

附录 B LINGO 使用简介

司守奎

烟台市, 海军航空大学

Email: sishoukui@163.com

LINGO 软件是美国 LINDO 系统公司开发的一套专门用于求解最优化问题的软件。它为求解最优化问题提供了一个平台, 主要用于求解线性规划、非线性规划、整数规划、二次规划、线性及非线性方程组等问题。它是最优化问题的一种建模语言, 包含有许多常用的函数供使用者编写程序时调用, 并提供了与其他数据文件的接口, 易于方便地输入, 求解和分析大规模最优化问题, 且执行速度快。由于它的功能较强, 所以在教学、科研、工业、商业、服务等许多领域得到了广泛的应用。


一个 LINGO 程序一般会包括以下几个部分:

(1) 集合段: 集部分是 LINGO 模型的一个可选部分。在 LINGO 模型中使用集之前, 必须在集部分事先定义。集部分以关键字 “sets:” 开始, 以 “endsets” 结束。一个模型可以没有集部分, 或有一个简单的集部分, 或有多个集部分。一个集部分可以放置于模型的任何部分, 但是一个集及其属性在模型目标函数或约束条件中被引用之前必须先定义。


(2) 数据段: 在处理模型的数据时, 需要为集部分定义的某些元素在 LINGO 求解模型之前为其指定值。数据部分以关键字 “data:” 开始, 以关键字 “enddata” 结束。

(3) 目标和约束段: 这部分用来定义目标函数和约束条件。该部分没有开始和结束的标记。主要是要用到 LINGO 的内部函数, 尤其是与集合有关的求和与循环函数等。

(4) 初始段: 这个部分要以关键字 “init:” 开始, 以关键字 “endinit” 结束, 它的作用是对集合的属性定义一个初值。在一般的迭代运算中, 如果可以给一个接近最优解的初始值, 会大大减少程序运行的时间。

(4) 计算段: 这一部分是以关键字 “calc:” 开始, 以关键字 “endcalc” 结束。它的作用是把原始数据  处理成程序模型需要的数据, 它的处理是在数据段输入完以后、开始正式求解模型之前进行的。在这个段中, 程序语句是顺序执行的。注意在计算段中不能有模型的决策变量。

B.1 LINGO 中集合的概念

在对实际问题建模的时候, 总会遇到一群或多群相联  对象, 比如消费者群体、交通工具和雇工等, LINGO 允许把这些相联系的对象聚集成集 (sets)。一旦把对象聚集成集, 就可以利用集来最大限度地发挥 LINGO 建模语言的优势。

1. 为什么使用集

集是 LINGO 建模语言的基础, 是程序设计最强有力的基本构件。借助于集能够用一个单一的、简明的复合公式表示一系列相似的约束, 从而可以快速方便地表达规模较大的模型。

2. 什么是集

集是一群相联系的对象, 这些对象也称为集的成员。一个集可能是一系列产品、卡车或雇员。每个集的成员可能有一个或多个与之有关联的特征, 把这些特征称为属性。属性值可以预先给定, 也可以是未知的, 有待于 LINGO 求解的。

LINGO 有两种类型的集: 原始集 (primitive set) 和派生集 (derived set)。一个原始集是由一些最基本的对象组成的。一个派生集是用一个或多个其他集来定义的, 也就是说, 它的成员来自于其他已存在的集。

3. 原始集的定义

为了定义一个原始集, 必须详细声明:

- 集的名称;
- 集的成员 (可选的);
- 集成员的属性 (可选的)。

定义一个原始集, 用下面的语法:

setname[/member_list]/[:attribute_list];

注 B.1 用 “[]” 表示该部分内容可选。同样的, 下面不再赘述。

setname 是用来标记集的名字, 最好具有较强的可读性。集名称必须严格符合标准命名

规则：以字母为首字符，其后由字母（A-Z）、下划线、阿拉伯数字（0，1，...，9）组成的总长度不超过 32 个字符的字符串，且不区分大小写。

注 B.2 该命名规则同样适用于集成员名和属性名等的命名。

`member_list` 是集成员列表。如果集成员放在集定义中，那么对它们可采取显式罗列和隐式罗列两种方式。如果集成员不放在集合定义中，那么可以在随后的数据部分定义它们。

(1) 当显式罗列成员时，必须为每个成员输入一个不同的名字，中间用空格或逗号隔开，允许混合使用。

例 B.1 定义一个名为 `friends` 的原始集，它具有成员 `John`、`Jill`、`Rose` 和 `Mike`，属性有 `sex` 和 `age`。

```
sets:
friends/John Jill, Rose Mike/: sex, age;
endsets
```

(2) 当隐式罗列成员时，不必罗列出每个集成员。可采用如下语法：

`setname/member1..memberN[: attribute_list];`

这里的 `member1` 是集合的第一个成员名，`memberN` 是集合的最末一个成员名。`LINGO` 将自动产生中间的所有成员名。`LINGO` 也接受一些特定的首成员名和末成员名，用于创建一些特殊的集合，如表 B.1。

表 B.1 隐式罗列成员示例

隐式成员列表格式	示例	所产生集成员
1..n	1..5	1,2,3,4,5
StringM..StringN	Car2..Car14	Car2,Car3,Car4,...,Car14
DayM..DayN	Mon..Fri	Mon,Tue,Wed,Thu,Fri
MonthM..MonthN	Oct..Jan	Oct,Nov,Dec,Jan
MonthYearM..MonthYearN	Oct2001..Jan2002	Oct2001,Nov2001,Dec2001,Jan2002

(3) 集成员不放在集定义中，而在随后的数据部分来定义。

例 B.2 集定义示例。

```
!集部分;
sets:
friends: sex, age;
endsets
!数据部分;
data:
friends, sex, age=John 1 16
Jill 0 14
Rose 0 17
Mike 1 13;
enddata
```

注 B.3 开头用感叹号 (!)，末尾用分号 (;) 表示注释，可跨多行。

在集部分只定义了一个集 `friends`，并未指定成员。在数据部分罗列了集成员 `John`、`Jill`、`Rose` 和 `Mike`，并对属性 `sex` 和 `age` 分别给出了值。

