9. 1stOpt 中巧用“Sum”、“Prod”、“For”、“MinIn”与“MaxIn”函数

“Sum”、“Prod”、“For”、“MinIn”和“MaxIn”在 1stOpt 快捷模式中是三个非常有用的函数，顾名思义，分别表示连续求和、连续求乘积和众多相同类别公式的简洁表达方式。通过三个实例介绍在 1stOpt 快捷模式下“Sum” 、“Prod”、“For”、“MinIn”与“MaxIn”函数的用法。

9.1 合金构成问题 — “Sum”与“For”的用法。

如下表，有 9 种市售合金，分别由铅，亚铅和锡以不同百分比构成，每种合金价格已知，现要求如何构成一种新的合金，其铅、亚铅和锡的比例分别为 30%，30%和 40%，且价格最低？

表 9.1 合金百分比构成数据

市售合金	1	2	3	4	5	6	7	8	9
铅（%）	20	50	30	30	30	60	40	10	10
亚铅（%）	30	40	20	40	30	30	50	30	10
锡（%）	50	10	50	30	40	10	10	60	80
价格（$/lb）	7.3	6.9	7.3	7.5	7.6	6.0	5.8	4.3	4.1

设 x1, x2,...x9 分别代表各种已知合金在新合金中的比例，则有目标函数：

$$\text{Min.}\quad 7.3x_1+\sin(6.9)x_2+7.3x_3+7.5x_4+7.6x_5+6.0x_6+5.8x_7+4.3x_8+4.1x_9$$

约束条件：

$$\text{S.t.}\quad \begin{cases} x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8+x_9=1 \\ 0.2x_1+0.5x_2+0.3x_3+0.3x_4+0.3x_5+0.6x_6+0.4x_7+0.1x_8+0.1x_9=0.3 \\ 0.3x_1+0.4x_2+0.2x_3+0.4x_4+0.3x_5+0.3x_6+0.5x_7+0.3x_8+0.1x_9=0.3 \\ 0.5x_1+0.1x_2+0.5x_3+0.3x_4+0.4x_5+0.1x_6+0.1x_7+0.6x_8+0.8x_9=0.4 \end{cases}$$

且所有 x 均大于 0

因为该问题是典型的线性约束优化问题，因此算法选择线性算法（Algorithm = LP;）。

常规代码写法 9-1

```
Algorithm = LP;
Parameter x(9)>0;
MinFunction 7.3*x1+6.9*x2+7.3*x3+7.5*x4+7.6*x5+6.0*x6+5.8*x7+4.3*x8+4.1*x9;
x1+x2+x3+x4+x5+x6+x7+x8+x9=1;
0.2*x1+0.5*x2+0.3*x3+0.3*x4+0.3*x5+0.6*x6+0.4*x7+0.1*x8+0.1*x9=0.3;
```

```
0.3*x1+0.4*x2+0.2*x3+0.4*x4+0.3*x5+0.3*x6+0.5*x7+0.3*x8+0.1*x9=0.3;
0.5*x1+0.1*x2+0.5*x3+0.3*x4+0.4*x5+0.1*x6+0.1*x7+0.6*x8+0.8*x9=0.4;
```

如果使用“Sum”和“For”函数，需先将已知相关数据写成一维或二维数组形式，便于调用。

代码：

```
Constant R1(3,9)=[20,50,30,30,30,60,40,10,10,
                  30,40,20,40,30,30,50,30,10,
                  50,10,50,30,40,10,10,60,80]/100,
         Cost=[7.3,6.9,7.3,7.5,7.6,6.0,5.8,4.3,4.1],
         R2(3)=[30,30,40]/100;
Algorithm = LP;
Parameter x(9)>0;
MinFunction Sum(i=1:9)(x[i]*Cost[i]);
            Sum(i=1:9)(x[i])=1;
            For(i=1:3)(Sum(j=1:9)(x[j]*R1[i,j])=R2[i]);
```

上述两种写法都可以得到同样结果，当数据规模比较大时，如有 100 种合金的组合，第二组写法的优势就更大了，仅需修改数据，主要代码可保持不变。

```
====== 结果输出 ======

迭代数: 9
计算用时(时:分:秒:微秒): 00:00:00:03
算法: 单纯形线性规划法
目标函数值(最小): 4.98

参数最优解为:
x1: 0
x2: 0
x3: 0
x4: 0
x5: 0
x6: 0.4
x7: 0
x8: 0.6
x9: 0

约束条件:
1: x1+x2+x3+x4+x5+x6+x7+x8+x9-1 = 0
2: x1*0.2+x2*0.5+x3*0.3+x4*0.3+x5*0.3+x6*0.6+x7*0.4+x8*0.1+x9*0.1-0.3 = 0
3: x1*0.3+x2*0.4+x3*0.2+x4*0.4+x5*0.3+x6*0.3+x7*0.5+x8*0.3+x9*0.1-0.3 = 0
4: x1*0.5+x2*0.1+x3*0.5+x4*0.3+x5*0.4+x6*0.1+x7*0.1+x8*0.6+x9*0.8-0.4 = 0
5: x1-0 = 0
6: x2-0 = 0
```

