第 1 章

绪　　论

1.1　概　　述

随着计算机科学和技术的快速发展，有限元法作为工程师分析的有效工具，可以较容易地对复杂问题进行建模分析，帮助工程师们对于一些初始设计方案，在实物原型出现前开展仿真评估和优化改进工作。它涉及的领域包括汽车、船舶、飞机、建筑等，处理的物理问题包括结构的静力问题，以及振动与声学、热流、液体流、电通量、磁通量等问题。它在科学研究中应用广泛，发挥了重要作用。

要完成这些仿真计算，需要我们充分了解有限元法的基本理论、建模技巧及实用计算方法等。有限元法求解的基本过程为：将模型的复杂几何区域离散为具有简单几何形状的单元，也就是有限单元；将单元的材料属性和控制方程通过单元节点处的未知量进行描述，通过单元集成、外载荷和约束条件的处理，得到模型整体的方程组，求解方程组就可以得到该模型中场变量的近似结果。

本章将简要介绍有限元法的发展历史，以及本书的主要内容和特点，之后给出数学软件 Mathematica 及有限元软件 Abaqus 的相关介绍。

1.2　有限元法的历史背景

有限元法的基本思想最初产生于航空结构中的分析需求。1941 年，Hrennikoff 采用框架变形功方法求解了弹性问题；Courant 于 1943 年发表论文，尝试应用在三角形区域上定义的分片连续函数和最小位能原理相结合的方法求解扭转问题；Argyris 于 1954 年和 1955 年发表的关于能量原理和矩阵方法的论文，为有限元法的进一步发展奠定了基础。Turner 等人于 1956 年将刚架分析中的位移法推广到平面问题中，首次给出了用三角形单元求解平面问题的正确解答，而三角形单元的特征矩阵和结构的求解方程是由弹性理论的方程通过直接刚度法（也称矩阵法）得到的。1960 年，Clough 在进一步求解平面弹性问题时，第一次提出了“有限单元法”的名称。在 20 世纪 60 年代初，工程师们采用有限单元法近似求解应力分析、流体流动、传热等领域的问题。Zienkiewicz 等人的第一本关于有限元的书籍于 1967 年出版。20 世纪 60 年代末到 70 年代初，有限元分析应用于非线性和大变形问题中，Oden 的非线性著作 *Continua* 于 1972 年出版，此时加权余量法（主要是其中的伽辽金法）也被用于建立有限元方程，与直接法和变分法相比，其适用性更好，从而拓展了有限元法的应

用领域。自 20 世纪 70 年代奠定了有限元法的数学基础后，其应用领域不断扩展，出现了多种新型单元，尤其是随着计算机的出现，有限元软件蓬勃发展，已成为解决工程问题的实用性工具。

1.3 本书的主要内容与特点

本书的重点是培养读者对线性有限元分析方法的清晰理解，考虑到有限元法所包括的矩阵法中涉及较多的数学推导和矩阵计算等，所以在介绍有限元法基本原理的同时，较早引入数学软件 Mathematica 辅助推导数学公式及实现有限元法编程，从而更好地实现方法原理的论述与例题的求解，另外还引入了商业有限元软件 Abaqus，便于读者体会和了解利用商业软件进行工程问题分析的建模过程，同时也对本书的部分例题进行了仿真校验。

1.4 Mathematica 8.0 介绍

Mathematica 是由 Wolfram 公司开发研制的一款用途广泛的科学计算软件，1998 年发布 1.0 版本，后经不断扩充和修改，于 2010 年 11 月推出了 Mathematica 8.0 版，其徽标如图 1.1 所示。本书以 Mathematica 8.0 为模板进行介绍。

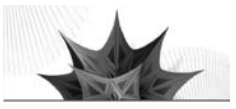

图 1.1 Mathematica 8.0 版本的徽标

Mathematica 是目前世界上应用最广泛的符号计算系统之一，具有符号运算、数值运算、数学图形绘制等多种功能。其主要特点有

(1) Mathematica 是人机对话式软件，使用者在软件的 Notebook 环境下输入命令后，系统可以立即进行处理，然后返回结果。用户不必关心中间的计算过程，其交互性能非常好。

