第 1 章　误差分析

本章主要介绍数值计算方法的基本过程、误差的基本概念及分类，并阐述算法设计的基本原则。

1.1　数值计算方法基本过程

数值计算方法是一门内涵丰富、思想和方法深刻、实用性非常强的数学类课程，主要研究使用计算机求解各种数学问题的数值方法，并对求得的解的精度进行评估，其研究对象主要是从科学与工程问题中简化抽象出来的数学问题。解决实际问题的步骤为：首先建立数学模型，然后选择适当的数值方法、编写相应的程序，最后利用计算机计算数值结果。

数值计算方法是一种研究并求解数学问题数值近似解的方法，也是在计算机上能够实现的求解数学问题的方法。在科学研究和工程技术中都要用到各种数值计算方法。例如，在石油开采、航天航空、地质勘探、汽车制造、桥梁设计和天气预报中都有数值计算方法的踪影。随着经济和社会的飞速发展，产生了一系列计算类的分支学科，各种数值计算方法已然成为解决实际问题的桥梁。

数值计算方法既有数学类课程的抽象性和严谨性，又有实用性和实验性的特点，因此数值计算方法是一门理论性和实践性都很强的学科。20 世纪 70 年代，大多数高等院校一般只在数学系开设数值计算方法这门课程，随着计算机技术的迅速发展，数值计算方法课程已经成为大部分理工科大学生的必修课程。

为何数值计算方法的应用如此之广？主要是由于在现实生活和科学计算中存在大量的问题不能从理论上进行求解。例如，

（1）不能理论求解的积分：

$$\int \sin x^2 \mathrm{d}x, \quad \int \frac{1}{\ln x}\mathrm{d}x, \quad \int \sqrt{1+x^3}\mathrm{d}x, \quad \int \mathrm{e}^{-x^2}\mathrm{d}x$$

（2）不能求解析解的方程：

$$(x-2)^{102}=0, \quad x^2+\sin x=1, \quad x^x+2^x=2$$

（3）很难求解的微分方程：

$$y''=x+y^2, \quad y'''+\sin(y')=1, \quad xy+y'+y''=x$$

数值计算方法的研究对象主要是微积分、线性代数、常微分方程中的数学问题。内容包

括：插值和拟合、数值微分和数值积分、求解线性方程组的直接法和迭代法、常微分方程数值解和最佳逼近等问题。

数值计算方法的关键过程如下：

实际应用问题→获取数据→建立数学模型→选择最优的数值计算方法→编写程序→计算数值结果→模型检验。

现实生活和科学计算过程中，由于研究的问题一般是比较复杂的、参数众多的、计算相对困难的，因此，在处理过程中如何选择一个相对最优的数值计算方法，就显得尤为关键。

1.2 误差分类

1.2.1 观测误差

研究实际问题过程中首先要对各个参数进行测量，测量过程中不可避免地会产生误差，这种误差称为观测误差。观测误差一般是不能避免的，只能尽量减小。例如，测量一段 1000 米的距离，观测过程中如果有 10 米的误差就比较大了，但是如果误差为 1 米，就是允许的，如果误差达到 0.1 米就更好了。

1.2.2 模型误差

研究实际问题时往往只考虑主要参数，利用主要参数建立数学模型，这种数学模型与现实问题之间会产生一定的误差，这种误差称为模型误差。模型误差也不能避免，但是可以不断修正数学模型，以获得最优的近似模型。

1.2.3 截断误差

在将无限问题有限化和连续问题离散化过程中，截取无限问题或连续问题的主体部分或有限项作为近似值时所产生的误差，这种误差称为截断误差。例如，

（1）求圆面积的三角形近似法，如图 1-1 所示。

（2）函数的近似计算：

$$\frac{1}{1-x} \approx 1 + x + x^2 + x^3$$

$$\sin x \approx x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!}$$

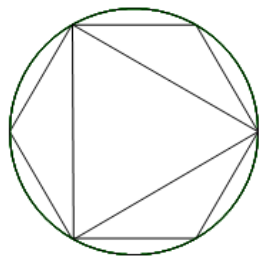

图 1.1 圆面积的三角形近似法

1.2.4 舍入误差

在日常生活和十进制运算中，四舍五入是我们最习惯和接受的近似方法，但是由于数值计算过程中每一步都要进行四舍五入，就会带来一定的数值误差，这种误差称为舍入误差。例如，

$$\pi \approx 3.141\ 592\ 7\ ,\quad e \approx 2.718\ 281\ 8\ ,\quad \frac{22}{7} \approx 3.142\ 857\ 1$$

