。每单击一次按钮 △，网格就会细分一次，可根据需要单击一、二次。网格越细，解题的时间越长。

在菜单 Mesh 下，选择 Show Node Labels 和 Show Triangle Labels 可以显示节点和三角形的编号，选择 Parameters 可以设定网格细分的各种参数。

5. 求解方程

在解方程之前，应该首先在菜单 Solve 下选择 Parameters 来指定解方程的方法及初始条件等。对椭圆型方程可以选择非线性解法或自适应解法，对双曲型和抛物型要给定初始条件和解方程的时间范围，对本征值方程要指定搜索本征值的间隔。然后用鼠标单击按钮 =，进入解方程模式（ Solve mode ）并解出方程，默认的输出是颜色表示的二维图形。

6. 将结果可视化

用鼠标单击按钮 ，进入作图模式 (Plot mode)，在打开的对话框中有多种输出结果的方式可供选择. 如绘图类型 (Plot type) 指彩色图、等值线图、箭头图等，输出的内容 (Property) 如方程的解、解的梯度的绝对值等，自选输出内容 (User entry) 可以根据用户的输入的表达式来输出结果，作图方式 (Plot style) 是设定作图时的各种属性. 默认的设置是将结果输出到界面窗口中，重新设置输出参数以后则将结果输出到一个单独的图形窗口. 可以同时使用几种作图方式的设定，如颜色、高度和矢量场. 对于双曲型和抛物型方程，可选用 Animation 作动画输出. 对于本征值问题可以按本征值来选择输出的本征模.

7. 放大视图

用鼠标单击按钮 ，再将光标移到图形上，单击鼠标左键可以将视图放大，单击鼠标右键可以将放大的图形缩小.

需要强调的是，每一种操作要在相应的模式之下才能进行. 每种操作完成之后，可以在菜单 File 下选择 Save as 将结果存为 M 文件以供下次直接调用，这类 M 文件叫模型 M 文件 (Model M — file)，在本节例 2 有介绍.

在每种操作完成之后，还可以选择各个菜单下的 Export... 项将相应的数据以变量形式直接输出到指令窗口的工作内存中，在指令窗口可以用变量名来调用这些数据，

下面介绍一个应用图形用户界面解方程的实例.

7.5.2 一个简单的例子

求解的问题是齐次的狄里克利边界条件下椭圆型方程 $-\Delta = 10$，求解的过程如下.

1. 画方程的求解区域

在菜单 Options 下，选 Grid 在作图区域中画上网线，选 Snap 使得所画的图形能与网线对齐. 用鼠标选上它们以后，在这两个指令前会出现一个勾. 再用鼠标单击 Grid Spacing 打开一个对话框如图 7.17 所示.

取消 Auto 框中的勾，输入 x, y 的间隔分别为 $-1.5 : 0.1 : 1.5$ 和 $-1 : 0.1 : 1$.

然后如图 7.18 画出矩形 R1 和 R2, 正方形 SQ1, 圆 C1,C2 和 C3, 椭圆 E1. 其中 R1,SQ1 和 C1 是用来画一个圆角. 所以位置要求准确一点. 可以用前面介绍的位置对话框来确定精确的位置. 其余图形的位置不要求很精确.



图 7.17 Grid Spacing 的对话框

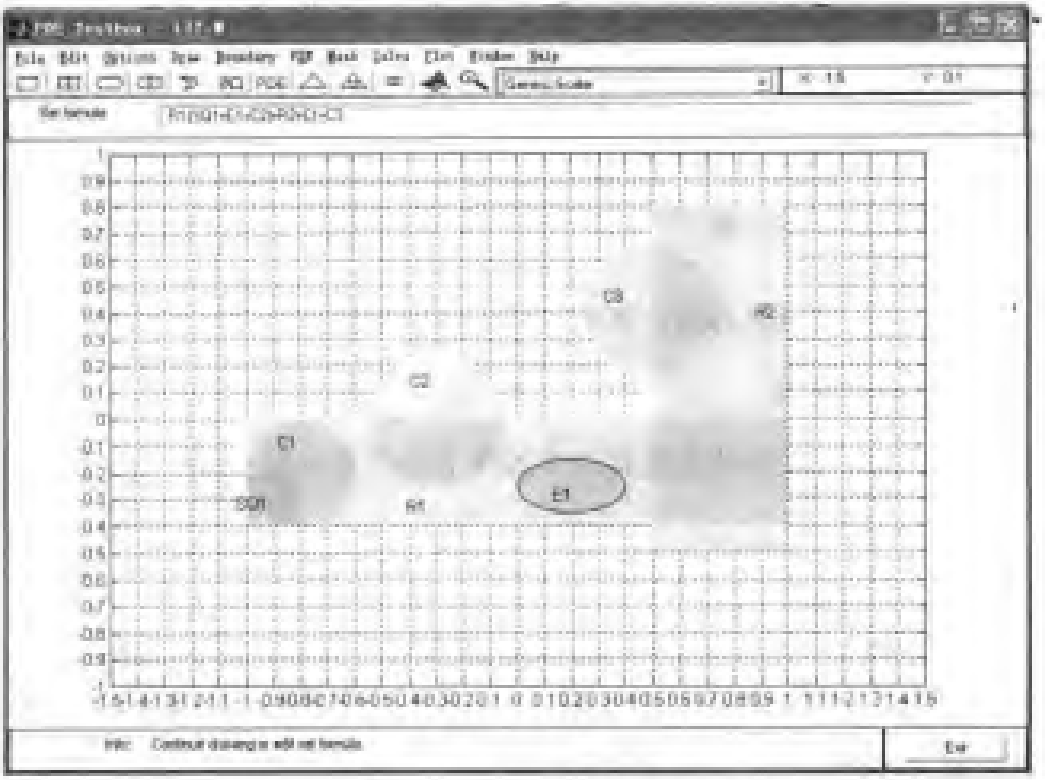


图 7.18 方程的求解区域

画好以后，在公式编辑框 (Set formula) 中，删除自动设置的公式，输入 R1−(SQ1+ E1 + C2) + R2 + C1 + C3 ，

2. 设置边界条件

单击菜单 Boundary/Boundary Mode, 进入边界模式. 单击 Remove All Subdomain Borders, 去掉所有不必要的子区域的边界, 结果如图所示.

选取所有的边界, 即按住 Shift 键, 依次单击所有的边界. 然后用鼠标单击菜单 Boundary / Specify Boundary Conditions, 打开对话框如图 7.19, 将边界条件设为齐次狄利克里条件. 实际上, 这是默认值, 所以无需改变.

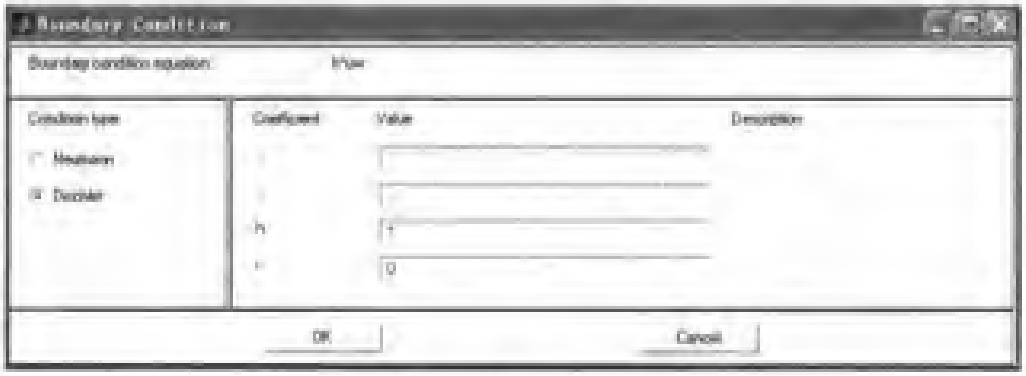


图 7.19 边界条件对话框

3. 设置方程

单击菜单 PDE/PDE MODE, 进入方程模式. 再单击 PDE Specification, 打开对话框如图 7.20 所示, 设置方程类型为椭圆型方程, 它就是默认值, 将系数 c, a, f 设为 $1, 0, 10$.

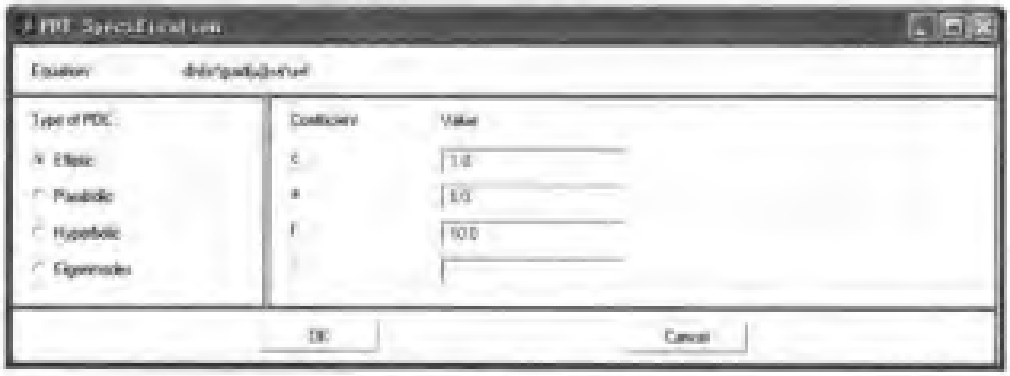


图 7.20 方程设置对话框

4. 网格分割

单击菜单 Mesh/Initialize Mesh, 进行初次网格划分. 结果如图 7.21 所示.

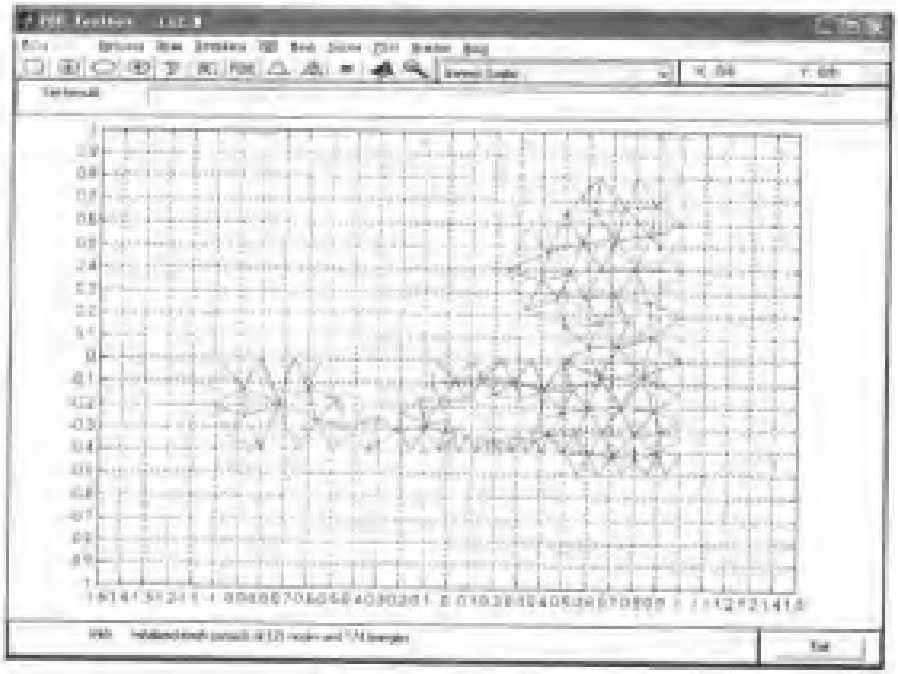


图 7.21 初次划分的网格

在用户界面的信息栏中，显示节点数目为 121，三角形的数目为 174 . 如果想显示节点的编号，可以选择 Show Node Labels 项. 如果想输出节点的坐标，单击 Export Mesh, 打开对话框如图 7.22 所示.

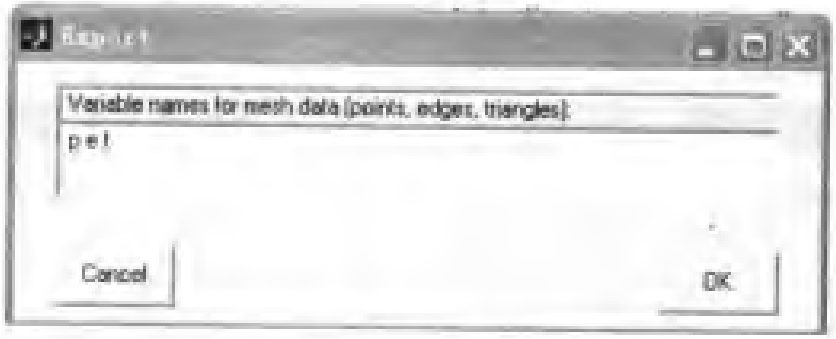


图 7.22 输出关于网格信息的变量

在打开的对话框中，默认的输出变量是

p(point) 节点坐标，是 $2 \times n$ 的矩阵， n 是节点数目.

e(edges) 边界线段的数据，是 $7 \times n$ 的矩阵， n 线段数目.

t(triangles) 三角形的数据，是 $2 \times n$ 的矩阵， n 是三角形的数目.

单击按钮 OK, 关闭对话框. 这些变量已经输出到指令窗的工作内存中. 你可以在程序中使用这些变量，也可以查看这些变量. 例如在指令窗中，依次键入 p,e,t, 就会在指令窗中显示这些数据.

单击菜单 Mesh/Refine Mesh, 再次进行网格细分, 结果如图 7.23 .

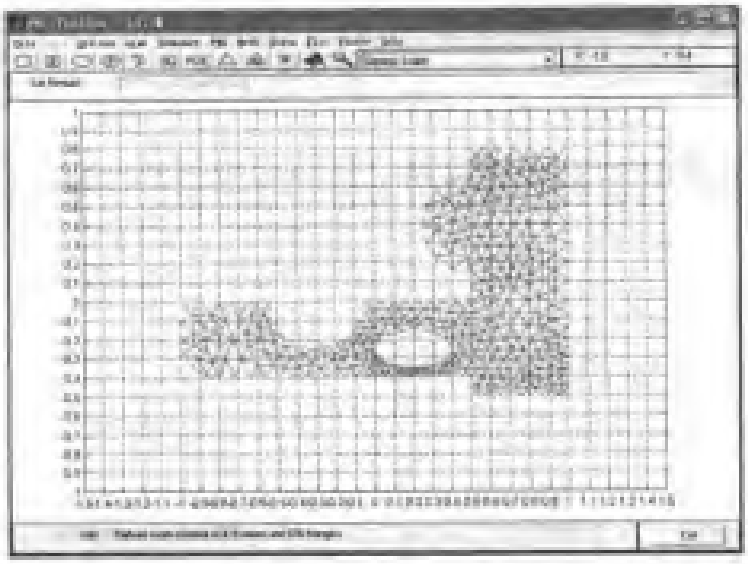


图 7.23 经过细分的网格

此时, 在用户界面的信息栏中, 显示节点数目已经变为 416, 三角形的数目为 696 .

5. 求解并将结果可视化

单击菜单 Solve/Solve PDE, 解偏微分方程并显示图形, 如图 7.24 所示.

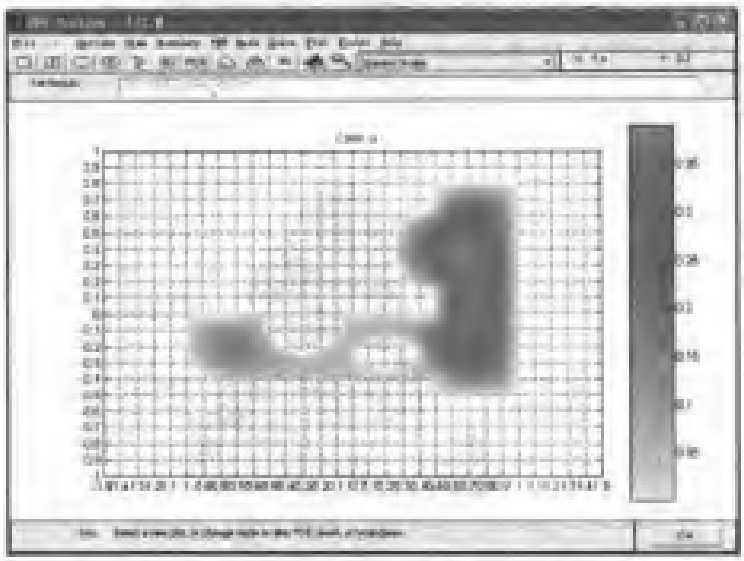


图 7.24 用色彩表示的方程的解

如果输出解的数据到指令窗中，可以单击 Solve/Export Solution，在打开的对话框中，默认的变量名是 u，如果不想改变，就关闭对话框。在指令窗中，输入 u，就能看到按节点编号排列的解的数据。

单击菜单 Plot/Parameters，打开对话框如图 7.25 。

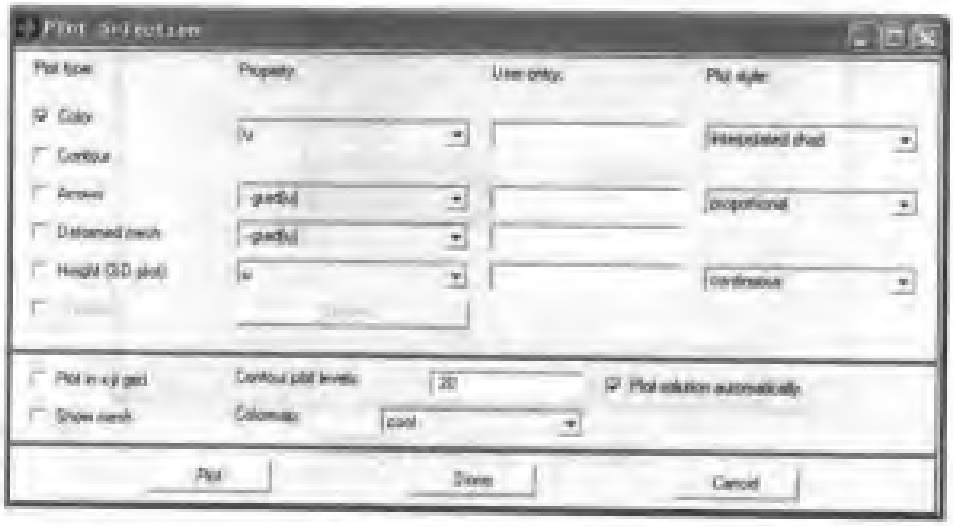


图 7.25 作图设置的对话框

在其中选取 Color,Height(3-D plot) 和 Show mesh，单击按钮 Plot，画出三维图形如图 7.26 所示。

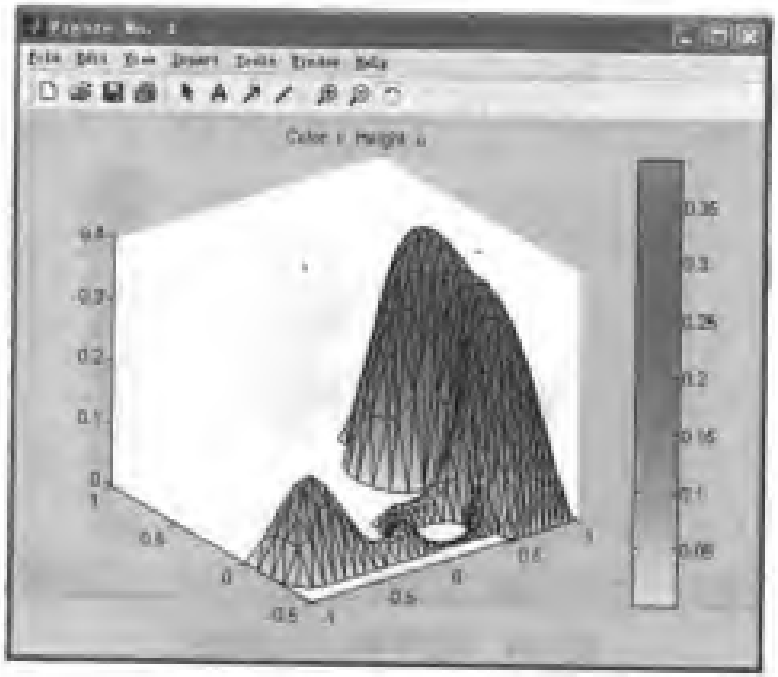


图 7.26 解的三维图形

如在对话框中选取 Contour 和 Arroows, 再单击 Plot, 可画解的等值线和矢量场图如图 7.27 所示。

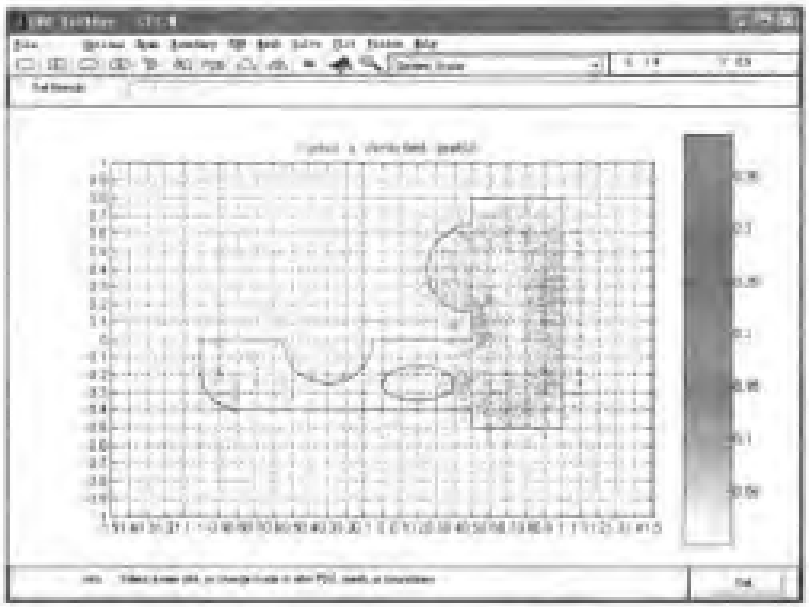


图 7.27 解的等值线和矢量场

7.6 图形用户界面窗口的菜单

虽然用图形界面上工具栏的图标可以完成解偏微分方程的工作，但是界面窗口上的菜单为解方程提供了更灵活的选择。下面详细介绍菜单中各个命令的用法。

1. File 菜单

用鼠标左键单击 File 菜单，会弹出如图 7.28 所示的子菜单。

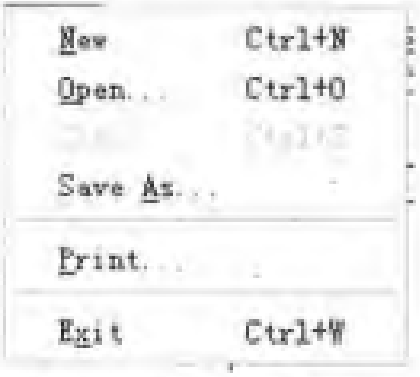


图 7.28 File 菜单的子菜单

New 新建一个组合的几何图形 (Constructive Solid Geometry,CSG)。

默认文件名为 'Untitled'，同时也将原有的 CSG 图形删除。

Open... 从硬盘打开 M 文件。

Save 将在 GUI 内完成的结果储存入当前使用的 M 文件。

Save As 将在 GUI 内完成的结果存入一个新的 M 文件。

Print... 将在偏微分方程工具箱中完成的图形送到打印机打印。

Exit 退出偏微分方程工具箱图形用户界面。

在其中，以下的指令在操作时会打开一个对话框。

Open 打开的对话框如图 7.29 所示。

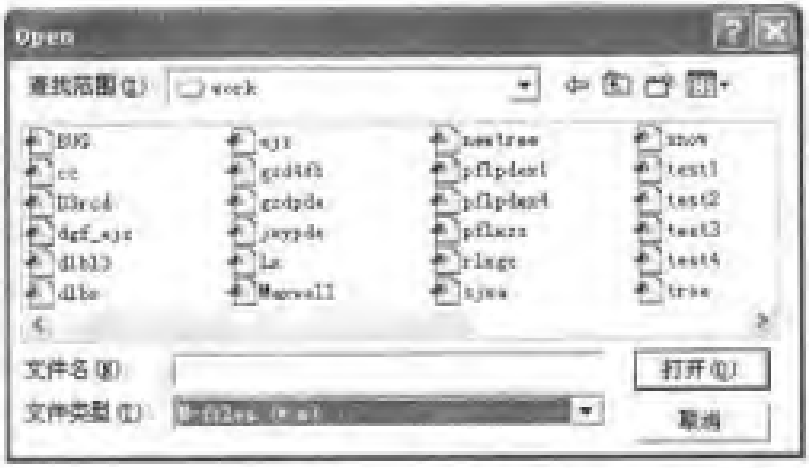


图 7.29 Open 指令的对话框

在对话框中，选定目录以后，会出现一个 M 文件列表，从中再选取要打开的文件。你应该选取在使用偏微分方程工具箱的过程中，由偏微分方程工具箱存入的 M 文件，它们的内容包括 CSG 图形，边界条件，网格的信息，求解的方程，解的结果以及可视化的输出。如果不是由偏微分方程工具箱存入的文件而是一般的程序文件，则它们不会载入到偏微分方程工具箱的用户界面上去，而是直接执行这个程序文件。

Save 打开的对话框与上图相同。

在对话框中，先选定目录，再输入所存入的文件名。所输入的文件名不必要加扩展名，MATLAB 会自动加上文件的扩展名。使用偏微分方程工具箱的任何一个步骤，都可以存为文件。下次打开这个文件时，这些操作的信息都会载入偏微分方程工具箱的用户界面。