(2) Mathematica 在进行数值计算时，会尽可能保持计算的精确度。它对整数运算的结果不受字段长的限制，而对实数运算结果则可以按使用者的需要给出足够多位的有效数字。

(3) Mathematica 的符号运算功能是非常突出的，符号计算过程是常量、变量、函数和计算公式到常量、变量、函数和计算公式的转换，即通常意义上的数学推导过程。当使用者输入一个表达式后，系统会在大量的对应法则中寻找最好的等价结果。

(4) Mathematica 不仅可以进行基本的运算，还可以进行图形处理，用 Mathematica 可以绘制出精美的二维和三维图形。Mathematica 也能对声音进行处理，它可以让一个函数产生一种声音，并且绘出表示该声音的波形。

(5) Mathematica 本身还是一门高级计算机语言，像其他计算机语言一样，可以编写 Mathematica 程序完成特定的任务。由于 Mathematica 的函数非常丰富，使得其程序编写更加容易。

Mathematica 的优点还在于它将各种功能有机地融合在一起，使用者可以非常轻松地在 Notebook 环境下完成数值计算、符号计算、图形绘制和程序设计等工作，如图 1.2 所示。

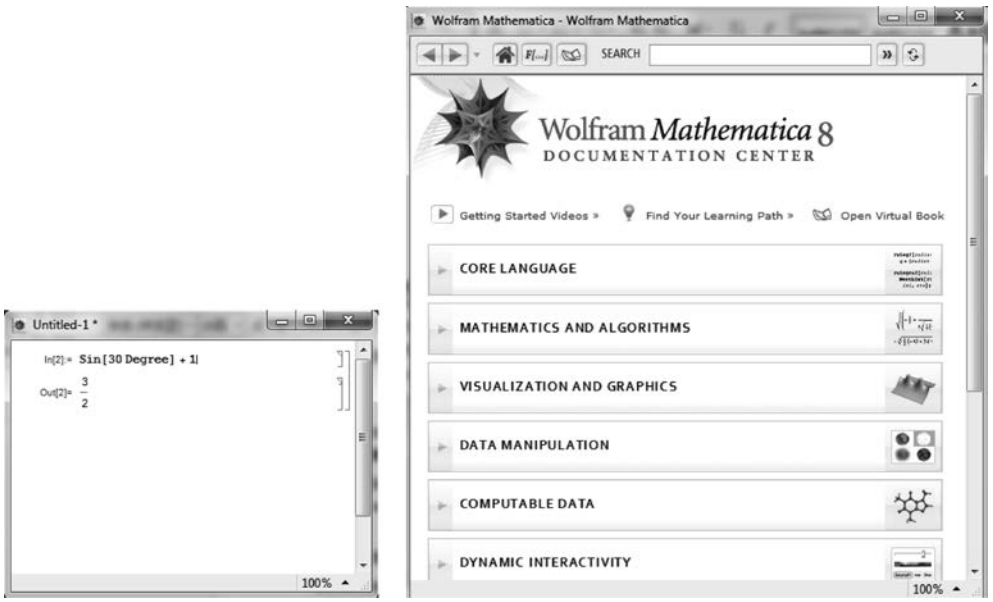

图 1.2　Mathematica 中的 Notebook 及帮助文档

1.4.1　界面

1. Mathematica 8.0 的启动、运行和退出

假设在 Windows 环境下已经安装好了 Mathematica 8.0，启动 Windows 后，在“开始”菜单的“程序”中单击 Mathematica 8.0，则在屏幕上显示 Mathematica 的 Notebook 窗口，系统暂时取名 Untitled-1，直到用户保存时重新命名为止。输入“1＋1”，然后按下 Shift＋Enter 键（或者小键盘上的 Enter 键），这时系统开始计算并输出计算结果，并给输入和输出附上次序标识 In[1]和 Out[1]，注意 In[1]是计算后才出现的；再输入第二个表达式，如 Plot[Sin[x],{x,0,2Pi}]，按 Shift＋Enter 键，绘出 $\sin x$ 在[0,2π]上的图形，则系统分别将其标识为 In[2]和 Out[2]。