集成员无论用何种字符标记，它的索引都是从 1 开始连续计数。在 `attribute_list` 可以指定一个或多个集成员的属性，属性之间必须用逗号隔开。

4. 定义派生集

为了定义一个派生集，必须详细说明集的名称和父集的名称，而集成员和集成员的属性是可选的。

可用下面的语法定义一个派生集：

`setname(parent_set_list)/[member_list]/[: attribute_list];`

`setname` 是集的名字。`parent_set_list` 是已定义集的列表，多个时必须用逗号隔开。如果没有指定成员列表，那么 `LINGO` 会自动创建父集成员的所有组合作为派生集的成员。派生

集的父集既可以是原始集，也可以是其他的派生集。

例 B.3 派生集定义示例。

```
sets:
product/A B/;
machine/M N/;
week/1..2/;
allowed(product, machine, week): x;
endsets
```

LINGO 生成了三个父集的所有组合共八组作为 allowed 集的成员。如表 B.2 所列。

表 B.2 集合 allowed 的成员

编号	成员
1	(A,M,1)
2	(A,M,2)
3	(A,N,1)
4	(A,N,2)
5	(B,M,1)
6	(B,M,2)
7	(B,N,1)
8	(B,N,2)



成员列表被忽略时，派生集成员由父集成员所有的组合构成，这样的派生集合称为稠密集。如果限制派生集的成员，使它成为父集成员所有组合构成的集合的一个子集，这样的派生集称为稀疏集。同原始集一样，派生集成员的声明也可以放在数据部分。一个派生集的成员列表有两种方式生成。

```
(1) 显式罗列派生集的成员
allowed(product, machine, week)/A M 1, A N 2, B N 1/;
(2) 设置成员资格过滤器
```

如果需要生成一个大的、稀疏的集，那么显式罗列就十分麻烦。但是许多稀疏集的成员都满足一些条件以和非成员相区分。可以把这些逻辑条件看作过滤器，在 LINGO 生成派生集的成员时把使逻辑条件为假的成员从稠密集中过滤掉。

例 B.4 稀疏集示例。

```
sets:
!学生集：性别属性 sex，1 表示男性，0 表示女性；年龄属性 age；
students/John,Jill,Rose,Mike/:sex,age;
!男学生和女学生的联系集：友好程度属性 friend，[0, 1]之间的数；
linkmf(students,students)|sex(&1)#eq#1 #and# sex(&2)#eq#0: friend;
!男学生和女学生的友好程度大于 0.5 的集；
linkmf2(linkmf)|friend(&1,&2)#gt#0.5: x;
endsets
data:
sex,age=1 16, 0 14, 0 17, 0 13;
friend=0.3, 0.5, 0.6;
enddata
```

用竖线(|)来标记一个成员资格过滤器的开始。#eq#是逻辑运算符，用来判断是否“相等”。&1 可看作派生集的第 1 个原始父集的索引，它取遍该原始父集的所有成员；&2 可看作派生集的第 2 个原始父集的索引，它取遍该原始父集的所有成员；&3, &4, …，以此类推。注意如果派生集 B 的父集是另外的派生集 A，那么上面所说的原始父集是集 A 向前回溯到最终的原始集，其顺序保持不变，并且派生集 A 的过滤器对派生集 B 仍然有效。因此，派生集的索引个数是最终原始父集的个数，索引的取值是从原始父集到当前派生集所作限制的总和。

总的来说，LINGO 可识别的集只有两种类型：原始集和派生集。

在一个模型中，原始集是基本的对象，不能再被拆分成更小的组分。原始集可以由显式罗列和隐式罗列两种方式来定义。当用显式罗列方式时，需在集成员列表中逐个输入每个成员。当用隐式罗列方式时，只需在集成员列表中输入首成员和末成员，而中间的成员由 LINGO 产生。

另一方面，派生集是由其它的集来创建。这些集被称为该派生集的父集（原始集或其它的派生集）。一个派生集既可以是稀疏的，也可以是稠密的。稠密集包含了父集成员的所有组合（有时也称为父集的笛卡尔乘积）。稀疏集仅包含了父集的笛卡尔乘积的一个子集，可通过显式罗列和成员资格过滤器这两种方式来定义，显式罗列方法就是逐个罗列稀疏集的成员，成员资格过滤器方法通过使用稀疏集成员必须满足的逻辑条件从稠密集成员中过滤出稀疏集的成员。不同集类型的关系如图 B.1 所示。

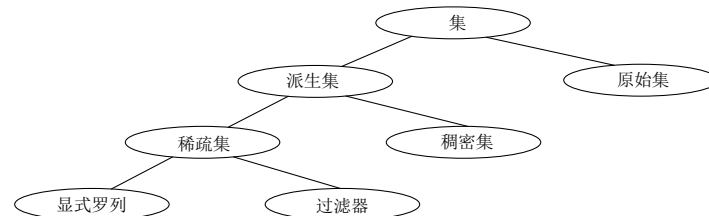


图 B.1 LINGO 集合关系图

B.2 LINGO 数据部分和初始部分

在处理模型的数据时，需要为集指派一些成员并且在 LINGO 求解模型之前为集的某些属性指定值。为此，LINGO 为用户提供了两个可选部分：输入集成员和数据的数据部分（Data Section）和为决策变量设置初始值的初始部分（Init Section）。

B.2.1 模型的数据部分

1. 数据部分入门

数据部分以关键字“data:”开始，以关键字“enddata”结束。在这里，可以指定集成员、集的属性。其语法如下：

object_list=value_list;

对象列（object_list）包含要指定值的属性名、要设置集成员的集名，用逗号或空格隔开。一个对象列中至多有一个集名，而属性名可以有任意多。如果对象列中有多个属性名，那么它们的类型必须一致。

数值列（value_list）包含要分配给对象列中的对象的值，用逗号或空格隔开。注意属性值的个数必须等于集成员的个数。

例 B.5

sets:

set1/A,B,C/: X,Y;

endsets

data:

X=1,2,3;

Y=4,5,6;

enddata

在集 set1 中定义了两个属性 X 和 Y。X 的三个值是 1、2 和 3，Y 的三个值是 4、5 和 6。也可采用如下例子中的复合数据声明（data statement）实现同样的功能。

例 B.6

sets:

set1/A,B,C/: X,Y;

endsets

data:

X,Y=1 4

2 5

3 6;

enddata

例 B.6 中可能会认为 X 被指定了 1、4 和 2 三个值，因为它们是数值列中前三个，而正确的答案是 1、2 和 3。假设对象列有 n 个对象，LINGO 在为对象指定值时，首先在 n 个对象的第 1 个索引处依次分配数值列中的前 n 个对象，然后在 n 个对象的第 2 个索引处依次分配数值列中紧接着的 n 个对象，…，以此类推。

2. 参数输入



在数据部分也可以指定一些标量变量（scalar variables）。当一个标量变量在数据部分确定时，称之为参数。例如，假设模型中用利率 8%作为一个参数，就可以输入一个利率作为参数。

例 B.7

```
data:
    interest_rate=0.08;
enddata
```

实际中也可以同时指定多个参数。

例 B.8

```
data:
    interest_rate,inflation_rate=0.08  0.03;
enddata
```

3. 实时数据处理

在某些情况，对于模型中的某些数据并不是定值。比如模型中有一个通货膨胀率的参数，如果在 2%~6%范围内，对不同的值求解模型，观察模型的结果对通货膨胀的依赖程度。那么把这种情况称为实时数据处理。

在本该放数的地方输入一个问号（?）。

例 B.9

```
data:
    interest_rate,inflation_rate=0.08  ?;
enddata
```

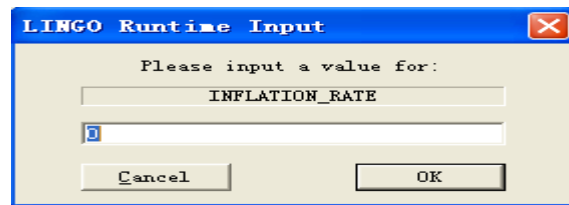


图 B.2 交互式输入对话框

每一次求解模型时，LINGO 都会提示为参数 inflation_rate 输入一个值。在 WINDOWS 操作系统下，将会接收到一个类似图 B.2 的对话框。

直接输入一个值后再单击 OK 按钮，LINGO 就会把输入的值指定给 inflation_rate，然后继续求解模型。

除了参数之外，也可以实时输入集的属性值，但不允许实时输入集成员名。

4. 指定属性为一个值

可以在数据声明的右边输入一个值来把所有的成员的该属性指定为一个值。看下面的例子。

例 B.10

```
sets:
    days /MO,TU,WE,TH,FR,SA,SU/:needs;
endsets
data:
    needs = 20;
enddata
```

例 B.10 中 LINGO 将用 20 指定 days 集的所有成员的 needs 属性。对于多个属性的情形，见下例。

例 B.11

```
sets:
    days /MO,TU,WE,TH,FR,SA,SU/:needs,cost;
endsets
data:
    needs cost = 20 100;
enddata
```

5. 数据部分的未知数值表示

有时只想为一个集的部分成员为某个属性指定值，而让其余成员的该属性保持未知，以便让 LINGO 去求出它们的最优值。在数据声明中输入一个逗号表示该位置对应的集成员的属性值未知。两个逗号间可以有空格。

例 B.12

```
sets:
    years/1..5/: capacity;
endsets
data:
    capacity = , 34, 20 , , , ;
enddata
```

属性 capacity 的第 2 个和第 3 个值分别为 34 和 20，其余的未知。

B.2.2 模型的初始部分

初始部分是 LINGO 提供的另一个可选部分。在初始部分中，与数据部分中的数据声明相同，可以输入初始声明 (initialization statement)。在实际问题建模时，初始部分并不起到描述模型的作用，在初始部分输入的值仅被 LINGO 求解器当作初始点来用，并且仅仅对非线性模型有用。这与数据部分指定变量的值不同，LINGO 求解器可以自由改变初始部分初始化的变量的值。

一个初始部分以 “init:” 开始，以 “endinit” 结束。初始部分的初始声明规则和数据部分的数据声明规则相同。也就是说，我们可以在声明的左边同时初始化多个集属性，可以把集属性初始化为一个值，也可以用问号实现实时数据处理，还可以用逗号指定未知数值。

例 B.13

```
init:
    X, Y = 0, .1;
endinit
Y=@log(X);
X^2+Y^2<=1;
```

好的初始点会减少模型的求解时间。

B.3 LINGO 函数

B.3.1 运算符及其优先级

LINGO 中的运算符可以分为 3 类：算法运算符、逻辑运算符和关系运算符。

1. 算术运算符

算术运算符是针对数值进行操作的。LINGO 提供了 5 种二元运算符：^ (求幂)、* (乘)、/ (除)、+ (加)、- (减)。

LINGO 唯一的一元算术运算符是取反运算 “-”。

运算符的运算次序为从左到右按优先级高低来执行。运算的次序可以用圆括号 “()” 来改变。

2. 逻辑运算符

在 LINGO 中，逻辑运算符主要用于集循环函数的条件表达式中，来控制在函数中哪些集成员被包含，哪些被排斥。在创建稀疏集时用在成员资格过滤器中。

LINGO 具有 9 个逻辑运算符，这些运算符分为两类：#and# (与)，#or# (或)，#not#，是参与逻辑值之间的运算，其结果还是逻辑值；#eq# (等于)，#ne# (不等于)，#gt# (大于)，#ge# (大于等于)，#lt# (小于)，#le# (小于等于) 是数与数之间的比较运算符，其结果为逻辑值。

3. 关系运算符

LINGO 中有 3 个关系运算符：< (等价于 <=, 小于等于)，> (等价于 >=, 大于等于)，= (等于)。

注意 LINGO 中优化模型的约束一般没有严格大于、严格小于。在 LINGO 中，关系运算符主要是被用在模型中，来指定一个表达式的左边是否等于、小于等于、或者大于等于右边，形成模型的一个约束条件。关系运算符与逻辑运算符 #eq#、#le#、#ge# 截然不同。

运算符的优先级如表 B.3 所示。

表 B.3 运算符的优先级

优先级	运算符
高级	#not#, - (取反) ^ *, / +, - #eq#, #ne#, #gt#, #ge#, #lt#, #le# #and#, #or#
最低	<, =, >

B.3.2 LINGO 函数简介

1.基本数学函数

LINGO 中有相当丰富的数学函数，这些函数的用法简单。表?? 对各个函数的用法做了简单介绍。

表 B.4 基本数学函数

函数调用格式	含义
@abs(x)	返回 x 的绝对值。
@sin(x)	返回 x 的正弦值 (x 的单位是弧度)。
@cos(x)	返回 x 的余弦值 (x 的单位是弧度)。
@tan(x)	返回 x 的正切值 (x 的单位是弧度)。
@exp(x)	返回 e^x 的值。
@log(x)	返回 x 的自然对数值。
@lgm(x)	返回 x 的伽玛 (Gamma) 函数的自然对数值。
@mod(x,y)	返回 x 对 y 取模的结果。
@sign(x)	返回 x 的符号值。
@pow(x,y)	返回 x^y 的值。
@sqr(x)	返回 x 的平方。
@sqrt(x)	返回 x 的正平方根值。
@floor(x)	返回 x 的整数部分。当 $x \geq 0$ 时，返回不超过 x 的最大整数；当 $x < 0$ 时，返回不低于 x 的最大整数。
@smax(x1,x2,...,xn)	返回 x1, x2, ..., xn 中的最大值。
@smin(x1,x2,...,xn)	返回 x1, x2, ..., xn 中的最小值。

例 B.14 给定一个直角三角形，求包含该三角形的最小正方形。

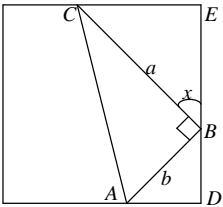


图 B.3 包含三角形的正方形

解 如图 B.3 所示。

$CE = a \sin x, \quad AD = b \cos x, \quad DE = a \cos x + b \sin x,$

求最小的正方形就相当于求如下的最优化问题：

$$\min_{0 \leq x \leq \frac{\pi}{2}} \max\{CE, AD, DE\}.$$

LINGO 代码如下：


```

model:
sets:
    object/1..3/: f;
endsets
data:
    a, b=3, 4; !两个直角边长, 修改很方便;
enddata
    f(1)=a*@sin(x);
    f(2)=b*@cos(x);
    f(3)=a*@cos(x)+b*@sin(x);
    min=@smax(f(1),f(2),f(3));
    @bnd(0,x,1.57); !变量 x 的下界为 0, 上界为 1.57;
end

```

注 B.4 上述 LINGO 程序中的语句

```
min=@smax(f(1),f(2),f(3));
```

可以替换为:

```
min=@max(object(i):f(i));
```

其中的函数@max 将在下面介绍。

2. 集操作函数

集操作函数是对集进行操作的函数, 主要有 4 种, 下面分别介绍它们的一般用法。

(1) @in(set_name, primitive_index_1 [, primitive_index_2, ...])

这个函数用于判断一个集中是否含有某个索引值, 它的返回值是 1 (逻辑值“真”) 或是 0 (逻辑值“假”)。

例 5.7 全集为 whole, part1 是 whole 的一个子集, part2 是 part1 的补集。

```

sets:
    whole/1..4/: a;
    part1(whole)/2/: b;
    part2(whole)|#not#@in(part1,&1): c;
endsets

```

(2) @index([set_name,] primitive_set_element)

这个函数给出 primitive_set_element 在集 set_name 中的索引值 (即按定义集时元素出现顺序的位置编号)。如果省略集 set_name, LINGO 按模型中定义的集顺序找到第一个含有元素 primitive_set_element 的集, 并返回索引值。如果找不到, 则产生一个错误。

例 B.15

```

sets:
    girls/debble,sue,alice/;
    boys/bob,joe,sue,fred/;
endsets
I1=@index(sue);
I2=@index(boys,sue);

```

可以看到女孩和男孩中都有一个 sue 的孩子, 这时调用函数@index(sue), 得到的值是 2。因为集 girls 在集 boys 之前, 索引函数只对集 girls 检索。若干想查找男孩中的 sue, 应该使用@index(boys,sue)。建议在使用@index 函数时最好指定集。

例 B.16 确定集成员(B,Y)是否属于派生集 S3。

```

sets:
    var1/A B C/;
    var2/X Y Z/;
    var3(var1,var2)/A X, A Z, B Y, C X/;
endsets
X=@in(var3,@index(var1,B),@index(var2,Y));

```

(3) @wrap(index,limit)

该函数返回 $\text{index}-k*\text{limit}$, 其中 k 是一个整数, 取适当值保证 j 落在区间[1, limit]上。该函数在循环、多阶段计划编制中特别有用。



(4) @size(set_name)

该函数返回集 set_name 的成员个数。在模型中，如果没有明确给出集的大小，使用该函数能够使模型中的数据变化和集的大小改变更加方便。

3. 集循环函数

集循环函数是指对集合上的元素（下标）进行循环操作的函数。其语法为

@function(setname[(set_index_list)[condition]]:expression_list);

其中，function 是集循环函数 for, max, min, prod, sum 五种之一；

setname 是集名称；

set_index_list 是集索引（可以省略）；

condition 是使用逻辑表达式描述的过滤条件（通常含有索引，可以省略）；

expression_list 是一个表达式（对@for 函数可以是一组表达式）。

下面对集循环函数的含义作如下解释：

@for（集元素的循环函数）：对集 setname 的每个元素独立生成表达式，表达式由 expression_list 描述；

@max（集属性的最大值）：返回集 setname 中的表达式的最大值；

@min（集属性的最小值）：返回集 setname 中的表达式的最小值；

@prod（集属性的乘积）：返回集 setname 中的表达式的积；

@sum（集属性的和）：返回集 setname 中的表达式的和。

例 B.17 求向量[5, 1, 3, 4, 6, 10]前 5 个数的和。



model:

sets:

number/1..6/:x;

endsets

data:

x = 5, 1, 3, 4, 6, 10;

enddata

s=@sum(number(i) | i #le# 5: x);

end

B.18 求向量[5, 1, 3, 4, 6, 10]前 5 个数的最小值，后 3 个数的最大值。

model:

sets:

number/1..6/:x;

endsets

data:

x = 5, 1, 3, 4, 6, 10;

enddata

minv=@min(number(i)|i#le#5: x);

maxv=@max(number(i)|i#ge#4: x);

end

例 B.19（职员时序安排模型）一项工作一周 7 天都需要有人（比如护士工作），每天（周一至周五）所需的最少职员数为 20、16、13、16、19、14 和 12，并要求每个职员一周连续工作 5 天，试求每周所需最少职员数，并给出安排。