7: x3-0 = 0
8: x4-0 = 0
9: x5-0 = 0
10: x6-0 = 0.4
11: x7-0 = 0
12: x8-0 = 0.6
13: x9-0 = 0

9.2 最小二乘拟合问题 — “Sum”的用法。

已知拟合自变量 x 和因变量 y 数据如下表：

表 9.2 拟合数据

x	234,235,242,244,252,266,269,269,269,269,273,276,277,280,281,284,285,288,291,292,294,294
y	383,379,370,368,359,349,344,346,347,345,346,349,350,353,354,357,359,362,368,369,371,372

拟合方程：

$$y = a_4 + \frac{a_1\left(a_2^2 + x^2 - 2xa_3 + a_3^2\right)^{\frac{1}{2}}}{a_2}$$

一般拟合代码 9-3

```
Parameter a(4);
Variable x,y;
Function y=a4+(a1*(a2^2+x^2-2*x*a3+a3^2)^(1/2))/a2;
Data;
234,235,242,244,252,266,269,269,269,269,273,276,277,280,281,284,285,288,291,292,294,294;
383,379,370,368,359,349,344,346,347,345,346,349,350,353,354,357,359,362,368,369,371,372;
```

如果自行构筑最小二乘模型公式，则有如下优化模型：

$$\text{Min.} \sum_{i=1}^{22}\left(\left(a_4 + \frac{a_1\left(a_2^2 + x_i^2 - 2x_i a_3 + a_3^2\right)^{\frac{1}{2}}}{a_2} - y_i\right)^2\right)$$

代码 9-4

```
Constant x=[234,235,242,244,252,266,269,269,269,269,273,276,277,280,281,284,285,288,291,292,294,294],
y=[383,379,370,368,359,349,344,346,347,345,346,349,350,353,354,357,359,362,368,369,371,372];
Parameter a(4);
MinFunction Sum(x,y)(sqr(a4+(a1*(a2^2+x^2-2*x*a3+a3^2)^(1/2))/a2-y));
```

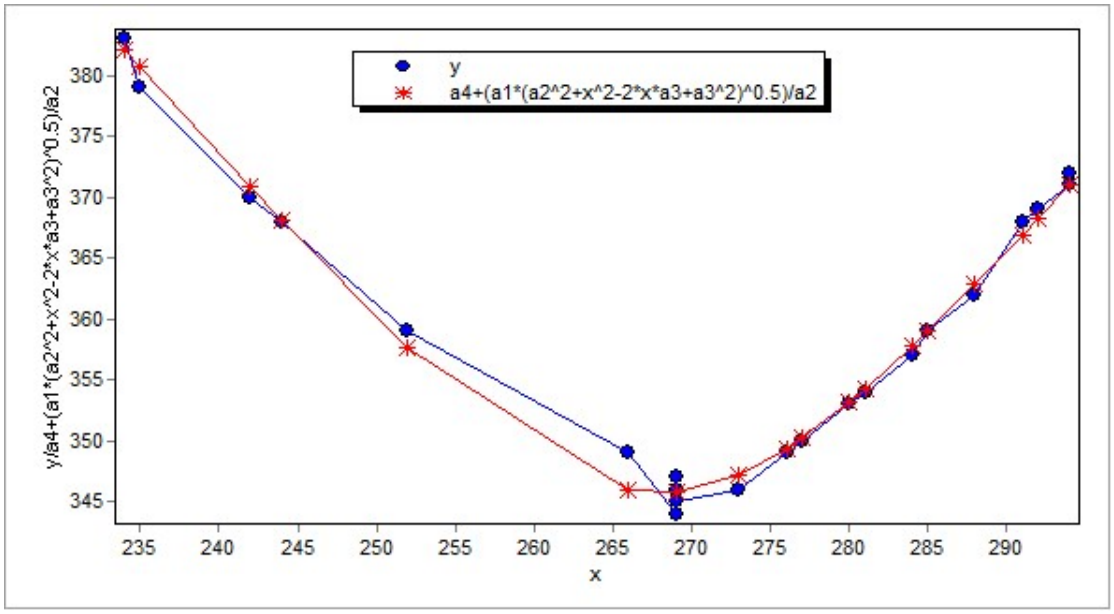
上述两段代码可得同样结果：

```
目标函数值(最小): 26.9589286673631
a1: 19.1175475421374
a2: 12.4570965310482
a3: 267.950793501975
a4: 326.644042820755
```

如果在第二段自行构造的最小二乘代码求解过程中，想实时显示计算与实测的因变量对比图，该如何实现？可用如下代码 9-5：

```
Constant x=[234,235,242,244,252,266,269,269,269,269,273,276,277,280,281,284,285,288,291,292,294,294],
y=[383,379,370,368,359,349,344,346,347,345,346,349,350,353,354,357,359,362,368,369,371,372];
ConstStr f=a4+(a1*(a2^2+x^2-2*x*a3+a3^2)^(1/2))/a2;
Plot x[x],y,f;
Parameter a(4);
MinFunction Sum(x,y)(sqr(f-y));
```