Print 打开的对话框如图 7.30 所示。

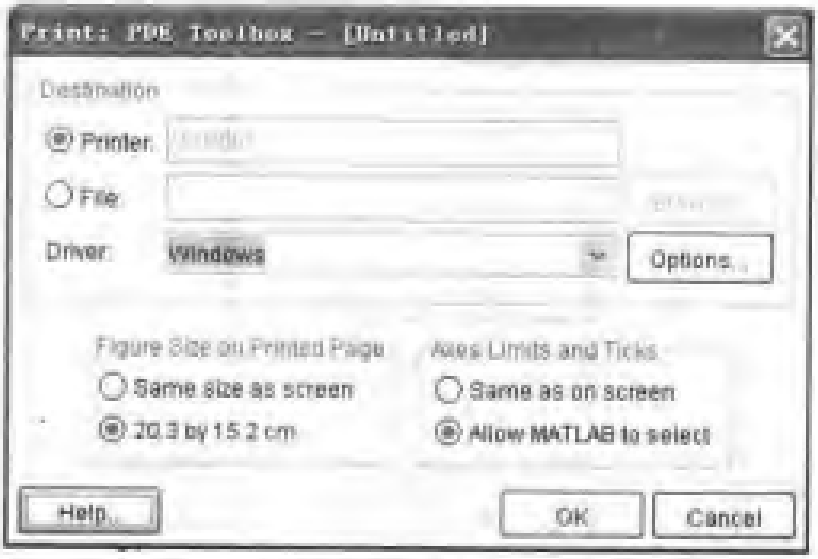


图 7.30 Print 指令的对话框

在对话框中，设置各种打印参数以后，就可以将结果打印出来。这时打印只是用户界面中间的图形部分，而不包括用户界面中上下的菜单。

2. Edit 菜单

用鼠标左键单击 Edit 菜单，会弹出如图 7.31 所示的子菜单。

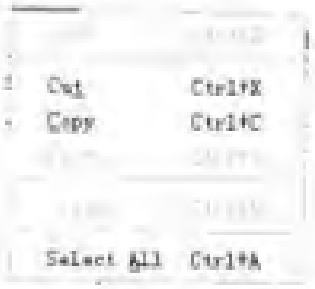


图 7.31 Edit 菜单的子菜单

Undo	在绘制多边形图形时退回到上一步的操作。
Cut	将选定的图形剪切到剪贴板。
Copy	将选定的图形复制到剪贴板。
Paste...	将剪贴板上的图形粘贴到当前的 CGS 图形上。
Clear	删除选定的图形。
Select All	删除当前所有的 CGS 图形，包括它的边界及子区域。

其中的 Paste 指令可以打开一个对话框如图 7.32 所示。



图 7.32 Paste 指令的对话框

在对话框中可以在 x, y 两个方向指定剪贴板的图形从原来的位置移动的距离，还可以指定重复复制的次数，默认值是移动距离为零而复制的次数为 1 。可以用鼠标移动画好的图形，方法是将光标指向要移动的图形，按住左键，然后移动光标，图形就会移动到新的位置。

3. Options 菜单

用鼠标左键单击 Options 菜单，会弹出如图 7.33 所示的子菜单。



图 7.33 Options 菜单下的子菜单

Grid	在绘图时添加或取消网格线
Grid Spacing...	改变网格线的间距
Snap	启动或关闭画图时与网格线对齐的功能
Axes Limits...	设置绘图时坐标轴的范围
Axes Equal	x 轴与 y 轴等比例 (此时画的正方形与圆在视觉上才像一个正方形与圆)
Turn Off Toolbar Help	关闭工具栏的按钮的文字帮助信息
Zoom	启动或关闭图形缩放功能
Application	选择应用模式

Refresh 重新显示图形用户界面中所有的图形

在其中，以下的指令在操作时会打开一个对话框。

Grid Spacing 打开的对话框如图 7.34 所示。



图 7.34 Grid Spacing 打开的对话框

在对话框中，如果取消 Auto 框中的勾，就可以改变 x 轴与 y 轴网格线的间距，这个间距是线性的，默认值是 $-1.5:0.5:1.5$ 和 $-1:0.2:1$，同时还可以任意增加额外的网格线。单击 Apply 按钮将应用输入的数值，单击按钮 Done 完成设置并退出对话框。

Axes Limits 打开的对话框，可以改变 x 轴与 y 轴的范围。

Application 会打开下一级子菜单，这些子菜单所包含的项目与工具栏的长方形框中所包含的项目是一样的，可以参看 7.4.4 节的说明。

4. Draw 菜单

用鼠标左键单击 Draw 菜单，会弹出如图 7.35 所示的子菜单。

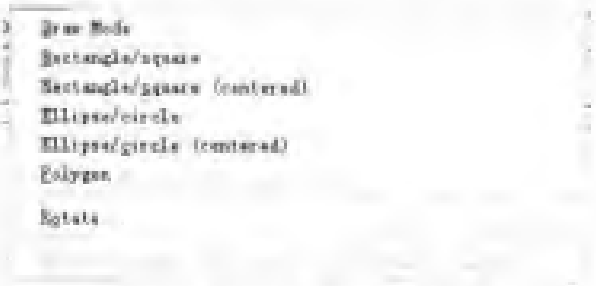


图 7.35 Draw 菜单下的子菜单

Draw Mode 进入作图模式。

Rectangle/square	从顶点开始画一个矩形 / 正方形。用鼠标左键单击并且拖动光标，就画出了一个矩形。用鼠标右键 (或者 Ctrl + 左键) 单击并且拖动光标，就画出了一个正方形。
Rectangle /square(centered)	从中心开始画一个矩形 / 正方形。用鼠标左键单击并且拖动光标，就画出了一个矩形。用鼠标右键 (或者 Ctrl + 左键) 单击并且拖动光标，就画出了一个正方形。
Ellipse/circle	从周边上一点开始画一个椭圆 / 圆。用鼠标左键单击并且拖动光标，就画出了一个椭圆。用鼠标右键 (或者 Ctrl+ 左键) 单击并且拖动光标，就画出了一个圆。
Ellipse /circle(centered)	从中心开始画一个椭圆 / 圆。用鼠标左键单击并且拖动光标。就画出了一个椭圆。用鼠标右键 (或者 Ctrl+ 左键) 单击并且拖动光标，就画出了一个圆。
Polygon	画多边形。用鼠标左键单击并且拖动光标，就画出了多边形的一条边。松开左键，将光标移动到下一个顶点，单击左键，就画出了下一条边。依次可以画出多边形的各条边。封闭多边形也可以直接单击右键。
Rotate...	旋转所选择的图形。
Export Geometry Description,Set Formula,Labels...	将几何描述矩阵 gd, 公式设定的字符串 sf, 已经命名的空间矩阵 ns(包括标志) 输出到主工作空间中去。

在其中，指令 Rotate 在操作时会打开一个对话框如图 7.36 所示。



图 7.36 Rotate 指令打开的对话框

选定一个要旋转的图形，打开对话框，输入旋转的角度，然后确认，即可旋转图形。默认的旋转中心是原图形的质心，如果取消 Use center-of-mass 框中的勾，则可以任意输入一个数组如 [−0.4, 0.3] 作为转动中心的 x, y 坐标。

5. Boundary 菜单

用鼠标左键单击 Boundary 菜单，会弹出如图 7.37 所示的子菜单。

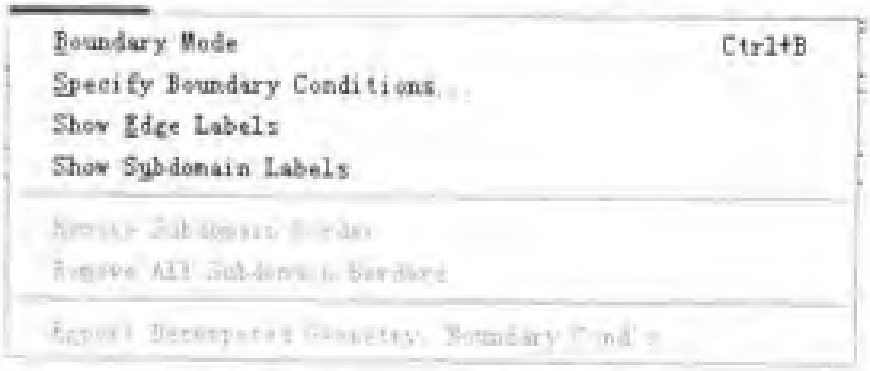


图 7.37 Boundary 菜单的子菜单

Boundary Mode	进入边界条件设置模式。
Specify Boundary Conditions...	对于选定的边界设置边界条件。如果没有选定边界，则是对所有的边界设置边界条件。
Show Subdomain Labels	显示边界（外边界和子区域的交界）标识。标识是根据分解几何矩阵的列数来确定。
Remove Subdomain Border	显示子区域的标识，标识是根据分解几何矩阵中的子域数来确定。
Remove All Subdomain Borders	取消所选定的子域边界。
Export Decomposed Geometry,	取消所有的子域边界。
Boundary Cond's...	输出分解几何矩阵 g 和边界条件矩阵 b。

在其中，Specify Boundary Conditions 指令在操作时会打开一个对话框如图 7.38 所示。

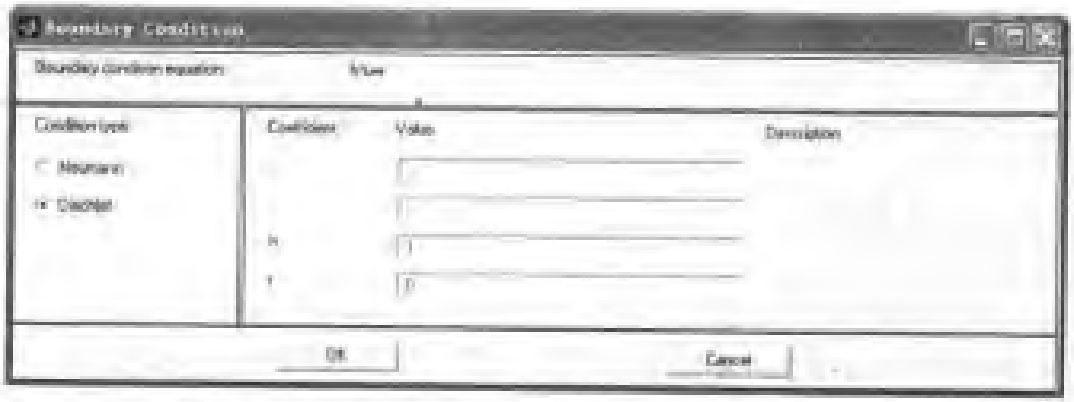


图 7.38 Specify Boundary Conditions 指令的对话框

在对话框中，可以对选定的边界线设定边界条件，有三种不同的边界条件类型：

(1) 广义的诺伊曼条件，它由下式

$$n \cdot (c\nabla u) + qu = g$$

中的系数 q 和 g 来决定。在方程组的情况下，q 是 2×2 矩阵而 g 是 2×1 的矢量。

(2) 狄里克利条件，它用等式 $hu = r$ 给出边界上的函数值，h 是权重因子，通常在归一化的情况下它为 1。在方程组的情况下，h 是 2×2 矩阵而 r 是 2×1 的矢量。

(3) 混合边界条件 (只是对方程组而言)，它是狄里克利边界条件和诺伊曼边界条件的组合。q 是 2×2 矩阵而 g 是 2×1 的矢量，h 是 1×2 矩阵而 r 是标量。

图 7.39 所示的对话框是在方程组的情况下设置边界条件。



图 7.39 解方程组的边界条件

在画好了一个区域后，在菜单下选 Option/Application/Generic System，然后进入边界条件设置模式，选菜单 Specify Boundary Conditions ... 就能打开这个对话框。

在输入边界条件时，以下的表达式是有效的：

(1) 二维的 x, y 坐标；

(2) 边界线段的长度参数 s，s 在边界线上沿着箭头的方向从 0 增加到 1；

(3) 外法向方向矢量的分量 n_x 和 n_y。如果需要切向矢量，可以通过 $t_x = -n_y$ 和 $t_y = n_x$ 来表示；

(4) 解函数 u；

(5) 时间 t，

注意，如果边界条件是解 u 的函数，就必须使用非线性求解算法。如果边界条件是时间 t 的函数，则必须选抛物型或双曲型方程。

例如，(100-80.*S) 和 cos(x.^2) 都是合法的表达式。

在非一般的应用模式下，在对话框中的 Decription 栏下包含了对边界条件参数的物理描述。

6. PDE 菜单

用鼠标左键单击 PDE 菜单，会弹出如图 7.40 所示的子菜单。

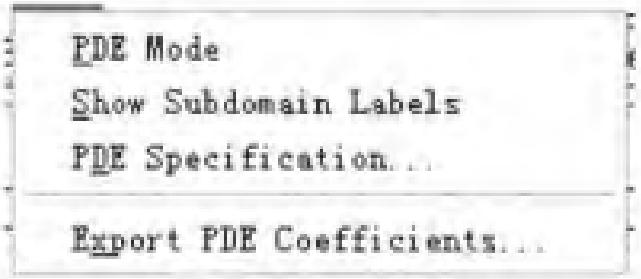


图 7.40 PDE 菜单打开的子菜单

PDE Mode	进入偏微分方程模式。
Show Subdomain Labels	显示子区域的标识。子区域是以它在分解几何矩阵中编号来标识的。
PDE Specification...	设置方程的系数和类型。
Export PDE Coefficients...	输出方程的系数。这些系数是字符变量。

在其中，PDE Specification 指令会打开一个对话框如图 7.41 所示。

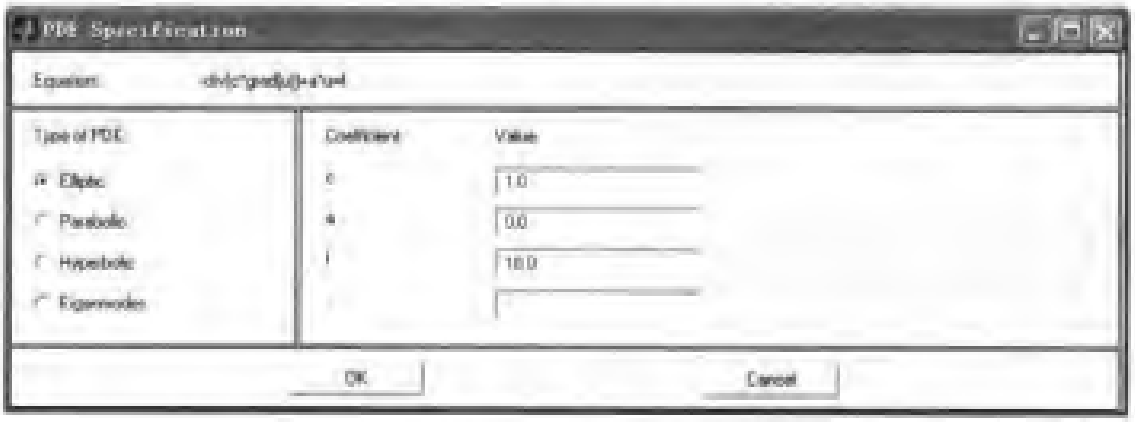


图 7.41 PDE Specification 指令打开的对话框

在对话框中，可以选择方程的类型及输入需要的参数。参数的维数取决于方程的维数。下面的描述适用于标量偏微分方程。如果选择的非一般的应用模式，则专门的偏微分方程和系数将替代标准的偏微分方程的系数。对不同的模式的具体应用，可参看后面的例子。

用 MATLAB 许可的表达式来表示的系数 c, a, f 和 d 将用于计算三角形的质心的参数。在表达式中，以下的变量是可以使用的：坐标 x, y 解 u，解的偏导数 u_x, u_y，时间 t 。也可以输入能接受变量 $p, t, u, time$ 的函数文件，如 circlef 。

注意，如果边界条件是解 u 的函数，就必须使用非线性求解算法。如果边界条件是时间 t 的函数，则必须选抛物型或双曲型方程。

系数 c 可以是标量或 2×2 的矩阵。矩阵 c 用于诸如各向异性的材料问题。如果 c 是两行，它们就是由 c_{11} 和 c_{22} 为对角元素组成的 2×2 矩阵 $\begin{pmatrix} c_{11} & 0 \\ 0 & c_{22} \end{pmatrix}$。如果 c 有三行，则它是由 c_{11},c_{12} 和 c_{22} 组成的对称的矩阵 (c_{12}=c_{21})，而 c_{22} 是 2×2 的矩阵 $\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$。如果 c 有四行，则 c_{11},c_{12},c_{21} 和 c_{22} 都是 2×2 的矩阵。

偏微分方程的类型有以下四种：

(1) 椭圆型。基本形式为

$$-\nabla\cdot(c\nabla u)+au=f$$

(2) 抛物型。基本形式为

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f$$

具有初始条件为 $u_0=u(t_0)$ 。

(3) 双曲型。基本形式为

$$d\frac{\partial^2 u}{\partial t^2}-\nabla\cdot(c\nabla u)+au=f$$

具有初始条件为 $u_0=u(t_0), u_0'=u'(t_0)$ 。

(4) 本征值方程。基本形式为

$$-\nabla\cdot(c\nabla u)+au=\lambda du$$

对方程组而言，c 是一个秩为 4 的张量，可以表示成 4 个 2×2 的矩阵 c_{11}, c_{12}, c_{21} 和 c_{22} 。可以是一行，二行，三行或四行的，详情见上面对标量情况的说明。 a 和

d 是 2×2 的矩阵，f 是 2×1 的矢量。这时 PDE Specification 指令打开的对话框如图 7.42 所示。

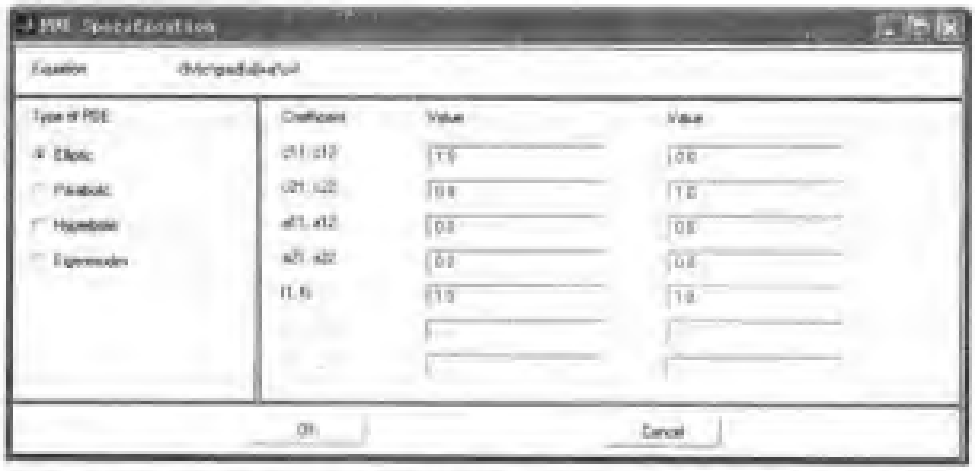


图 7.42 方程组情况下 PDE Specification 指令打开的对话框

7. Mesh 菜单

用鼠标左键单击 Mesh 菜单，会弹出如图 7.43 所示的子菜单

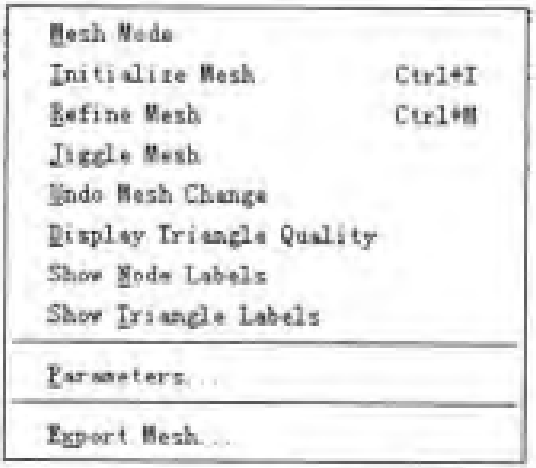


图 7.43 Mesh 菜单打开的子菜单

Mesh Mode	进入网格模式。
Initialize Mesh	建立和显示初始化的三角形网格。
Refine Mesh	细分当前的三角形网格。
Jiggle Mesh	优化网格。
Undo Mesh Change	取消上次对网格的操作。所有对网格的操作都会被保存，所以重复操作 Undo Mesh Change 可以

	回到最初的网格。
Display Triangle Quality	用颜色表示各个三角形网格的质量。颜色代表的数值在 0 ~ 1 之间。数值大于 0.6 的三角形的网格是可以接受。更详细地解释可参阅指令 pdetriq 的说明。
Show Node Labels	显示网格节点的标识。节点的标识是根据节点矩阵 p 的列数。
Show Triangle Labels	显示三角形网格的标志。三角形网格的标识是根据三角形网格矩阵的 t 的列数。
Parameters...	输入网格生成的参数。
Export Mesh	输出节点矩阵 p、边界矩阵 e 和三角形网格矩阵 t。

在其中，指令 Parameters 在操作时会打开一个对话框如图 7.44 所示。



图 7.44 指令 Parameters 打开的对话框

在对话框中输入的参数可对网格初始化算法 Initmesh 中的参数进行调整。各个参数的含义是：

Maximum edge size 三角形最大的边长 (近似值)。这个参数可以任意设定，但必须是正数。

Mesh growth rate 网格增长的比例。网格的大小是从图形的小的部分开始增长。其值在 1 ~ 2 之间。默认值为 1.3，也即增长的大小为 30%。

Jiggle mesh 启动自动优化初始网格的算法 jiggling。

在网格优化的算法 jigglemesh 中使用的参数有以下几种：

Jiggle mode 带有一个下拉的菜单，其中的选项有 on,optimize minimumt, optimize mean.on 只对网格作一次优化. optimize minimumt 是将网格优化的过程反复进行直到最小的三角形的质量不再提高或者达到优化次数的极限。同样，optimize mean 也是将网格优化的过程反复进行，不过它提高的是三角形的平均质量。

Number of jiggle optimize minimumt, optimize mean 模式下优化的次数，默认值是 20 .

Refinement 用于网格细分的算法 refinemesh 的参数，有 regular 和 longest 两种，默认值是 refular, 它产生均匀的网格. 而 longest 则总是将每个三角形的最长边细分.

8. Solve 菜单

用鼠标左键单击 Solve 菜单，会弹出如图 7.45 所示的子菜单

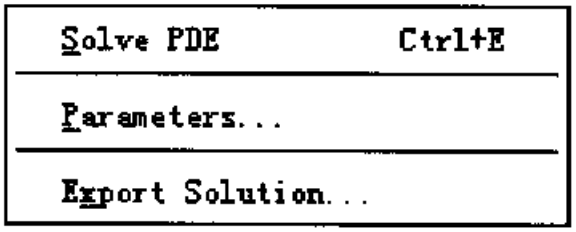


图 7.45 Solve 菜单的子菜单

Solve PDE	根据当前的组合几何图形，三角形网格以及作图 (自动作图的功能可以取消) 的设置来求解偏微分方程.
Parameters...	输入解方程所需的参数.
Export Solution...	输出解矢量 u . 如果有的话，还可以输出本征值.

在其中， Parameters 指令在操作时会打开一个对话框。由于不同的方程要设定不同的参数，所以打开的对话框随方程的不同而不同。有以下几种情况：

1) 椭圆型方程

在默认的情况下，不需要设置参数. 椭圆型方程是用基本的椭圆型方程解法 assempde . 另一种解法是用自适应网格生成和解法 adaptmesh . 对于自适应模式，打开的对话框是如图 7.46 所示。



图 7.46 椭圆型方程的参数设置对话框

可应用的参数有以下几种:

Adaptive mode 打开自适应模式.

Maximum number of triangles 新的三角形网格的最大数目 (可以是无穷大). 默认值是根据当前的网格数来计算的.

Maximum number of refinements 设置随后的对网格细分的数目.

triangle selection method 三角形网格的选择方法. 偏微分方程工具箱本身可以提供两种方法, 你还可以提供你自己的方法. 下面加以详细说明.

Worst triangles 最坏的三角形. 这个方法是设定一个最差的三角形的分数值 (默认值是 0.5), 然后剔除所有的比这个三角形更差的三角形. 详情可参看指令 pdeadworst .

Relative tolerance 相对误差. 这个方法是根据相对误差的判据 (默认值是 1E−3) 来剔除三角形. 详情参看指令 pdeadgsc .

User-defined function 输入用户自定义的三角形网格选择方法的名称. 演示程序 pdedemo7 就是一个用户自定义三角形网格选择法的例子.

Function parameter 函数参数. 它能更细致地调节三角形网格的选择方法. 对于 Worst triangles (pdeadworst) 法, 调整的是最差的分数值, 用它来确定要细分的三角形. 对于 Relative tolerance 法, 调整的是误差参数, 以此来使解更好地符合方程.

Refinement method 细分的方法, 有 regular(均匀的) 和 longest(最长边的) 两种. 详情参看对菜单 Mesh 下的对话框 Parameters 的说明.