注：Mathematica 的 Notebook 是一个集成环境，Enter 键的作用是输入换行，而 Shift＋Enter 键则是向计算机发出计算指令。

Mathematica 第一次计算时因为要启动内核（kernel），所需时间长一些，也可以在 Mathematica 启动后第一次计算之前，手动启动内核，方法是用鼠标单击菜单：Evaluation→Kernel→Start Kernel→Local，这样第一次计算就很快了。在 Mathematica 工作过程中，可以随时退出内核，方法是选择菜单 Evaluation→Quit Kernel→Local。

Mathematica 中的单元（Cell）的概念很重要，Mathematica 的一个输入或输出即为一个 Cell。Mathematica 退出时按 Alt＋F4 键或用鼠标单击窗口右上角的关闭按钮。

2. 帮助菜单的使用

任何时候都可以用鼠标单击 Help 菜单中的 Documentation Center 命令，打开帮助中心，查找需要的帮助或者选择 Virtual Book，里面有完整的 Mathematica 使用指南，单击菜单命令，可获得相关的帮助信息。Documentation Center 主要用来查找一些函数的使用方法，Virtual Book 则主要用来介绍 Mathematica 的主要功能，主要选项有 Introduction，Core Language，Mathematicas and Algorithms，Visualization and Graphics 等。读者可自行打开帮助菜单浏览 Mathematica 的各项功能，以便对 Mathematica 有一个整体的认识。

1.4.2 基本运算

1. 常量和变量

在运算过程中保持不变的量称为常量，常量也称为常数。Mathematica 提供许多常用的数学常数，如：Pi 表示圆周率，Degree 表示度数，Infinity 表示无穷大，E 表示自然对数的底，I 表示虚数单位。

为了便于计算、保存或引用在运算中的一些中间结果，常常需要引入变量，在 Mathematica 中变量名以英文字母开头，后跟字母或数字，变量名字符的长度不限。例如：hijk，A，x3 都是合法的变量名；而 2t 和 u v(u 和 v 之间有空格)不能作为变量名。英文字母大小写的意义不同，因此 A 和 a 表示两个不同的变量。在 Mathematica 中常量、函数都用大写字母开头的标识符来表示，为了避免混淆，变量名通常都以小写字母开头。对矩阵等变量，用户可以用大写字母，这样更符合数学表示的习惯。

在 Mathematica 中，变量即取即用，不需要先说明变量的类型再使用，而且变量不仅可以存放一个实数或复数，还可以存放一个多项式或复杂的图形等。

数值有类型，变量也有类型。通常在运算中不需要对变量进行类型说明，系统根据变量初值会做出正确的处理。在定义函数和程序设计中也允许对变量进行类型说明。

2. 算术运算

Mathematica 最基本的功能是进行算术运算，包括加、减、乘、除、乘方、阶乘等，运算顺序遵循数学习惯，先阶乘，后乘方，再乘除，最后加减，同级运算遵循从左到右的顺序。Mathematica 作为符号计算软件，和 C，Fortran 等语言对于数的处理有明显的不同，最大区别在于 Mathematica 引入了任意精度实数，并且可以精确表示出非常大的整数。Mathematica 的实数分为两类：一类是任意精度实数，如不带小数点的分数和无理数等；另一类是机器精度实数，主要指带小数点且小数位数小于或等于 16 的小数，如果小数位数大于 16，则 Mathematica 将其解释为任意精度实数。Mathematica 的机器精度实数相当于 C，Fortran 等高级语言中的双精度实数，在运算时会产生由于数据的舍入导致的浮点数误差，但运算速度较快。Mathematica 的任意精度实数运算类似于通常意义上的数学运算，计算准确，但要花费比较长的时间，机器精度数通过命令 SetAccuracy 可以转化为任意精度的实数。巧妙地运用 Mathematica 的任意精度实数和机器精度实数运算，有助于更好地理解数