解 用 $i=1,2,\dots,7$ 分别表示周一、周二、…、周日，设第 i 天需要的人数为 x_i ($i=1,2,\dots,7$)，建立如下的线性规划模型：

$$\min \quad z = \sum_{i=1}^7 x_i,$$

$$\text{s.t.} \begin{cases} x_4 + x_5 + x_6 + x_7 + x_1 \geq 20, \\ x_5 + x_6 + x_7 + x_1 + x_2 \geq 16, \\ x_6 + x_7 + x_1 + x_2 + x_3 \geq 13, \\ x_7 + x_1 + x_2 + x_3 + x_4 \geq 16, \\ x_1 + x_2 + x_3 + x_4 + x_5 \geq 19, \\ x_2 + x_3 + x_4 + x_5 + x_6 \geq 14, \\ x_3 + x_4 + x_5 + x_6 + x_7 \geq 12. \end{cases}$$

计算的 LINGO 程序如下：

```
model:
sets:
    days/mon..sun/: a,x;
endsets
data:
    a= 20, 16, 13, 16, 19, 14, 12; !每天所需的最少职员数;
enddata
min=@sum(days: x); !最小化每周所需职员数;
@for(days(j): @sum(days(i) | i#le#5:x(@wrap(j+i-5,7))) >= a(j));
end
```

求得最小需要的总人数为 22 人，其中每天的安排如表 B.5 所示。

表 B.5 每天的人数安排

星期	星期一	星期二	星期三	星期四	星期五	星期六	星期日
人数	8	2	0	6	3	3	0

4. 变量界定函数

变量界定函数能够实现对变量取值范围的附加限制，共 4 种：

@bin(x)：限制 x 为 0 或 1；

@bnd(L,x,U)：限制 $L \leq x \leq U$ ；

@free(x)：取消对变量 x 的默认下界为 0 的限制，即 x 可以取任意实数；

@gin(x)：限制 x 为整数。

在默认情况下，LINGO 规定变量是非负的，也就是说下界为 0，上界为 $+\infty$ 。@free 取消了默认的下界为 0 的限制，使变量也可以取负值。@bnd 用于设定一个变量的上下界，它也可以取消默认下界为 0 的约束。

5. 概率函数

(1) @pbn(p,n,x)

二项分布的分布函数在 x 点的取值。当 n 和 (或) x 不是整数时，用线性插值法进行计算。

(2) @pcx(n,x)

自由度为 n 的 χ^2 分布的分布函数在 x 点的取值。

(3) @peb(a,x)

当到达负荷 (平均服务强度) 为 a，服务系统有 x 个服务台，且系统容量无限时的 Erlang 繁忙概率。

(4) @pel(a,x)

当到达负荷 (平均服务强度) 为 a，服务系统有 x 个服务台，且系统容量有限时的 Erlang 繁忙概率。

(5) @pfd(n,d,x)

自由度为 n 和 d 的 F 分布的分布函数在 x 点的取值。

(6) @pfs(a,x,c)

当负荷上限为 a，顾客数为 c，平行服务台数量为 x 时，有限源的 Poisson 服务系统的等待或返修顾客数的期望值。a 是顾客数乘以平均服务时间，再除以平均返修时间。当 c 和 (或) x 不是整数时，采用线性插值进行计算。

(7) @phg(pop,g,n,x)

超几何 (Hypergeometric) 分布的分布函数在 x 点的取值。pop 表示产品总数, g 是正品数。从所有产品中任意取出 n ($n \leq \text{pop}$) 件, 取出的正品数为 x 。pop, g , n 和 x 都可以是非整数, 这时采用线性插值进行计算。

(8) @ppl(a,x)

Poisson 分布的线性损失函数, 即返回 $\max(0, Z - x)$ 的期望值, 其中随机变量 Z 服从均值为 a 的 Poisson 分布。

(9) @pps(a,x)

均值为 a 的 Poisson 分布的分布函数在 x 点的取值。当 x 不是整数时, 采用线性插值进行计算。

(10) @psl(x)

单位正态线性损失函数, 即返回 $\max(0, Z - x)$ 的期望值, 其中随机变量 Z 服从标准正态分布。

(11) @psn(x)

标准正态分布的分布函数在 x 点的取值。

(12) @ptd(n,x)

自由度为 n 的 t 分布的分布函数在 x 点的取值。

(13) @qrand(seed)

产生 $(0,1)$ 区间上的拟随机数向量。@qrand 只允许在模型的数据部分使用, 它将用拟随机数填满集属性。通常, 声明一个 $m \times n$ 的二维表, m 表示运行实验的次数, n 表示每次实验所需的随机数的个数。在行内, 随机数是独立分布的; 在行间, 随机数是非常均匀的。这些随机数是用“分层取样”的方法产生的。

例 B.20

```
model:
sets:
    rows/1..4/;
    cols/1..2/;
    table(rows,cols): x;
endsets
data:
    X=@qrand(); !@qrand(seed)中没有指明种子, LINGO用系统时间构造种子;
enddata
end
```

(14) @rand(seed)

返回 0 和 1 间的一个伪随机数, 依赖于指定的种子。典型用法是 $U(I+1)=@rand(U(I))$ 。注意如果 seed 不变, 那么产生的随机数也不变。

例 B.21 利用 @rand 产生 15 个服从标准正态分布的随机数和自由度为 2 的 t 分布的随机数。

```
model:
sets:
    series/1..15/: u, znorm, zt;
endsets
u(1)=@rand(0.1234); !产生第一个(0,1)区间上随机数;
@for(series(i)|i#gt# 1:u(i)=@rand(u(i-1))); !产生其余的(0,1)区间上的随机数;
@for(series(i):
    @psn(znorm(i))=u(i); !正态分布随机数;
    @ptd(2,zt(i))=u(i); !自由度为2的t分布随机数;
    @free(znorm(i)); @free(zt(i)); !ZNORM 和 ZT 可以是负数;
end
```

6. 金融函数

目前 LINGO 提供了两个金融函数。

(1) @fpa(I,N)

返回如下情形的净现值：单位时段利率为 I ，连续 N 个时段支付，每个时段支付单位费用。若每个时段支付 x 单位的费用，则净现值可用 x 乘以 @fpa(I,N) 算得。@fpa 的计算公式为

$$@fpa(I, N) = \sum_{n=1}^N \frac{1}{(1+I)^n} = \left(1 - \left(\frac{1}{1+I} \right)^N \right) / I.$$

例 B.22 贷款总金额 50000 元，贷款年利率 5.31%，采取分期付款方式（每年年末还固定金额，直至还清）。问拟贷款 10 年，每年需偿还多少元？

解 设贷款的总额为 A_0 元，年利率为 r ，总贷款时间为 N 年，每年的等额还款额为 x 元。设第 n 年的欠款为 A_n ($n=0,1,\dots,N$)，则有递推关系

$$A_{n+1} = (1+r)A_n - x, \quad n=0,1,\dots,N-1.$$

于是有

$$\begin{aligned} A_1 &= (1+r)A_0 - x, \\ A_2 &= (1+r)A_1 - x, \\ &\dots\dots\dots, \\ A_N &= (1+r)A_{N-1} - x, \end{aligned}$$

可以递推地得到

$$\begin{aligned} A_N &= A_0(1+r)^N - x[(1+r)^{N-1} + (1+r)^{N-2} + \dots + (1+r) + 1] \\ &= A_0(1+r)^N - x \frac{(1+r)^N - 1}{r}, \end{aligned}$$

因而得到贷款总额 A_0 、年利率 r 、总贷款时间 N 年、每年的还款额 x 的如下关系

$$A_N = A_0(1+r)^N - x \frac{(1+r)^N - 1}{r} = 0, \quad (1)$$

所以每年的还款额

$$x = \frac{A_0(1+r)^N r}{(1+r)^N - 1}. \quad (2)$$

代入数据，计算得每年需偿还 $x = 6573.069$ 元。

计算的 LINGO 程序如下：

A0=50000; r=0.0531; N=10;

A0=x1*@fpa(r,N); !利用 LINGO 函数解方程计算;

x2=A0*(1+r)^N*r/((1+r)^N-1); !利用还款额公式(2)计算

(2) @fpl(I,N)

返回如下情形的净现值：单位时段利率为 I ，第 N 个时段支付单位费用。@fpl(I,N) 的计算公式为

$$(1+I)^{-N}.$$

这两个函数间的关系为

$$@fpa(I, N) = \sum_{k=1}^N @fpl(I, k).$$

例 B.23 验证 $@fpa(0.05,10) = \sum_{k=1}^{10} @fpl(0.05,k)$ 。

验证的 LINGO 程序如下：

sets:

num/1..10/;

endsets

r=0.05; a=@fpa(r,10);

b=@sum(num(i):@fpl(r,i));

B.4 LINGO 与其他文件的数据传递

B.4.1 通过文本文件传递数据

在LINGO软件中，通过文本文件输入数据使用的是@file函数，输出结果使用的是@text函数。下面介绍这两个函数的详细用法。

1.通过文本文件输入数据

@file 函数通常可以在集合段和数据段使用，但不允许嵌套使用。这个函数的一般用法是

@file(filename);

其中 filename 为存放数据的文件名，文件名可以包含完整的路径名，没有指定路径时表示在当前目录下寻找这个文件。该文件必须是文本（或 ASCII 码文件），可以用 Windows 附件中的写字板或记事本创建，文件中可以包含多个记录，记录之间用“~”分开，同一记录内的多个数据之间用逗号或空格分开。执行一次@file，读入一个记录的数据。下面通过一个简单的例子来说明。

例 B.24 @file 函数的用法示例。

假设存放数据的文本文件 myfile.txt 的内容如下：

Seattle,Detroit,Chicago,Denver~

COST,NEED,SUPPLY,ORDERED~

12,28,15,20~

1600,1800,1200,1000~

1700,1900,1300,1100

现在，在 LINGO 模型窗口中建立如下 LINGO 模型：

model:

sets:

myset/@file(myfile.txt)/: @file(myfile.txt);

endsets

data:

cost=@file(myfile.txt); !LINGO是不区分大小写字母的;

need=@file(myfile.txt);

supply=@file(myfile.txt);

enddata

end

运行上述 LINGO 模型的结果为：文本文件 myfile.txt 中第一行的 4 个字符串赋值给集合 myset 的 4 个成员，第二行的 4 个字符串 COST,NEED,SUPPLY,ORDERED（或 cost,need,supply,ordered）成为集合 myset 的 4 个属性，第三行的 4 个数值赋值给属性 cost，第四行的 4 个数值赋值给属性 need，第五行的 4 个数值赋值给属性 supply，未赋值的属性 ordered 作为决策向量。

显然，当仅仅是输入数据改变了，只需要改变输入文件 myfile.txt，而程序无需改变，这是非常有利的，因为这样就做到了程序与数据的分离。

2.通过文本文件输出数据

@text 函数用于文本文件输出数据，通常只在数据段使用这个函数。这个函数的语法为：

@text([filename,['a']])

它用于数据段中将解答结果输出到文本文件 filename 中，当省略 filename 时，结果送到标准的输出设备（通常就是屏幕）。当有第二个参数 'a' 时，数据是以追加（append）的方式输出到文本文件，否则，是新建一个文本文件（如果文件已经存在，则其中的内容将会被覆盖）供输出数据。

@text 函数的一般调用格式为：

@text('results.txt')=属性名;

其中 results.txt 是文件名，它可以由用户按自己的意愿命名，该函数的执行结果是把属性名对应的取值输出到文本文件 results.txt 中。

例 B.25 已知某种商品 6 个仓库的存货量，8 个客户对该商品的需求量，单位商品运价如表 B.6 所示。试确定 6 个仓库到 8 个客户的商品调运数量，使总的运输费用最小。

表 B.6 商品信息数据表

单位运价 仓库 \ 客户	V1	V2	V3	V4	V5	V6	V7	V8	存货量
W1	6	2	6	7	4	2	5	9	60
W2	4	9	5	3	8	5	8	2	55
W3	5	2	1	9	7	4	3	3	51
W4	7	6	7	3	9	2	7	1	43
W5	2	3	9	5	7	2	6	5	41
W6	5	5	2	2	8	1	4	3	52
需求量	35	37	22	32	41	32	43	38	

解 设 x_{ij} ($i=1,2,\dots,6; j=1,2,\dots,8$) 表示第 i 个仓库运到第 j 个客户的商品数量， c_{ij} 表示第 i 个仓库到第 j 个客户的单位运价， d_j 表示第 j 个客户的需求量， e_i 表示第 i 个仓库的存货量，建立如下线性规划模型

$$\begin{aligned} \min \quad & \sum_{i=1}^6 \sum_{j=1}^8 c_{ij} x_{ij}, \\ \text{s.t.} \quad & \begin{cases} \sum_{j=1}^8 x_{ij} \leq e_i, & i=1,2,\dots,6, \\ \sum_{i=1}^6 x_{ij} = d_j, & j=1,2,\dots,8, \\ x_{ij} \geq 0, & i=1,2,\dots,6; j=1,2,\dots,8. \end{cases} \end{aligned}$$

在例 B.25 的运输问题中，使用文本文件输入和输出数据。

```

model:
sets:
    warehouses/1..6/: e;
    vendors/1..8/: d;
    links(warehouses,vendors): c,x;
endsets
data: !数据部分;
e=@file(gd251.txt);
d=@file(gd251.txt);
c=@file(gd251.txt);
@text(gd252.txt)=@table(x); !把求解结果以表格形式输出到文本文件gd252. txt中;
enddata
min=@sum(links(i,j): c(i,j)*x(i,j)); !目标函数;
@for(warehouses(i):@sum(vendors(j): x(i,j))<=e(i)); !约束条件;
@for(vendors(j):@sum(warehouses(i): x(i,j))=d(j));
end

```

其中文本文件 gd251.txt 中的内容如下：

```
60 55 51 43 41 52~
35 37 22 32 41 32 43 38~
6 2 6 7 4 2 5 9
4 9 5 3 8 5 8 2
5 2 1 9 7 4 3 3
7 6 7 3 9 2 7 1
2 3 9 5 7 2 6 5
5 5 2 2 8 1 4 3
```

注 B.5 文本文件 gd251.txt 必须放在 LINGO 程序所在目录下。

注 B.6 在例 B.25 数学模型中 $c = (c_{ij})_{6 \times 8}$ 是一个矩阵，即二维向量，在上述 LINGO 程序中，属性 c 数据（LINGO 中通过 c(i,j) 引用属性的值）的排列方式为

$c(1,1), c(1,2), \dots, c(1,8), c(2,1), c(2,2), \dots, c(2,8), \dots, c(6,1), c(6,2), \dots, c(6,8);$

即 LINGO 中的数据是逐行排列的。

B.4.2 LINGO 与 EXCEL 文件之间的数据传递

LINGO 通过 @ole 函数实现与 EXCEL 文件传递数据，使用 @ole 函数既可以从 EXCEL 文件中输入数据，也能把计算结果输出到 EXCEL 文件。

1. 通过 EXCEL 文件输入数据

@ole 函数只能用在模型的集定义段、数据段和初始段。使用格式为

`object_list = @OLE(['spreadsheet_file'] [, range_name_list]);`

其中 spreadsheet_file 是电子表格文件的名称，应当包括扩展名（如 *.xls, *.xlsx 等），还可以包含完整的路径名，只要字符数不超过 64 即可；range_name_list 是指文件中包含数据的单元范围（单元范围的格式与 EXCEL 中工作表的单元范围格式一致）。其中 spreadsheet_file 和 range_name_list 都是可以缺省的。

具体地说，当从 EXCEL 中向 LINGO 模型中输入数据时，在集定义段可以直接采用“@ole(…)”的形式读入集成员，但在数据段和初始段应当采用“属性=@ole(…)”的赋值形式。

2. 通过 EXCEL 文件输出数据

@ole 函数能把数据输出到 EXCEL 文件，调用格式为

`@OLE(['spreadsheet_file'] [, range_name_list]) = object_list;`

其中对象列表 object_list 中的元素用逗号分隔，spreadsheet_file 是输出值所保存到的 EXCEL 文件名，如果文件名缺省，默认的文件名是当前 EXCEL 软件所打开的文件。域名列表 range_name_list 是表单中的域名，所在的单元用于保存对象列表中的属性值，表单中的域名必须与对象列表中的属性一一对应，并且域名所对应的单元大小（数据块的大小）不应小于变量所包含的数据，如果单元中原来有数据，则 @ole 输出语句运行后原来的数据将被新的数据覆盖。

要注意 @ole 函数用于输出和输入之间的差异，只要记住

`@ole(...) = object_list;` ↔ 输出，

`object_list = @ole(...);` ↔ 输入。

例 B.26（续例 B.25）在例 B.25 的运输问题中，使用 EXCEL 文件输入和输出数据。

首先，我们用 EXCEL 建立一个名为 gd26.xlsx 的 EXCEL 数据文件，参见图 B.4。为了能够通过 @ole 函数与 LINGO 传递数据，我们需要对这个文件中的数据进行命名，具体做法是：我们用鼠标选中这个表格的 B4: B7 单元，然后选择 EXCEL 的菜单命令“插入→名称→定义”，这时将会弹出一个对话框，请您输入名字，例如可以将它命名为 cost（LINGO 禁

止使用 c 作为域名), 同理, 我们将 J2: J7 单元命名为 e, 将 B4: I8 单元命令为 d, 将 B10: I15 单元命名为 x。一般来说, 这些单元取什么名字是无所谓的, 只要 LINGO 调用时使用对应的名字就可以了。但最好是这些单元的名称 (称为域名) 与 LINGO 对应的属性名同名, 将来 LINGO 调用时就可以省略域名。

	A	B	C	D	E	F	G	H	I	J
1		V1	V2	V3	V4	V5	V6	V7	V8	存量
2	W1	6	2	6	7	4	2	5	9	60
3	W2	4	9	5	3	8	5	8	2	55
4	W3	5	2	1	9	7	4	3	3	51
5	W4	7	6	7	3	9	2	7	1	43
6	W5	2	3	9	5	7	2	6	5	41
7	W6	5	5	2	2	8	1	4	3	52
8	需求量	35	37	22	32	41	32	43	38	

图 B.4 运输问题的已知数据

```

model:
sets:
    warehouses/1..6/: e;
    vendors/1..8/: d;
    links(warehouses,vendors): c,x;
endsets
data: !数据部分;
e=@ole(gd26.xlsx); !域名与属性名相同时, 调用时省略域名;
d=@ole(gd26.xlsx);
c=@ole(gd26.xlsx,cost); !域名与属性名不相同, 调用时必须提供域名;
@ole(gd26.xlsx)=x; !把求解结果输出到EXCEL文件;
enddata
min=@sum(links(i,j): c(i,j)*x(i,j)); !目标函数;
@for(warehouses(i): @sum(vendors(j): x(i,j))<=e(i)); !约束条件;
@for(vendors(j): @sum(warehouses(i): x(i,j))=d(j));
end
  
```

求解结果的输出内容如图 B.5 所示。

	A	B	C	D	E	F	G	H	I	J
1		V1	V2	V3	V4	V5	V6	V7	V8	存量
2	W1	6	2	6	7	4	2	5	9	60
3	W2	4	9	5	3	8	5	8	2	55
4	W3	5	2	1	9	7	4	3	3	51
5	W4	7	6	7	3	9	2	7	1	43
6	W5	2	3	9	5	7	2	6	5	41
7	W6	5	5	2	2	8	1	4	3	52
8	需求量	35	37	22	32	41	32	43	38	
9										
10		0	19	0	0	41	0	0	0	
11		1	0	0	32	0	0	0	0	
12		0	11	0	0	0	0	40	0	
13		0	0	0	0	0	5	0	38	
14		34	7	0	0	0	0	0	0	
15		0	0	22	0	0	27	3	0	

图 B.5 求解结果的输出数据

注 B.7 LINGO 要输入外部 EXCEL 文件中的数据, 必须预先用 EXCEL 软件把要操作的 EXCEL 文件打开, 否则 LINGO 是无法输入数据的。

例 B.27（续例 B.26）使用 EXCEL 的单元范围作为域名，输入或输出属性的值。

我们使用 EXCEL 的单元范围作为域名，不需要像例 B.26 那样，需要预先定义输入数据的域名，才能向 LINGO 输入或输出数据。

```

model:
sets:
    warehouses/1..6/: e;
    vendors/1..8/: d;
    links(warehouses,vendors): c,x;
endsets
data: !数据部分;
e=@ole(gd26.xlsx,J2:J7);
d=@ole(gd26.xlsx,B8:I8);
c=@ole(gd26.xlsx,B2:I7);
@ole(gd26.xlsx,B10:I15)=x; !把求解结果输出到EXCEL文件;
enddata
min=@sum(links(i,j): c(i,j)*x(i,j)); !目标函数;
@for(warehouses(i): @sum(vendors(j): x(i,j))<=e(i)); !约束条件;
@for(vendors(j): @sum(warehouses(i): x(i,j))=d(j));
end

```

习题

1.为了破坏敌方海上交通线，根据交通线情况设置了 5 个潜艇活动海域（阵地） B_j ($j=1,2,\dots,5$)。根据交通线运输频繁程度和海区的开阔程度，确定了各活动海域内潜艇的数量分别为： $b_1=2$ ， $b_2=2$ ， $b_3=3$ ， $b_4=3$ ， $b_5=4$ 。潜艇则由 A_i ($i=1,2,\dots,5$) 5 个基地派出，各基地能够派出的潜艇数量分别为 $a_1=3$ ， $a_2=3$ ， $a_3=3$ ， $a_4=3$ ， $a_5=2$ 。从海图上可以量取各基地到各阵地之间的航程见表 B.7。求使总航程最短的最优兵力派出方案。

表 B.7 潜艇机动距离（单位：km）

潜艇阵地 潜艇基地		B_1	B_2	B_3	B_4	B_5
		$b_1=2$	$b_2=2$	$b_3=3$	$b_4=3$	$b_5=4$
A_1	$a_1=3$	516	645	1290	1204	1634
A_2	$a_2=3$	451	537	1110	989	1505
A_3	$a_3=3$	323	258	688	645	1075
A_4	$a_4=3$	344	278	708	655	1095
A_5	$a_5=2$	1268	1032	1296	796	258

2.设我水面舰艇编队组织 10 座相同口径的主炮（双联装，可以双管发射）同时对敌岸上目标实施破坏性炮火射击。目标按其类型与分布（如岸炮群、碉堡群、铁桥、指挥所）分为 4 组。每组目标被摧毁必须命中炮弹数 n_i ($i=1,2,3,4$) 及我跑群对其射击的单发命中概率 p_i ($i=1,2,3,4$) 见表 B.8。我炮弹发射率为单管 25 发/min，要求每个目标至少有一座主炮去射击，每座主炮只能射击一个目标，如何进行兵力配置才能使战斗时间最短。

表 B.8 平均必须命中炮弹数与单发命中概率

目标 指标	1	2	3	4
n_i	100	5	9	15
p_i	0.2	0.005	0.03	0.02

3.求解下列线性规划问题

$$\begin{aligned} \max \quad & z = 13x_{11} + 13x_{12} + 9x_{21} + 9x_{22}, \\ \text{s.t.} \quad & \begin{cases} x_{11} + x_{12} \leq 5, \\ x_{21} + x_{22} \leq 5, \\ 6(x_{11} + x_{12}) + 2(x_{21} + x_{22}) \leq 32, \\ 7(x_{11} + x_{12}) + 8.5(x_{21} + x_{22}) \leq 60, \\ x_{11} + x_{21} \leq 1.5(x_{12} + x_{22}), \\ 0 \leq x_{11} \leq 4, 0 \leq x_{12} \leq 3, 0 \leq x_{21} \leq 2, 0 \leq x_{22} \leq 4. \end{cases} \end{aligned}$$

4.求解下列线性规划问题

$$\begin{aligned} \max \quad & z = x_1 - 3x_2 - 5x_3, \\ \text{s.t.} \quad & \begin{cases} x_1 + 2x_2 + x_3 \leq 4, \\ x_1 - 6x_2 + 4x_3 \leq 5, \\ -x_1 + x_2 + 2x_3 \leq 10, \\ 2x_1 + 3x_2 + 3x_3 = -7, \\ x_1 \geq 0, x_2 \leq 0, x_3 \text{可正可负}. \end{cases} \end{aligned}$$