对于非线性问题，也即当偏微分方程的参数是直接依赖于解 u 的时候，必须应用非线性解法 (非线性解法是可以和自适应模式一起使用的)。可应用的参数有以下几种：

Use nonlinear solver 启动非线性算法。

Nonlinear tolerance 设置非线性算法的误差参数。

Initial solution 初始的猜测值，可以是一个常数，也可以是 MATLAB 允许的一个 x, y 的表达式，表达式是在当前的网格节点上赋值。例如 1 和 $e^{x \cdot *y}$，默认值是 0。

Jacobian 雅可比逼近法，包括定点迭代法 (默认值)，分块的，近似为 "分块的" (对角的) 以及完全的雅可比法。

Norm 范数。各种范数是用来计算残数的。对于能量范数即输入能量值，为了给定 l_p 范数，则输入实标量 p，默认值为无穷，即无穷范数。

2) 对抛物型方程

打开的对话框如图 7.47 所示。



图 7.47 抛物型方程的参数设置对话框

求解的参数有以下几种：

Time 指定求解方程的时间范围。相关的时间间隔取决于问题的运动状况。如可取 0 : 10 或 logspace(−2, 0, 20)。

u(t0) 解方程的所需的初值 u(t0)，它可以是一个常数或在当前网格的节点上赋值的一个列矢量。

Relative tolerance 相对误差，为常微分方程的解法而设置。此处这个解法是用来求解抛物型方程的时间部分。

Absolute tolerance 绝对误差，为常微分方程的解法而设置。此处这个

解法是用来求解抛物型方程的时间部分。

3) 对双曲型方程

打开的对话框如图 7.48 所示：

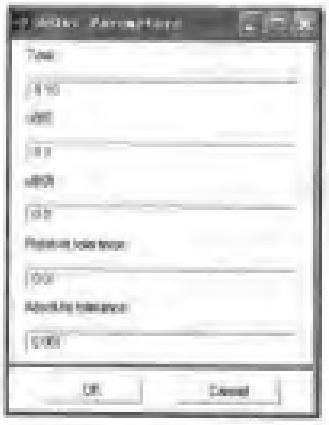


图 7.48 双曲型方程的参数设置对话框

求解的参数有以下几种：

Time 指定求解方程的时间范围。相关的时间间隔取决于问题的运动状况。如可取 $0:10$ 或 $\mathrm{logspace}(-2,0,20)$。

u(t0) 解方程的所需的初值 u(t0)，它可以是一个常数或在当前网格的节点上赋值的一个列矢量。

u'(t0) 解方程的所需的初值 u'(t0)，设置方法同 u(t0)。

Relative tolerance 相对误差，为常微分方程的解法而设置。此处这个解法是用来求解双曲型方程的时间部分。

Absolute tolerance 绝对误差，为常微分方程的解法而设置。此处这个解法是用来求解双曲型方程的时间部分。

4) 对本征值方程

打开的对话框如图 7.49 所示。



图 7.49 本征值方程的参数设置对话框

在对话框中，可设置的参数仅仅是 Eigenvalue search range，一个仅有两个元素的矢量，它定义了在实轴上搜索本征值的范围，左端可设为 −Inf(无穷大)，如 [0 100], [−Inf, 50]。

9. Plot 菜单

用鼠标左键单击 Plot 菜单，会弹出如图 7.50 所示的子菜单。

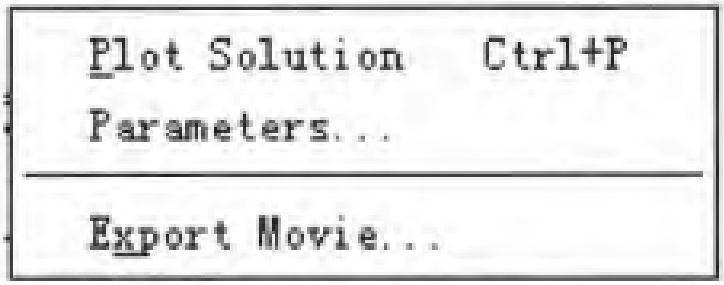


图 7.50 Plot 菜单的子菜单

Plot Solution 显示解的图形。
Parameters... 打开作图选项的对话框。
Export Movie 输出录制的动画矩阵 M。

在其中，Parameters 指令在操作时会打开一个对话框如图 7.51 所示，在对话框中可设置控制作图与可视化的各个选项。

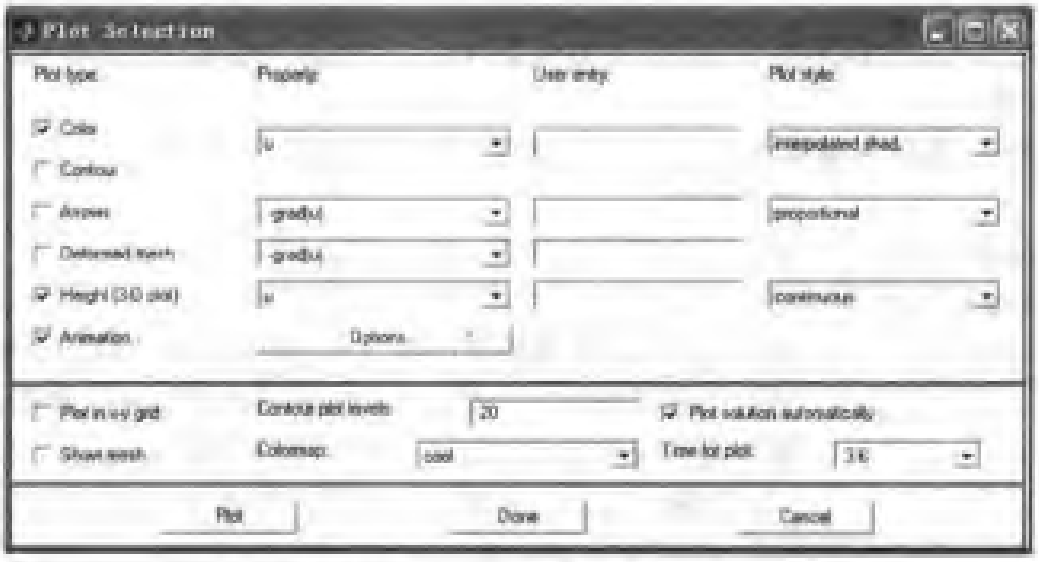


图 7.51 Parameters 指令的对话框

对话框的上半部共有 4 列，分述如下：

1) Plot type 有 6 种不同的方式作图来实现可视化.

Color 在物体表面上着色以表示对象的标量属性。

Contour 用带色的等值线表示对象的标量属性. 如果 Color 和 Contour 都选，则它会加强 Color 可视化的效果，此时，等值线是画成黑的。

Arrows 用箭头来表示对象的矢量属性.

Deformed mesh 用表示矢量属性的网格的形变来实现矢量属性的可视化. 在问题涉及的区域内，形变的尺度将会自动控制在 10% . 这种作图方式主要用于结构力学中，表示 x,y 方向的位移 (u,v) . 如果不选取其他作图方式，则变形的三角形网格就自动显现出来。

Height(3-D plot) 用三维图形中的高度 (z 轴) 来表示对象的标量属性. 三维图形是画在一个新的图形窗口中. 如果没有选取 Color 和 Contour 的作图方式，则三维图形只是一个简单的网格图形. 如果同时使用 Color 和 (或)Contour 的作图方式，则会出现另一种三维表面图与等值线图合成的图形.

Animation 生成动画，适用于含时的抛物型方程和双曲型方程的解. 如果选这一项，方程的解就会记录下来，并用指令 movie 在另一个窗口中放映。

2) Property 作图方式所表示的属性. 共有 4 个弹出式菜单，菜单中的列表就是可以用这种作图方式表示的属性. 在第一个弹出式菜单中，是用 Color 和 (或)Contour 的作图方式能表示的属性. 在第二，第三个弹出式菜单中，是用 Arrows 和 Deformed mesh 作图方式可以表示的矢量属性. 第四个弹出式菜单是用 Height(3-D plot) 作图方式可以表示的标量属性. 属性的列表与你当前所选的应用模式有关.

对于一般的标量模式，可用于可视化的标量属性有以下几种：

u	方程的解.
abs(grad(u))	每个三角形中心的 ∇u 绝对值。
abs(c*grad(u))	每个三角形中心的 $c*\nabla u$ 绝对值.
user entry	在 user entry 编辑框中可输入由用户定义的 MATLAB 的表达式. 它返回的是当前的三角形网格的节点或三角形上的数据矢量. 可使用的变量有，解 u 及其导数 u_x,u_y,$c*\nabla u$ 的 x,y 分量 cu_x,cu_y, 还有 x,y 本身. 例如 $u.*u$, $x+y$.

对于一般的标量模式，可用于可视化的矢量属性有以下几种：

-grad(u)	解 u 的负梯度 $-\nabla u$.
−c*grad(u)	c 乘以解 u 负梯度 $-c\cdot\nabla u$.

user entry 在 user entry 编辑框中可输入由用户定义的 MATLAB 的表达式 $[px;py]$。它返回的是定义在当前网格的三角形上的一个 $2\times n$ 的数据矩阵。n 是当前网格的三角形数目。三角形中心的值是由节点插值得到的。可使用的变量有，解 u 及其导数 u_x, u_y，$c*\nabla u$ 的 x,y 分量 cu_x, cu_y，还有 x,y 本身。如 $[ux;uy]$，$[x;y]$。

对于一般的方程组模式，可用 Color,Contour 和 Height(3-D plot) 表示的属性有 **u, v, abs(u,v), user entry**，而用 Arroows 和 Deformed mesh 表示的属性有 **u, v, user entry** 。在结构力学中，u,v 分别表示 x,y 方向的位移。

对于其他应用模式的可视化 (关于应用模式的介绍参看 7.4.4 节)。注意，对 user entry 编辑框中可输入由用户定义的 MATLAB 的表达式而言，无论是所有的标量应用模式还是方程组模式，可使用的变量都相同。在标量应用模式下，解总是用 u 表示，而在方程组模式，则用 v 表示。

3) **User entry** 包含 4 个编辑框，用户可以在其中输入自己的表达式。只有在编辑框的左边的弹出式菜单选择了 user entry，才能在编辑框中输入自己的表达式。比如可以输入方程解析解的表达式，然后画解析解的图，也可输入方程数值解与解析解之差的表达式，然后画二者的误差图形。

4) **Plot style** 包括三个弹出式菜单，分别用于控制 color,arrow 和 height plot 三种作图方式。其中控制表面颜色的对 color 的设置有以下几种：

interpolated shad 采用选定的色图进行插值着色，即对于每个三角形区域用线性插值着色 (默认值)。

flat shading 采用选定的色图进行平坦方式着色，即每个三角形区域都用相同的颜色。

对 arrow 的设置有以下几种：

propotional 箭头的长度对应着可视化的属性的大小 (默认值)。

normalized 所有的箭头具有相同的长度。它的好处在于，如果你感兴趣的只是矢量场的方向，那么，即使有某些区域的矢量场的值很小，它的方向也是很清楚的。

对 height plot 的设置有以下几种：

continuous 从三角形的中点到网格的节点用插值法产生一个连续的曲面 (默认值)。

discontnuous 产生一个不连续的曲面，即每个三角形的数据和 z 方向的高度是个常数。

方程的解总共有三个属性，两个标量属性和一个矢量属性。它们可以同时显

示. 如果关闭 height plot 作图方式, 那么画出的解的图形是二维的, 并且是画在偏微分方程工具箱的图形用户界面内. 否则就是在一个另外的窗口画出一个三维的图形. 如果你喜欢在一个新的图形窗口作图, 只要在 MATLAB 指令窗口中键入指令 figure 即可.

在对话框的中部, 还有下面一些辅助的作图选项:

Plot in x–y grid 选取此项以后, 会将三角形的网格转变为 $x-y$ 方向的矩形网格. 它对作动画特别有用, 因为它能明显地加快处理每幅动画的速度.

Show mesh 选取此项以后会将曲面图中的网格画成黑色, 而默认的设置是隐藏网格.

Contour plot levels 在等值线图中所画的等值线的条数. 比如可以输入 15 或 20, 默认值是 20 . 也可以直接输入一个矢量, 用以指明画等值线的位置. 例如 [0 : 100 : 1000],logspace[−1, 1, 30] .

Colormap 是一个弹出式菜单, 从中可以选定一个色图. 可选项有: cool, gray, bone, pink, copper, hot, jet, hsv 和 prism .

Plot solution automatically 通常都要选定此项, 如果关闭它, 则方程解的图形不会即时的显现出来. 然而, 新的解还可以用这个对话框画出来.

对于抛物型方程和双曲型方程, 在对话框的右下部, 还有一个弹出式菜单

Time for plot 在弹出式菜单中显示是一个时间序列, 从中可以选择一个时间, 然后画出与其对应的解来. 默认的设置是画到最后一个解.

如果启动了作动画的指令 Animation, 就会看到一个按钮 Options..., 按下这个按钮, 又可以弹出一个对话框如图 7.52 所示.



图 7.52 Animation Options 对话框

在其中可以设置如下参数以控制所作的动画.

Animation rats(fps)	每秒播放动画的帧数 (fps) .
Number of repeats	循环播放动画的次数.
Replay movie	重复播放动画而不用重新录制. 如果没有已经存好的动画, 则这项不可用.

对于本征值问题, 对话框的右下部有一个弹出式菜单, 其中有全部的本征值. 所作的本征矢量的图形是对应你在其中选定的本征值. 默认的设置是选择最小的本征值.

对于三维图形, 单击它并按住鼠标右键, 然后移动鼠标, 就可以旋转图形. 为了引导旋转, 会出现一个包围图形的方框. 松开鼠标, 图形就会按照新的视角重新画出. 而最初画图的视角是水平转角为 $-37.5°$, 天顶角为 $30°$.

在完成以上步骤以后, 单击对话框中的 Plot 按钮, 方程的解就会按照当前的设置画出来. 如果没有现成的解可供利用, 则偏微分方程工具箱会先解方程, 然后画出新解. 而对话框则会保留在屏幕上.

若单击 Down 按钮, 则对话框会关闭. 当前的设置会保存但没有新图形产生.

若单击 Cancel 按钮, 对话框也会关闭, 上次作图时的设置不会被改变.

10. Window 菜单

在这个菜单中, 你可以从所有已经打开的 MATLAB 的图形窗口挑选你要的图形窗口, 将它移到前台.

11. Help 菜单

Help...	显示帮助信息.
About...	显示版本信息.

7.7 工具箱的指令

虽然偏微分方程工具箱有一个十分方便的操作界面, 但是使用偏微分方程工具箱的指令能够完成一些更灵活的任务, 比如想画出比直线、椭圆、多边形更复杂的几何区域, 设定非标准形式的边界条件, 指定更复杂的方程系数等.

不同的指令使用不同的数据, 要学会使用指令就要弄清数据结构与指令间的关系.

工具箱求解的问题的定义与求解已经反映在图形用户界面的设计中, 多种多样的数据结构定义了问题的不同侧面和处理过程的不同阶段, 在每个阶段都会从旧的数据产生新的数据.

图 7.53 所示的流程图表现了求解过程. 在流程图上, 矩形框内是指令, 椭圆是用矩阵或 M 文件表示的数据, 箭头表示指令所要用的数据.

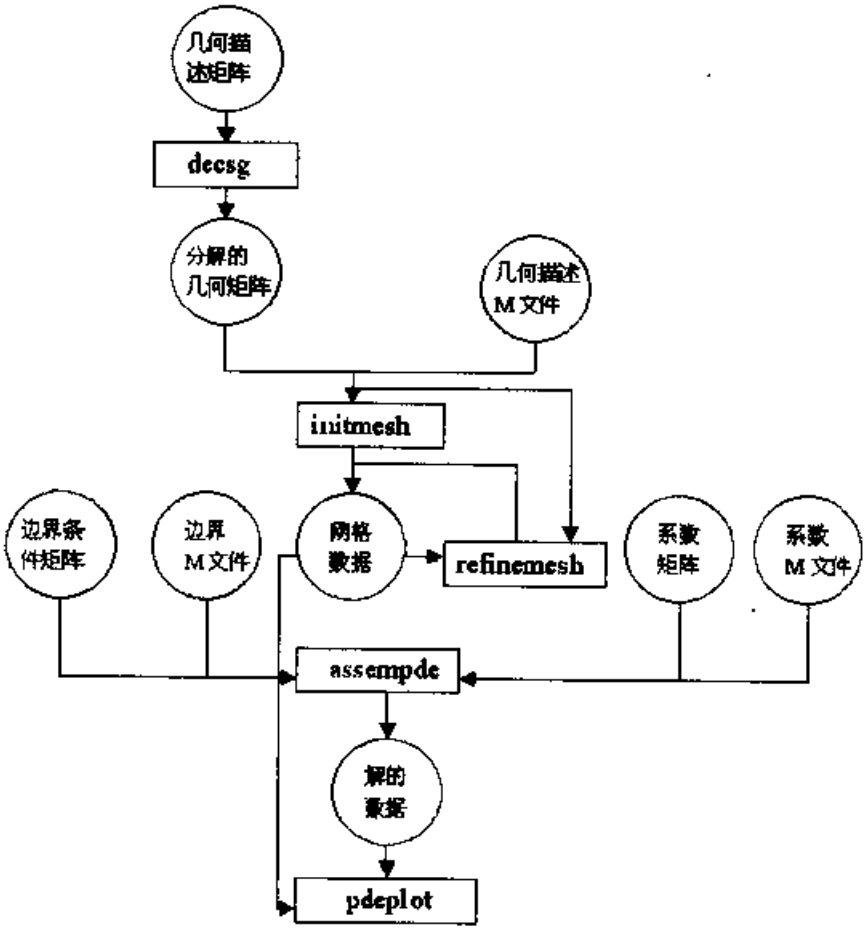


图 7.53 指令求解偏微分方程的流程图

框图中有确定的方向, 用户可以用自己提供的数据来取代其中的某一部分, 然后再执行后面的步骤. 下面对不同的数据结构和 M 文件给予简要的描述.

(1) 几何区域内部 (CSG Model)

用一个几何描述矩阵、一组公式、或一个命名的空间矩阵来加以说明. 这些数据结构可以参看指令 decsg 的用法. 在这个层次上, 问题的几何描述是一组重迭的实体. 它也可以由图形用户界面在画图模式下输出.

(2) 分解的几何区域

由分解几何矩阵或几何 M 文件来描述. 在这个层次上, 几何区域被描述为一组由边界线和交界线所分开的最小区域. 它也是由指令 decsg 所建立. 也可以由图形用户界面输出. 与分解几何矩阵等效的几何 M 文件可以由指令 wgeom 建立. 它们的数据结构可以参看指令 decsg 和 pdegeom .

(3) 边界条件

由矩阵或 M 文件来表示。边界条件由边界线上的函数来定义。矩阵可由图形用户界面输出。 M 文件可由指令 wbound 建立。边界条件的数据结构参看指令 assempde 和 pdebound 。

(4) 方程系数

由系数矩阵或系数 M 文件描述每个方程的 c,a,f,d 。系数是每个子区域上的函数。它也可由图形用户界面输出。

其数据结构参看指令 assempde 。

(5) 网格

一个三角形网格由网格数据来描述，它包括节点矩阵、边界矩阵和三角形矩阵.在网格中,最小区域分割为三角形的子区域而边界或交界则是它们的边缘.由分解区域建立的网格数据可由指令 initmesh 实现，再由指令 refinemesh 和 jigglemesh 进一步细化。指令 adaptmesh 建立的网格是求解过程的一部分。网格可以由指令 pdemesh 画出。网格数据的结构参看指令 initmesh 。

(6) 解

方程的解由解矢量来表示。它包括每个独立变量在网格节点上的值，它可能是几个不同时刻的值或者对应不同本征值的值.解矢量是指令 assempde, pdenolin, adapmesh, parabolic, hyperbolic, pdeeig 。根据网格、边界条件和方程系数解出。也可以由图形用户界面输出。由于解矢量的意义依赖于相应的数据网格，所以它们总是要一起使用。有关解矢量的细节参看指令 assempde 。

(7) 数据的最后处理和表现

得到网格及其相应的解以后，有多种工具用以处理数据和将其可视化。指令 pdeintrp 和 pdeprtni 可将定义在三角形节点上的函数和定义在三角形中点上的函数进行转换， tri2grid 将三角形网格上的函数转换为矩形网格上的函数， pdegrad 和 pdecgrad 计算解的梯度， pdecont ， pdesurf 是 pdeplot 的一种简便用法.

上述各个步骤都可以从图形用户界面输出，应该尽量使用图形用户界面的功能来完成这些工作.因为它提供的数据结构有正确的语法，并且可以根据需要来修改它们.

下面列出偏微分方程工具箱的指令及其简要说明，限于篇幅不详细介绍每个指令的具体用法，读者可以用 help 指令查看它们的用法，或者查看软件所带的帮助文件.

数值求解偏微分方程指令

adaptmesh	生成自适应网格并求解偏微分方程
assema	组装积分区域的贡献

assemb	组装边界条件的贡献
assempde	组装刚度矩阵和方程右边的函数
hyperbolic	解双曲型偏微分方程
parabolic	解抛物型偏微分方程
pdeeig	解本征值偏微分方程
pdenonlin	解非线性偏微分方程
poisolv	在矩形网格上快速求解泊松方程

用户界面算法

pdecirc	画圆
pdeellip	画椭圆
pdemdlcv	转换 PDE Toolbox 1.0 版本的 M 文件
pdepoly	画多边形
pderect	画矩形
pdetool	调用偏微分方程工具箱

区域几何描述算法

csgchk	检查几何描述矩阵的有效性
csgdel	删除最小区域之间的交界线
decsg	将构成的实心几何区域分解成最小的区域
initmesh	建立初始的三角形网格
jigglemesh	微调三角形网格内部的点
pdearcl	在参数表达式与弧长之间插值
poimesh	在矩形区域上建立规则的网格
refinemesh	细分三角形网格
wbound	写边界条件的说明文件
wgeom	写几何区域的说明文件

画图指令

pdecont	绘制等值线的快捷命令
pdegplot	画出偏微分方程的几何区域
pdemesh	画偏微分方程的三角形网格
pdeplot	一般的偏微分方程的作图指令
pdesurf	画表面图的快捷命令

通用算法

dst	离散正弦变换
idst	逆离散正弦变换
pdeadgsc	用相对误差判据选择的三角形
pdeadworst	相对于最差值选择的三角形
pdecgrad	计算偏微分方程的解的通量
pdeent	与给定的三角形的相邻的三角形的索引
pdegrad	偏微分方程解的梯度
pdeintrp	从节点到三角形中点的插值
pdejmps	自适应方法的误差估计
pdeprtni	从三角形中点到节点的插值
pdesde	在子区域集合中边界线的索引
pdesdp	在子区域集合中点的索引
pdesdt	在子区域集合中三角形的索引
pdesmech	计算结构力学张量函数
pdetrg	三角形几何数据
pdetriq	三角形质量的测量
poiasma	泊松方程快速解的边界点矩阵的贡献
poicalc	在矩形网格上的泊松方程快速解
poiindex	矩形网格上正规次序中点的索引
sptarn	解一般的稀疏矩阵的本征值问题
tri2grid	从偏微分方程的三角形网格到矩形网格的插值计算

用户定义算法

pdebound	建立边界 M 文件
pdegeom	建立几何区域的描述 M 文件

演示程序

pdedemo1	单位圆上泊松方程的精确解
pdedemo2	解亥姆霍兹方程并研究反射波
pdedemo3	解最小表面问题
pdedemo4	用子区域分解法解偏微分方程
pdedemo5	解热传导方程
pdedemo6	解波动方程
pdedemo7	点源问题的自适应解法
pdedemo8	在矩形网格上解泊松方程

7.8 例 题

7.8.1 散射问题

1. 题目分析

如图 7.54 所示，平面电磁波照射在物体上就会发生散射。下面研究这种散射波在空间的运动。这就是 7.1.2 节的演示程序。这里考虑的是二维的散射现象。

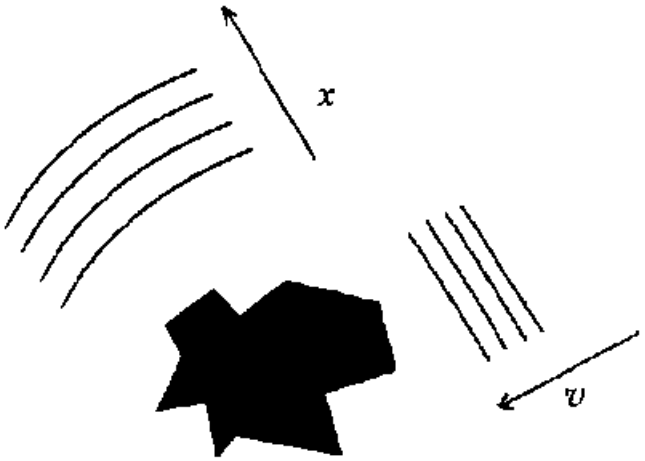


图 7.54 物体表面的散射

假定介质是均匀的，所以光速 c 为常数。又假定从物体散射的电磁波是频率为 ω 的单色波

$$U(x,y,t)=u(x,y)\mathrm{e}^{-\mathrm{i}\omega t}$$

将它代入波方程

$$\frac{\partial^2 U}{\partial t^2}-c^2\Delta^2 U=0$$

得亥姆霍兹方程

$$\Delta u+k^2u=0$$

其中 $k=\omega/c=2\pi/\lambda$,λ 是电磁波波长。

下面来确定边界条件. 设照射的电磁波是平面波 $V(x,y,t)$，传播方向就是波矢 k 的方向，是 $\boldsymbol{a}=(\cos(a),\sin(a))$, 则有

$$V(x,y,t)=\mathrm{e}^{\mathrm{i}(k\boldsymbol{a}\cdot\boldsymbol{r}-\omega t)}=v(x,y)\mathrm{e}^{-\mathrm{i}\omega t}$$

其中 $v(x,y)=\mathrm{e}^{\mathrm{i}k\boldsymbol{a}\cdot\boldsymbol{r}}$.

u 是 v 与反射波 w 之和.

$$u=v+w$$

在反射物体的边界，边界条件是 $u=0$, 也即

$$w=-v(x,y)$$

它表示有反射波从物体反射回来。

反射波从物体出来，在到达计算区域的外边界时，不会再发生反射，即反射波可以全部通过。这种边界条件叫无反射边界条件，我们将使用经典的索莫菲辐射条件来表示。当 $r\to\infty$ 时，则 w 近似满足单向的波方程

$$\frac{\partial w}{\partial t}+c\boldsymbol{n}\cdot\nabla w=0$$

这时波只会沿着 $\boldsymbol{n}$ 的正方向前进， $\boldsymbol{n}$ 是到物体的径向距离。如果取解的形式为 $w\mathrm{e}^{-\mathrm{i}\omega t}$ ，由此得到

$$\boldsymbol{n}\cdot\nabla w=\mathrm{i}kw$$

在边界上，它表示广义的诺伊曼边界条件。为了简单，取计算区域的边界的外法线方向近似作为 $\boldsymbol{n}$ 的方向。

下面是一个具体的例子：平面上有一方形金属片，被平面电磁波 $\mathrm{e}^{-\mathrm{i}kx-\mathrm{i}\omega t}$ 所照射. 按照上面的分析，由此得到的关于散射波 w 的定解问题是

$$\begin{cases}\Delta w+k^2w=0\\ w=-\mathrm{e}^{-\mathrm{i}kx} & \text{(内边界)}\\ w_n=-\mathrm{i}k & \text{(外边界)}\end{cases}$$

其中取 $k=60,\lambda=0.1$.

2. 用图形用户界面求解

打开偏微分方程工具箱的图形用户界面，选择应用格式为 Scalar mode .

利用菜单 Options/Axes Limits, 将 x 轴的范围设为 $[0,1.5]$, y 轴的范围设为 $[0,1]$.

用按钮  画一个圆，用鼠标双击这个圆 C1, 打开对话框，设置圆心在 $(0.8,0.5)$, 半径为 0.45 .