值算法。

3. Mathematica 中的函数

Mathematica 的所有功能均通过函数实现，Mathematica 的函数包含了通常意义上的数学函数和所有命令，例如三角函数、打印命令和绘图功能等。Mathematica 函数采用见名知义的方式命名，要求首字母和复合函数名的各个单次首字母大写，且函数中的变量放在方括号中，例如 Sin[x]和 Cos[x]分别表示正弦函数和余弦函数，ArcSin[x]和 ArcCos[x]分别表示反正弦函数和反余弦函数等。同样，我们不难猜出下面这些函数的数学意义，如 Abs[x]，Sqrt[x]，Exp[x]，Log[x]，Log[b，x]，Tan[x]，Cot[x]，ArcTan[x]，Sinh[x]，Cosh[x]，Tanh[x]，Coth[x]，ArcTanh[x]等。

1.4.3　符号计算功能

1. 求解代数方程和方程组

因为 Mathematica 把方程看作逻辑语句，对于数学方程式 $x^2-2x+1=0$，在 Mathematica 中应表示为“x^2－2x＋1＝＝0”。方程的解显示为未知量替换形式。例如，用 Solve 命令求方程 $x^2-3x+2=0$ 的根，显示为

```
Sol=Solve[x^2－3x＋2＝＝0,x]
{{x→1},{x→2}}                                    (＊输出结果为未知量替换形式＊)
```

对于以上方程的解，若需在其他表达式中应用，则使用 Mathematica 的符号替换功能，即使用符号“/.”，例如，对于上面方程的解 Sol，有：

```
{x^2＋4/.Sol[[1]],x^2＋4/.Sol}                   (＊结果{5,{5,8}}＊)
r=Solve[2－4x＋x^5＝＝0,x]                       (＊没有整数或分数形式的解，结果如下＊)
{{x→Root[2－4＃1＋＃1^5δ,1]},{x→Root[2－4＃1＋＃1^5δ,2]},{x→Root[2－4＃1＋＃1^5δ,3]},
{x→Root[2－4＃1＋＃1^5δ,4]},{x→Root[2－4＃1＋＃1^5δ,5]}}
r//N                                             (＊N 函数后置形式，求小数形式的解＊)
{{x→－1.51851},{x→0.508499},{x→1.2436},{x→－0.116792－1.43845i},{x→－0.116792＋
1.43845i}}
```

或直接使用 NSolve 命令求方程的小数解，例如：

```
NSolve[2－4x＋x^5＝＝0,x]                        (＊结果同上＊)
```

使用 Solve 命令也可以求方程组的解，例如求解

$$\begin{cases}2x+y=0\\ x+3y=3\end{cases}\quad 和\quad \begin{cases}x^2+y^2=1\\ x+3y=0\end{cases}$$

Mathematica 命令分别为

```
Solve[{2x+y==0,x+3y==3},{x,y}]        (*结果{{x→-(3/5),y→6/5}}*)
Solve[{x^2+y^2==1,x+3y==0},{x,y}]
{{x→-3/√10,y→1/√10},{x→3/√10,y→-1/√10}}
```

2. 微积分运算

Mathematica 的微积分运算包括求函数导数和求不定积分、定积分，下面对这些功能分别予以说明。

在 Mathematica 中，计算函数的导数命令为 D[f,x]，表示求函数 f 关于 x 的导数或偏导数，该命令的常用形式见表 1.1。

表 1.1 常用的求导函数形式

求导函数	功能
D[f,x]	计算 $f'(x)$ 或 $\frac{\partial f}{\partial x}$
D[f,x_1,x_2,…,x_n]	计算多重偏导数 $\frac{\partial}{\partial x_1}\frac{\partial}{\partial x_2}\cdots\frac{\partial}{\partial x_n}$
D[f,{x,n}]	计算高阶导数 $f^{(n)}(x)$

例 1.1 计算 $\sin x, x^x$ 的一阶导数和 $\sin x\tan x$ 的二阶导数；计算 $\frac{\partial}{\partial x}\frac{\partial}{\partial y}z\sin(x^2y^3)$。

```
{D[Sin[x],x],D[x^x,x],D[Sin[x]Tan[x],{x,2}]
{Cos[x],x^x(1+Log[x]),2Sec[x]-Sin[x]Tan[x]+2sec[x]tan[x]^2}
D[zSin[x^2*y^3],x,y]
6xy^2zCos[x^2y^3]-6x^3y^5zSin[x^2y^3]
```