可以实时得到计算对比图：



9.3 方程求解问题 — “Sum”与“Prod”的用法。

如下图示，已知：n=50，A=10，Z=10，该方程组由 n=50 个方程组成，且每个方程的构成有一定的规律。

$$\begin{cases} x_1 = \dfrac{\sqrt{(A-x_1)^2+z^2}}{x_1-A} \\ x_2 = \dfrac{\sqrt{\left(A-(x_1+x_2)\right)^2+z^2}}{x_1+x_2-A} \\ x_3 = \dfrac{\sqrt{\left(A-(x_1+x_2+x_3)\right)^2+z^2}}{x_1+x_2+x_3-A} \\ \vdots \\ x_n = \dfrac{\sqrt{\left(A-(x_1+x_2\ldots+x_n)\right)^2+z^2}}{x_1+x_2\ldots+x_n-A} \end{cases}$$

如果一个一个方程书写，不仅费时费力，还容易出错，而用“For”和“Sum”函数，代码仅需两句：

代码 9-6

```
Constant n=50,A=10,Z=10;
Function For(i=1:n)(x[i]=sqrt((A-Sum(j=1:i)(x[j]))^2+Z^2)/(Sum(j=1:i)(x[j])-A));
```

在代码本里按快捷键“Ctrl+t”，上述代码里的公式如下：

$$x_i = \frac{\sqrt{\left(10-\sum_{j=1}^{i}(x_j)\right)^2+10^2}}{\sum_{j=1}^{i}(x_j)-10} \quad i=1..50$$

结果：

```
目标函数值(最小): 1.66646866227939E-29
x1: -1.333607529306
x2: -1.27628133438355
x3: -1.23361448275201
x4: -1.20076342180857
x5: -1.17479544308883
.
.
.
x45: -1.01468965146078
x46: -1.01419366299101
x47: -1.01372253697351
x48: -1.0132746290347
x49: -1.01284842930399
```

x50: -1.01244254935425

如上面求解方程变为如下含有连乘的形式：

$$\begin{cases} x_1 = \dfrac{\sqrt{(A-x_1)^2+z^2}}{x_1-A} \\ x_2 = \dfrac{\sqrt{\left(A-(x_1\cdot x_2)\right)^2+z^2}}{x_1+x_2-A} \\ x_3 = \dfrac{\sqrt{\left(A-(x_1\cdot x_2\cdot x_3)\right)^2+z^2}}{x_1+x_2+x_3-A} \\ \vdots \\ x_n = \dfrac{\sqrt{\left(A-(x_1\cdot x_2 \ldots \cdot x_n)\right)^2+z^2}}{x_1+x_2 \ldots + x_n-A} \end{cases}$$

或简写为：

$$x_i = \frac{\sqrt{\left(10-\prod_{j=1}^{i}(x_j)\right)^2+10^2}}{\sum_{j=1}^{i}(x_j)-10} \quad i = 1..50$$

代码 9-7

```
Constant n=50,A=10,Z=10;
Function For(i=1:n)(x[i]=sqrt((A-Prod(j=1:i)(x[j]))^2+Z^2)/(Sum(j=1:i)(x[j])-A));
```

结果：

```
目标函数值(最小): 3.11345227756522E-25
x1: -1.3336075293062
x2: -1.06302167636053
x3: -1.13233615041634
x4: -0.91063301625705
x5: -0.985078562974792.
.
.
x45: -0.383978495370469
x46: -0.380056666720802
x47: -0.376252220060503
x48: -0.372559425196937
x49: -0.36897293556197
x50: -0.365487755841811
```

9.4 “MinIn”与“MaxIn”

“MinIn”与“MaxIn”函数是分别求多个不等式中的最小值和最大值，其用法与“Sum”和“Prod”函数相同：

MinIn(i=1:n)(f(i))

会求出 f1, f2,…fn 表达式中的最小值，相对于：

Min(f[1], Min(f[2],… Min(f[n-1], f[n])))

例如下优化问题：

其中 n = 5.

约束条件：

$$\begin{cases} x_i \geq 0 \\ \mathrm{Min}(\mathrm{i} \cdot \sin(x_i)) \geq 0.5 \end{cases} \qquad i = 1 \ldots n$$

代码 9-8

```
Constant n=5;
Parameter x(n)>0;
MinFunction Sum(i=1:n,x)(x);
            MinIn(i=1:n,x)(sin(x)*i)>0.5;
```

结果：

```
函数表达式: ((((x1)))+(((x2)))+(((x3)))+(((x4)))+(((x5))))
目标函数值(最小): 1.16922594977644
x1: 0.523600382086322
x2: 0.252681031956438
x3: 0.16744859233476
x4: 0.125328215213072
x5: 0.100167728185847

约束函数:
1: min(sin(x1)*1,min(sin(x2)*2,min(sin(x3)*3,min(sin(x4)*4,sin(x5)*5))))-(0.5) = 1.39125879405899E-6
```

9.5 结论

“Sum”、“Prod”、“For”、“MinIn”和“MaxIn”函数在 1stOpt 的快捷模式下非常有用，不仅可以提高代码书写效率，而且可防止人为出错。