再用按钮  画一个正方形 SQ1, 用鼠标双击这个正方形，打开对话框，设置 Left 为 0.75,Bottom 为 0.45,Width 和 Height 都是 0.1 . 于是得到一个边长为 0.1, 中心在 $(0.8,0.5)$ 的矩形. 选择菜单 Draw/Rotate, 打开对话框，选择 Use center-of-mass, 即以物体中心为旋转中心，输入旋转角度为 45 ° ，然后单击按钮 OK, 关闭对话框，得到一个旋转了 45 ° 的正方形。

在公式设置栏 (Set Formula) 中，输入 C1-SQ1，表示定解问题的求解区域是一个有方孔的圆域。

用按钮 Ω 将圆周上的边界设为 Neumann 边界条件，$q=-i60, g=0$，它对应的波数 $k=60$，波长为 0.1。这是前面推导的外边界的边界条件在这里的应用，边界上函数值是个常数，这里实际上是取 1。

对矩形的边界条件则设为 Dirichlet 边界条件，h =1, r = −exp(−i*60*x)。

用按钮 PDE 将方程类型设为椭圆型，方程的系数为 $c=1, a=-k^2=-3600$, $f=0$。

用按钮 △ 产生初始化的网格。

用按钮 △ 对网格作两次细分以达到所需的解的近似程度。

用按钮 = 求解方程，并将方程的解用颜色图形表示出来。

方程所得的解为复数，对于复数解，只有解实部会画出来同时给出警告提示框如图 7.55 所示。

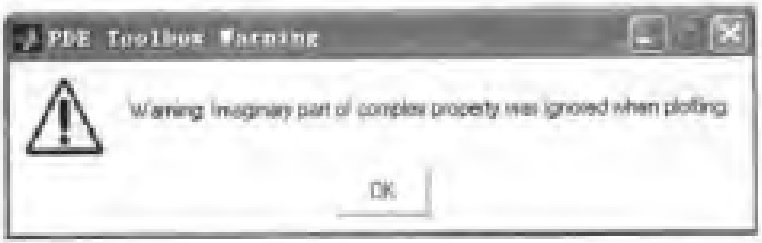


图 7.55 画图时的警告框

只要单击按钮 OK 将它关闭即可。所得的图形如图 7.56 所示。

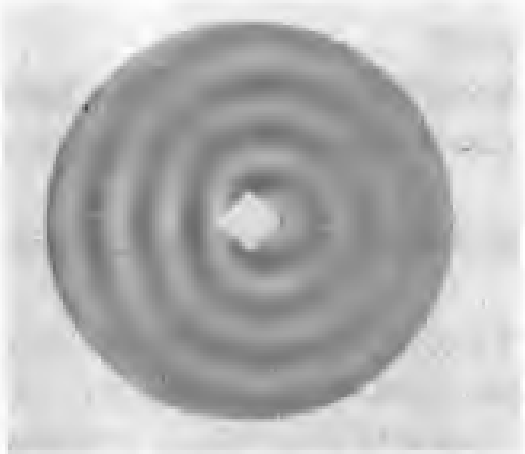


图 7.56 物体表面反射波的形状

选择菜单 File/Save As, 将这些操作存成 M 文件如下:

```
[pde_fig,ax]=pdeinit; pdetool('appl_cb',1);
set(ax,'DataAspectRatio',[1 1 1]);
set(ax,'PlotBoxAspectRatio',[1.5 1 2]); set(ax,'XLim',[0
1.5]); set(ax,'YLim',[0 1]);
set(ax,'XTickMode','auto');
set(ax,'YTickMode','auto');

pdecirc(0.80000000000000004,0.5,0.45000000000000001,'C1');
pdepoly([ 0.87071067811865477, 0.80000000000000004,...
0.72928932188134521, 0.79999999999999993,],...
[ 0.5, 0.42928932188134522, 0.5, 0.57071067811865472,], 'SQ1');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval'),'Striag','C1-SQ1')

pdetool('changemode',0)
pdesetbd(8,'neu',1,'-60i','0')
pdesetbd(7,'neu',1,'-60i','0')
pdesetbd(6,'neu',1,'-60i','0')
pdesetbd(5,'neu',1,'-60i','0')
pdesetbd(4,'dir',1,'1','-exp(-i*60*x)')
pdesetbd(3,'dir',1,'1','-exp(-i*60*x)')
pdesetbd(2,'dir',1,'1','-exp(-i*60*x)')
pdesetbd(1,'dir',1,'1','-exp(-i*60*x)')

setuprop(pde_fig,'Hgrad',1.3);
setuprop(pde_fig,'refinemethod','regular');
pdetool('initmesh') pdetool('refine')
pdetool('refine')

pdeseteq(1,'1.0','(-3600)','0','1.0','0:10','0.0','0.0','[0 100]');
setuprop(pde_fig,'currparam',['1.0';'(-3600)';'0 ';'1.0 '])

setuprop(pde_fig,'solveparam',...
str2mat('0','6528','10','pdeadworst',...
```

```
'0.5','longest','0','1E-4','','fixed','Inf'))

setuprop(pde_fig,'plotflags',[1 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1]);
setuprop(pde_fig,'colstring','');
setuprop(pde_fig,'arrowstring','');
setuprop(pde_fig,'deformstring','');
setuprop(pde_fig,'heightstring','');

pdetool('solve')
```

为了作出反射波的动画，先选择菜单 Mesh/Export Mesh 将网格数据 p,e,t 输出到指令窗口，再选择菜单 Solve/Export Solution 将解 u 输出到指令窗口。然后键入下述命令或将它们编成文件 sswt.m 再运行，就可以看到动画演示的解.

```
h=newplot;
hf=get(h,'Parent');
set(hf,'Renderer','zbuffer')
axis tight
set(gca,'DataAspectRatio',[1 1 1]);
axis off
M=moviein(10,hf);
maxu=max(abs(u));
colormap(cool)
for j=1:10,
   ur=real(exp(-j*2*pi/10*sqrt(-1))*u);
   pdeplot(p,e,t,'xydata',ur,'colorbar','off','mesh','off');
   caxis([-maxu maxu]);
   axis tight,
   set(gca,'DataAspectRatio',[1 1 1]);
   axis off
   M(:,j)=getframe;
end
movie(hf,M,10);
```

7.8.2 放射性杆中的热传导问题

这个热传导问题是一个三维的抛物型方程问题. 在柱坐标下, 它可以简化为二维问题.

考虑一个圆柱形放射性杆, 左边有热流进入, 右边保持恒温, 侧面与环境有热交换. 同时, 由于放射性, 在整个杆上会有均匀的热量产生. 假定杆的初始温度为零度. 这个问题可以用下述方程来描述:

$$\rho C\frac{\partial u}{\partial t}-\nabla\cdot(k\nabla u)=f$$

其中 ρ 为密度, C 是杆的热容量, k 是导热系数, f 是放射性热源的强度. 计算中取 ρ=7800 kg/m^3, C= 500 Ws/kg $\cdot$°C, k=40 W/m $\cdot$°C, f=20000 W/m^3, 此外, 左端的热流强度为 5000 W/m^2, 右端恒温为 100°C, 侧面环境温度为 100°C, 热交换系数为 50 W/m^2 $\cdot$°C .

设杆的长度为 3, 半径为 0.2 . 以圆柱的轴为 z 轴, 以圆柱的中心为原点, 在柱坐标 (r,θ,z) 下, 考虑到柱体的轴对称性, 问题与 θ 无关, 方程简化为

$$r\rho C\frac{\partial u}{\partial t}-\frac{\partial}{\partial r}\left(kr\frac{\partial u}{\partial r}\right)-\frac{\partial}{\partial z}\left(kr\frac{\partial u}{\partial z}\right)=fr$$

按题意, 边界条件表述如下:

在杆的左端, $z=-1.5$,$\boldsymbol{n}\cdot(k\nabla u)=5000$, 由于在偏微分方程工具箱中, 边界条件是表述为 $\boldsymbol{n}\cdot(C\nabla u)+qu=g$, 其中 $C=kr$, 所以边界条件是 $\boldsymbol{n}\cdot(C\nabla u)=5000r$.

在杆的右端, $z=1.5$, $u=100$.

在杆的侧面, $r=0.2$,$\boldsymbol{n}\cdot(k\nabla u)=50(100-u)$, 按照上面的讨论, 它应该写成 $\boldsymbol{n}\cdot(C\nabla u)+50r\cdot u=50r\cdot 100$.

在杆的轴心, $r=0$, 在最初的问题中它不是边界, 但是在简化的二维问题中, 它已经成了一个边界, 我们为它指定一个人为的边界条件, $\boldsymbol{n}\cdot(C\nabla u)=0$.

初始温度为 $u(t=0)=0$.

下面在偏微分方程工具箱的用户界面中解这个问题, 取 $0\leqslant y\leqslant 0.2$, $-1.5\geqslant x\geqslant 1.5$, 画一个矩形区域如图, 矩形的顶点为 $(-1.5,0),(1.5,0),(1.5,0.2),(-1.5,0.2)$. 它是圆柱剖面的一半区域.

按前面的讨论设置边界条件, 在左边界有诺伊曼边界条件, 取 $g=5000*y$, $q=0$, 右边界是狄里克利边界条件, 取 $h=1,r=100$, 上下边界都是诺伊曼边界条件, 分别取 $g=50*y*100,q=50*y$ 和 $g=0,q=0$.

方程的类型为抛物型方程, 取 $C=40*y,a=0,f=20000*y,d=7800*500*y$.

在解方程的参数设置中, 取 Time= 0 : 100 : 20000,u(t0)=0, 其余用默认设置.

在作网格划分时, 除了网格初始的划分之外, 最好将网格细分一次.

为了将结果可视化，可选参数为 Color,Contour，显示的图形是 $t = 20000$ 时的温度分布的彩色的等值线图如图 7.57 所示。

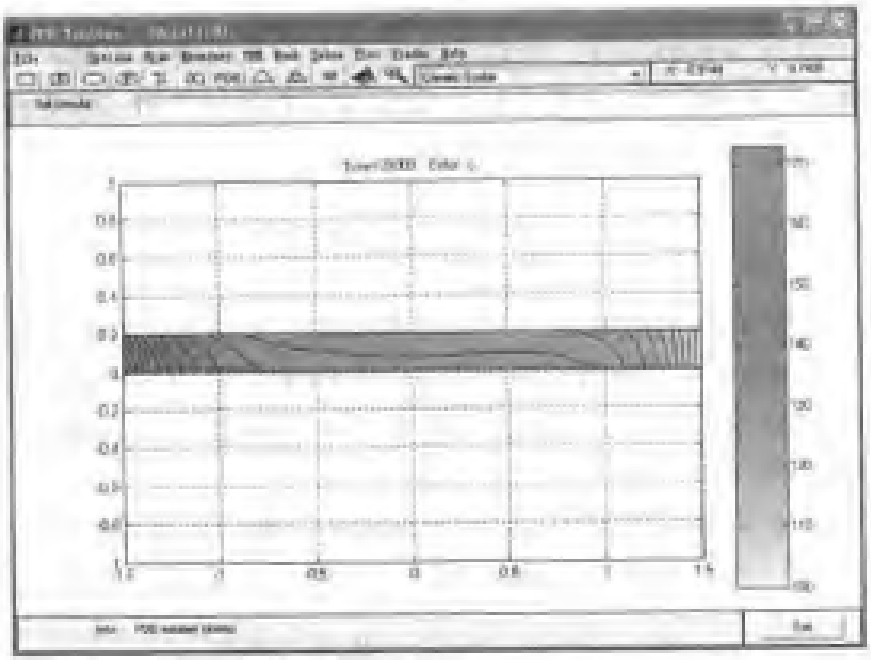


图 7.57 放射性杆的温度分布

如果在 Plot Selection 对话框中选上 Animation，则可以看到用等值线表现的热量流动图。

如果将方程设置为椭圆型方程，则所得结果是 $t \to \infty$ 抛物型方程的解。

7.8.3 薄膜振动问题

本题研究左右两侧固定，上下两侧自由的薄膜的振动，相应的定解问题是

$$\begin{cases} \dfrac{\partial^2 u(x,y,t)}{\partial t^2} - \Delta u(x,y,t) = 0 \qquad (-1 \leqslant x \leqslant 1, -1 \leqslant y \leqslant 1) \\ u(x,y,0) = \arctan\left(\cos\dfrac{\pi}{2}x\right) \\ u_t(x,y,0) = 3\sin(\pi x)\mathrm{e}^{\sin(\frac{\pi}{2}y)} \\ u(-1,y,t) = 0, \quad u(1,y,t) = 0 \\ u_t(x,-1,t) = 0, \quad u_t(x,1,t) = 0 \end{cases}$$

在使用图形用户界面求解时，选择 Generic Scalar 应用模式，画一个顶点为 (-1,-1), (-1,1), (1,-1), (1,-1) 的正方形，按照题意定义边界条件，画好三角形网格。选择双曲型方程的系数为 $c = 1, a = 0, f = 0, d = 1$，求解之前要在选择菜单

Solve/Paramaters，在对话框中输入初始条件，然后解出方程。

为了更好地显示薄膜的运动，可以在选择菜单 Plot/Parameters，然后在对话框中选择 Height(3-D plot),Animation,Show mesh,Plot solution automatically 等项，最后作图，所得的就是动画。

全部用指令编写的解这个问题的演示程序是 pdedemo6，程序如下：

```
  %%定义问题
g='squareg';   %%调用画矩形的文件
b='squareb3';   %%调用设置边值条件的文件
c=1;  a=0;   f=0;   d=1;   %%设置方程系数

[p,e,t]=initmesh('squareg');   %%初始化网格

  %%设置初始条件
x=p(1,:)';   y=p(2,:)';  %%从网格数据中提取 x,y 的值

u0=atan(cos(pi/2*x));   %%初始位移
ut0=3*sin(pi*x).*exp(sin(pi/2*y));   %%初始速度

  %%设定时间范围为 0～5，分成 31 步计算
n=31;
tlist=linspace(0,5,n);

uu=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d);  %%解双曲型方程

  %%为了提高画图速度，采用矩形网格
delta=-1:0.1:1;
[uxy,tn,a2,a3]=tri2grid(p,t,uu(:,1),delta,delta);
gp=[tn;a2;a3];

  %%动画演示
newplot;
```

```
M=moviein(n);
umax=max(max(uu));
umin=min(min(uu));
for i=1:n,
  if rem(i,10)==0,
    fprintf('
  end,
  pdeplot(p,e,t,'xydata',uu(:,i),'zdata',uu(:,i),'zstyle','continuous',...
        'mesh','off','xygrid','on','gridparam',gp,'colorbar','off');
  axis([-1 1 -1 1 umin umax]);
  caxis([umin umax]);
  M(:,i)=getframe;
  if i==n,
    fprintf('done \ n');
  end,
end

nfps=5;   %%回放次数
movie(M,10,nfps);   %%电影回放
```

7.8.4 本征值问题

本题求 L 形薄膜的本征值与本征函数，这是所有 MATLAB 用户都感兴趣的一个问题，因为 MathWorks 公司的商标就是画的它的第一个本征函数．你不妨对比这里解得的结果与 MATLAB 中的演示商标图形的指令 membrane 画的图形．

下面计算的是边界固定的 L 形薄膜的所有小于 100 的本征值及其对应的本征函数．方程的形式为

$$-\Delta u = \lambda u$$

再用图形用户界面求解方程．应用模式选择 Generic Scalar ．

在窗口中画一个顶点在 $(0,0),(-1,0),(-1,-1),(1,-1),(1,1),(0,1)$ 的多边形，输入边界条件 $u=0$ ．由于这些都是图形用户界面的默认设置，实际上可以直接进入下面的步骤，即建立初始化的网格，并对网格作两次细分．

单击PDE按钮, 方程类型选择 Eigenmodes, 由于默认的系数 $c=1, a=0, d=1$ 也是所要的系数, 所以不必改动.

在菜单 Solve 下选择 Parameters, 打开对话框, 选择本征值的搜索范围为 0 ~ 100, 再解方程, 则会显示第一个本征函数的图像, 并在窗口的下方提示, 一共找到 19 个本征值.

在菜单 Plot 下选择 Parameters, 打开对话框, 改变 Eigenvalue 所对应的值, 就可以画出其他本征函数的图像.

还可以用指令解这个问题, 程序 lbm.m 如下. 其中 lshapeg 是 MATLAB 中已经有的画 L 形图形及设定其边界条件的文件. 所以程序直接进入网格初始化和两次细分网格, 然后调用指令 pdeeig 解本征值方程, 解出的结果保存在文件 membrane, 最后分区画图.

```
%program lbm.m
[p,e,t]=initmesh('lshapeg');
[p,e,t]=refinemesh('lshapeg',p,e,t);
[p,e,t]=refinemesh('lshapeg',p,e,t);
[v,l]=pdeeig('lshapeb',p,e,t,1,0,1,[0 100]);
subplot(2,2,1)
pdesurf(p,t,v(:,1))
subplot(2,2,2)
membrane(1,20,9,9)
subplot(2,2,3)
pdesurf(p,t,v(:,12))
subplot(2,2,4)
membrane(12,20,9,9)
```

程序所画出的图形为如图 7.58 所示, 图形中画出的第 1 个和第 12 个本征函数. 程序使用了 MATLAB 本身的文件来建立 L 型薄膜的 PDE 模型的几何数据. 指令 pdesurf 画的是解, 指令 membrane 可以直接画出 L 型薄膜前 12 个本征函数.

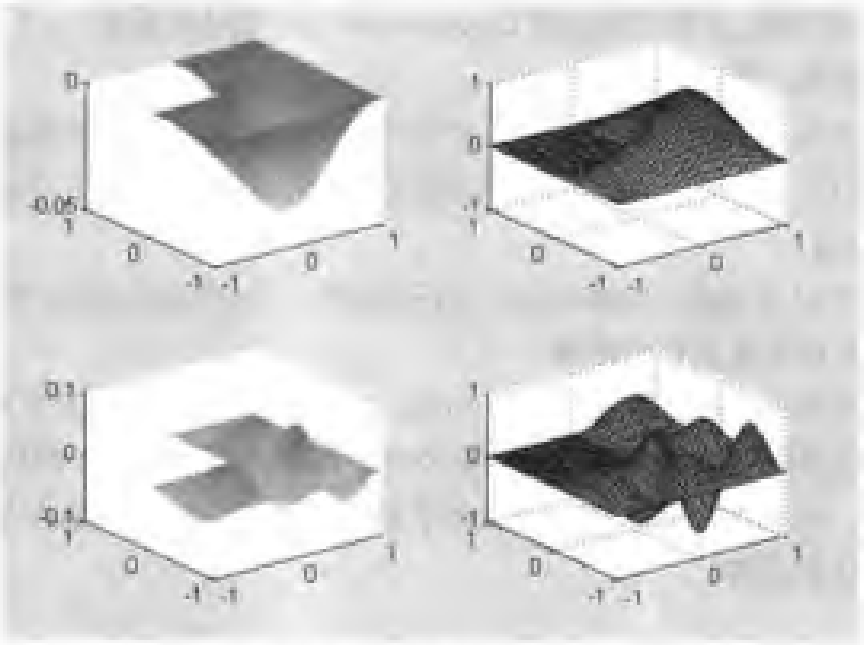


图 7.58 L 形膜的本征函数

7.8.5 导电介质中的直流电流

本题是图形用户界面应用模式的一个例子. 两个圆的金属导体薄片放置在一个平面介质上, 平面介质是盐水浸湿的纸. 那么描述电势的方程为

$$-\nabla \cdot (\sigma \nabla V) = 0$$

设边界条件为, 对于左边的圆导体, 电势为 $V = 1$, 对于右边的圆导体, 电势为 $V = -1$, 在外边界上, $\frac{\partial V}{\partial n} = 0$, 介质的电导率 σ 为 1 .

用图形用户界面解方程的过程如下, 选择应用模式为 Conductive Media DC , 画一个顶点在 $(-1.2, -0.6)$, $(1.2, -0.6)$, $(1.2, 0.6)$, $(-1.2, 0.6)$ 矩形 R1 来代表介质所在的区域, 画两个半径为 0.3 的圆 C1 和 C2 代表导体, 圆心分别是 $(-0.6, 0)$, $(0.6, 0)$, 在 Set formula 用公式 R1−(C1+C2) 来表示计算的区域.

在矩形边界上设置诺伊曼边界条件, g=0,q=0. 在两个圆的边界上设置狄里克利边界条件, 分别是 h=1,r=1 和 h=1,r=−1 . 在设置方程类型时, 首先选用应用模式为 Conductive Media DC, 这时的方程形式为

$$-\nabla \cdot (\sigma \nabla V) = Q$$

取 q=0,$\sigma = 1$.

建立初始化网格, 并作两次细分, 最后用菜单 Mesh 下的 Jiggle Mesh 对网格作优化, 最后解方程.

如果要画出等电势线，在对话框 Plot Selection 中，在 Plot type 下选择 Contour，在 Property 下对应位置选择 electric potential．如果要将电流密度 $\boldsymbol{J}$ 可视化，则在 Property 下都选择 current density，然后在 Plot type 下选择 Contour 画电流密度 $\boldsymbol{J}$ 的绝对值，选择 Arrow 画电流密度 $\boldsymbol{J}$ 的矢量场．图 7.59 是其中的一个例子．

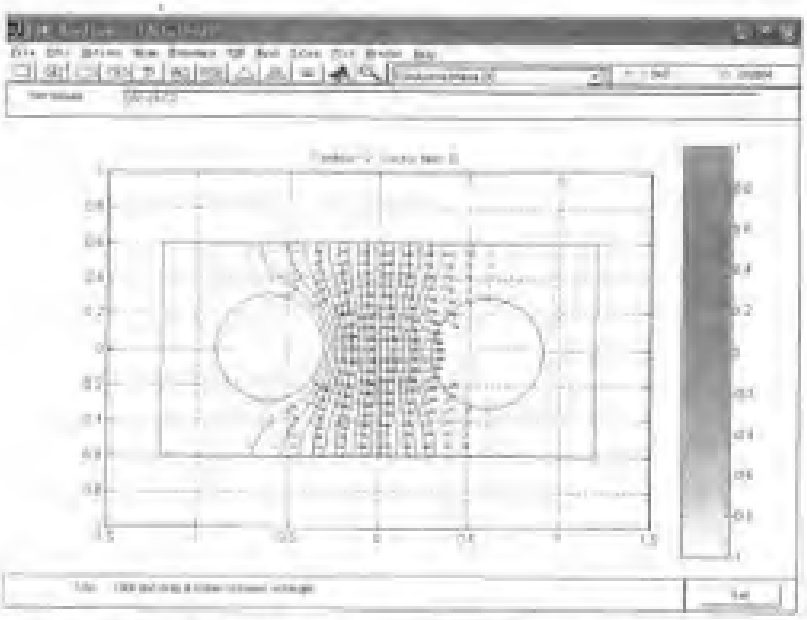


图 7.59　两个金属导体之间的等电势线和电流密度矢量场

7.8.6　热传导问题

本题也是图形用户界面应用模式的一个例子．应用模式 Heat Transfer 可解的方程为

$$\rho C\frac{\partial T}{\partial t}-\nabla\cdot(k\nabla T)=Q+h\cdot(T_{ext}-T)$$

它描述了平面上或轴对称情形下的热传导现象，其中 T 为温度，其余的参数是：ρ 为密度，C 为比热，k 为热传导系数，Q 为热源，h 为对流传热系数，T_{ext} 为外界环境温度．

$h\cdot(T_{ext}-T)$ 描述的是周围环境产生的横向热传导，它常用于描述薄板的热传导现象．

在稳恒情况下，方程成为椭圆型

$$-\nabla\cdot(k\nabla T)=Q+h\cdot(T_{ext}-T)$$

边界条件可以是狄利克里型、诺伊曼型和广义诺伊曼型．

可视化的量有温度场、温度场的梯度、热流量 $k\nabla T$，作图选择有等温线和热流矢量。

下面的例题是解不同材料参数之间的热传导。在正方形区域内包含一个菱形（将一个小的正方形转 45° ）区域。正方形区域的热传导系数为 10，密度为 2，菱型区域包含一个均匀的热源 Q=4，它的热传导系数为 2，密度为 1。两个区域的热容量都是 0.1。

用图形用户界面解方程的过程如下，选择应用模式为 Heat Transfer，将 x 轴和 y 轴的范围都设为 $[-0.5, 3.5]$，并在菜单 Option 下选择 Axis Equal，画一个顶点在（0, 0），（3, 0），（3, 3），（0, 3）的矩形 R1，再画一个顶点在 (1.5, 0.5), (2.5, 1.5), (1.5, 2.5), (0.5, 1.5) 的菱形 C1，用公式 R1+C1 来表示计算的区域。

在矩形边界上温度为零，这正好是默认的边界条件。在矩形区域指定方程类型为抛物型，参数为 $\rho = 2, C = 0.1, k = 10, Q = 0, h = 0, T_{ext} = 0$，在菱形区域指定方程类型为抛物型，参数为 $\rho = 1, C = 0.1, k = 2, Q = 4, h = 0, T_{ext} = 0$。

所解的偏微分方程是含时的方程，需要指定初值和解方程的时间范围，用菜单 Solve 下的 Parameters 打开 Solve Parameters 对话框，输入时间为 logspace (−2, −1, 10)，这是对数标度的时间，它给出的时间范围为 $0.01 = 10^{-2} \sim 0.1 = 10^{-1}$ 之间的 10 个对数值，因为温度在 0.1 单位时间就会趋于稳恒，输入初始温度为零即 u(t0)=0。

解方程以后默认的输出为最后时刻的温度分布，更好的显示方式是动态地显示温度变化，为此，可以在 Plot Selection 对话框中选择 Animation, Heigh(3-D plot), show mesh，另外选择 Plot in x-y grid 是用矩形网格代替三角网格，可以加快模拟处理的速度，如果要画等温线，可选择 contour，画热流矢量可选 arrow，图 7.60 和图 7.61 是其中的实例。

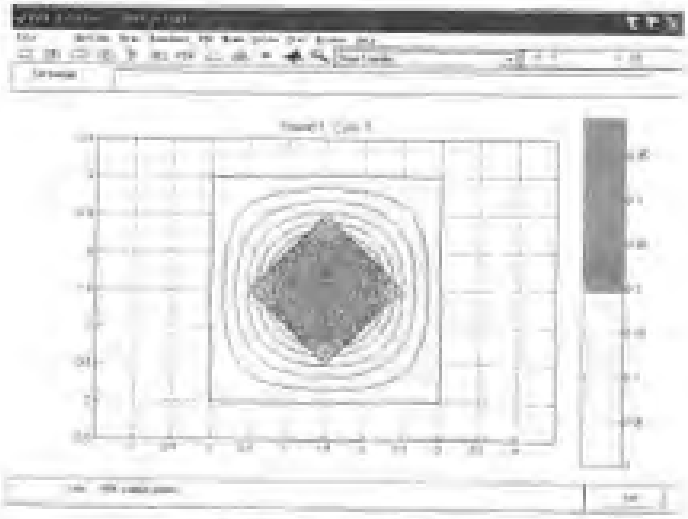


图 7.60 温度场的等温线

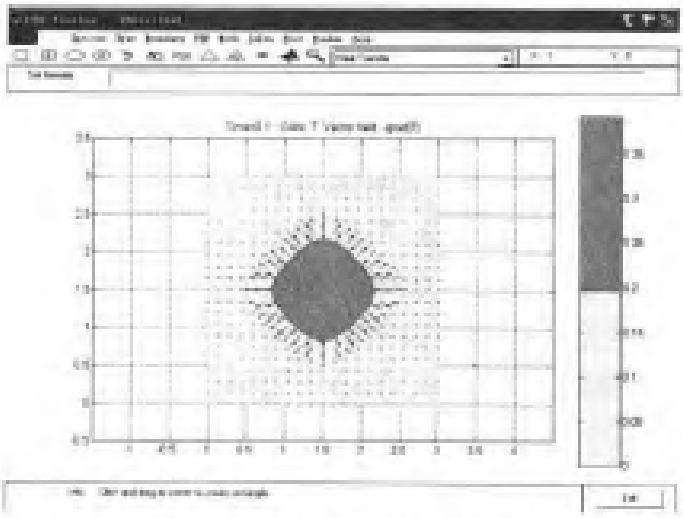


图 7.61　温度场的热流矢量场

第 8 章　解微分方程的其他方法

在 MATLAB 中，专门用于解偏微分方程的工具除了偏微分方程工具箱之外，还有几个专用的指令，如 pdepe 解一维椭圆型或抛物型方程组的初值边值问题的，bvp4c 解一维边值问题或本征值问题。另外也可以通过 MATLAB 语言编程，用差分法或有限元法来解偏微分方程。本章对这些内容作些介绍，以使读者对 MATLAB 有更多的了解。在用差分法或有限元法时， MATLAB 编程的方法与其他语言如 FORTRAN， C 语言相似，但在 MATLAB 中编程会更简单一些。在指令窗口运行指令 odeexamples，可以看到这几个指令的演示程序。

8.1　指令 bvp4c 解本征值问题

在 MATLAB 中，一维的边值问题或本征值问题可用指令 bvp4c 来求解，二维的本征值问题可用偏微分方程工具箱来求解。后者已经在第 1 章介绍过，这里再介绍一下 bvp4c 的用法。使用 bvp4c 会用到三个函数，即用 odefun 定义要求解的方程，用 bcfun 设置边界条件，用 solinit 给定一个初始的猜测解。

8.1.1　用指令 bvp4c 解一维本征值问题

1. 指令 bvp4c 的用法

常微分方程的两点边值问题可能没解，可能有多个解，也可能有无限个解。在使用指令求解时，必须给出一个猜测解。猜测解的好坏会严重影响求解过程。

使用 bvp4c 的语句格式有三种：

sol = bvp4c(odefun,bcfun,solinit)
sol = bvp4c(odefun,bcfun,solinit,options)
sol = bvp4c(odefun,bcfun,solinit,options,p1,p2...)

语句中各个量的含义为

odefun	由被求解的微分方程表示的函数 $f(x,y)$。
bcfun	表示边界条件的函数 $bc(y(a),y(b))$。
solinit	用以设置初始的猜测解的结构数组。
options	可选的积分参数。可以用 bvpset 来建立的结构数组。
p1,p2...	p1,p2... 可选参数。是已知的参数，由算法指令将它们传递给 odefun,bcfun, 以及所有由用户在 options 中指定的函数。

下面会对 odefun,bcfun,solinit 作更详细的说明。

2. bvp4c 三种用法的比较

三种语句中的输入量不同，下面作一说明.

(1) **sol = bvp4c(odefun,bcfun,solinit)**

该语句在区间 [a,b] 求解形式为

$$y' = f(x, y)$$

常微分方程组，要求解满足两端点边界条件

$$bc(y(a), y(b)) = 0$$

bvp4c 也能解带有未知参数的形式为

$$y' = f(x, y, p)$$

$$bc(y(a), y(b), p) = 0$$

的方程，这里 p 代表参数. 边界条件必须能够确定 p, 并且首先要对未知参数给定一个初始的猜测值，而 bvp4c 所解得的结果在 sol.parameters 中.

bvp4c 所得的解及其一阶导数在区间 [a,b] 上是连续的，用指令 deval 及 bvp4c 可以求得区间 [a,b] 中任何一点 xint 的值 sxint = deval(sol,xint) .

由 bvp4c 返回的解 sol 是一个结构数组，其中包含以下的域:

sol.x	bvp4c 使用的变量的数据点.
sol.y	与数据点 sol.x 对应的近似 y 值.
sol.yp	与数据点 sol.x 对应的近似 y' 值.
sol.parameters	由 bvp4c 返回的未知参数的值.
sol.solver	bvp4c 中使用的算法.

结构数组 sol 的命名可以是任意的， bvp4c 会自动建立它的域名 x, y, yp, parameters 和 solver .

(2) **sol = bvp4c(odefun,bcfun,solinit,options)**

用法相同，对积分属性的要求不是默认的，而是由 options 指定的，这是由指令 bvpset 所建立的结构数组. 详情参看 bvpset 的帮助文件. 附带说明一下，指令 bvpset 是进入或修改 bvp4c 解方程的各种选项 (option structure), 指令 bvpget 是获得用 bvpset 所建立的各种选项的属性.

(3) **sol = bvp4c(odefun,bcfun,solinit,options,p1,p2...)**

将已知的参量 p1, p2, ..., 传递给 odefun, bcfun 和所有用户在 options 指定的函数. 采用 options = [ ] 表示不改变原来的设置.

3. 微分方程函数文件 odefun

要求解的微分方程用函数文件 odefun 来设定，要求写成一阶的微分方程组

$$y' = f(x, y)$$

的形式，它的写法与解常微分方程时用的函数文件相同，有以下几种：

dydx = odefun(x,y)
dydx = odefun(x,y,p1,p2,...)
dydx = odefun(x,y,parameters)
dydx = odefun(x,y,parameters,p1,p2,...)

x 是一个标量， y 一个列矢量， parameters 是代表未知量的矢量， p1,p2,... 是未知的参数. 输出量 dydx 是列矢量.

4. 边界条件函数文件 bcfun

边界条件是函数或者其导数在边界上的值，有两种类型：

$$bc(y(a), y(b)) = 0$$

$$bc(y(a), y(b), p) = 0$$

它是用函数文件 bcfun 来设定的. 它的形式为

res = bcfun(ya,yb)
res = bcfun(ya,yb,p1,p2,...)
res = bcfun(ya,yb,parameters)
res = bcfun(ya,yb,parameters,p1,p2,...)

ya 和 yb 是分别对应 $y(a)$ 和 $y(b)$ 的列矢量. parameters 是未知参量的列矢量. 而 p1,p2,... 是已知参量. 输出 res 是列矢量.

5. 指令 bvpinit 的用法

指令 bvpinit 建立微分方程的初始猜测解 solinit . 这是一个结构数组，结构数组可以任意命名，但是它的域名必须是 x,y 和 parameters .

指令 bvpinit 的语句格式有以下几种：

solinit = bvpinit(x,v)
solinit = bvpinit(x,v,parameters)
solinit = bvpinit(x,v,parameters,p1,p2...)

solinit = bvpinit(sol,[anew bnew])
solinit = bvpinit(sol,[anew bnew],parameters)

solinit 是具有如下域名的结构数组，它本身可以任意命名，但它包含的域名必须是 x, y, and parameters 。几个域名的含义是

x 变量初始的有序的数据点．边界条件由 a = solinit. x(1) and b = solinit. x(end) 确定。

y 一个猜测的初始解， solinit.y(:,i) 是这个猜测的初始解在节点 solinit.x(i) 的值。

parameters 可选项，它是矢量，给出未知参量一个初始的猜测值．

下面对这几种语句格式加以解释。

(1) solinit = bvpinit(x,v)

在一个通用的环境中生成 bvp4c 要用的初始猜测值 $v.x$ 是初始指定的数据矢量．如果你希望在区域 [a, b] 上解边值问题，则有 $x(1)$ = a and $x(end)$ = b. bvp4c 将在求解中使用这个数据，一般来说，猜测值取 x = linspace(a,b,10) 就够了．然而在复杂的情形下，在方程的解变化剧烈的地方必须要多放置一些数据点．各个 x 必须是排序的面且各不相同，所以如果 a<b, 则有 $x(1) < x(2) < \cdots < x(end)$, 而对 a>b , 则大小的顺序也相反．

v 是一个猜测解．它可以是矢量或函数：

如果是矢量，则对于解的每个分量，bvpinit 将复制矢量对应的分量作为所有网格点上常数猜测值．也即 $v(i)$ 是解的第 i 个分量 $y(i,:)$ 在 x 网格上所有点的猜测值．

如果是函数则对于给定的数据点网格，函数将返回一个矢量，它的元素是解的对应分量的一个猜测值．例如，如果对于 [a,b] 之中的任何 x,yfun(x) 返回解 $y(x)$ 的一个猜测值，则可以使用语句 solinit = bvpinit(x,@yfun) .

(2) solinit = bvpinit(x,v,parameters)

这个格式使用了未知参数．当使用参数矢量时是对所有未知参数提供一个猜测值．

(3) solinit = bvpinit(x,v,parameters,p1,p2...)

传递附加的参量给猜测的函数 v, 这时未知的参数已经给定．这种格式只有在 v 函数时才能使用．

(4) solinit = bvpinit(sol,[anew bnew])

从区域 [a, b] 上的解 sol 生成的在新区域 [anew bnew] 的一个猜测解 solinit .

新的区域必须大于原有的区域. 可以是 anew<=a < b <= bnew 或者 anew >= a > b >= bnew . 原来的解是外推到新的区域的. 如果在 sol 中含有参数, 则它们会复制到 solinit 中.

(5) **solinit = bvpinit(sol,[anew bnew],parameters)**

用法与 (3) 相同, 但是使用猜测的参数以替代未知的参数.

8.1.2 有两个解的边值问题

边值问题可能有多重解, 初始猜测的一个目的就是指定我们想要的解. 下面的二阶常微分方程

$$y'' + |y| = 0$$

有两个解析解满足边值条件

$$y(0) = 0, \quad y(4) + 2 = 0$$

我们用 bvp4c 来求这两个解, 这个问题说明不同的初始猜测值会导致不同的解.

引入 $y_1 = y, y_2 = y'$, 将方程改写为

$$\begin{cases} y_1' = y_2 \\ y_2 = -|y_1| \end{cases}$$

这就是 bvp4c 编程所需要的如下形式 $y' = f(x, y)$, $bc(y(a), y(b)) = 0$. 它们分别对应下面程序中的子函数文件 twoode 和 twobc .

在程序中, 语句

solinit = bvpinit(linspace(0,4,5),[1 0]);

在 [0,4] 之间 5 个等间距的点组成了一个初始的变量数据网格, 而猜测值则是

$$y_1(x) = 1, \quad y_2(x) = 0$$

解方程的语句是

sol = bvp4c(@twoode,@twobc,solinit);

以下的语句是求出在 100 个等间距上的点的数值解并作图

x = linspace(0,4);

y = deval(sol,x);

plot(x,y(1,:));

使用另一个猜测, 将得到另外一个解:

solinit = bvpinit(linspace(0,4,5),[−1 0]);

完整的程序如下:

```
function twobvp
% One solution is obtained using an initial guess of y1(x)=1,
% y2(x)=0
solinit = bvpinit(linspace(0,4,5),[1 0]);
sol = bvp4c(@twoode,@twobc,solinit);
x = linspace(0,4);
y1 = deval(sol,x);
figure;
plot(x,y1(1,:));
xlabel('x');
ylabel('y');

% The other solution is obtained using an initial guess of y1(x)=-1,
% y2(x)=0
solinit = bvpinit(linspace(0,4,5),[-1 0]);
sol = bvp4c(@twoode,@twobc,solinit);
y2 = deval(sol,x);

% Plot both solutions
figure;
plot(x,y1(1,:),x,y2(1,:));
xlabel('x');
ylabel('solution y');
title('A BVP with two solutions');

function res = twobc(ya,yb)
res = [    ya(1)
        yb(1) + 2 ];

function dydx = twoode(x,y)
dydx = [    y(2)
         -abs(y(1)) ];
```

程序画出的图形如图 8.1 所示.

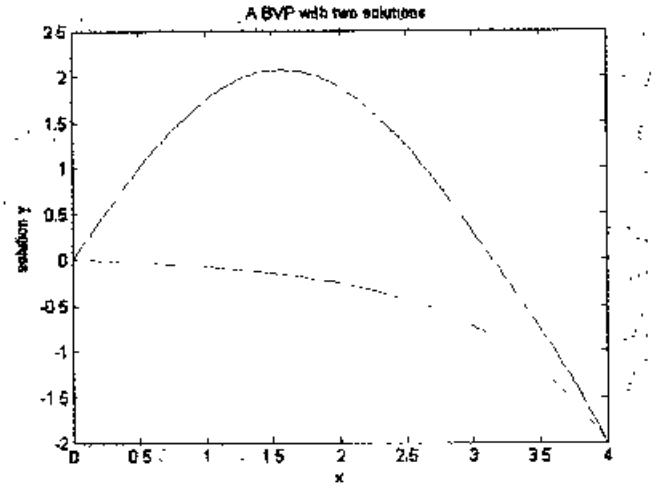


图 8.1 有两个不同解的边值问题

8.1.3 Mathieu 方程的本征值

我们要计算的是 Mathieu 方程

$$\begin{cases} y'' + [(\lambda - 2q\cos 2x)]y = 0 \\ y'(0) = 0, \quad y'(\pi) = 0 \end{cases}$$

的第 4 个本征值，其中 $q = 5$ 。

要用专门的编程来计算斯特姆 - 刘维本征值问题。而 BVP4C 只能求出最接近猜测值的本征值。可以通过使猜测的函数具有与第 4 个本征函数一样的定性的行为来使做到这一点。

所求得的本征函数会差一个常数因子，所以增加一个规范化的条件 $y(0) = 1$ 来得到一个特定的函数。用 bvp4c 所得到的网格作图并不能产生一条光滑的曲线。解 $S(x)$ 是连续的，其导数也是连续的。为了得到光滑的曲线，可以用 deval 来计算所增加的点上的数据。

在程序中，子函数 mat4ode 是转换成一阶常微分方程组的微分方程，子函数 mat4bc 是边界条件。由于出现了未知参量，这些函数都必须接受三个输入参量，即使是某些量可能没有用到。猜测的初始结构数组 solinit 由子函数 bvpinit 来构造。任何一个初始的猜测解都必须由函数 mat4init 所决定的形式来提供。我们选择 $y = \cos 4x$ 是因为它和它的导数满足边界条件并且有正确的定性行为 (正确的符号改变次数)，bvpinit 用以在 10 个等间隔的点上产生一个初始的猜测值。在调用 bvpinit 时，第三个量 ($\lambda = 15$) 给出了本征值的初始猜测值。

bvp4c 返回的解是结构数组 sol，本征值在域名 sol.parameters 下。这里，域 sol.parameters 返回的值 $\lambda = 17.097$，而图形则是与此本征值相联的本征函数的图形。程序如下，画出的图形见图 8.2 。

```
function mat4bvp
lambda = 15;
solinit = bvpinit(linspace(0,pi,10),@mat4init,lambda);

sol = bvp4c(@mat4ode,@mat4bc,solinit);

fprintf('The fourth eigenvalue is approximately...
     %7.3f.\n',sol.parameters)

xint = linspace(0,pi);
Sxint = deval(sol,xint);
figure;
plot(xint,Sxint(1,:));
axis([0 pi -1 1.1]);
title('Eigenfunction of Mathieu''s equation.');
xlabel('x');
ylabel('solution y');

function dydx = mat4ode(x,y,lambda)
q = 5;
dydx = [              y(2)
         -(lambda - 2*q*cos(2*x))*y(1) ];

function res = mat4bc(ya,yb,lambda)
res = [  ya(2)
         yb(2)
        ya(1)-1 ];

function yinit = mat4init(x)
yinit = [   cos(4*x)
          -4*sin(4*x) ];
```

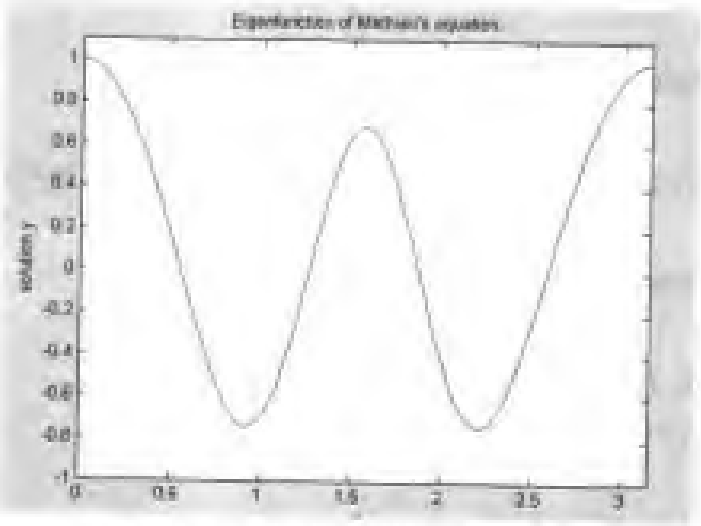


图 8.2 Mathieu 方程的本征函数图形

8.1.4 艾登 (emden) 方程

要解的方程是艾登 (emden) 方程，形式如下。

$$y'' + \frac{2}{x}y' + y^5 = 0 \quad (0 \leqslant x \leqslant 1)$$

这个方程叫 emden 方程，来自于一个气体球的模型，这是一个有奇异项的方程，由于对称性而简化成上述常微分方程。 $x = 0$ 是系数 $2/x$ 的奇点，但是，对称性意味 $y'(0) = 0$，因此，根据这个边界条件，当 $x \to 0$ 时， $(2/x)y'(x)$ 是有定义的。而另一端的边界条件是 $y(1) = \sqrt{3/4}$，所以这个边值问题有解析解。

指令 bvp4c 能解形式为

$$y' = \frac{S}{x}y + f(x, y)$$

的具有奇异性的问题。矩阵 S 必须是常数，并且 $x = 0$ 的边界条件必须与必要条件 $Sy(0) = 0$ 自洽。 S 是作为 odeset 中的属性 'SingularTerm' 的值传递给 bvp4c 的。在其他方面，这个问题的解法与没有 $(S/x)y$ 的方程一样。

求解时，定义 $y(1) = y, y(2) = y'$，将方程写成一阶方程组，而 $S = [0\,0;\,0\,-2]$，即

$$\begin{pmatrix} y'(1) \\ y'(2) \end{pmatrix} = \frac{1}{x}\begin{pmatrix} 0 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} y(1) \\ y(2) \end{pmatrix} + \begin{pmatrix} y(2) \\ y^5(1) \end{pmatrix}$$

由于 $x=0$ 的边界条件为 $y(2)=0$，所以有

$$Sy(0)=\begin{bmatrix}0\\-2y(2)\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

在程序中，guess 是一个满足边界条件的猜测常数，它表示猜测的解是

$$y(1)=\sqrt{3}/2,\quad y(2)=0$$

最后在对计算结果画图时，与这个问题的解析解 truy 作了对比.

```
function emdenbvp

S = [0  0
     0 -2];
options = bvpset('SingularTerm',S);

guess = [sqrt(3)/2; 0];
solinit = bvpinit(linspace(0,1,5),guess);

sol = bvp4c(@emdenode,@emdenbc,solinit,options);

x = linspace(0,1);
truy = 1 ./ sqrt(1 + (x.^2)/3);

figure;
plot(x,truy,sol.x,sol.y(1,:),'ro');
title('Emden problem -- BVP with singular term.')
legend('Analytical','Computed');
xlabel('x');
ylabel('solution y');

% ------------------------------------------------------

function dydx = emdenode(x,y)
dydx = [  y(2)
         -y(1)^5 ];
```

```
% ------------------------------------------------------------

function res = emdenbc(ya,yb)
res = [ ya(2)
        yb(1) - sqrt(3)/2 ];
```

此解的图如图 8.3 所示.

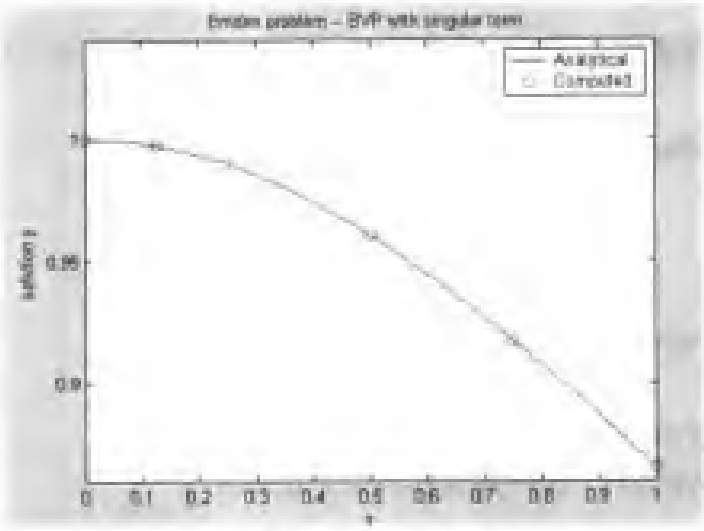


图 8.3 艾登 (emden) 方程图形

8.1.5 Falkner-Skan 边值问题

Falkner-Skan 边值问题是解在薄板上的粘性的不可压缩的层流. 方程是

$$\begin{cases} f''' + ff'' + \beta(1 - (f')^2) = 0 \\ f(0) = 0, \quad f'(0) = 0, \\ f'(\infty) = 1 \end{cases}$$

其中 $\beta = 0.5$, 这个问题是在不断的改变边界点.

这个问题的求解区域是 $0 \leqslant x < \infty$. 在求解时, 无穷远的边界是被代之以有限远点, 不断地改变这个边界点, 使它在足够大时能够收敛. 并且从收敛的一致性验证结论的可靠性. 在初值条件函数文件中, 从一个足够大的边界点开始, 求出一个解, 然后改变边界点, 使解也得到不断的伸展.

代替无穷远边界点的的第一个猜测值是 3, 它能满足边界条件也能得到很好的收敛. 作为对比, 还给出了原始文献中结果 $f''(0) = 0.92768$.

```
function fsbvp
infinity = 3;
maxinfinity = 6;

solinit = bvpinit(linspace(0,infinity,5),[0 0 1]);
sol = bvp4c(@fsode,@fsbc,solinit);
eta = sol.x;
f = sol.y;

fprintf('\n');
fprintf('Cebeci \& Keller report that f''''(0) = 0.92768.\n')
fprintf('Value computed using infinity = %g is...
        %7.5f.\n',infinity,f(3,1))

figure
plot(eta,f(2,:),eta(end),f(2,end),'o');
axis([0 maxinfinity 0 1.4]);
title('Falkner-Skan equation, positive wall shear, \beta = 0.5.')
xlabel('\eta')
ylabel('df/d\eta')
hold on
drawnow
shg

for Bnew = infinity+1:maxinfinity

  solinit = bvpinit(sol,[0 Bnew]);   % Extend the solution to Bnew.
  sol = bvp4c(@fsode,@fsbc,solinit);
  eta = sol.x;
  f = sol.y;

  fprintf('Value computed using infinity = %g is %7.5f.\n',Bnew,f(3,1))
  plot(eta,f(2,:),eta(end),f(2,end),'o');
  drawnow

end
hold off

% ------------------------------------------------
```

```
function dfdeta = fsode(eta,f)
beta = 0.5;
dfdeta = [ f(2)
           f(3)
          -f(1)*f(3) - beta*(1 - f(2)^2) ];

% ------------------------------------------------------------

function res = fsbc(f0,finf)
res = [f0(1)
       f0(2)
       finf(2) - 1];
```

此解的图如图 8.4 所示.

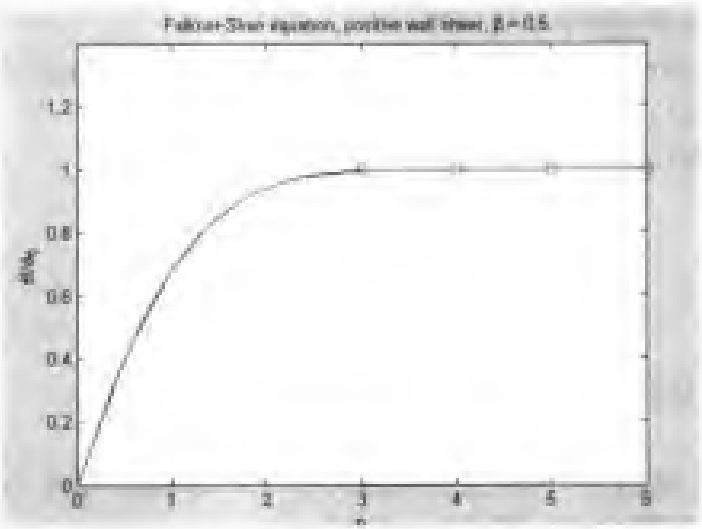


图 8.4 Falkner-Skan 边值问题

8.1.6 在 $x=0$ 处有突变的问题

所解的方程是

$$\begin{cases} \varepsilon y'' + xy' = -\varepsilon\pi^2\cos\pi x - \pi x\sin\pi x \quad (-1 \leqslant x \leqslant 1) \\ y(-1) = -2,\ \ y(1) = 0 \end{cases}$$

其中 $0 < \varepsilon \ll 1$,ε 是各个子函数共用的一个参数, bvp4c 要把它的值传递给所有的函数, 如 shockODE, shockBC, shockJac and shockBCJac .

解在 $x=0$ 处有一个突变，本例介绍一种通过网格选择的技巧来解决这个问题。本例表明，在数值解上很困难的问题却可以借助于连续性来成功地解决。在这个问题中，解析的偏导数很容易求出，可以用它来帮助解题。

将方程写成一阶的常微分方程组，然后用 shockODE and shockBC 对它和边界条件分别编程。它们的偏导数分别用 shockJac and shockBCJac 编程并且用 options 传递给算法指令。选项'Vectorized' 提示算法指令，微分方程是矢量化的，也即 shockODE([x1 x2 ...],[y1 y2 ...],e) 返回的是 [shockODE(x1,y1,e) shockODE(x2,y2,e)...]. 这种编程会提高程序的效率。

```
function shockbvp
options = bvpset('FJacobian',@shockJac,'BCJacobian',...
          @shockBCJac,'Vectorized','on');

sol = bvpinit([-1 -0.5 0 0.5 1],[1 0]);   %%猜测初始网格与初始解
  %%由连续性得到的当 e = 0.01,0.001,0.0001 时的解
for i=2:4
    e = e/10;
    sol = bvp4c(@shockODE,@shockBC,sol,options,e);
end

figure;
plot(sol.x,sol.y(1,:));
axis([-1 1 -2.2 2.2]);
title(['There is a shock at x = 0 when \epsilon =' sprintf('%.e',e) '.']);
xlabel('x');
ylabel('solution y');
   function dydx = shockODE(x,y,e)
  %%计算矢量化的 ODE 函数
pix = pi*x;
dydx = [ y(2,:)
         -x/e.*y(2,:)  - pi^2*cos(pix) - pix/e.*sin(pix) ];
function res = shockBC(ya,yb,e)  %%计算边界条件
res = [ ya(1)+2
        yb(1) ];

function jac = shockJac(x,y,e)          %%计算雅可比行列式
jac = [ 0 1
```

```
0 -x/e ];

function [dBCdya,dBCdyb] = shockBCJac(ya,yb,e)
  %%计算边界条件的偏导数
dBCdya = [ 1 0
           0 0 ];

dBCdyb = [ 0 0
           1 0 ];
```

此解的图如图 8.5 所示.

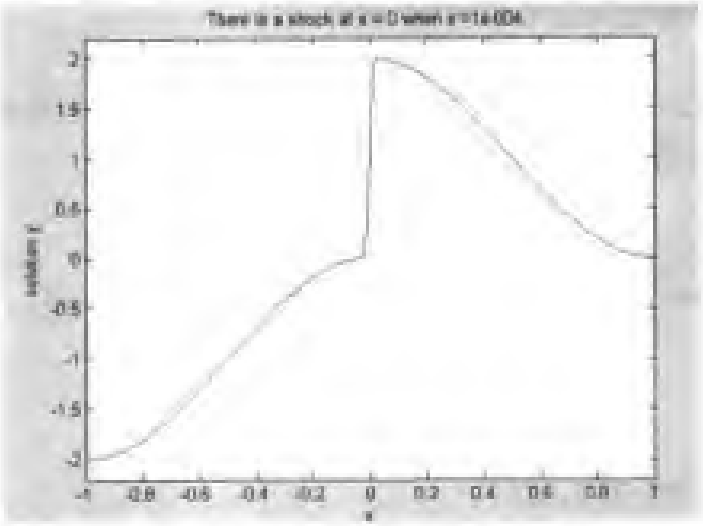


图 8.5 在 $x=0$ 处有突变的问题的解

8.2 用 pdepe 解一维初值边值问题

MATLAB 有专门的指令 pdepe 解一维的椭圆型或抛物型方程组的初值边值问题，使用指令 pdepe 要写三个辅助函数文件，即微分方程函数文件，初始条件函数文件，边界条件函数文件.

8.2.1 指令 pdepe 的用法

1. 指令 pdepe 的功能

可以用 MATLAB 的专门指令 pdepe 求解如下形式的一维的椭圆型或抛物型方程组的初值边值问题.

$$c\left(x,t,u,\frac{\partial u}{\partial t}\right)\frac{\partial u}{\partial t}=x^{-m}\frac{\partial}{\partial x}\left(x^m f\left(x,t,u,\frac{\partial u}{\partial x}\right)\right)+s\left(x,t,u,\frac{\partial u}{\partial x}\right)$$

其中 $t_0 \leqslant t \leqslant t_f$, $a \leqslant x \leqslant b$, 并且 [a,b] 必须是有限的区间. 如果 $m > 0$ 必须有 $a \geqslant 0$. $f\left(x,t,u,\dfrac{\partial u}{\partial x}\right)$ 是通量项, $s\left(x,t,u,\dfrac{\partial u}{\partial x}\right)$ 是源项. 对 t 的偏导数的耦合仅限于与对角矩阵 $c\left(x,t,u,\dfrac{\partial u}{\partial x}\right)$ 相乘中. 这个矩阵的元素或者全为零或者为正. 为零的元素对应椭圆型方程, 而其他的元素对应抛物型方程. 至少必须要有一个抛物型方程. 在某些孤立的 x 的数据网格的点上, 对应于抛物型方程的 c 的元素可以为零. 如果网格点在材料的交界面上, 则 c 和 s 由于材料界面而产生的不连续性也是可以的.

当 $t = t_0$ 时, 对所有的 x, 解满足初始条件

$$u(x,t_0) = u_0(x)$$

对所有的 t, 当 $x = a$ 或者 $x = b$ 时解满足的边界条件是

$$p(x,t,u) + q(x,t)f\left(x,t,u,\frac{\partial u}{\partial x}\right) = 0$$

q 的元素或者全为零或者全不为零. 注意边界条件是用通量项 f 表述而不是用 $\dfrac{\partial u}{\partial x}$ 表述的. 此外, 在两个系数中, 只有 p 与 u 有关.

2. 指令 pdepe 的语句格式

调用 pdepe 的语句有三种格式:

SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan)

SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan,options)

SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan,options,p1,p2...)

其中各个参量的含义为

m	m=0,1,2 分别对应 0 平面对称性, 1 柱对称性, 2 球对称性,
pdefun	所求解的各个方程;
icfun	定义初始条件的函数文件;
bcfun	定义边界条件的函数文件;
xmesh	矢量 [x0, x1, ..., xn] 是空间变量 x 的取值 要求 $x0 < x1 < ... < xn$, 且元素数目 $\geqslant 3$;
tspan	矢量 [t0, t1, ..., tf] 是时间变量 t 的取值, 要求 $t0 < t1 < ... < tf$, 且元素数目 $\geqslant 3$;
options	在 ode 算法指令中可用的某些选项如 RelTol, AbsTol, Norm-

Control,InitialStep,and MaxStep 在 pdepe 中也可使用，请参看 odeset;

p1,p2,... 传递给 pdefun,icfun,bcfun 的参数。

3. 几种语句格式的比较

下面对几种格式的用法加以比较。

(1) **SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan)**

这种格式用于解一维的椭圆型或抛物型方程组的初值边值问题，方程的变量为 x,t . 对 tspan 指定的每一个时刻，由空间变量离散化所得的常微分方程通过积分得到近似解。 pdepe 指令会返回解在 x 网格上每一点的值。

在这个格式中， xmesh(1) 和 xmesh(end) 对应边界点 a 和 b, tsapn(1) 和 tspan(end) 对应时间的起点 t_0 和终点 t_f .

pdepe 返回多维矩阵形式的解 sol . ui=sol(:,:,i) 是解矢量 u 的第 i 个分量. 元素 ui(j,k)=sol(j,k,i) 是解矢量 ui 在 (t,x)=(tspan(j),xmesh(k)) 处的近似解。 ui = sol(j, :, i) 是解的第 i 个分量在时间 T(j) 和 x 网格 xmesh(:) 的点的近似值. 使用 pdeval 可以计算 xmesh 没有包括的点上的 u_i 和 $\partial u_i/\partial x$ 值。 pdeval 的语句格式是

[uout,duoutdx]=pdeval(m,x,ui,xout)

其中 x, ui 原有的变量值与函数值， $xout$ 是新的变量值， $uout, duoutdx$ 是求出的函数与导数值.

(2) **SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan,options)**

用法同上，只是默认的积分参数被 options 中选项所代替. options 中的选项由 odeset 建立，其中只有以下的项可以改变， RelTol, AbsTol, NormContral, InitialStep, MaxStep . 由输入量 options 所保留的默认值一般是足够用的。

(3) **SOL = pdepe(m,pdefun,icfun,bcfun,xmesh,tspan,options,p1,p2...)**

用法相同，还可传递参数 p1,p2 给函数 pdefun,icfun,,bcfun . 使用 options = [ ] 表示不改变原来的设定. 详情见 odeset .

4. 辅助函数文件

微分方程函数文件 pdefun 计算方程中的各项系数 c, f, s 各项，它的形式为

[c,f,s]=pdefun(x,t,u,dudx)

输入量是标量 x,t, 以及矢量 $u.dudx$. 它是 u 及其对 x 偏导数的近似解。c,f,s 都是列矢量。 c 存储的是矩阵 C 的对角元.

初始条件函数文件 icfun 计算初始条件的值，其形式为

u=icfun

当用 x 调用时， icfun 计算与返回的是在列矢量形式的 u 中在 x 处的分量.

边界条件函数文件 bcfun 计算的是边界条件中 p,q 的值，它的形式是

[pl,ql,pr,qr]=bcfun(xl,ul,xr,ur,t)

ul 是在左边界 xl=a 处的近似解， ur 是在右边界 xr=b 处的近似解， pl,ql 是对应于 p,q 在 xl 处的值的列矢量，类似地， pr, qr 对应于 xr 。当 $m>0$ 且 $a=0$, 在接近 $x=0$ 处的解的有界性，要求通量 f 在 $a=0$ 为零。 pdepe 会自动地强加这种边界条件而略去在 pl, ql 中的返回值.

5. 几点说明

tspan 指令 pdepe 用 ODE 的算法指令来作积分， ODE 的算法指令是动态地选择时间步长与公式。 tspan 的元素仅仅是说明，你想在何处获得答案，效果而与 tspan 的长度依赖甚弱.

xmesh 解的二阶近似是在 xmesh 中指定的网格上完成的。一般来说，在解变化剧烈的地方最好使用较密的网格。 pdepe 不会自动的选择 x 的网格。用户要在 xmesh 中提供合适的固定的网格，解的效果强烈地依赖于 xmesh 的长度。当 $m>0$ 时，就不需要在 $x=0$ 时使用很细的网格来考虑坐标变量的奇异性.

时间积分是用 ode15s 实现的， pdepe 使用 ode15s 的功能来解微分代数方程，它产生于当方程 (2-1) 包含椭圆方程时，并且用它来处理稀疏雅可比矩阵。

在离散化以后，椭圆方程产生了代数方程。如果对应于椭圆方程的初始条件中的元素与离散化不相洽， pdepe 会尝试在开始时间积分以前调整它们。因此，返回的解对初始条件在与其他时间点相比较来说，有离散误差。如果这个网格足够细小， pdepe 可能找到与给定的初始条件相一致的条件，如果 pedpe 显示信息，表示很难找到相洽的初始条件，那么请尝试进一步细分网格。

对于抛物型方程，则不必要对初始条件矢量的元素进行调整.

8.2.2 例题

本节介绍几个使用 pdepe 的实例。

例 1

$$\begin{cases} \pi^2 u_t = u_{xx} \quad (0 \leqslant x \leqslant 1, t \geqslant 0) \\ u(x,0) = \sin \pi x \\ u(0,t) = 0, \quad \pi e^{-t} + u_x(1,t) = 0 \end{cases}$$

这是一个最简单的例子，只有一个偏微分方程而且有解析解 $u(x,t) = \exp(-t) * \sin(\pi * x)$，用来说明如何表示公式，可以直接计算和画图．显然，初值条件是 $\sin(\pi * x)$．边值条件选择了两种形式，以便练习边值条件的写法．为了对比，还画出了解析解与数值值的误差．实际程序如下：

```
function  pdex1
m = 0;
x = linspace(0,1,20);
t = linspace(0,2,5);

sol = pdepe(m,@pdex1pde,@pdex1ic,@pdex1bc,x,t);
u = sol(:,:,1);

subplot(1,2,1)
surf(x,t,u);
title('Numerical solution computed with 20 mesh points.');
xlabel('Distance x');
ylabel('Time t');

subplot(1,2,2)
surf(x,t,exp(-t)'*sin(pi*x)-u);
title('Erro of  true solution and numerical solution.');
xlabel('Distance x');
ylabel('Time t');

% --------------------------------------------------------------

function [c,f,s] = pdex1pde(x,t,u,DuDx)
c = pi^2;
f = DuDx;
s = 0;

% --------------------------------------------------------------
```

```
function u0 = pdex1ic(x)
u0 = sin(pi*x);

% ---------------------------------------------------------------
function [pl,ql,pr,qr] = pdex1bc(xl,ul,xr,ur,t)
pl = ul;
ql = 0;
pr = pi * exp(-t);
qr = 1;
```

所画图形如图 8.6 所示。

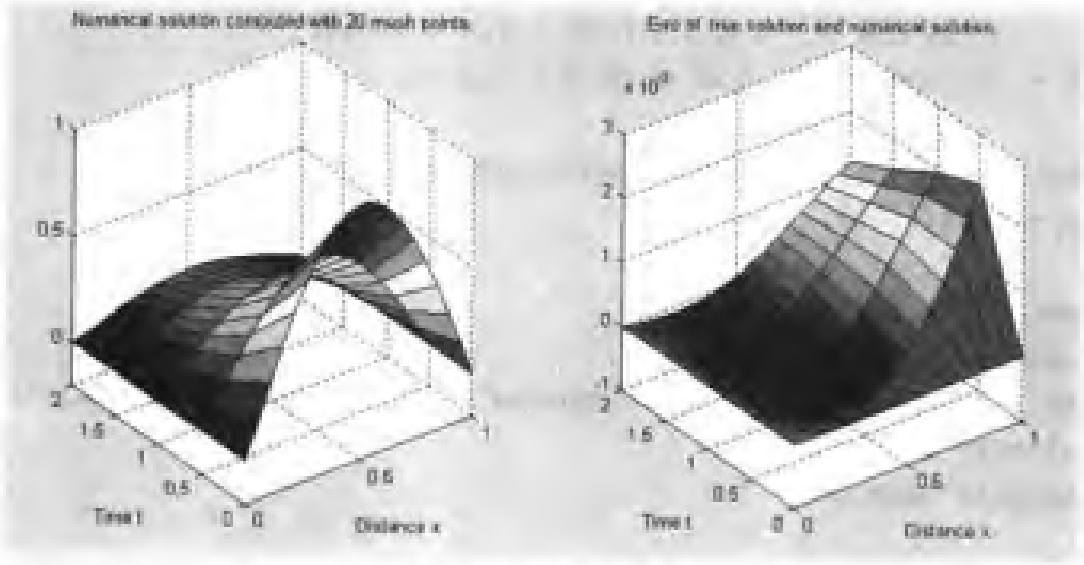


图 8.6 例 1 的图形

例 2

$$\begin{cases} \dfrac{\partial u}{\partial t} = x^{-2}\dfrac{\partial}{\partial x}\left[5x^2\dfrac{\partial u}{\partial x}\right] - 1000\mathrm{e}^u & (0 \leqslant x \leqslant 0.5) \\ \dfrac{\partial u}{\partial t} = x^{-2}\dfrac{\partial}{\partial x}\left[x^2\dfrac{\partial u}{\partial x}\right] - \mathrm{e}^u & (0.5 \leqslant x \leqslant 1) \\ u(x,0) = 0, & (0 <= x < 1), \quad u(1,0) = 1 \\ \dfrac{\partial u(0,t)}{\partial x} = 0, \quad u(1,t) = 1 \end{cases}$$

这是一个具有材料界面的例子，因此，在 $x = 0.5$ 有不连续性，除此之外，在 $x = 1$ 端，初始条件也有不连续性。另外还有来自球对称性的奇异性。

通量 f 在 $x = 0.5$ 应该是连续的，所以在材料的界面上偏导数要乘以因子 5。

左边界条件 $\partial u/\partial x = 0$ 表明球对称的解是以原点为界。 pdepe 会自动地加上这个边界条件而忽略由边界函数 PDEX2BC 所指定的边界条件。右边界条件是 $u(1,t) = 1$，仍然要写成边界条件函数指定的格式。

为了包括界面，空间网格应该包括 $x = 0.5$ 这个点，还要包括接近 1.0 的点，因为初始条件与边界条件的不连续性在这里消失了。对很小变化的 t，解会发生急剧的变化。在本例中，t 选择阶梯式的数值，就是为了求解这个急剧变化的问题。为了在图形中看出这个变化，输出的时间要要合理地选择。这里画出的图形很明显地表现了解在界面的变化。

```
function pdex2
m = 2;
x = [0 0.1 0.2 0.3 0.4 0.45 0.475 0.5 0.525...
     0.55 0.6 0.7 0.8 0.9 0.95 0.975 0.99 1];
t = [0 0.001 0.005 0.01 0.05 0.1 0.5 1];

sol = pdepe(m,@pdex2pde,@pdex2ic,@pdex2bc,x,t);
u = sol(:,:,1);

subplot(1,2,1)
surf(x,t,u);
title('Numerical solution computed using a nonuniform mesh.');
xlabel('Distance x');
ylabel('Time t');

subplot(1,2,2)
plot(x,u,x,u,'*');
xlabel('Distance x');
title('Solution profiles at a selection of times.');

% --------------------------------------------------------------

function [c,f,s] = pdex2pde(x,t,u,DuDx)
c = 1;
if x <= 0.5
  f = 5*DuDx;
  s = -1000*exp(u);
else
```

```
  f = DuDx;
  s = -exp(u);
end

% ---------------------------------------------------------------------

function u0 = pdex2ic(x)
if x < 1
  u0 = 0;
else
  u0 = 1;
end

% ---------------------------------------------------------------------

function [pl,ql,pr,qr] = pdex2bc(xl,ul,xr,ur,t)
pl = 0;  % pdepe will use pl=ql=0 because this
ql = 0;  % problem has a singularity: xl=0 and m>0
pr = ur - 1;
qr = 0;
```

所画图形如图 8.7 所示。

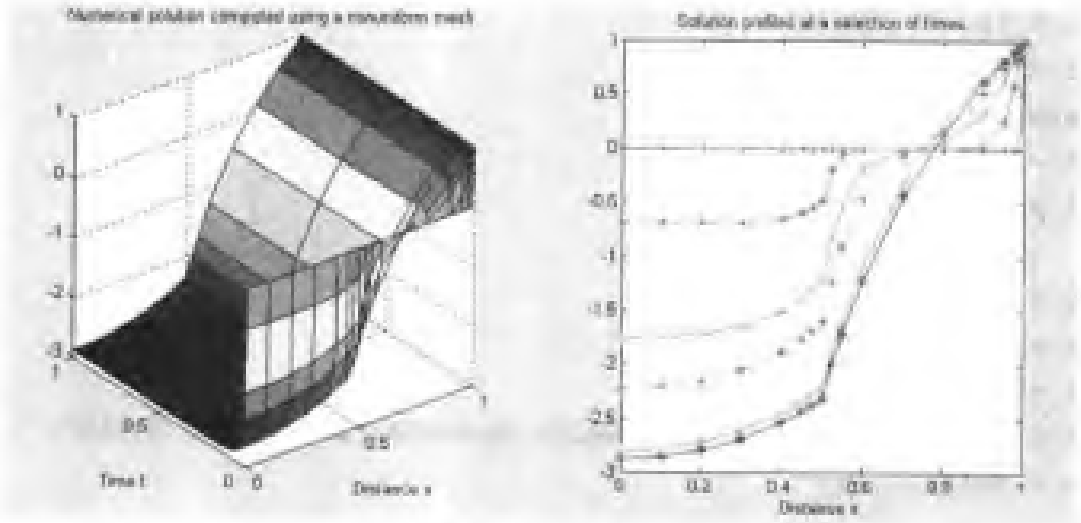


图 8.7 例 2 的图形

例 3

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{\partial}{\partial x}\left(d\dfrac{\partial u}{\partial x}\right) - \dfrac{\eta}{L}\dfrac{\partial u}{\partial x} \quad (0 \leqslant x \leqslant L, t \geqslant 0) \\ u(x,0) = \dfrac{K \cdot L}{d}\dfrac{1 - \mathrm{e}^{-\eta(1-x/L)}}{\eta} \\ u(0,t) = 0, \quad u(L,t) = 0 \end{cases}$$

这里， d, η, K 是物理常数，且取 $L = 1$.

这个例子是用以说明用 pdeval 获得偏导数的值，这是问题的解的一部分。为了得到精确的偏导数的值，比较细的网格是必须的．获得的值是在较大的 t 的范围内.

这个问题来自跃迁理论，用以说明具有级数解的偏微分方程．物理上感兴趣的量是发射器的放电电流

$$I(t) = I_p \frac{d}{K}\frac{\partial u(0,t)}{\partial x}$$

这里 I_p 是另一个物理常数，它只在 $t > 0$ 时有意义，因为在边界 $x = 0$ 处，对 $t = 0$ 与 $t > 0$ 是矛盾的．这表明，级数解在 $t = 0$ 时是不收敛的．方程中的常数是任意指定的． $I(t)$ 变化很快，所以比较大范围的 t 的输出是需要．回想起 t 的数目与位置对计算的结果的影响较小．此外，近似解只是空间变量的二阶导数，而对时间的偏导数只是一阶．因此，在计算发射器的放电电流与级数解中获得的结果有一定的合理性。

```
function pdex3
L = 1;
d = 0.1;
eta = 10;
K = d;
I_p = 1;

m = 0;
x = linspace(0,L,41);
t = linspace(0,1,51);
sol = pdepe(m,@pdex3pde,@pdex3ic,@pdex3bc,x,t,[],L,K,d,eta);
u = sol(:,:,1);

subplot(1,2,1)
surf(x,t,u);
title('Numerical solution computed with 41 mesh points.');
xlabel('Distance x');
```

```
ylabel('Time t');

nt = length(t);
I = zeros(1,nt);
seriesI = zeros(1,nt);
iok = 2:nt;
for j = iok
  % At time t(j), compute Du/Dx at x = 0.
  [dummy,I(j)] = pdeval(m,x,u(j,:),0);
  seriesI(j) = serex3(t(j),L,d,eta,I_p);
end
% I(t) = (I_p*d/K)*Du(0,t)/Dx
I = (I_p*d/K)*I;

subplot(1,2,2)
plot(t(iok),I(iok),t(iok),seriesI(iok));
legend('From PDEPE','From series');
title('Emitter discharge current I(t)');
xlabel('Time t');

% --------------------------------------------------------------

function [c,f,s] = pdex3pde(x,t,u,DuDx,L,K,d,eta)
c = 1;
f = d*DuDx;
s = -d*eta*DuDx;

% --------------------------------------------------------------

function u0 = pdex3ic(x,L,K,d,eta)
u0 = (K*L/d)*(1 - exp(-eta*(1 - x)))/eta;

% --------------------------------------------------------------

function [pl,ql,pr,qr] = pdex3bc(xl,ul,xr,ur,t,L,K,d,eta)
pl = ul;
ql = 0;
pr = ur;
qr = 0;
```

```
% ------------------------------------------------------------------

function It = serex3(t,L,d,eta,I_p)
%  Approximate I(t) by 40 terms of a series expansion.

It = 0;
for n = 1:40
  temp = (n*pi)^2 + 0.25*eta^2;
  It = It + ((n*pi)^2 / temp)* exp(-(d/L^2)*temp*t);
end
It = 2*I_p*((1 - exp(-eta))/eta)*It;
```

所画图形如图 8.8 所示.

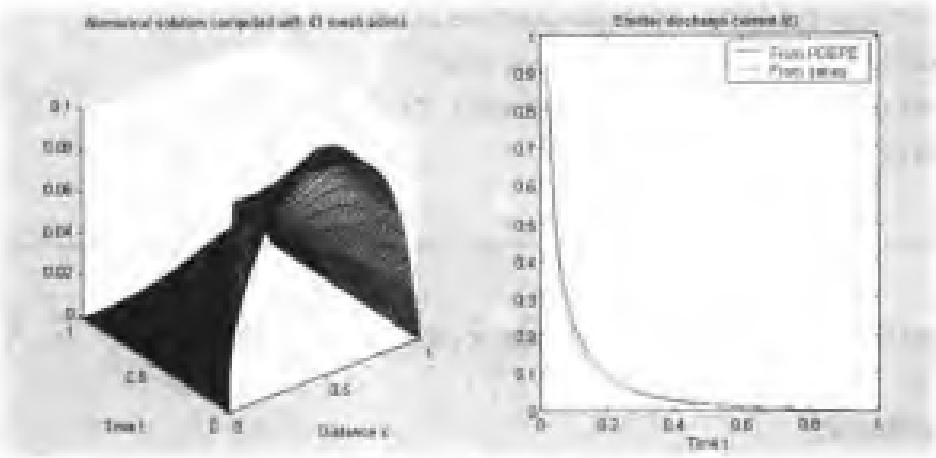


图 8.8 例 3 的图形

例 4

要解的偏微分方程组是

$$\frac{\partial u_1}{\partial t} = 0.024\frac{\partial^2 u_1}{\partial x^2} - F(u_1 - u_2)$$

$$\frac{\partial u_2}{\partial t} = 0.170\frac{\partial^2 u_2}{\partial x^2} + F(u_1 - u_2)$$

其中

$$F(y) = \exp(5.73y) - \exp(-11.46y)$$

且 $0 \leqslant x \leqslant 1, t \geqslant 0$

初始条件是

$$u_1(x,0)=1,\quad u_2(x,0)=0$$

边界条件是

$$\frac{\partial u_1}{\partial x}(0,t)=0,\quad u_1(1,t)=1$$
$$u_2(0,t)=0,\quad \frac{\partial u_2}{\partial x}(1,t)=0$$

这是一个偏微分方程组的例子，它来自电动力学．在区间的两端有一个边界层，当 t 变化很小时，解就会剧烈的变化。

按照 pdepe 要求的形式，把求解的方程改写为

$$\begin{bmatrix}1\\1\end{bmatrix}.*\frac{\partial}{\partial t}\begin{bmatrix}u_1\\u_2\end{bmatrix}=\frac{\partial}{\partial x}\begin{bmatrix}0.024(\partial u_1/\partial x)\\0.170(\partial u_2/\partial x)\end{bmatrix}+\begin{bmatrix}-F(u_1-u_2)\\F(u_1-u_2)\end{bmatrix}$$

u 的偏导数在边界的行为要写成通量的形式，这也是 pdepe 要求的．因此，在左边界的边界条件是

$$\begin{bmatrix}0\\u_2\end{bmatrix}+\begin{bmatrix}1\\0\end{bmatrix}.*\begin{bmatrix}0.024(\partial u_1/\partial x)\\0.170(\partial u_2/\partial x)\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

而在右边界的边界条件的要求是

$$\begin{bmatrix}u_1-1\\0\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}.*\begin{bmatrix}0.024(\partial u_1/\partial x)\\0.170(\partial u_2/\partial x)\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

在边界上，对很小的 t 解就会发生急剧的变化．程序是通过选择时间步长来解这个问题．为了在图形上能看到这种行为，还要合理地选择输出时间．解在 $[0,1]$ 两端都有一个边界层，所以要在 0 和 1 附近放置一个网格来表现这种急剧变化．通常需要借助经验来选择合适的网格．

```
function pdex4
m = 0;
x = [0 0.005 0.01 0.05 0.1 0.2 0.5 0.7 0.9 0.95 0.99 0.995 1];
t = [0 0.005 0.01 0.05 0.1 0.5 1 1.5 2];

sol = pdepe(m,@pdex4pde,@pdex4ic,@pdex4bc,x,t);
```

```
u1 = sol(:,:,1);
u2 = sol(:,:,2);

subplot(1,2,1)
surf(x,t,u1);
title('u1(x,t)');
xlabel('Distance x');
ylabel('Time t');

subplot(1,2,2)
surf(x,t,u2);
title('u2(x,t)');
xlabel('Distance x');
ylabel('Time t');

function [c,f,s] = pdex4pde(x,t,u,DuDx)
c = [1; 1];
f = [0.024; 0.17] .* DuDx;
y = u(1) - u(2);
F = exp(5.73*y)-exp(-11.47*y);
s = [-F; F];

function u0 = pdex4ic(x);
u0 = [1; 0];

function [pl,ql,pr,qr] = pdex4bc(xl,ul,xr,ur,t)
pl = [0; ul(2)];
ql = [1; 0];
pr = [ur(1)-1; 0];
qr = [0; 1];
```

程序中的子函数 pdex4pde, pdex4ic 和 pdex4bc 分别是偏微分方程、初始条件和边界条件. 程序画出的图形如图 8.9 所示.

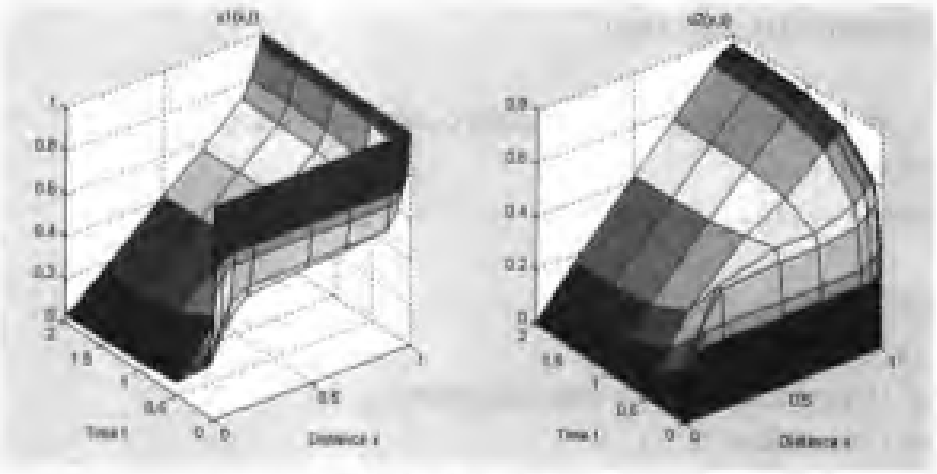


图 8.9 例 4 的图形

例 5

这是一个血管瘤生长初步阶段的数学模型，所解方程组是

$$\begin{cases} \dfrac{\partial n}{\partial t} = \dfrac{\partial}{\partial x}\left(d\dfrac{\partial n}{\partial x} - a\cdot n\dfrac{\partial c}{\partial x}\right) + s\cdot r\cdot n(N-n) \\ \dfrac{\partial c}{\partial t} = \dfrac{\partial}{\partial x}\left(\dfrac{\partial c}{\partial x}\right) + s\left(\dfrac{n}{n+1} - c\right) \end{cases}$$

写成 PDEPE 要求的形式是

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} .* \frac{\partial}{\partial t}\begin{bmatrix} n \\ c \end{bmatrix} = \frac{\partial}{\partial x}\begin{bmatrix} d\dfrac{\partial n}{\partial x} - a\cdot n\dfrac{\partial c}{\partial x} \\ \dfrac{\partial c}{\partial x} \end{bmatrix} + \begin{bmatrix} s\cdot r\cdot n(N-n) \\ s\left(\dfrac{n}{n+1} - c\right) \end{bmatrix}$$

计算中各参数取值为 $d = 1e-3, a = 3.8, s = 3, r = 0.88, N = 1$ 。

初始条件是对持续不变的态 $n = 1, c = 0.5 (0 \leqslant x \leqslant 1)$ 的一个微扰。线性稳定性分析预测，系统会朝一个空间非均匀的解演化。在初始条件函数文件中，一个阶梯函数作为初始值用来激发这种演化。

在区间 [0,1] 的两端，解中的通量都是为零，所以边界条件可以写成

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} .* \begin{bmatrix} d\dfrac{\partial n}{\partial x} - a\cdot n\dfrac{\partial c}{\partial x} \\ \dfrac{\partial c}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

结果表明，为了看到极限的分布状态，要取较长的时间间隔。此外，$c(x,t)$ 的极限分布的变化仅有 0.1%，所以要采用较细的网格。

```
m = 0;
x = linspace(0,1,41);
```

```
t = linspace(0,200,10);

sol = pdepe(m,@pdex5pde,@pdex5ic,@pdex5bc,x,t);
n = sol(:,:,1);
c = sol(:,:,2);

figure;
surf(x,t,c);
title('Distribution of fibronectin--c(x)');
xlabel('Distance x');
ylabel('Time t');

figure;
surf(x,t,n);
title('Distribution of ECs--n(x)');
xlabel('Distance x');
ylabel('Time t');

figure;
plot(x,n(end,:));
title('Final distribution of n(x).');

figure;
plot(x,c(end,:));
title('Final distribution of c(x).');

% ------------------------------------------------------------

function [c,f,s] = pdex5pde(x,t,u,DuDx)
d = 1e-3;
a = 3.8;
S = 3;
r = 0.88;
N = 1;

c = [1; 1];
f = [ d*DuDx(1) - a*u(1)*DuDx(2)
            DuDx(2)             ];
s1 = S*r*u(1)*(N - u(1));
```

```
s2 = S*(u(1)/(u(1) + 1) - u(2));
s = [s1; s2];

% --------------------------------------------------------------

function u0 = pdex5ic(x)
u0 = [1; 0.5];
if x >= 0.3 & x <= 0.6
  u0(1) = 1.05 * u0(1);
  u0(2) = 1.0005 * u0(2);
end

% --------------------------------------------------------------

function [pl,ql,pr,qr] = pdex5bc(xl,ul,xr,ur,t)
pl = [0; 0];
ql = [1; 1];
pr = [0; 0];
qr = [1; 1];
```

程序画出的图形如图 8.10 所示.

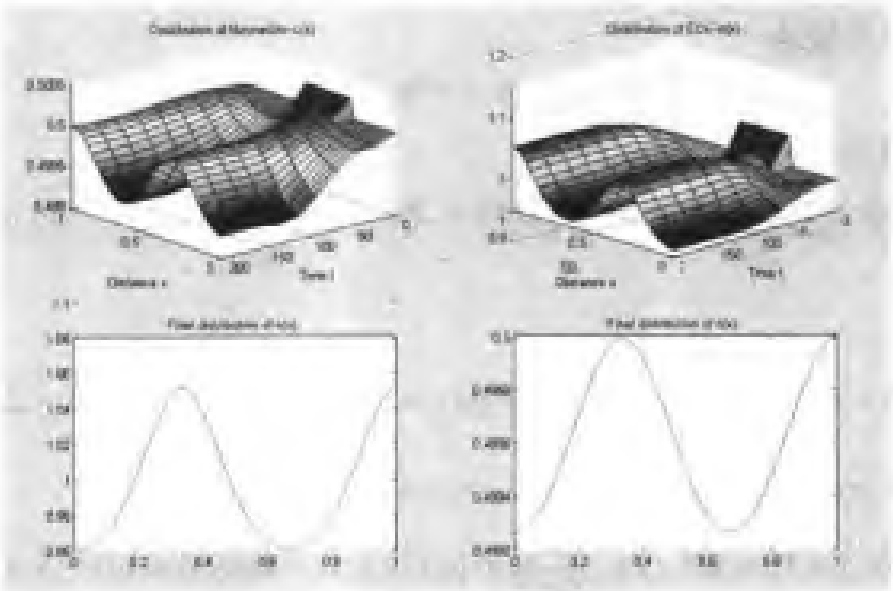


图 8.10 例 5 的图形

8.3 差 分 法

本节对书中用到的差分方程及解法作一简单的说明.

1. 导数的差分公式

在 x 附近对 $f(x)$ 展开, 由泰勒展开公式

$$f(x+h) \approx f(x) + f'(x)h$$

得到前差公式为

$$f'(x) = \frac{f(x+h) - f(x)}{h}$$

同理, 也可以得到后差公式

$$f'(x) = \frac{f(x) - f(x-h)}{h}$$

由后差公式可以得到二阶导数的差分公式为

$$f''(x) = \frac{f'(x+h) - f'(x)}{h} = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

叫中心差分公式.

利用这些公式可以将微分方程写成差分方程.

2. 热传导方程的差分公式

热传导方程是

$$u_t = a^2 u_{xx}$$

可以写成差分形式

$$\frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} \approx a^2 \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{(\Delta x)^2}$$

即

$$u(x,t+\Delta t) \approx u(x,t) + \frac{\Delta t}{(\Delta x)^2} a^2 \left[u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)\right]$$

令 $x = i\Delta x, t = i\Delta t, i = 0,1,2,\cdots,n-1$, 上式可写成

$$u(i,j+1) = u(i,j) + \frac{\Delta t}{(\Delta x)^2} a^2 \left[u(i+1,j) - 2u(i,j) + u(i-1,j)\right]$$

可以证明, 上式的稳定条件为 $\Delta t \leqslant \frac{(\Delta x)^2}{2a^2}$, 即 $\frac{\Delta t}{(\Delta x)^2} a^2 \leqslant \frac{1}{2}$. 稳定且非振荡的条件为 $\frac{\Delta t}{(\Delta x)^2} a^2 \leqslant \frac{1}{4}$, 截断误差为 $O((\Delta x)^2, \Delta t)$, 这个格式称为显式格式.

另一种格式为

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t}$$

$$\approx a^2\frac{u(x+\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x-\Delta x,t+\Delta t)}{(\Delta x)^2}$$

即

$$u(i-1,j+1)+\left[-2-\frac{(\Delta x)^2}{a^2\Delta t}\right]u(i,j+1)+u(i+1,j+1)=-\frac{(\Delta x)^2}{a^2\Delta t}u(i,j)$$

该式称为隐式格式. 两式的计算工作量相当，截断误差为 $O((\Delta x)^2,\Delta t)$, 对任何步长都是恒稳定的. 在 Δt 上取值的唯一限制是，要将截断误差保持在合理的程度上从而节约计算时间.

3. 二维拉普拉斯方程的差分公式

二维拉普拉斯方程是

$$u_{xx}+u_{yy}=0$$

可以利用二阶导数的中心差分公式写成

$$\frac{u(x+\Delta x,y)-2u(x,y)+u(x-\Delta x,y)}{(\Delta x)^2}$$

$$+\frac{u(x,y+\Delta y)-2u(x,y)+u(x,y-\Delta y)}{(\Delta y)^2}\approx 0$$

记

$$\Delta x=\Delta y=h,\quad x=ih,\quad y=jh\quad (i,j=0,1,2,\cdots,n-1)$$

得

$$u(i,j)=\frac{1}{4}\left[u(i+1,j)+u(i-1,j)+u(i,j+1)+u(i,j-1)\right]$$

稳定条件是 $\frac{(\Delta y)^2}{(\Delta x)^2}\leqslant 1$.

同样也有隐式格式的公式为

$$u(i,j+1)-2u(i,j)+u(i,j-1)=-\frac{(\Delta y)^2}{(\Delta x)^2}$$

$$[u(i+1,j+1)-2u(i,j+1)+u(i-1,j+1)]$$

它的截断误差是 $O((\Delta x)^2,(\Delta y)^2)$, 该式是恒稳定的.

4. 弦振动方程的差分公式

弦振动方程的是

$$u_{tt} = a^2 u_{xx}$$

可以写成

$$\frac{u(x,t+\Delta t)-2u(x,t)+u(x,t-\Delta t)}{(\Delta t)^2} = a^2 \frac{u(x+\Delta x,t)-2u(x,t)+u(x-\Delta x,t)}{(\Delta x)^2}$$

令 $x = i\Delta x$, $t = j\Delta t$, 按中心差分的表达式, 可将偏微分变为差分式表示

$$\frac{\partial^2 u}{\partial t^2} = \frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{(\Delta t)^2}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{(\Delta x)^2}$$

得

$$u_{i,j+1} = c(u_{i+1,j}+u_{i-1,j}) + 2(1-c)u_{i,j} - u_{i,j-1}$$

其中

$$c = a^2 \frac{(\Delta t)^2}{(\Delta x)^2}$$

这就是显式差分格式. 其截断误差为 $O(\Delta x^2, \Delta t^2)$, 式中 c 值对双曲线型偏微分方程数值解是很重要的. 当 $c<1$ 时其解是稳定的, 当 $c=1$ 时, 可得到正确的数值解. 当 $c>1$ 时, 其解是不稳定的. 和前面讨论的结果一样, 这意味着, δt 较小, 所得结果较好, δx 较小则所得结果较差.

在解题过程中, 需要利用初始条件, 设 $t=0$ 时, $\frac{\partial u_{i,1}}{\partial t}=0$ 用中心差分可表示为

$$\frac{\partial u_{i,1}}{\partial t} = \frac{u_{i,2}-u_{i,0}}{2\Delta t} = 0$$

于是得 $u_{i,0} = u_{i,2}$ 把它代入上述方程中的最后一项, 并取 $j=1$ 即对应 $t=0$, 得出

$$u_{i,2} = \frac{1}{2}[c(u_{i+1,1}+u_{i-1,1}) + 2(1-c)u_{i,1}]$$

对初位移为零, 初速为 ψ, 初条件可以表为

$$\frac{\partial u_{i1}}{\partial t} = \frac{u_{i2}-u_{i1}}{\Delta t} = \psi$$

由 $u_{i1}=0$ 得

$$u_{i2} = \psi \Delta t$$

8.4 有 限 元 法

本节以泊松方程为例，介绍有限元法的基本概念.

8.4.1 变分原理

设要解的问题是

$$\begin{cases} \nabla^2\phi = -\dfrac{\rho}{\varepsilon} \\ \left(\dfrac{\partial\phi}{\partial n} + \alpha\phi\right)\Big|_L = g \end{cases}$$

求解的区域是 D, L 是 D 的边界线，下面证明上式的解与问题

$$W(\phi) = \int\int_D \left\{\frac{\varepsilon}{2}\left[\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2\right] - \rho\phi\right\}\mathrm{d}x\mathrm{d}y + \oint_L \varepsilon\left(\frac{1}{2}\alpha\phi^2 - g\phi\right)\mathrm{d}l$$

等价.

对上式求变分

$$\begin{aligned}\delta W(\phi) = & \int\int_D \left\{\varepsilon\left[\left(\frac{\partial\phi}{\partial x}\right)\left(\frac{\partial\delta\phi}{\partial x}\right) + \left(\frac{\partial\phi}{\partial y}\right)\left(\frac{\partial\delta\phi}{\partial y}\right)\right] - \rho\delta\phi\right\}\mathrm{d}x\mathrm{d}y \\ & + \varepsilon\oint_L (\alpha\phi\delta\phi - g\delta\phi)\mathrm{d}l\end{aligned}$$

利用边界条件以及格林公式

$$\int\int_D (u\nabla^2 v + \nabla v\cdot\nabla u)\mathrm{d}x\mathrm{d}y = \oint_L u\nabla v\cdot\boldsymbol{n}\mathrm{d}l$$

$\boldsymbol{n}$ 为 D 区域边界 L 上的外法线方向的单位矢量，将上述式子写成

$$\delta W(\phi) = -\int\int_D (\varepsilon\nabla^2\phi + \rho)\delta\phi\mathrm{d}x\mathrm{d}y$$

由泛函的极值条件 $\delta W(\phi) = 0$ 和 $\delta\phi$ 的任意性，就可以得到原来的场方程.

与此类似，对第一类边界条件即上述边界条件中的 $\left.\dfrac{\partial\phi}{\partial n}\right|_L = 0$ 相应的变分泛函为

$$W(\phi) = \int\int_D \left\{\frac{\varepsilon}{2}\left[\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2\right] - \rho\phi\right\}\mathrm{d}x\mathrm{d}y$$

对第二类边值问题即上述边界条件中的 $\alpha = 0$ 时，相应的变分泛函为

$$W(\phi) = \int\int_D \left\{\varepsilon\left[\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2\right] - \rho\phi\right\}\mathrm{d}x\mathrm{d}y - \varepsilon\oint_L g\phi\mathrm{d}l$$

当边界为导体面时，边界面是等电位，为常数，则上式为

$$W(\phi)=\int\int_D\left\{\varepsilon\left[\left(\frac{\partial\phi}{\partial x}\right)^2+\left(\frac{\partial\phi}{\partial y}\right)^2\right]-\rho\phi\right\}\mathrm{d}x\mathrm{d}y-g\phi$$

8.4.2 计算格式

在有限元法中，将二维的场域划分为三角形，每个三角形称为一个元素，划分之后，要对所有的三角形和节点编号．一般规定，各元素的节点编号取逆时针方向，另外注意，尽量使各元素中最大的编号与最小的编号之差尽可能小．这样可以使最后形成的代数方程组的系数矩阵带宽较小，以减少计算机的存储量．另外应该使每个边界元素只有一条边落在边界上．

计算格式的建立过程如下．

设任一元素 e(即一个小三角形) 上的函数值 $\phi(x,y)$ 可以表示成

$$\phi=\phi(x,y)=\alpha_1+\alpha_2x+\alpha_3y$$

在该元素的三个节点 $(x_l,y_l),(l=i,j,m)$ 上有

$$\begin{cases}\phi_i=\phi(x_i,y_i)=\alpha_1+\alpha_2x_i+y_i\\ \phi_j=\phi(x_j,y_j)=\alpha_1+\alpha_2x_j+\alpha_3y_j\\ \phi_m=\phi(x_m,y_m)=\alpha_1+\alpha_2x_m+\alpha_3y_m\end{cases}\tag{8.4.1}$$

由此解得

$$\begin{cases}\alpha_1=\dfrac{1}{2\Delta}\displaystyle\sum_{l=i,j,m}a_l\phi_l\\ \alpha_2=\dfrac{1}{2\Delta}\displaystyle\sum_{l=i,j,m}b_l\phi_l\\ \alpha_3=\dfrac{1}{2\Delta}\displaystyle\sum_{l=i,j,m}c_l\phi_l\end{cases}\tag{8.4.2}$$

其中

$$\begin{cases}a_i=x_jy_m-x_my_j\\ b_i=y_j-y_m\\ c_i=x_m-x_j\end{cases}$$

轮换坐标可得 $a_j, b_j, c_j, a_m, b_m, c_m$ 的表达式. Δ 为该三角形元素的面积.

$$\Delta = \frac{1}{2}\begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{vmatrix} = \frac{1}{2}(b_i c_j - b_j c_i)$$

将 (8.4.2) 式代入 (8.4.1) 式得到三角形元素内一点的函数值的插值公式

$$\phi = \frac{1}{2\Delta}\left[\sum_{l=i,j,m}(a_l + b_l x + c_l y)\phi_l\right]$$

由它可得到 ϕ 的两个偏微分为

$$\frac{\partial \phi}{\partial x} = \frac{1}{2\Delta}\sum_{l=i,j,m} b_l \phi_l$$

$$\frac{\partial \phi}{\partial y} = \frac{1}{2\Delta}\sum_{l=i,j,m} c_l \phi_l$$

引入记号

$$(\nabla\phi)_e = \begin{pmatrix} \dfrac{\partial \phi}{\partial x} \\ \dfrac{\partial \phi}{\partial y} \end{pmatrix}, \quad (\Phi)_e = \begin{pmatrix} \phi_i \\ \phi_j \\ \phi_l \end{pmatrix}, \quad (B)_e = \frac{1}{2\Delta}\begin{pmatrix} b_i & b_j & b_m \\ c_i & c_j & c_m \end{pmatrix} \tag{8.4.3}$$

则上式可以写成

$$(\nabla\phi)_e = (B)_e(\Phi)_e$$

下面推导前面的泛函问题的有限元素表示法，以第一边值问题为例，这个问题是

$$\begin{cases} \delta W(\phi) = 0 \\ W(\phi) = \displaystyle\int\int_D \left\{\frac{\varepsilon}{2}\left[\left(\frac{\partial \phi}{\partial x}\right)^2 + \left(\frac{\partial \phi}{\partial y}\right)^2\right] - \rho\phi\right\} \mathrm{d}x\mathrm{d}y \\ \phi|_L = \phi_0 \end{cases}$$

显然，总的泛函可以视为各个元素上的泛函之和，即

$$\begin{aligned} W(\phi) &= \sum_{e=1}^{e_0}\left\{\int\int_e \frac{\varepsilon}{2}\left[\left(\frac{\partial \phi}{\partial x}\right)^2 + \left(\frac{\partial \phi}{\partial y}\right)^2\right]\mathrm{d}x\mathrm{d}y - \int\int_e \rho\phi \mathrm{d}x\mathrm{d}y\right\} \\ &= \sum_{e=1}^{e_0}[W_{1e}(\phi) + W_{2e}(\phi)] = W_1(\phi) + W_2(\phi) \end{aligned}$$

引入 (8.4.3) 定义的符号，得

$$\begin{aligned} W_{1e}(\phi) &= \int\int_e \frac{\varepsilon}{2}(\nabla\phi)_e^{\mathrm{T}}(\nabla\phi)_e \mathrm{d}x\mathrm{d}y \\ &= \frac{\varepsilon}{2}\int\int_e [(B)_e(\Phi)_e]^{\mathrm{T}}[(B)_e(\Phi)_e]\mathrm{d}x\mathrm{d}y \\ &= \frac{\varepsilon}{2}(\Phi)_e^{\mathrm{T}}\left[\int\int_e (B)_e^{\mathrm{T}}(B)_e \mathrm{d}x\mathrm{d}y\right](\Phi)_e \\ &= \frac{1}{2}(\Phi)_e^{\mathrm{T}}(K)_e(\Phi)_e \end{aligned}$$

其中 $(\Phi)_e$ 不是坐标的函数，可移出积分号外，并且定义

$$(K)_e = \int\int_e \varepsilon(B)_e^{\mathrm{T}}(B)_e \mathrm{d}x\mathrm{d}y$$

同样，$(B)_e$ 也不是坐标的函数，也可移出积分号外. 这样就很容易得出 $(K)_e$ 的表达式.

$$\begin{aligned} (K)_e &= \frac{\varepsilon}{4\Delta}\begin{pmatrix} K_{ii}^e & K_{ij}^e & K_{im}^e \\ K_{ji}^e & K_{jj}^e & K_{jm}^e \\ K_{mi}^e & K_{mj}^e & K_{mm}^e \end{pmatrix} \\ &= \frac{\varepsilon}{4\Delta}\begin{pmatrix} b_i^2+c_i^2 & b_ib_j+c_ic_j & b_ib_m+c_ic_m \\ b_jb_i+c_jc_i & b_j^2+c_j^2 & b_jb_m+c_jc_m \\ b_mb_i+c_mc_i & b_mb_j+c_mc_j & b_m^2+c_m^2 \end{pmatrix} \end{aligned}$$

由此可见，$(K)_e$ 是一个三阶的对称方阵，其一般形式为

$$K_{rs}^e = K_{sr}^e = \frac{\varepsilon}{4\Delta}(b_rb_s + c_rc_s) \qquad (r,s = i,j,m)$$

最后得到 $W_1(\phi)$ 的向量表示为

$$W_1(\phi) = \sum_{e=1}^{e_0} W_{1e}(\phi) = \frac{1}{2}\sum_{e=1}^{e_0}(\Phi)_e^{\mathrm{T}}(K)_e(\Phi)_e \tag{8.4.4}$$

为了将 $W_2(\phi)$ 表示为向量，定义

$$(N)_e = \begin{pmatrix} N_i \\ N_j \\ N_m \end{pmatrix} = \begin{pmatrix} \dfrac{1}{2\Delta}(a_i+b_ix+c_iy) \\ \dfrac{1}{2\Delta}(a_j+b_jx+c_jy) \\ \dfrac{1}{2\Delta}(a_m+b_mx+c_my) \end{pmatrix}$$

则 ϕ 的插值表示式可以写成

$$\phi = (\Phi)_e^{\mathrm{T}}(N)_e$$

所以

$$W_{2e}(\phi) = \int\int_e \rho_e(\Phi)_e^{\mathrm{T}}(N)_e \mathrm{d}x\mathrm{d}y \tag{8.4.5}$$

假定三角形元素足够小，ρ 值可取为 ρ_i, ρ_j, ρ_m 的平均值 ρ'，于是上式可以写成

$$W_{2e}(\phi) = (\Phi)_e^{\mathrm{T}} \int\int_e \rho'_e(N)_e \mathrm{d}x\mathrm{d}y = (\Phi)_e^{\mathrm{T}}(p)_e$$

由欧拉公式

$$\int\int_\Delta N_1^a N_2^b N_3^c \mathrm{d}x\mathrm{d}y = \frac{a!b!c!}{(a+b+c+2)!}2\Delta$$

可以求得

$$W_{2e}(\phi) = (\Phi)_e^{\mathrm{T}}(p)_e = (\Phi)_e^{\mathrm{T}}\frac{\Delta}{3}\rho'_e\begin{pmatrix}1\\1\\1\end{pmatrix}$$

即得到

$$(p)_e = \begin{pmatrix}p_i^e\\p_j^e\\p_m^e\end{pmatrix} = \frac{\Delta}{3}\rho'_e\begin{pmatrix}1\\1\\1\end{pmatrix}$$

最后得到 $W_2(\phi)$ 的向量记法为

$$W_2(\phi) = \sum_{e=1}^{e_0} W_{2e}(\phi) = \sum_{e=1}^{e_0}(\Phi)_e^{\mathrm{T}}(p)_e \tag{8.4.6}$$

综合 (8.4.4) 式和 (8.4.6) 式，就得到泛函表示式为

$$W(\phi) = W_1(\phi) - W_2(\phi) = \frac{1}{2}\sum_{e=1}^{e_0}(\Phi)_e^{\mathrm{T}}(K)_e(\Phi)_e - \sum_{e=1}^{e_0}(\Phi)_e^{\mathrm{T}}(p)_e \tag{8.4.7}$$

引入一个 $3\times n$ 阶的辅助矩阵 $(R)_e$, n 为总节点数,

$$(R)_e=\begin{pmatrix} 0 & \cdots & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 1 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 & \cdots & 0 \\ & & i & & j & & m & & \end{pmatrix}$$

则有

$$(\Phi)_e=(R)_e(\Phi)$$

其中 (Φ) 是场域上所有节点上的函数值向量, 它表示为

$$(\Phi)=(\phi_1,\phi_2,\cdots,\phi_n)^{\mathrm{T}}$$

这样就可以把公式 (8.4.7) 和泛函表示式重新表示为

$$\begin{aligned} W(\phi) &= \frac{1}{2}(\Phi)^{\mathrm{T}}\left[\sum_{e=1}^{e_0}(R)_e^{\mathrm{T}}(K)_e(R)_e\right](\Phi)-(\Phi)^{\mathrm{T}}\left[\sum_{e=1}^{e_0}(R)_e^{\mathrm{T}}(p)_e\right] \\ &= \frac{1}{2}(\Phi)^{\mathrm{T}}(K)(\Phi)-(\Phi)^{\mathrm{T}}(p) \end{aligned}$$

对上式取泛函极值, 则要满足条件

$$\frac{\mathrm{d}}{\mathrm{d}\phi_i}W(\phi)=0 \quad (i=1,2,\cdots,n)$$

可得到必须满足的线性代数方程组为

$$(K)(\Phi)=(p)$$

显然对 $\rho=0$ 时, 对应的方程为拉普拉斯方程公式中的向量 p 为 (0), 则拉普拉斯方程的有限元素方程为齐次的方程组

$$(K)(\Phi)=(0)$$

这里, 我们对总系数矩阵 (K) 和向量 (p) 作一些说明. 由以上的讨论可以看到, (K) 的矩阵元素是由所有的相关的三角形元素对该矩阵元的贡献之和. 具体地讲, 其对角线上某矩阵元素 K_{ll} 是以 l 为节点的各三角形元素对该矩阵元的贡献之和. 矩阵元素 K_{lm} 是以线段 lm 为邻边的某两个三角形元素的贡献 K_{lm}^e 之和. 因此若和 l 节点相邻的节点有 $m_1,m_2,\cdots,m_s$, 那么 (K) 的第 l 行中除了对角矩阵元 K_{ll} 和与第 $m_1,m_2,\cdots,m_s$ 列相交处的矩阵元非零外, 其他的均为零. 所以 (K) 是大型稀疏矩阵. 又由于 $(K)_e$ 是正定对称的, 所以 (K) 也是正定对称的. 同样可知, (p) 的各分量也是各个相关的三角形元素贡献之和.

8.4.3 强加边界条件处理

为了满足第一类边界条件 $\phi|_L = \phi_0$, 可以在对节点编号时，使得 n 个节点中前 n_0 个为内部节点，从 n_0+1 到 n 为边界节点，即

$$\phi_{n_0+i} = \phi_0 \qquad (i = 1, 2, \cdots, n - n_0) \tag{8.4.8}$$

式 (8.4.8) 可以写成向量形式，为此定义

$$(\Phi_2) = (\phi_{n_0+1}, \phi_{n_0+2}, \cdots, \phi_n)^{\mathrm{T}}$$

$$(\Phi_0) = (\phi_{01}, \phi_{02}, \cdots, \phi_{0(n-n_0)})^{\mathrm{T}}$$

上式即为

$$(\Phi_2) = (\Phi_0)$$

再定义

$$(\Phi_1) = (\phi_1, \phi_2, \cdots, \phi_{n_0})^{\mathrm{T}}$$

再把 $(K), (p)$ 都写成相应的分块形式，则 (8.4.8) 式可写成

$$\begin{pmatrix} (K_{11}) & (K_{12}) \\ (K_{21}) & (K_{22}) \end{pmatrix} \begin{pmatrix} (\Phi_1) \\ (\Phi_2) \end{pmatrix} = \begin{pmatrix} (p_1) \\ (p_2) \end{pmatrix}$$

它的第一个方程

$$(K_{11})(\Phi_1) = (p_1) - (K_{12})(\Phi_2)$$

和边界条件 $(\Phi_2) = (\Phi_0)$, 就是强加边界条件以后的有限元方程。

$$\begin{aligned} &(K_{11})(\Phi_1) = (p_1) - (K_{12})(\Phi_2) \\ &(\Phi_2) = (\Phi_0) \end{aligned} \tag{8.4.9}$$

其中 (K_{11}) 是 $n_0 \times n_0$ 阶的对称方阵，(p_1) 是 n_0 维列向量

$$(p_1) = (P_1, P_2, \cdots, P_{n_0})$$

写出方程中的各个项，得到

$$\begin{pmatrix} K_{11} & K_{12} & \cdots & K_{1n_0} \\ K_{21} & K_{22} & \cdots & K_{2n_0} \\ \vdots & \vdots & & \vdots \\ K_{n_0 1} & K_{n_0 2} & \cdots & K_{n_0 n_0} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{n_0} \end{pmatrix}$$

$$= \begin{pmatrix} P_1 - K_{1(n_0+1)}\phi_{01} - K_{1(n_0+2)}\phi_{02} - \cdots - K_{1n}\phi_{0(n-n_0)} \\ P_2 - K_{2(n_0+1)}\phi_{01} - K_{2(n_0+2)}\phi_{02} - \cdots - K_{2n}\phi_{0(n-n_0)} \\ \vdots \\ P_{n_0} - K_{n_0(n_0+1)}\phi_{01} - K_{n_0(n_0+2)}\phi_{02} - \cdots - K_{nn}\phi_{0(n-n_0)} \end{pmatrix}$$

它可以简单地记为

$$(K_1)(\Phi_1) = (p_1)$$

所得的有限元方程可以用高斯 - 塞德尔迭代法和超松弛法求解。对于不是十分复杂的情况，可以用 MATLAB 的解线性代数方程组的指令左除来求解.

8.4.4 有限元方法的例子

下面用有限元方法计算十字形截面的方形同轴电缆线内的电势，电缆的外导体的电势为零，芯线的电势为 10V. 芯线的截面如图 8.11 所示.

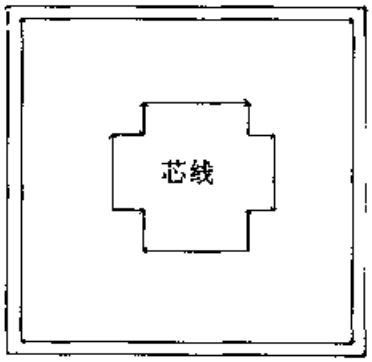


图 8.11 电缆芯线的截面图

这可以看成二维问题，由于问题的对称性，只要解第一象限 (四分之一区域) 就行，求解的定解问题是

$$\begin{cases} \Delta u = 0 \\ \dfrac{\partial u}{\partial n} = 0, & \text{在对称面处} \\ u = 0, & \text{在外导体} \\ u = 10, & \text{在芯线} \end{cases}$$

区域内网格的划分如图 8.12 所示。

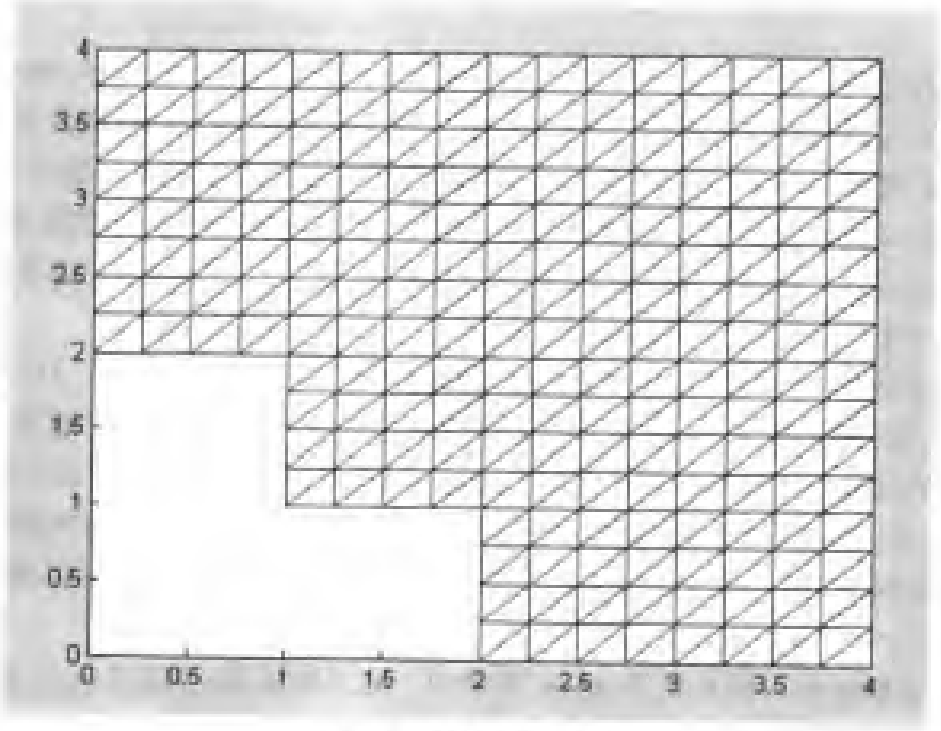


图 8.12 区域的网格划分

求解时，节点的编号如下所示：

```
241 240 239 238 237 236 235 234 233 232 231 230 229 228 227 226 225
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 224
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 223
144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 222
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 221
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 220
 96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 219
 80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95 218
```

208	207	206	205	204	69	70	71	72	73	74	75	76	77	78	79	217
0	0	0	0	203	58	59	60	61	62	63	64	65	66	67	68	216
0	0	0	0	202	47	48	49	50	51	52	53	54	55	56	57	215
0	0	0	0	201	36	37	38	39	40	41	42	43	44	45	46	214
0	0	0	0	200	199	198	197	196	29	30	31	32	33	34	35	213
0	0	0	0	0	0	0	0	195	22	23	24	25	26	27	28	212
0	0	0	0	0	0	0	0	194	15	16	17	18	19	20	21	211
0	0	0	0	0	0	0	0	193	8	9	10	11	12	13	14	210
0	0	0	0	0	0	0	0	192	1	2	3	4	5	6	7	209

所用的程序包含了四个子程序，在主程序 xdx.m 中调用子程序 setelm，给节点和三角形元素编号，并设定节点的坐标．调用子程序 hwg，用指令 gplot 画网格．调用子程序 mmat 编写有限元公式左边的系数矩阵 M，调用子程序 Gcolm 编写有限元公式右边的矢量 G．然后在主程序中使用指令左除来解有限元方程，得到各节点场的值，并将结果画成等势线，此外还输出了节点的编号和各节点的场的数据．所有公用的变量都设成全局变量．

全局变量有：描述网线纵向数目与横向数目的变量 I1,I2, Imax, J1, J2, Jmax; 描述节点坐标的变量 X,Y, 描述节点编号的变量 NN, 描述各基点局域节点的矩阵 NE, 活动节点数 na, 内外界面节点区分 ns2, 总节点数 ndm, 基元数 nel 等．

程序 mmat 在给节点编号时，顺序是由下而上，先左后右，先内部的活动节点，然后是外部节点．在编号的同时，也计算节点的坐标．最后还要将节点按三角形元素标记．

在程序 hwg 中，使用指令 gplot 画网线，使用格式是 gplot(A,xy), 其中 xy 是节点的坐标，A 是用子描述节点连结信息的稀疏矩阵，其中对角线上的元素 A(i,i) 表示编号为 i 的节点与其他节点的连结的数目，A(i,j) 表示节点 i 与节点 j 连结状况，−1 表示相连，0 表示不相连．下面的小程序是一个例子．它是由 4 个节点画的网线，如图 8.13 所示．

```
xy=[0,0;  0,1;  1,1;  1,0];
a=[3,-1,-1,-1;
    -1, 2,-1,0;
    -1, -1, 3,-1;
    -1,0, -1, 2];
A=sparse(a);
gplot(A,xy)
```

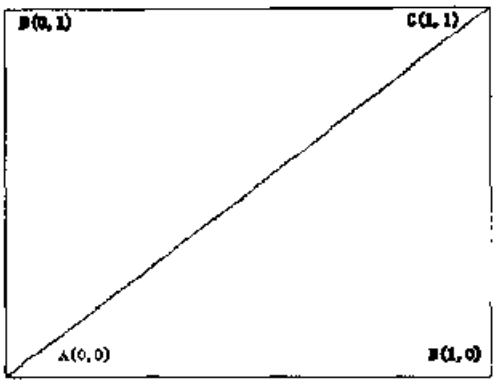


图 8.13 指令 gplot 画的图形

在程序 hwg 中，活动节点与边界节点所连结的节点数目是不一样的，必须分开计算。在边界上，不同位置的节点与其他节点连结的方式也不一样，尤其在拐角点，所连结的节点的数目与方向都不相同，必须一一分开计算。程序 MMAT 和程序 Gcolm 是直接使用有限元的方程编写的，对照公式很容易理解。

实际的程序如下所示：

```
function xdx   %%输电线
global I1 I2 J1 J2 Imax Jmax X Y NN NE na ns2 ndm nel
Imax=16; I1=4; I2=8; X0=4/Imax;
Jmax=16; J1=4; J2=8; Y0=4/Jmax;
V1=10; V2=0; J=0;
  %%设定网格与基元，活动节点数 na，节点数 ndm，
  %%内外界面节点区分点 ns2，基元数 nel
setelm(X0,Y0);
hwg   %%画划分三角形元素的网线
M=mmat;   %%系数矩阵 M
G=Gcolm(V1,V2,J);   %%右方矢量 G
F=M\G;   %%求解

fi=zeros(ndm,1);
fi(1:na)=F(1:na);
fi(na+1:ns2)=V1;
fi(ns2+1:ndm)=V2;
```

```
for j=0:Jmax
    for i=0:Imax,     n=NN(i+1,j+1);
        if n<=0, n=na+1; end
        NNV(i+1,j+1)=fi(n);
    end
end
[XX,YY]=meshgrid(0:Imax,0:Jmax);
figure(2)
contour(XX,YY,NNV);   %%画等电势线
```

%%用文本输出数据，矩阵数据以单变量形式输出，输出文件在当前工作目录下

```
fid=fopen('out.txt','w');
fprintf(fid,'\n 节点编号 \n \n');
Nna=fliplr(NN);
fprintf(fid,'%4d%4d%4d%4d%4d%4d%4d%4d%4d%4d...
%4d%4d%4d%4d%4d%4d%4d%4d\n',Nna)
fprintf(fid,'\n 各节点的电势 \n \n ')
NNV=fliplr(NNV);
fprintf(fid,'%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f...
%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f\ n',NNV)
fprintf(fid,'\ n 节点 phi \ n \ n ');
L=[1:ndm]';
fprintf(fid,'%6d%12.5f\n', [L  fi]');
fclose(fid)
```

```
function setelm(X0,Y0)
global I1 I2 J1 J2 Imax Jmax X Y NN NE na ns2 ndm nel
nlm=Imax*Jmax; X=nlm:1; Y=nlm:1;
NN=zeros(Imax+1,Jmax+1);
n1=0;   %%活动节点编号
for j=0:Jmax-1
    Is=0;     if j<=J2, Is=I1+1; end
              if j<=J1, Is=I2+1; end
    for i=Is:Imax-1,
```

```
        n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0;
        NN(i+1,j+1)=n1;
    end
end

na=n1;
i=I2;
for j=0:J1-1,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
j=J1;
for i=I2:-1:I1+1,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
i=I1;
for j=J1:J2-1,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
j=J2;
for i=I1:-1:0,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
ns2=n1;
i=Imax;
for j=0:Jmax-1,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
j=Jmax;
for i=Imax:-1:0,
    n1=n1+1; X(n1)=i*X0; Y(n1)=j*Y0; NN(i+1,j+1)=n1;
end
ndm=n1; NE=zeros(3,2*ndm); n1=0;
for j=0:Jmax-1
    for i=0:Imax-1
        n=NN(i+1,j+1);
```

```
        if n>0,
            n1=n1+1; NE(1,n1)=n;
            NE(2,n1)=NN(i+2,j+1); NE(3,n1)=NN(i+2,j+2);
            n1=n1+1; NE(1,n1)=n;
            NE(2,n1)=NN(i+2,j+2); NE(3,n1)=NN(i+1,j+2);
        end
    end
end
nel=n1;

function M=mmat
global X Y NE T na ns2 ndm nel
T=zeros(ndm,ndm);
for n=1:nel,
    n1=NE(1,n); n2=NE(2,n); n3=NE(3,n);
    s=abs((X(n2)-X(n1))*(Y(n3)-Y(n1))-...
    (X(n3)-X(n1))*(Y(n2)-Y(n1)))/2;
    for k=1:3
        if n1<=na|n2<=na
            T(n1,n2)=T(n1,n2)+(Y(n2)-Y(n3))*(Y(n3)-Y(n1))+...
            (X(n3)-X(n2))*(X(n1)-X(n3))/(4*s);
            T(n2,n1)=T(n1,n2);
            T(n1,n1)=T(n1,n1)+(Y(n2)-Y(n3)) ^2+(X(n3)-X(n2))^2/(4*s);
        end
        k=n1; n1=n2; n2=n3; n3=k;
  %%轮换足标用 k 作为过渡变量
    end
end
M=T(1:na,1:na);

function G=Gcolm(V1, V2, J)   %%确定 G 矢量
global X Y NE T na ns2 ndm nel
G=zeros(na,1);
for n=1:nel
```

```
    n1=NE(1,n); n2=NE(2,n); n3=NE(3,n);
    s=abs((X(n2)-X(n1))*(Y(n3)-Y(n1))-...
    (X(n3)-X(n1))*(Y(n2)-Y(n1)))/2;
    for k=1:3
        if n1<= na, G(n1)=G(n1)+J*s/3; end
        n4=n1; n1=n2; n2=n3; n3=n4;
      %%轮换足标, 用 n4 作为过渡变量
    end
end

for n=1:na
    for ns=na+1:ndm, V=V1;
        if ns>ns2, V=V2; end
        G(n)=G(n)-T(n,ns)*V;
    end
end


function hwg
global I1 I2 J1 J2 Imax Jmax X Y NN NE

for j=1:Jmax+1
    for i=1:Imax+1
        L=0;
        if  NN(i,j)>0,    k5= NN(i,j)
            if  i-1>0&j-1>0&NN(i-1,j-1)>0,
                L=L+1; k1= NN(i-1,j-1); AA(k5,k1)=-1 ;
            end
            if  j-1>0&NN(i,j-1)>0,
                L=L+1; k2= NN(i,j-1);   AA(k5,k2)=-1 ;
            end
            if  i-1>0&NN(i-1,j)>0,
                L=L+1; k4= NN(i-1,j);   AA(k5,k4)=-1 ;
            end
            if  i+1<=Imax+1&NN(i+1,j)>0,
```

```
                L=L+1; k6= NN(i+1,j);   AA(k5,k6)=-1 ;
            end
            if  j+1<=Jmax+1&NN(i,j+1)>0,
                L=L+1; k8= NN(i,j+1) ;  AA(k5,k8)=-1 ;
            end
            if  i+1<=Imax+1&j+1<=Jmax+1&NN(i+1,j+1)>0,
                L=L+1; k9= NN(i+1,j+1); AA(k5,k9)=-1;
            end
            AA(k5,k5)=L ;
        end
    end
end

AA=sparse(AA);
    figure(1)
    xy=[X/Y/];
    size(xy)
    gplot(AA,xy)
```

输出的各节点的电势如下：

```
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 1.25  1.24  1.23  1.20  1.13  1.02  0.90  0.80  0.70  0.59  0.47  0.36  0.27  0.19  0.12  0.06  0.00
 2.49  2.49  2.47  2.41  2.27  2.03  1.81  1.61  1.40  1.17  0.94  0.72  0.53  0.37  0.23  0.11  0.00
 3.74  3.73  3.70  3.62  3.43  3.05  2.70  2.41  2.10  1.76  1.40  1.07  0.79  0.55  0.35  0.17  0.00
 4.99  4.98  4.95  4.86  4.61  4.06  3.60  3.21  2.81  2.34  1.86  1.42  1.05  0.73  0.46  0.22  0.00
 6.24  6.23  6.20  6.11  5.82  5.06  4.48  4.02  3.53  2.93  2.32  1.77  1.30  0.90  0.57  0.28  0.00
 7.49  7.49  7.46  7.38  7.10  6.03  5.34  4.83  4.27  3.52  2.77  2.10  1.54  1.07  0.67  0.32  0.00
 8.75  8.74  8.73  8.68  8.47  6.96  6.19  5.65  5.02  4.11  3.20  2.42  1.76  1.22  0.77  0.37  0.00
10.00 10.00 10.00 10.00 10.00  7.80  7.00  6.47  5.81  4.70  3.63  2.72  1.98  1.37  0.86  0.41  0.00
10.00 10.00 10.00 10.00 10.00  8.47  7.78  7.32  6.66  5.29  4.03  3.00  2.17  1.50  0.94  0.45  0.00
10.00 10.00 10.00 10.00 10.00  9.03  8.54  8.16  7.60  5.86  4.40  3.25  2.35  1.62  1.02  0.49  0.00
10.00 10.00 10.00 10.00 10.00  9.53  9.27  9.06  8.68  6.40  4.79  3.48  2.50  1.72  1.08  0.52  0.00
10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00  6.86  5.02  3.67  2.63  1.81  1.13  0.55  0.00
10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00  7.17  5.23  3.82  2.74  1.88  1.18  0.57  0.00
10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00  7.35  5.39  3.93  2.81  1.93  1.21  0.58  0.00
10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00  7.46  5.48  3.99  2.86  1.96  1.23  0.59  0.00
10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00 10.00  7.49  5.51  4.02  2.87  1.97  1.23  0.59  0.00
```

所得的结果见图 8.14 所示.

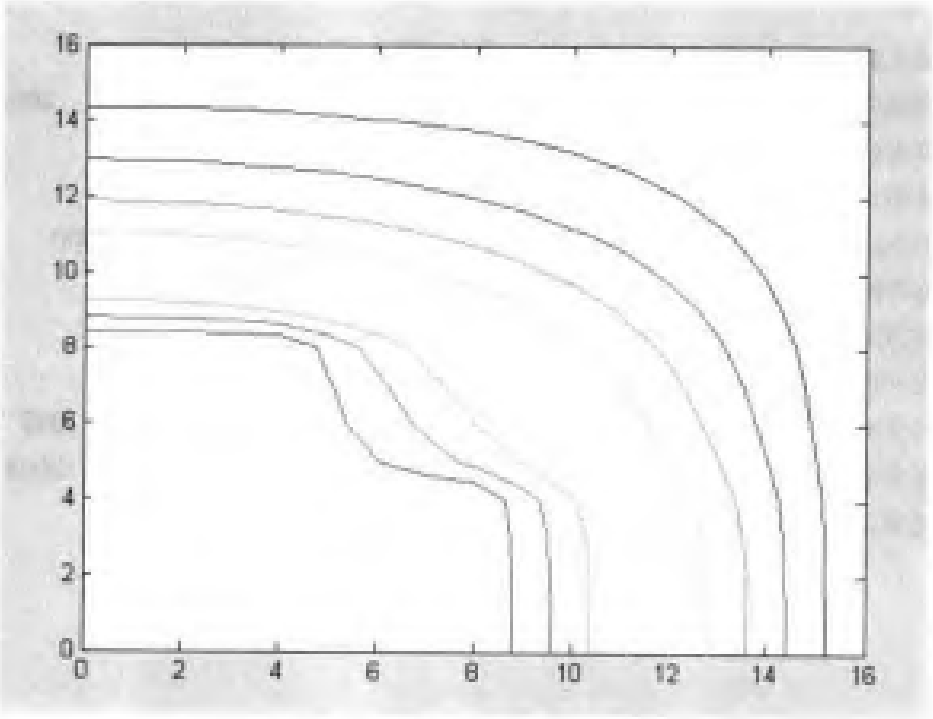


图 8.14 输电线的等电势线

参考文献

[1] 梁昆淼. 数学物理方法, 第三版. 北京: 高等教育出版社, 1998

[2] 胡嗣柱, 倪光炯. 数学物理方法. 第二版. 北京: 高等教育出版社, 2002

[3] 吴崇试. 数学物理方法, 第二版. 北京: 北京大学出版社, 2003

[4] 姚端正. 数学物理方法学习指导. 北京: 科学出版社, 2003

[5] 王永成. 数学物理方程, 第二版. 北京: 北京师范大学出版社, 2000

[6] 彭芳麟等. 理论力学计算机模拟. 北京: 清华大学出版社, 2002

[7] 陈世民. 电动力学简明教程. 北京: 高等教育出版社, 2004

[8] 张开明, 顾昌鑫. 计算物理学. 上海: 复旦大学出版社, 1987

[9] 马文淦, 张子平. 计算物理学. 合肥: 中国科学技术大学出版社, 1992

[10] 张志涌. 精通 MATLAB6.5 版. 北京: 北京航天航空大学出版社, 2003

[11] 吴崇试. 数学物理方法. 北京: 北京大学出版社, 2003