在 Mathematica 中计算不定积分的命令为 Integrate[f,x]，当然也可以使用工具栏直接输入不定积分式计算函数的不定积分。对于一些手工计算相当复杂的不定积分，Mathematica 能轻易求得。注意，Mathematica 的输出结果中省略了积分常数。

例 1.2 计算积分 $\int 3ax^2\,dx$，$\iint(3x^2+y)\,dx\,dy$，$\int\sqrt{\tan x}\,dx$，$\int\frac{u\sqrt{1+u^2}}{2+11u^2}du$。

```
{Integrate[3ax^2,x],Integrate[3x^2+y,x,y]}
{ax^3,x^3y+(xy^2)/2}
ix=Integrate[Sqrt[Tan[x]],x]          (*积分函数复杂,略*)
D[ix,x]                               (*对 ix 求一阶导数,结果更加复杂,略*)
Simplify[%]                           (*对上述结果化简,得积分原函数√Tan[x] *)
Integrate[u Sqrt[1+u^2]/(2+11u^2),u]  (*请运算*)
```

Integrate 命令还可以计算函数的定积分，一般形式为 Integrate[f,{x,a,b}]，表示求解 $\int_a^b f(x)\,dx$。

若 $f(x)$的原函数无法求出，则使用 NIntegrate[f,{x,a,b}]计算 $\int_a^b f(x)\mathrm{d}x$ 的数值积分。

例 1.3　计算积分 $\int_a^b(\cos^2 x+\sin^3 x)\mathrm{d}x$，$\int_0^{+\infty}\exp(-x^2)\mathrm{d}x$，$\int_0^{\pi}\sin\sin x\,\mathrm{d}x$。

```
Integrate[Cos[x]^2+Sin[x]^3,{x,0,1}]
  1/12(14-9Cos[1]+Cos[3]+3Sin[2])
Integrate[Exp[-x^2],{x,0,Infinity}]          (*结果为√π/2*)
Integrate[Sin[Sin[x]],{x,0,Pi}]
  πStruveH[0,1]                              (*积分结果为特殊函数*)
NIntegrate[Sin[Sin[x]],{x,0,Pi}]             (*计算数值积分*)
  1.78649
```

1.4.4　矩阵计算

1. 向量和矩阵的生成

在 Mathematica 中，向量和矩阵是一类特殊形式的表，向量是一维表，矩阵是每行元素个数相同的二维表，所以向量和矩阵可以使用以下方法生成。

方法一，直接按表的方式输入向量或矩阵，例如：

```
b={0,7,9,20}
   {0,7,9,20}
M={{2,5,-1},{0,-6,4},{4,7,1}}
   {{2,5,-1},{0,-6,4},{4,7,1}}
MatrixForm[M]                                (*将表显示为矩阵形式,略*)
```

注：在 Mathematica 中，向量不分行向量和列向量，Mathematica 自动根据向量所处环境将其解释为行向量或列向量。另外，矩阵 ***M*** 也可以通过单击“Palettes”菜单中的“Basic Math Assistant”矩阵工具栏选项按照传统方式输入，Ctrl+Enter 键表示增加行，“Ctrl,”键表示增加列。

方法二，使用 Table 和 Array 命令，这是生成矩阵最常用的命令。例如：

```
A=Table[x^(i+j),{i,0,2},{j,0,4}]             (*或*)
A=Array[x^(#1+#2-2)δ,{3,5}]
{{1,x,x^2,x^3,x^4},{x,x^2,x^3,x^4,x^5},{x^2,x^3,x^4,x^5,x^6}}
MatrixForm[A]                                (*将二维表 A 显示为矩阵形式*)
```

结果为

$$\begin{bmatrix}1 & x & x^2 & x^3 & x^4\\ x & x^2 & x^3 & x^4 & x^5\\ x^2 & x^3 & x^4 & x^5 & x^6\end{bmatrix}$$

```
a=1;A=Table[If[i>=j,a++,0],{i,3},{j,3}];     (*If 判断语句*)
a=1;B=Table[If[i>=j,++a,0],{i,3},{j,3}];
{MatrixForm[A],MatrixForm[B]}
```