1.3 误差基本概念

1.3.1 绝对误差与相对误差

1. 绝对误差

定义 1.3.1 准确值 x 与其近似值 x^* 的差称为近似值 x^* 的绝对误差，用 $e(x^*)$ 表示，即

$$e(x^*) = x - x^* \tag{1-3-1}$$

如果 $|e(x^*)| \leqslant \varepsilon(x^*)$，则称 $\varepsilon(x^*)$ 为近似值 x^* 的绝对误差限。

2. 相对误差

绝对误差能够精确知道准确值和近似值之间的误差，但是很多时候它的实际意义不大。例如，假设测量一段 1 000 km 的距离，绝对误差是 10 m，这个误差在实际应用中是可以接受的，但是如果测量 100 m 的距离，绝对误差还是 10 m，这个近似值显然不可以接受。因此，我们又引入一个更有意义的误差——相对误差。

定义 1.3.2 绝对误差与准确值 x 的比值称为近似值 x^* 的相对误差，用 $e^r(x^*)$ 表示，即

$$e^r(x^*) = \frac{x - x^*}{x} \tag{1-3-2}$$

如果 $|e^r(x^*)| \leqslant \varepsilon^r(x^*)$，则称 $\varepsilon^r(x^*)$ 为近似值 x^* 的相对误差限。由于准确值 x 一般是未知的，为了方便，经常将相对误差表示为

$$e^r(x^*) = \frac{x - x^*}{x^*} \tag{1-3-3}$$

例 1.3.1 在工程测量中获得两个测量数据 $x^* = 120 \pm 0.1$，$y^* = 12\ 000 \pm 0.5$，求 x^* 和 y^* 的相对误差。

解 利用相对误差的定义，有 $e^r(x^*) = \dfrac{x - x^*}{x}$。于是

$$|e^r(x^*)| = \left|\frac{x - x^*}{x}\right| = \frac{0.1}{120} = \frac{1}{1\ 200} \approx 0.000\ 833$$

$$|e^r(y^*)| = \left|\frac{y - y^*}{y}\right| = \frac{0.5}{12\ 000} = \frac{1}{24\ 000} \approx 0.000\ 041\ 67$$

因此 x^* 和 y^* 的相对误差分别为 0.000 833 和 0.000 041 67。

从这个例子可以看出来，相对误差的结果往往比绝对误差更有说服力。

1.3.2 有效数字与可靠数字

定义 1.3.3 设准确值 x 的近似值可以用科学计数法表示为

$$x^* = \pm 0.x_1x_2x_3\cdots x_n\cdots\times 10^m \tag{1-3-4}$$

这里 m 为整数，所有的 x_i 都是 0 ~ 9 之间的自然数且 $x_1 \neq 0$，如果近似值 x^* 的绝对误差 $|e(x^*)|=|x-x^*| \leqslant \frac{1}{2}\times 10^{m-n}$，则称 x_n 为有效数字，近似值 x^* 为 n 位有效数字。

定义 1.3.4 如果准确值 x 的近似值 x^* 可以表示成式（1-3-4），并且

$$|e(x^*)|=|x-x^*| \leqslant 10^{m-n} \tag{1-3-5}$$

则称 x_n 为可靠数字。

根据上述两个定义，可以建立如下的两个重要关系：

（1）有效数字与相对误差限的关系。

定理 1.3.1 设 x 的近似值 x^* 可以表示成式（1-3-4），

（i）如果近似值 x^* 具有 n 位有效数字，那么相对误差 $e^r(x^*)$ 满足

$$|e^r(x^*)| \leqslant \frac{1}{2x_1}\times 10^{-(n-1)} \tag{1-3-6}$$

（ii）如果相对误差 $e^r(x^*)$ 满足

$$|e^r(x^*)| \leqslant \frac{1}{2(x_1+1)}\times 10^{-(n-1)} \tag{1-3-7}$$

则近似值 x^* 具有 n 位有效数字。

证明 根据式（1-3-4），可得

$$x_1\times 10^{m-1} \leqslant |x^*| \leqslant (x_1+1)\times 10^{m-1}$$

又因为近似值 x^* 具有 n 位有效数字，有

$$|x-x^*| \leqslant \frac{1}{2}\times 10^{m-n}$$

于是得到

$$|e^r(x^*)| = \frac{|x-x^*|}{|x^*|} \leqslant \frac{\frac{1}{2}\times 10^{m-n}}{x_1\times 10^{m-1}} = \frac{1}{2x_1}\times 10^{-(n-1)}$$

反之，如果相对误差 $e^r(x^*)$ 满足 $|e^r(x^*)| \leqslant \frac{1}{2(x_1+1)}\times 10^{-(n-1)}$，则

$$|x-x^*| = |e^r(x^*)|\cdot|x^*| \leqslant \frac{1}{2(x_1+1)}\times 10^{-(n-1)}\times(x_1+1)\times 10^{m-1} = \frac{1}{2}\times 10^{m-n}$$

这就说明近似值 x^* 具有 n 位有效数字。

（2）可靠数字与相对误差限的关系。

类似于定理 1.3.1 的证明，我们可以证明如下结果：

定理 1.3.2 设 x 的近似值 x^* 可以表示成式（1-3-4），

（i）如果 x_n 为可靠数字，那么相对误差 $e^r(x^*)$ 满足

$$|e^r(x^*)| \leqslant \frac{1}{x_1} \times 10^{-(n-1)} \tag{1-3-8}$$

（ii）如果相对误差 $e^r(x^*)$ 满足

$$|e^r(x^*)| \leqslant \frac{1}{x_1+1} \times 10^{-(n-1)} \tag{1-3-9}$$

则 x_n 为可靠数字。

1.4 算法设计基本原则

在数值计算过程中，大多数误差都是客观存在，且不可避免的，如何有效控制误差，使误差相对较小，这是我们在设计数值算法之初就应该考虑的。为了有效控制计算误差，大致有这样几个基本原则：

1. 防止除数太小

在数值计算过程中，如果除数太小，可能导致整个比值的绝对误差很大，这是要尽量避免的。例如，

$$\frac{2.457\ 2}{0.001} = 2\ 457.2$$

如果对分母稍微改变一点，增加 0.0001 后，

$$\frac{2.457\ 2}{0.001\ 1} = 2\ 233.8$$

由此可以看到除数的微小变化对绝对误差产生的巨大影响，所以在实际计算过程中要尽量避免除数太小。

2. 防止大数吃小数

由于计算机都会有位数的限制，如果两个数字的大小区别太大，可能导致有效数字的丢失，最终产生较大的误差。

例 1.4.1 假设计算机具有 16 位浮点数，即计算机能够保留 16 位有效数字，若 $a=10^{17}$，$b=6$，要求计算 $a+\sum_{i=1}^{10^8} b$。

解 如果直接计算，计算机是按照求和的先后顺利来处理，首先计算$a+b$，

$$a=1.000\ 000\ 000\ 000\ 000\ 00\times10^{17}$$

因为计算机的位数限制，

$$b=6=0.000\ 000\ 000\ 000\ 000\ 06\times10^{17}\approx0.000\ 000\ 000\ 000\ 000\ 0\times10^{17}$$

因此

$$a+b=1.000\ 000\ 000\ 000\ 000\ 00\times10^{17}$$

继续加入b，但是b的最末位依然被舍去，所以最终

$$a+\sum_{i=1}^{10^8}b=1.000\ 000\ 000\ 000\ 000\ 00\times10^{17}$$

但是这明显是不对的。

正确的方法是：将$\sum_{i=1}^{10^8}b$作为整体先计算，然后再计算最终的结果。显然

$$\sum_{i=1}^{10^8}b=0.000\ 000\ 000\ 000\ 000\ 06\times10^{17}\times10^{8}=0.000\ 000\ 006\times10^{17}$$

因此

$$a+b=1.000\ 000\ 006\ 000\ 000\ 00\times10^{17}$$

3. 避免相近两数作差

如果计算过程中$x_1\approx x_2$，计算x_1-x_2将会使有效数字位数产生巨大的损失，这是在计算中应该尽量避免的。

例 1.4.2 已知$\sqrt{63}\approx7.94$，求方程$x^2-16x+1=0$两个正根中较小的根，要求至少保留 3 位有效数字。

解 利用一元二次方程的求根公式得到两根分为$x_1=8-\sqrt{63}$，$x_2=8+\sqrt{63}$，显然要求的是$x_1=8-\sqrt{63}$。因为$\sqrt{63}\approx7.94$，直接代入则$x_1=8-\sqrt{63}=0.06$，只有 1 位有效数字。

如果换另一种方式

$$x_1=8-\sqrt{63}=\frac{1}{8+\sqrt{63}}\approx0.062\ 7$$

就有 3 位有效数字，满足题目要求。

例 1.4.3 若$\pi_1=3.141\ 593$，$\pi_2=3.141\ 592$，分别求π_1，π_2，$\pi_1-\pi_2$的有效数字的位数。

解 利用有效数字的定义，π_1的有效数字是 7 位，π_2的有效数字是 6 位，然而$\pi_1-\pi_2=0.000\ 001$只有 1 位有效数字，由此可看出两个相近的数字作差会极大损失有效数字的位数。

例 1.4.4 求方程$x^2-116x+1=0$的较小正根，要求至少有 3 位有效数字。

解 利用求根公式$x_1=58+\sqrt{3\ 363}$，$x_2=58-\sqrt{3\ 363}$，所以较小的正根为$x_2=58-\sqrt{3\ 363}$，

直接计算达不到精度要求，取 $x_2=\dfrac{1}{x_1}\approx 8.621\,3\times 10^{-3}$，具有 4 位有效数字。

为了应用方便，我们还介绍如下几种其他常采取的转换方法：

$$\sqrt{x}-\sqrt{x-1}=\frac{1}{\sqrt{x}+\sqrt{x-1}},\quad \lg x-\lg y=\lg\frac{x}{y},\quad \frac{1-\cos x}{\sin x}=\frac{\sin x}{1+\cos x}$$

4. 简化运算过程

在计算过程中，计算效率十分关键，如何简化计算过程，提高计算速度，在数值计算中就显得非常重要。

例 1.4.5 合理简化多项式

$$P(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$$

的运算次数。

解 如果考虑先计算每一项 a_ix^i 再求和，计算每一项需要 i 次乘法，则总共就需要计算 n 次加法和 $1+2+\cdots+n=\dfrac{n(n+1)}{2}$ 次的乘法，显然这样计算比较繁琐，乘法的计算步骤也非常多，需要非常大的计算量。

可以考虑先修改一下计算过程：

$$P(x)=(((a_nx+a_{n-1})x+a_{n-2})x+\cdots+a_1)x+a_0$$

通过这样一个简单的处理，乘法的次数就减少为 n 次，加法也是 n 次，明显比第一种思路的计算量减少了很多。

1.5 习题 1

1. 设 x 的相对误差是 ε，$x>0$，求 $\ln x$ 的绝对误差？
2. 在算法设计过程中，应简化运算步骤，减少_____。
3. 在算法设计过程中，要防止有效数字丢失，应该尽量避免相近两数_____。
4. 设 $x^*=3.246\,73$ 是由四舍五入得到的近似值，则 x^* 有_____有效数字。
5. 设 $\pi=3.141\,592\,6\cdots$，$\pi_1=3.141\,5$，$\pi_2=3.141$，求 π_1，π_2 分别有几位有效数字？
6. 二次方程 $x^2-16x+1=0$，$x_1=8-\sqrt{63}$，$\sqrt{63}\approx 7.937$，求 x_1 的近似值使其具有 4 位有效数字。
7. 为了使计算 $y=4+\dfrac{5}{x}+\dfrac{1}{x^2}-\dfrac{3}{x^3}$ 的乘除法运算次数尽量地少，应将该表达式改写为_____。
8. 求方程 $x^2-322x+1=0$ 的较小正根，要求至少有 3 位有效数字。
9. 已知近似值 $x_1=1.420$，$x_2=0.0142$，$x_3=1.420\times 10^{-4}$ 的绝对误差限均为 0.5×10^{-3}，那么它们各有几位有效数字？

10. 已知 $\sqrt{87}\approx 9.327\ 4$，求方程 $x^2-56x+1=0$ 的两个根，要求这两个根至少具有 4 位有效数字。

1.6 Matlab 程序设计（一）

1.6.1 基础实验

Matlab 软件是由美国 MathWorks 公司开发的一款商业数学软件，主要用于算法开发、数据可视化、数据分析和数值计算的高级计算机语言和交互式环境，主要包括 Matlab 和 Simulink 两大部分。

Matlab 是由 Matrix 和 Laboratory 两个词组合简化而成，意思是矩阵实验室，主要提供科学计算、可视化及交互式程序设计的计算环境。它将数值计算、矩阵计算、科学数据可视化及非线性动态系统的建模和仿真等诸多强大功能集成在一个易于使用的视窗环境中，为科学研究、工程设计以及必须进行有效数值计算的众多科学领域提供了一种全面的解决方案，并在很大程度上摆脱了传统非交互式程序设计语言的编辑模式，代表了当今国际科学计算软件的先进水平。

Matlab 软件和 Mathematica 软件、Maple 软件并称为三大数学软件。它在数学类科技应用软件中的数值计算方面首屈一指。Matlab 软件可以进行矩阵运算、绘制函数图像、实现算法、创建用户界面、连接其他编程语言的程序等，主要应用于控制设计、工程计算、信号处理与通讯、图像处理、金融建模设计等领域。

Matlab 软件的基本数据单位是矩阵，它的指令表达式与数学、工程中常用的形式十分相似，故用 Matlab 软件来求解相同问题要比用 C，Fortran 等语言简捷得多，并且 Matlab 软件也吸收了 Maple 软件的优点，使 Matlab 软件成为一个强大的数学软件。

Matlab 软件是当前国际上理工科专业最常用数学类计算机软件。学习 Matlab 软件可更深入理解和掌握数学问题数值计算方法的基本思想，提高数值求解应用问题的能力，为今后其他专业课程的学习提供重要支撑。这里我们主要介绍 Matlab 软件的一些常用运算和画图功能，如果读者有兴趣，可以参考一些其他的 Matlab 软件教材来继续学习。

1. 向量、矩阵和数组

1）向量输入

向量的基本形式为 $x=x_0:step:x_n$，其中 x_0 是初值，x_n 是终值，$step$ 是步长，默认步长是 1，例如，

```
>> a=1:10
a = 1   2   3   4   5   6   7   8   9   10
>> a=1:3:15
a = 1   4   7   10   13
```

2）矩阵输入

矩阵的基本形式 $x=[;]$，同行元素用空格或逗号“,”分隔，行与行之间用分号“;”分隔，例如，

```
>> a=[2  6 7 8;3  2 9 6;1  2 6 0]
a = 2     6     7     8
    3     2     9     6
    1     2     6     0
>> b=[12,16;24,28;38,91;35,78]
b = 12    16
    24    28
    38    91
    35    78
```

3）**数组输入**

数组的基本形式和矩阵的基本形式是完全一样的，其建立和存储完全等同于矩阵。只能在同型矩阵之间进行的一些运算称为数组运算。

```
>> a11=[1  23 34;35 67 79;21 45 27]
a11 =   1    23    34
       35    67    79
       21    45    27
>> b11=[23 45 57;12 23 36]
b11 = 23    45    57
      12    23    36
```

2. 向量、矩阵和数组运算

1）**向量运算**

向量运算包括加（减）、数加（减）、数乘、点积和叉积。

```
>> a1=0:2:10;
>> b1=1:2:11;
>> a=a1+b1                          %向量的加运算
a = 1   5   9   13   17   21
>> b=a1-b1                          %向量的减运算
b = -1   -1   -1   -1   -1   -1
>> c=a1*2                           %向量的数乘运算
c = 0   4   8   12   16   20
>> d=dot(a1,b1)                     %向量的点积运算,表示对应分量乘积的和
d = 250
>> e=cross([1 2 3],[3 4 5])         %向量的叉积运算
e = -2   4   -2
```

2）矩阵运算

矩阵运算主要包括加减、乘除（左除、右除）和乘方。

```
>> A=[1  5 7 8;5 8 7 6;3 9 12 0];
>> B=[0 8 3 7;9 2 1 5;7 8 6 3];
>> C=A+B                          %矩阵的加运算
C =   1    13    10    15
      14   10     8    11
      10   17    18     3
>> D=A-B                          %矩阵的减运算
D =   1    -3     4     1
      -4    6     6     1
      -4    1     6    -3
>> A=[1  5 7 8;5 8 7 6;3 9 12 0];
>> B=[1 3 7;2 1 5;7 6 3;2 3 4];
>> E=A*B                          %矩阵与矩阵相乘
E =    76    74    85
       82    83   120
      105    90   102
>> A=[1  5 7 8;5 8 7 6;3 9 12 0];
>> B=[0 8 3 7;9 2 1 5;7 8 6 3];
>> F=A\B                          %矩阵的左除 A^-1*B
F =   2.0929   -1.3552   -0.5574   -0.1803
         0         0         0         0
      0.0601    1.0055    0.6393    0.2951
     -0.3142    0.2896   -0.1148    0.6393
>> A=[1  5 7 8;5 8 7 6;3 9 12 0];
>> B=[0 8 3 7;9 2 1 5;7 8 6 3];
>> G=A/B                          %矩阵的右除 A*B^-1
G =   0.8049    0.0731    0.1442
      0.4352    0.0469    0.6929
     -0.2046   -1.0422    1.8373
>> A=[1  2 3;1 0 1;2  1 0];
>> H=A^2                          %矩阵的乘方运算
H = 9     5    10
    3     3     3
    3     4     7
```

3）数组运算

数组运算主要包括加减、数乘、乘法和乘方。

```
>> a1=[2 3 6 9;0 3 5 4;2 1 0 4];
>> b1=[1 2 7 6;9 2 1 5;7 8 6 3];
>> c1=a1+b1                          %数组的加运算
c1 =    3     5    13    15
        9     5     6     9
        9     9     6     7
>> d1=a1-b1                          %数组的减运算
d1 = -1     1    -4     3
     -9     1     4    -1
     -5    -7    -6    -1
>> e1=a1.*2                          %数组的数乘
e1 = 4     6    12    18
     0     6    10     8
     4     2     0     8
>> a1=[1 5 7;8 7 6;3 9 1];
>> b1=[2 1 5;7 6 3;2 3 4];
>> f1=a1.*b1                         %数组与数组相乘
f1 =    2     5    35
       56    42    18
        6    27     4
>> g1=a1.^2                          %数组的乘方运算
g1 =   1    25    49
      64    49    36
       9    81     1
```

3. 常用数学函数

Matlab 软件中有许多内置数学函数，例如，指数函数：exp()；开平方函数：sqrt()；正弦函数：sin()；余弦函数：cos()；正切函数：tan()；绝对值函数：abs()；对数函数：log()；反正弦函数：asin()；反余弦函数：acos()；求和函数：sum()。

```
>> a=2*sin(pi/2)
a = 2.0000
>> b=12*asin(tan(exp(1)))
b = -5.6086
>> c=sqrt(cos(pi/16))
c = 0.9903
```

4. 绘制二维图像

绘制二维图像的常用命令是函数 plot()。

例 1.6.1 绘制区间 $0 \leqslant x \leqslant 2\pi$ 内的函数 $y = 2e^{-0.5x} \cdot \sin 2\pi x$ 的图像。

程序：

```
>> x=0:pi/100:2*pi;
>> y=2*exp(-0.5*x).*sin(2*pi*x);
>> plot(x,y)
```

运行结果如图 1.2 所示。

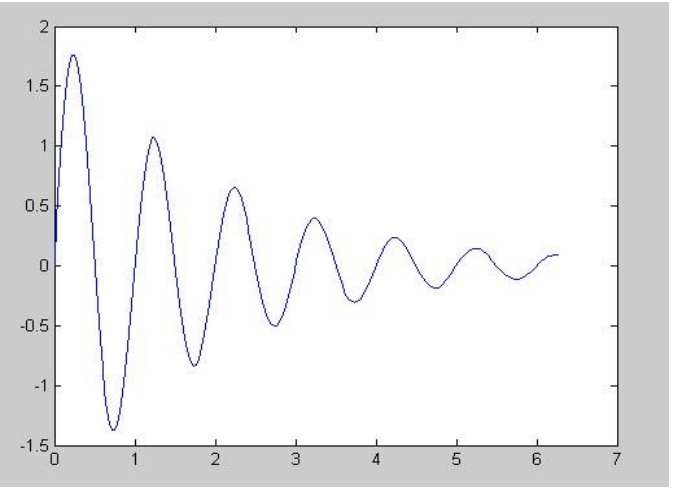

图 1.2

例 1.6.2 用不同标度在同一坐标内绘制曲线 $y_1 = e^{-0.5x}\sin(2\pi x)$ 及曲线 $y_2 = 1.5e^{-0.1x}\sin x$ 。

程序：

```
>> x1=0:pi/100:2*pi;
>> x2=0:pi/100:3*pi;
>> y1=exp(-0.5*x1).*sin(2*pi*x1);
>> y2=1.5*exp(-0.1*x2).*sin(x2);
>> plotyy(x1,y1,x2,y2);
```

运行结果如图 1.3 所示。

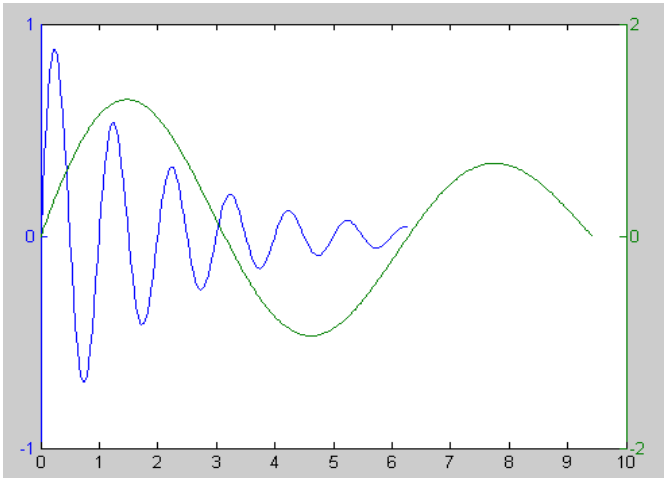

图 1.3

例 1.6.3 画出隐函数 $f(x,y)=x^2\sin(x+y^2)+y^2e^{x+y}=0$ 的函数图像。

程序：

```
>> ezplot('x.^2.*sin(x+y.^2)+y.^2.*exp(x+y)')
```

运行结果如图 1.4 所示。

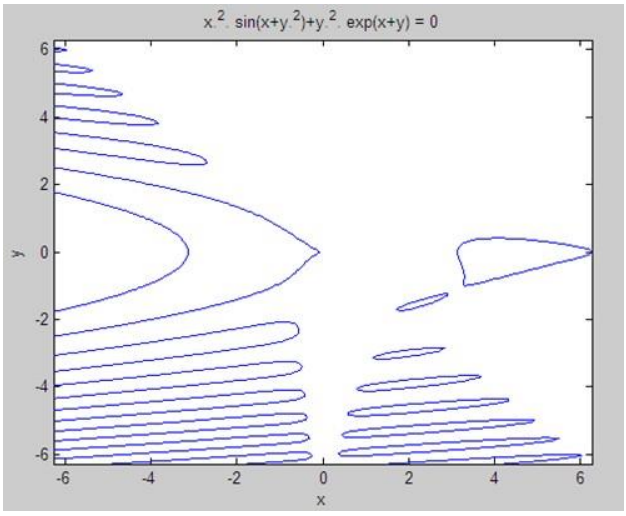

图 1.4

例 1.6.4 画出隐函数 $f(x,y)=4x^2\sin(x+y^2)+9y^2e^{x+y}=0$ 的函数图像。

程序：

```
>> ezplot('4*x.^2*sin(x+y.^2)+9*y.^2*exp(x+y)',[-20  20])
```

运行结果如图 1.5 所示。

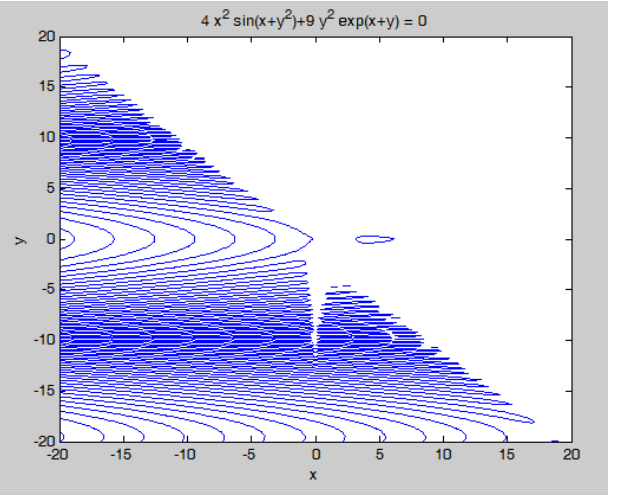

图 1.5

例 1.6.5 在一个图形窗口中以子图形式同时绘制正弦、余弦、正切、余切曲线。

程序：

```
>> x=linspace(0,2*pi,60);
>> y=sin(x);
>> z=cos(x);
>> t=sin(x)./(cos(x)+eps);
>> ct=cos(x)./(sin(x)+eps);
>> subplot(2,2,1);
>> plot(x,y);
>> title('sin(x)');axis([0,2*pi,-1,1]);
>> subplot(2,2,2);
>> plot(x,z);
>> title('cos(x)');axis([0,2*pi,-1,1]);
>> subplot(2,2,3);
>> plot(x,t);
>> title('tan(x)');axis([0,2*pi,-40,40]);
>> subplot(2,2,4);
>> plot(x,ct);
>> title('cot(x)');axis([0,2*pi,-40,40]);
```

运行结果如图 1.6 所示。

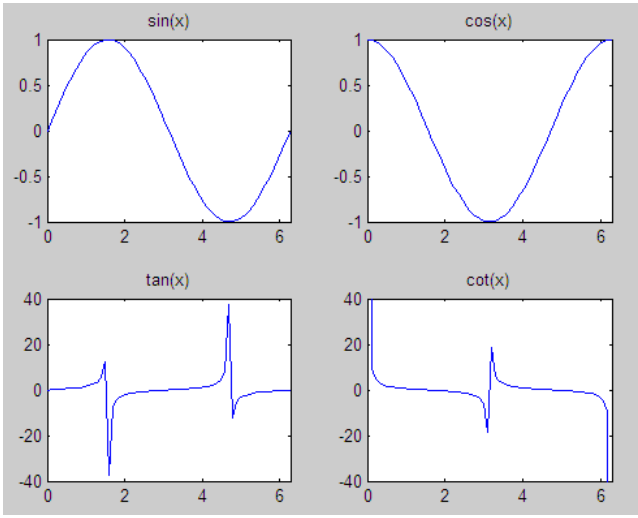

图 1.6

例 1.6.6 在同一直角坐标系内绘制函数 sin(3*x*) 和它的导函数在 $0 \leqslant x \leqslant \pi$ 范围内的函数图像。

程序：

```
>> x=linspace(0,3*pi);
>> y1=sin(3*x);
>> y2=3*cos(3*x);
>> plot(x,y1,'r-',x,y2,'b-.');
>> xlabel('x');
>> ylabel('y');
>> legend('f(x)','d/dx  f(x)');
```

运行结果如图 1.7 所示。

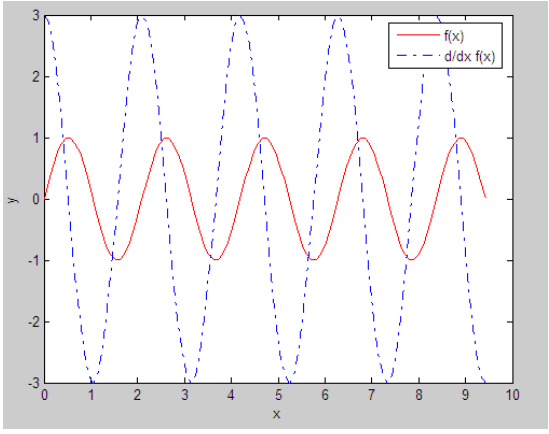

图 1.7