结果为

$$\boldsymbol{A}=\begin{bmatrix}1&0&0\\2&3&0\\4&5&6\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}2&0&0\\3&4&0\\5&6&7\end{bmatrix}$$

这里注意 a++和++a 的区别，它们的功能均是自加 1，但前者先使用，后加 1；++a 则是先加 1，再使用。

```
A=RandomInteger[{0,9},{4,5}]//MatrixForm
                                    (*生成一个元素在0～9之间的4行5列随机整数矩阵*)
```

注：MatrixForm[A]的功能是将表 A 显示为矩阵形式，此形式不能用于计算，所以，通常情况下，采用方法二中的使用方式，此时矩阵 $\boldsymbol{A}$ 仍为一个二维表。

方法三，应用特殊矩阵命令生成特殊矩阵，常用的命令有

IdentityMatrix[n]，表示 n 阶单位矩阵。

DiagonalMatrix[list]，表示对角线元素为表 list 中元素的对角矩阵。

ConstantArray[a,{m,n}]，表示元素均为 a 的 $m\times n$ 矩阵。

HilbertMatrix[n]，表示 n 阶希尔伯特矩阵。

Normal[SparseArray[{{i1,j1}→v1,{i2,j2}→v2,…},{m,n}]]，表示创建一个 $m\times n$ 稀疏矩阵，第{ik,jk}位置元素为 vk，k=1,2,…，其他位置元素为 0。

以上命令示例如下：

```
DiagonalMatrix[{1,2,3}]                    (*生成对角矩阵*)
  {{1,0,0},{0,2,0},{0,0,3}}
HilbertMatrix[2]//MatrixForm               (*二阶希尔伯特矩阵*)
SparseArray[{{1,3}→a,{3,2}→b},{3,4}]
SparseArray[<2>,{3,4}]
Normal[SparseArray[{{1,3}→a,{3,2}→b},{3,4}]]
{{0,0,a,0},{0,0,0,0},{0,b,0,0}}
```

SparseArray 是一个非常有用的稀疏矩阵生成命令，其功能非常强大，请读者自行查找相应的帮助信息。

2. 向量和矩阵的运算

在 Mathematica 中有许多对向量和矩阵进行操作的命令，这些命令使得复杂的向量和矩阵运算变得十分简单，主要命令见表 1.2。

表 1.2 常用的向量和矩阵命令

向量和矩阵操作函数	说　明
A+c	$\boldsymbol{A}$ 为矩阵，c 为标量，c 与 $\boldsymbol{A}$ 中每个元素相加
A+B	$\boldsymbol{A}$、$\boldsymbol{B}$ 为同类型矩阵或向量，$\boldsymbol{A}$ 与 $\boldsymbol{B}$ 的对应元素相加
c*A	$\boldsymbol{A}$ 为矩阵，c 为标量，c 与 $\boldsymbol{A}$ 中每个元素相乘
A.B	矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 相乘，要求 $\boldsymbol{A}$ 的列数等于 $\boldsymbol{B}$ 的行数
U.V	向量 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的内积

续表

向量和矩阵操作函数	说　明
Cross[a,b]或 a×b	向量 $\boldsymbol{a}$,$\boldsymbol{b}$ 的矢量积
Outer[Times,U,V]	列向量 $\boldsymbol{U}$ 乘以行向量 $\boldsymbol{V}$
Normalize[v]	与 $\boldsymbol{v}$ 同方向的单位向量
Projection[u,v]	向量 $\boldsymbol{u}$ 正交投影到向量 $\boldsymbol{v}$ 上得到的向量
Det[A]	计算矩阵 $\boldsymbol{A}$ 的行列式
Transpose[A]	计算矩阵 $\boldsymbol{A}$ 的转置
Inverse[A]	计算矩阵 $\boldsymbol{A}$ 的逆矩阵
Tr[A]	计算矩阵 $\boldsymbol{A}$ 的迹
MatrixRank[A]	计算矩阵 $\boldsymbol{A}$ 的秩
MatrixPower[A,n]	计算矩阵 $\boldsymbol{A}$ 的 n 次方
Dimensions[A]	给出矩阵 $\boldsymbol{A}$ 的行、列数
Eigenvalues[A]	以表的形式给出矩阵 $\boldsymbol{A}$ 的全部特征值
Eigenvectors[A]	以表的形式给出矩阵 $\boldsymbol{A}$ 的全部特征向量
Eigensystem[A]	计算矩阵 $\boldsymbol{A}$ 所有的特征值和特征向量

1.4.5　程序设计

1. 函数定义

一个函数或一个命令即对应一个变换规则,例如,求和函数 Sum 和绘图命令 Plot 等都可看成一个变换规则。在 Mathematica 中,可以认为定义函数就是定义了一条规则,例如,对数学函数 $f(x)=2x-1$ 则用定义形式"f[x_]:=2x-1",其中 f[x_]称为模式(pattern),出现在 f[x_]中的 x_是一类重要实体,它表示函数定义中的变量,可以看成高级语言函数定义的形式参数。x_可为实数、向量或矩阵。如果用"f[x]=expr"定义函数,那么这个规则仅对具体对象 x 才有意义。例如,f[x]=2x-1,只对符号 x 才有定义值。

在命令行中用"f[x_]=."则表示清除函数 f[x_]的定义,用 Clear[f]表示清除所有以 f 为函数名的所有函数定义。例如:

```
Clear[f];x=6;{f[x]=x-2,f[3]}              (*结果为 4 和 f[3]*)
Clear[f];x=6; f[x_]:= x-2;{f[x],f[3]}     (*结果为 4 和 1*)
```

2. 条件

在进行编程时,常用到条件语句,Mathematica 提供了多种设置方法,包括 If,Which 和 Switch 等,下面分别给出其语句结构:

```
If[逻辑表达式,表达式 1]
```

当逻辑表达式的值是真(True)时,计算表达式 1,表达式 1 的值就是整个 If 结构的值。

```
If[逻辑表达式,表达式 1,表达式 2]
```

当逻辑表达式的值是 True 时,计算表达式 1,并将表达式 1 的值作为整个结构的值;当逻辑表达式的值是 False 时,计算表达式 2,并将表达式 2 的值作为整个结构的值。

```
If[逻辑表达式,表达式 1,表达式 2,表达式 3]
```

当逻辑表达式的值是 True 时,转向计算表达式 1,当逻辑表达式的值是 False 时转向计算表达式 2,当逻辑表达式的值非 True 非 False 时,计算表达式 3,并将所计算表达式的值作为整个结构的值。

Which 语句结构为

```
Which[条件 1,表达式 1,条件 2,表达式 2,…]
Which[条件 1,表达式 1,条件 2,表达式 2,…,True,表达式]
```

依次计算条件 i,计算对应第一个条件为 True 的表达式的值,作为整个结构的值,如果所有条件的值都为 False,则不做任何运算,用 True 作为 Which 的最后一个条件时,用以处理其他情况,相当于 C 语言的 Switch 语句中的 default 的作用。

例如,计算

$$h(x)=\begin{cases}-x, & x<0\\ \sin(x), & 0\leqslant x<6\\ x/2, & 16\leqslant x<20\\ 0, & \text{其他}\end{cases}$$

使用 Which 语句表示为

```
h[x_] := Which[x<0,-x,x>=0&&x<6,Sin[x],x>=16&&x<20,x/2,True,0];
{h[-12],h[3],h[17],h[50]}        (*结果为{12,Sin[3],17/2,0}*)
```

Switch 语句结构为

```
Switch[表达式; 形式 1,结果 1,形式 2,结果 2,…]
```

表达式与每一个形式 i 比较,给出第一个匹配形式所对应的结果。

3. 循环

Mathematica 中共有三种描述循环的语句,它们是 Do,While 和 For,其中 Do 循环根据循环描述先计算循环次数,再做循环体,常用于有确定循环次数的循环结构。在 While 和 For 中,做一次条件确认后,计算一次循环体的表达式。类似于 C 语言中的 While 和 For 语句,在用逻辑表达式构造条件时,要注意避免空循环和死循环。

Do 语句的一般形式为 Do[循环体,循环范围]。

Do 语句有下列形式: