1stOpt 10.0 主要新增功能

1stOpt 10.0 版于 2022 年 11 月 1 日正式发布，主要新增功能及改进如下。12 月 1 日开始符合免费升级或打算升级的用户可申请办理。

1. 神经网络拟合工具箱及相关命令

不同于已有的相对独立的神经网络工具箱 NeuralPower，全新的神经网络拟合工具箱 NNFit 与 1stOpt 完全融为一体，提供可视化神经网络设计并可产生 1stOpt 标准运行代码，用户不仅可以完全掌控神经网络拟合模型的设计过程，还可以清晰了解详细全面的计算流程甚至完整的拟合模型公式（虽然公式有时非常长），打破神经网络“黑箱”操作的谬传，拟合训练后的模型可方便迁移和再现，实用性显著增强。

在熟悉并掌握可视化设计后，仅用一句代码即可实现复杂神经网络拟合计算，再结合 1stOpt 特有的全局优化算法，在拟合模型公式不定的情况下，几乎可以拟合任何复杂形式的数据并得到满意的结果。

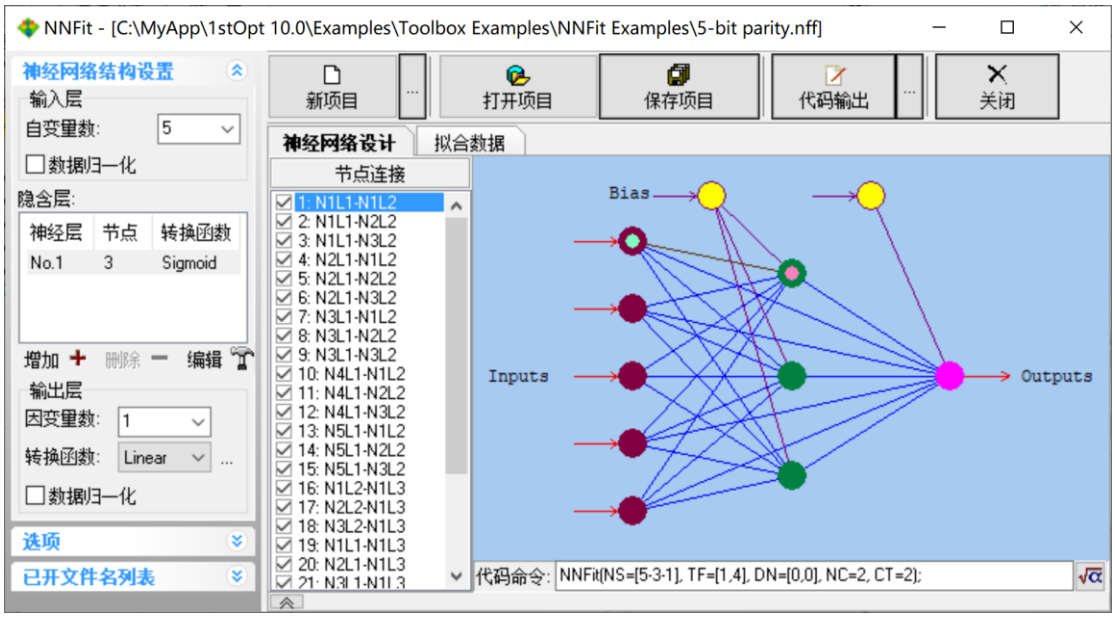

图-1. 神经网络可视化设计

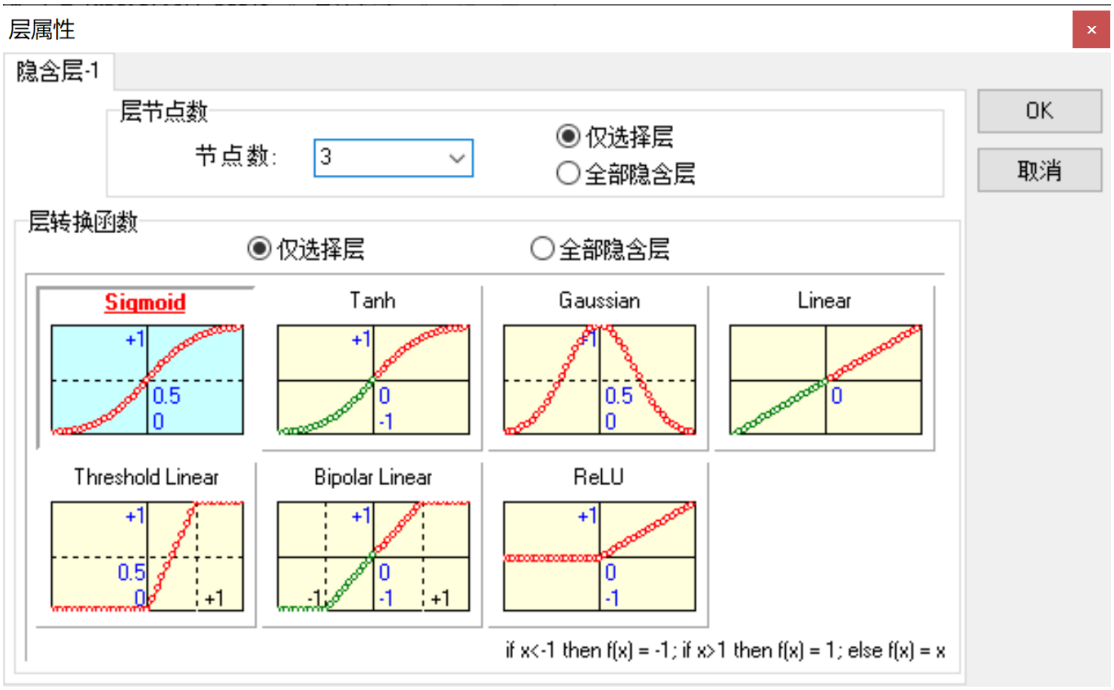

图-2. 神经网络转换（激活）函数

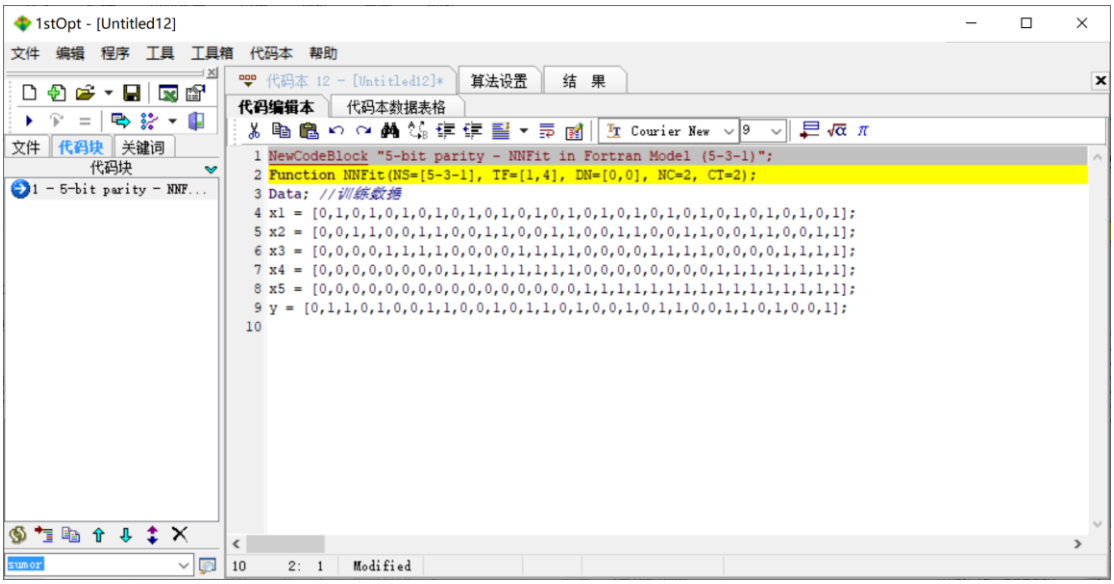

图-3. 代码自动生成（精简格式）

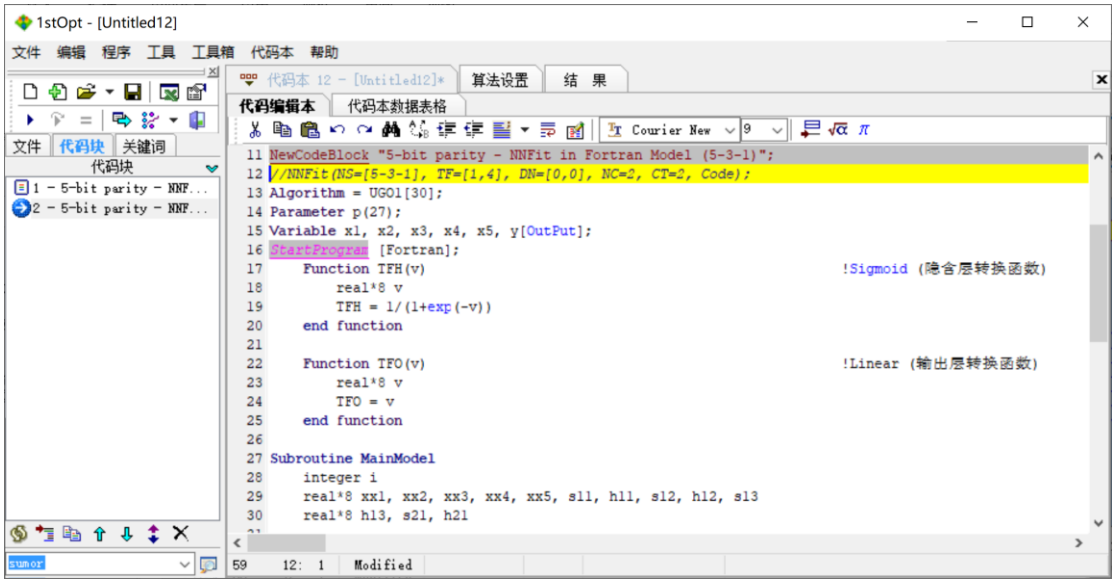

图-4. 代码自动生成（完全格式）

上述神经网络模型对应的拟合公式：

$$y = p_{16} \cdot \left(\frac{1}{1+\exp\left(-\left(p_1 \cdot x_1 + p_4 \cdot x_2 + p_7 \cdot x_3 + p_{10} \cdot x_4 + p_{13} \cdot x_5 + p_{24}\right)\right)}\right) + p_{17}$$
$$\cdot \left(\frac{1}{1+\exp\left(-\left(p_2 \cdot x_1 + p_5 \cdot x_2 + p_8 \cdot x_3 + p_{11} \cdot x_4 + p_{14} \cdot x_5 + p_{25}\right)\right)}\right) + p_{18}$$
$$\cdot \left(\frac{1}{1+\exp\left(-\left(p_3 \cdot x_1 + p_6 \cdot x_2 + p_9 \cdot x_3 + p_{12} \cdot x_4 + p_{15} \cdot x_5 + p_{26}\right)\right)}\right) + p_{27}$$
$$+ p_{19} \cdot x_1 + p_{20} \cdot x_2 + p_{21} \cdot x_3 + p_{22} \cdot x_4 + p_{23} \cdot x_5$$

简单的 1-3-1 神经网络结构，如图-5，即可拟合复杂多变的数据如图-6

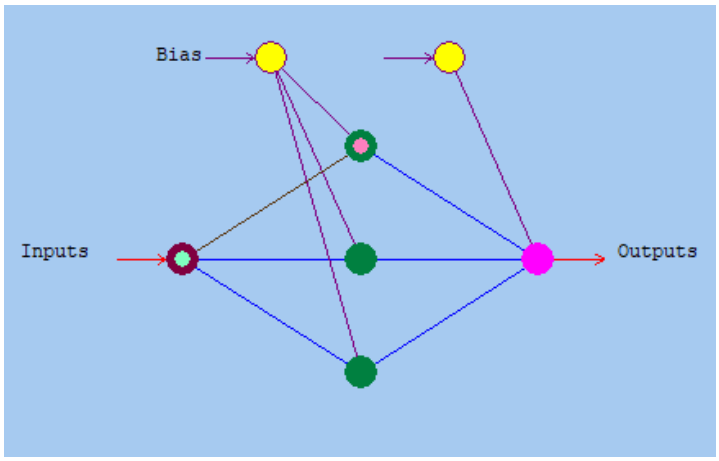

图-5. 1-3-1 神经网络结构

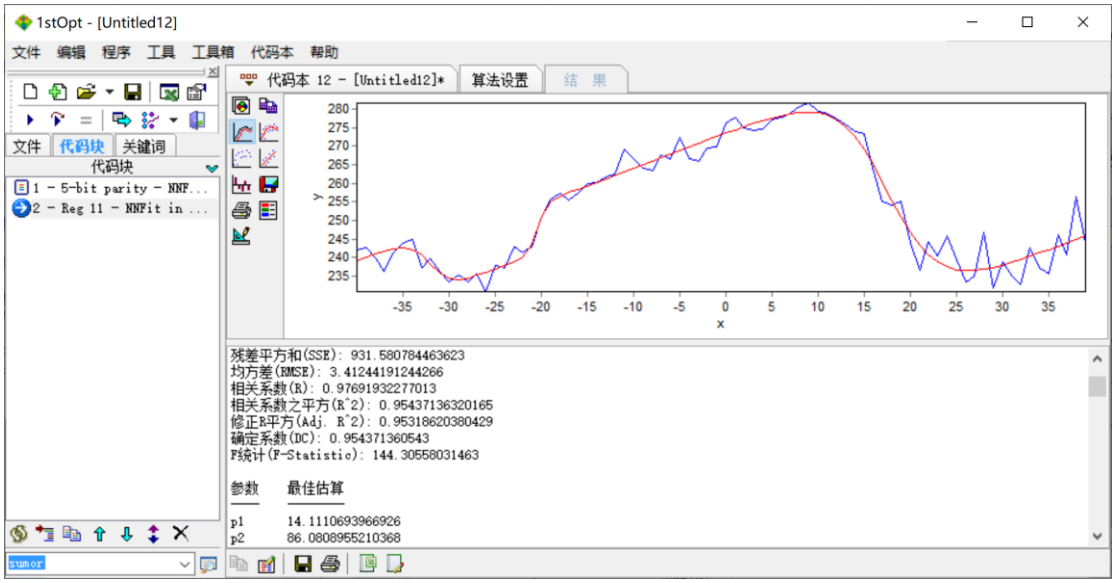

图-6. 复杂数据拟合

2. 拟合计算可输出参数标准差及置信度；

拟合计算时可选择是否输出参数标准差及置信度，即可通过设置面板设置（如图-7），也可通过在代码本添加命令代码如“ConfidenceLevel = 0.95;”达到同样的效果。该功能的实现避免了之前版本当需要置信度数据时用户不得不再用第三方软件重新计算的繁琐。

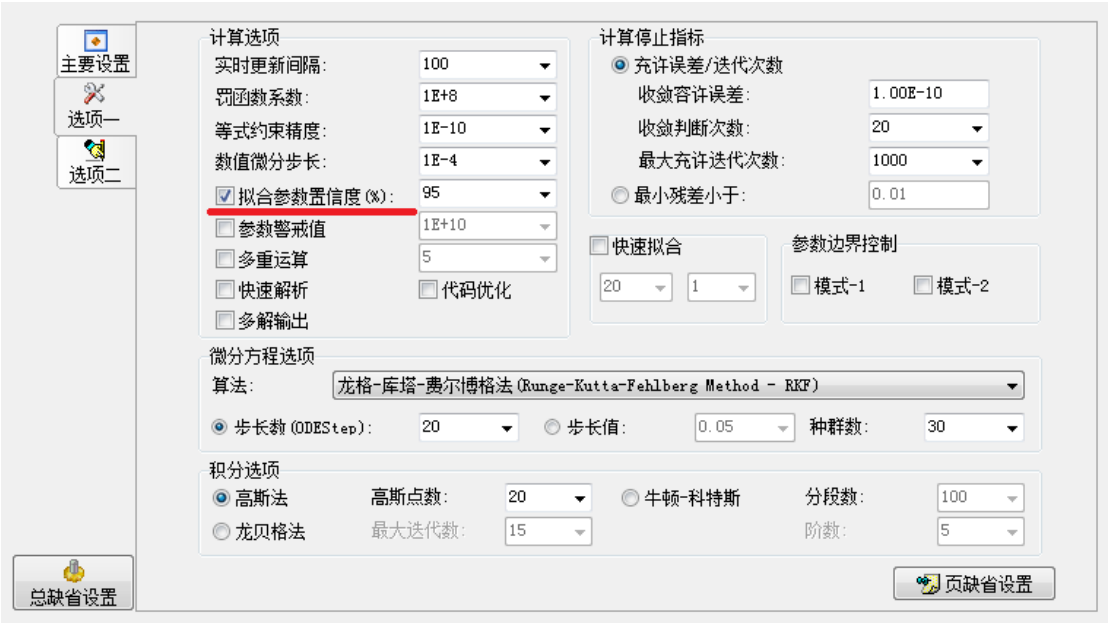

图-7. 拟合置信度设置面板

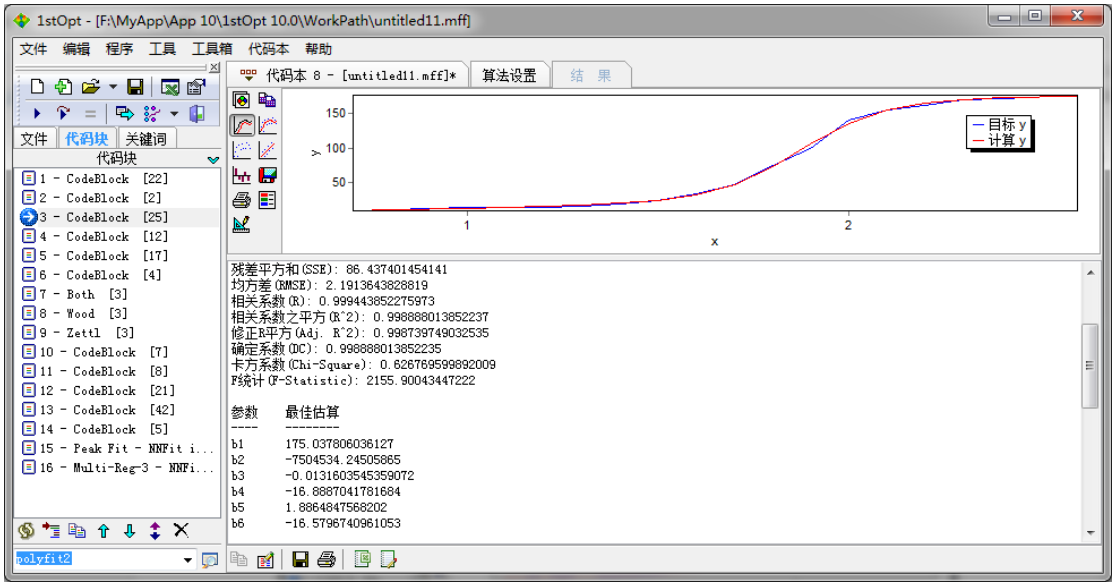

图-8. 无置信度结果输出

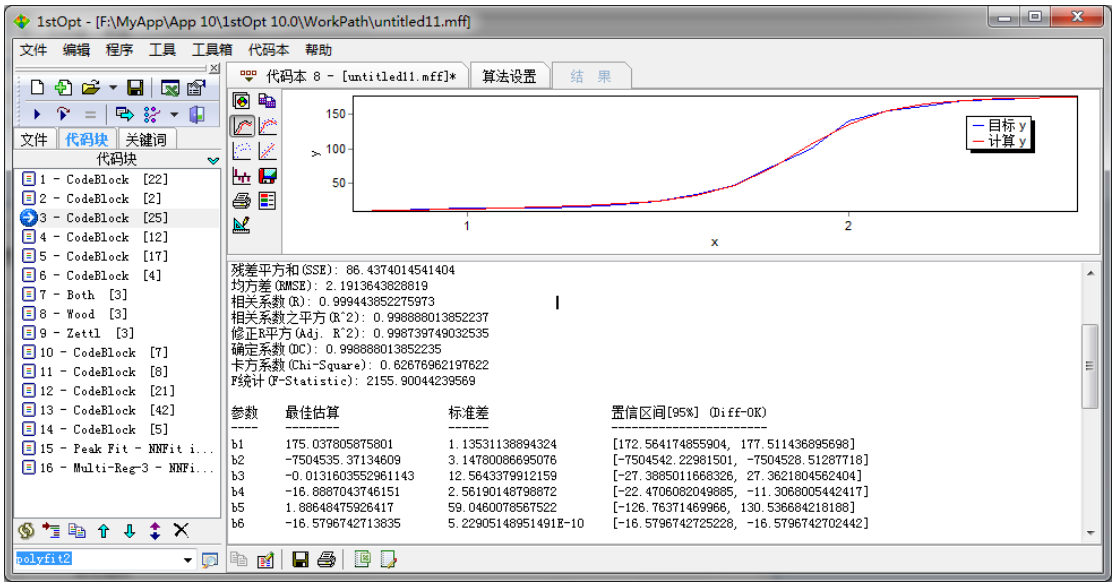

图-9. 有置信度结果输出

3. 可定义三维、四维及五维参数（Parameter）、传递参数（PassParameter）及常数；

之前版本不论是参数（Parameter）、传递参数（PassParameter）或常数（Constant），最多只能定义二维，新版本最高可定义至五维，应用场景及范围显著扩大。如排班规划问题，因为涉及到三维数组参数，老版本将无力处理，而新版本则可轻松应对。

```
Constant D(7,3)=[4,4,3,
                 3,3,2,
                 3,3,2,
```

```
                3,2,3,
                4,3,3,
                2,2,1,
                3,2,2];
BinParameter x(12,7,3);//工人 i 被安排到第 j 天第 k 个班次
Algorithm = LP;
PassParameter w=Sum(i=1:12)(Sum(j=1:7)(x[i,j,1]));
MinFunction Sum(i=1:12)(Sum(j=1:7)(x[i,j,1]));
        For(i=1:12)(For(j=1:7)(Sum(k=1:3)(x[i,j,k])<=1));//每人每天最多工作一个班次
        For(i=1:12)(For(j=1:6)(x[i,j,3]+x[i,j+1,1]<=1)); //每人在任何时候都不能连续两个班次工作
        For(i=1:12)(x[i,7,3]+x[i,1,1]<=1);
        For(i=1:12)(Sum(j=1:7)(Sum(k=1:3)(x[i,j,k]))<=5);//每人每周至少休息两天
        for(j=1:7)(for(k=1:3)(Sum(i=1:12)(x[i,j,k])>=d[j,k]));//每个班次人数满足需求
```

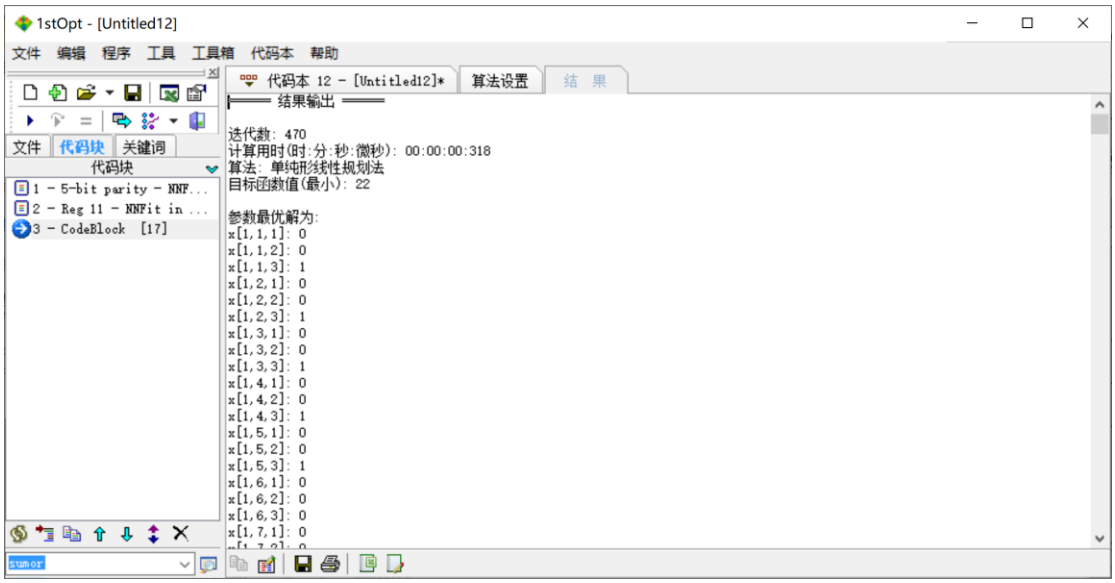

图-10. 三维数组计算结果

4. Latex 及 MathXL 公式支持

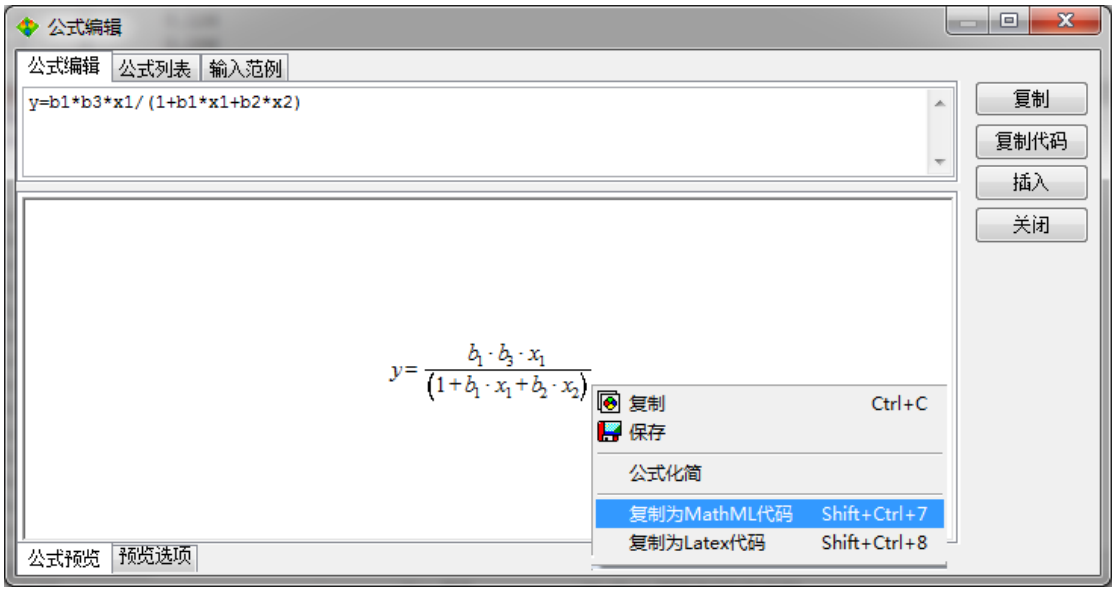

图-11. Latex 及 MathXL 公式格式输出

支持公式代码直接转换成 Latex 代码及 MathXL 格式。

例：拟合公式代码：y=b1*b3*x1/(1+b1*x1+b2*x2)

1）直接生成 Latex 代码：

{y=\frac{b_{1} \cdot b_{3} \cdot x_{1}}{1+b_{1} \cdot x_{1}+b_{2} \cdot x_{2}}}

该代码生成的 Latex 公式形式如下：

$$y=\frac{b_{1} \cdot b_{3} \cdot x_{1}}{1+b_{1} \cdot x_{1}+b_{2} \cdot x_{2}}$$

2）直接生成 MathXL 格式

该代码可直接复制黏贴至 Word 文档，成为即时可编辑的 Word 公式形式

$$y=\frac{b_{1} \cdot b_{3} \cdot x_{1}}{1+b_{1} \cdot x_{1}+b_{2} \cdot x_{2}}$$

5. 拟合预测可以计算曲线一阶至五阶导数值；

拟合计算完成后，可查看并输出最高至 5 解的拟合函数导数值，还可输出各点弧微分、曲率、曲线长度及对应面积值。

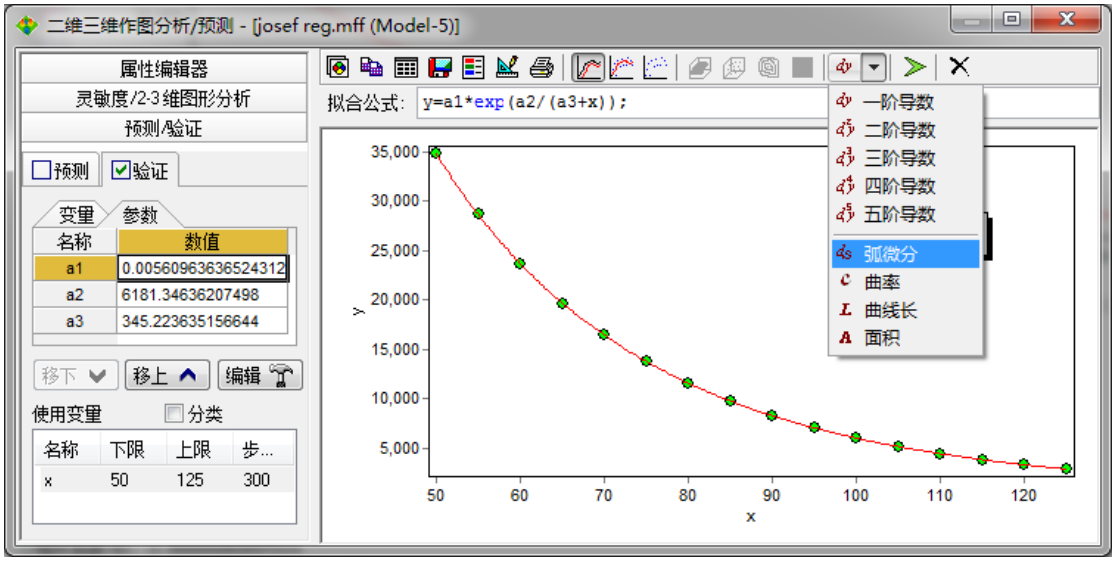

图-12. 拟合函数可输出最高至 5 阶导数值

6. “SharingPar”关键字

批量数据拟合时（BatchFileModel）或循环常数计算时（LoopConstant），后一组数据或后一次循环计算的拟合参数初值来自上一组数据拟合或循环计算上

一次得到的最优值，或前面已完成的各组数据最优参数值的平均值，可以提高批量数据的整体拟合精度并缩短计算时间。

1）SharingPar = 0，无作用；

2）SharingPar = 1，前一最优解作为后一的参数初值，对所有参数，无论该参数最初是否设定有初值；

3）SharingPar = 2，对没有设定初值的参数，前一最优解作为后一的参数初值，对最初已设定了初值得参数，仍使用最初设定的参数；

4）SharingPar = 3，前面所有计算参数最优解的平均值作为后一的参数初值，对所有参数，无论该参数最初是否设定有初值；

5）SharingPar = 4，对没有设定初值的参数，前面所有组参数最优解的平均值作为后一组的参数初值，对最初已设定了初值得参数，仍使用最初设定的参数；

7. 高价微分方程简写

1stOpt 描述微分方程阶数时用“’”号表示，一阶用一个“’”，二阶用两个“’”，以此类推，阶数更高时书写不便，同时也容易出错，如：y’’’’’=3+x*y; 可以简写成: y(6’)=3+x*y; 当阶数比较高时方便书写同时减少出错；

例：12 阶微分方程如下：

$$\frac{d^{12}u}{dx^{12}} = 2 \cdot \exp(x) \cdot \frac{d^2u}{dx^2} + \frac{d^3u}{dx^3}$$

边界条件：

$$\begin{cases} u^{2\cdot k}(0) = 1, k = 0,1,2,3,4,5 \\ u^{2\cdot k}(1) = \dfrac{1}{e}, k = 0,1,2,3,4,5 \end{cases}$$

Code 1 (原写法)	Constant e=exp(1); Variable x=[0,1],u=[1,1/e],u''=[1,1/e],u''''=[1,1/e],u''''''=[1,1/e],u''''''''=[1,1/e],u''''''''''=[1,1/e]; Plot x[x],u,u',u'',u'''; ODEFunction u’’’’’’’’’’=2*exp(x)*u''+u''';
Code 2 (现写法)	Constant e=exp(1); Variable x=[0,1],u=[1,1/e],u(2')=[1,1/e],u(4')=[1,1/e],u(6')=[1,1/e],u(8')=[1,1/e],u(10')=[1,1/e]; Plot x[x],u,u',u'',u'''; ODEFunction u(12')=2*exp(x)*u''+u''';

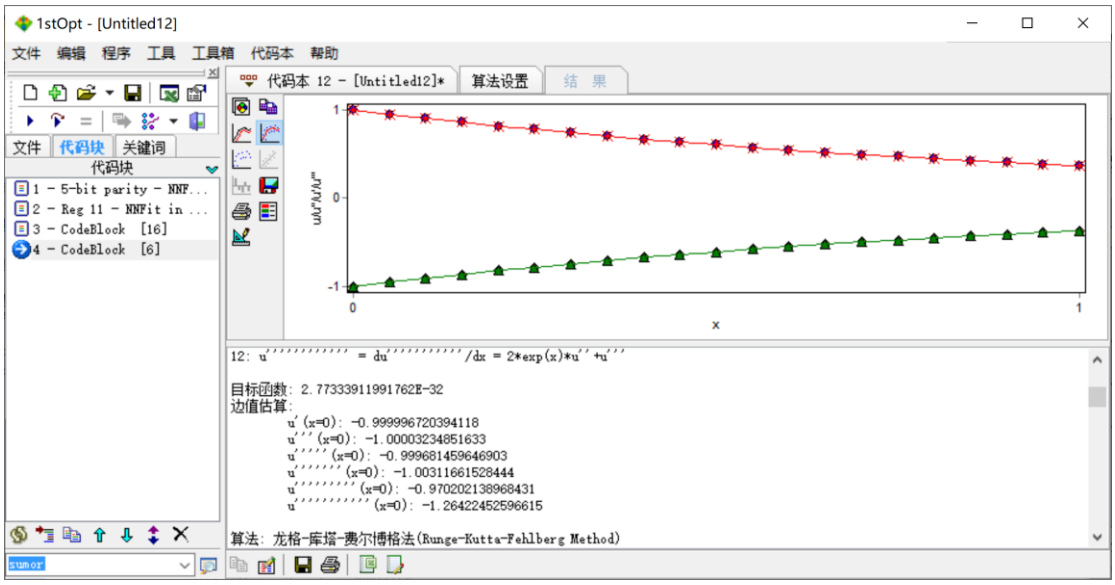

图-13. 高阶微分方程边值问题求解

8. 增加“SumAnd”和“SumOr”函数；

快捷模式下 if 语句连续多个“and”或“or”判断时的简写方式

原写法：

```
Constant n=15;                                                    //天数
Constant a=[25,16,24,17,29,25,22,28,21,17,28,14,12,25,11];        //人数
Constant p=[208,212,213,217,206,203,212,214,200,204,208,209,219,216,214]; //费用
IntParameter x(n);
Algorithm = LP;
MinFunction Sum(i=1:n,x)(x*p[i]);
For(i=1:n)(Sum(j=1:n)(if((j<>Wrap(i+1,n))and(j<>Wrap(i+2,n))and(j<>Wrap(i+3,n))and(j<>Wrap(i+4,n))and(j<>
Wrap(i+5,n)),x[j]))>=a[i]);
```

简化写法：

```
Constant n=15;                                                    //天数
Constant a=[25,16,24,17,29,25,22,28,21,17,28,14,12,25,11];        //人数
Constant p=[208,212,213,217,206,203,212,214,200,204,208,209,219,216,214]; //费用
IntParameter x(n);
Algorithm = LP;
MinFunction Sum(i=1:n,x)(x*p[i]);
        For(i=1:n)(Sum(j=1:n)(if(SumAnd(k=1:5)(j<>Wrap(i+k,n)),x[j]))>=a[i]);
```

9. 批量定义范围：LowBound，UpBound

可以批量定义待求参数的上下范围。

```
Parameter x1=[-2.3,2.3], x2=[-2.3,2.3], x3=[-3.2,3.2], x4=[-3.2,3.2], x5=[-3.2,3.2];
```

```
MinFunction exp(Prod(i=1:5)(x[i]));
            Sum(i=1:5)(x[i]^2)-10=0;
            x2*x3-5*x4*x5=0;
            x1^3+x2^3+1=0;
```

上述代码也可写成如下：

```
Parameter x(5);
LowBound = -[2.3,2.3,3.2,3.2,3.2];
UpBound = [2.3,2.3,3.2,3.2,3.2];
MinFunction exp(Prod(i=1:5)(x[i]));//+a1+a2;
            Sum(i=1:5)(x[i]^2)-10=0;
            x2*x3-5*x4*x5=0;
            x1^3+x2^3+1=0;
```

均可得到相同的结果

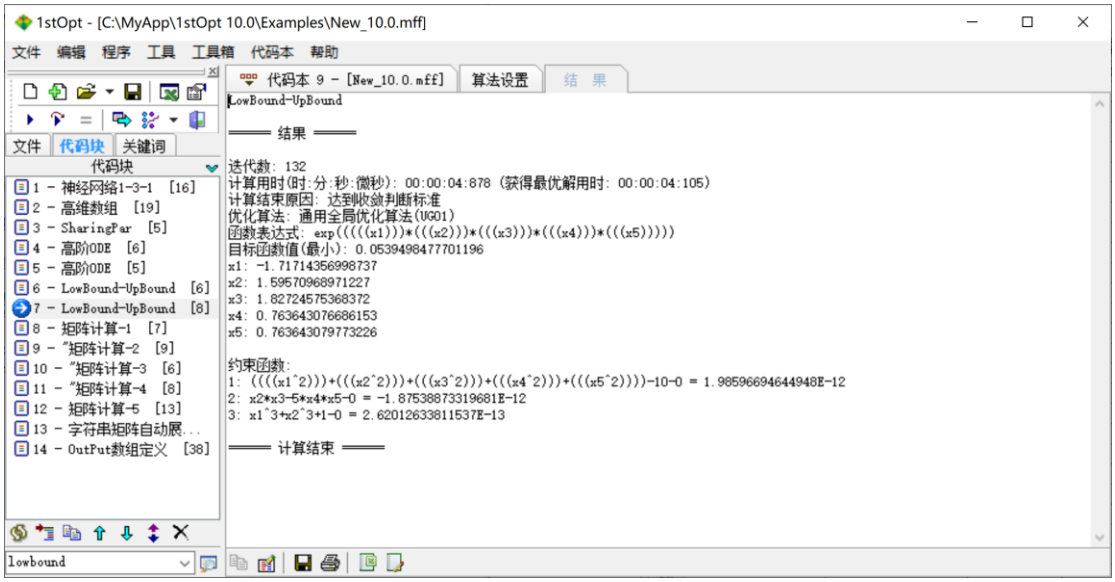

图-14. “LowBound”与“UpBound”案例计算结果

10. “?”定义 Sheet 或 CodeSheet 范围，可自动获取数据范围

从表格读取数据时，可用“?”代表数据下限范围，程序可以自动搜索判断数据范围，这样即使表格数据范围有所改变，代码本中的代码也不用改动，方便使用。

如：DataFile "CodeSheet1[C3:D38]";

可以写成：DataFile "CodeSheet1[C3:D?]";

11. 矩阵计算功能强化

矩阵相加：AddMat()；

矩阵相减：SubMat()；

矩阵相乘：MultMat()；

矩阵求逆：InvMat()；

矩阵转置：TransMat()；

矩阵表达式展开：MatrixStr()；

矩阵操作：Matrix()；

例：矩阵方程如下，试求解该方程组

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1.1 & 1.1^2 & 1.1^3 & 1.1^4 & 1.1^5 & 1.1^6 & 1.1^7 & 1.1^8 & 1.1^9 & 1.1^{10} \\ 1 & 1.2 & 1.2^2 & 1.2^3 & 1.2^4 & 1.2^5 & 1.2^6 & 1.2^7 & 1.2^8 & 1.2^9 & 1.2^{10} \\ 1 & 1.3 & 1.3^2 & 1.3^3 & 1.3^4 & 1.3^5 & 1.3^6 & 1.3^7 & 1.3^8 & 1.3^9 & 1.3^{10} \\ 1 & 1.4 & 1.4^2 & 1.4^3 & 1.4^4 & 1.4^5 & 1.4^6 & 1.4^7 & 1.4^8 & 1.4^9 & 1.4^{10} \\ 1 & 1.5 & 1.5^2 & 1.5^3 & 1.5^4 & 1.5^5 & 1.5^6 & 1.5^7 & 1.5^8 & 1.5^9 & 1.5^{10} \\ 1 & 1.6 & 1.6^2 & 1.6^3 & 1.6^4 & 1.6^5 & 1.6^6 & 1.6^7 & 1.6^8 & 1.6^9 & 1.6^{10} \\ 1 & 1.7 & 1.7^2 & 1.7^3 & 1.7^4 & 1.7^5 & 1.7^6 & 1.7^7 & 1.7^8 & 1.7^9 & 1.7^{10} \\ 1 & 1.8 & 1.8^2 & 1.8^3 & 1.8^4 & 1.8^5 & 1.8^6 & 1.8^7 & 1.8^8 & 1.8^9 & 1.8^{10} \\ 1 & 1.9 & 1.9^2 & 1.9^3 & 1.9^4 & 1.9^5 & 1.9^6 & 1.9^7 & 1.9^8 & 1.9^9 & 1.9^{10} \\ 1 & 2.0 & 2.0^2 & 2.0^3 & 2.0^4 & 2.0^5 & 2.0^6 & 2.0^7 & 2.0^8 & 2.0^9 & 2.0^{10} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \pi \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

矩阵解法-1	Constant c(11,11)=[1,1,1,1,1,1,1,1,1,1,1,
	1,1.1,1.1^2,1.1^3,1.1^4,1.1^5,1.1^6,1.1^7,1.1^8,1.1^9,1.1^10,
	1,1.2,1.2^2,1.2^3,1.2^4,1.2^5,1.2^6,1.2^7,1.2^8,1.2^9,1.2^10,
	1,1.3,1.3^2,1.3^3,1.3^4,1.3^5,1.3^6,1.3^7,1.3^8,1.3^9,1.3^10,
	1,1.4,1.4^2,1.4^3,1.4^4,1.4^5,1.4^6,1.4^7,1.4^8,1.4^9,1.4^10,
	1,1.5,1.5^2,1.5^3,1.5^4,1.5^5,1.5^6,1.5^7,1.5^8,1.5^9,1.5^10,
	1,1.6,1.6^2,1.6^3,1.6^4,1.6^5,1.6^6,1.6^7,1.6^8,1.6^9,1.6^10,
	1,1.7,1.7^2,1.7^3,1.7^4,1.7^5,1.7^6,1.7^7,1.7^8,1.7^9,1.7^10,
	1,1.8,1.8^2,1.8^3,1.8^4,1.8^5,1.8^6,1.8^7,1.8^8,1.8^9,1.8^10,
	1,1.9,1.9^2,1.9^3,1.9^4,1.9^5,1.9^6,1.9^7,1.9^8,1.9^9,1.9^10,
	1,2.0,2.0^2,2.0^3,2.0^4,2.0^5,2.0^6,2.0^7,2.0^8,2.0^9,2.0^10];
	Constant p(11,1)=[0,0,0,0,0,pi,0,0,0,0,0];
	d=InvMat(c);
	x=MultMat(d,p);

矩阵解法-2	Constant c(11,11)=[1,1,1,1,1,1,1,1,1,1,1, 1,1.1,1.1^2,1.1^3,1.1^4,1.1^5,1.1^6,1.1^7,1.1^8,1.1^9,1.1^10, 1,1.2,1.2^2,1.2^3,1.2^4,1.2^5,1.2^6,1.2^7,1.2^8,1.2^9,1.2^10, 1,1.3,1.3^2,1.3^3,1.3^4,1.3^5,1.3^6,1.3^7,1.3^8,1.3^9,1.3^10, 1,1.4,1.4^2,1.4^3,1.4^4,1.4^5,1.4^6,1.4^7,1.4^8,1.4^9,1.4^10, 1,1.5,1.5^2,1.5^3,1.5^4,1.5^5,1.5^6,1.5^7,1.5^8,1.5^9,1.5^10, 1,1.6,1.6^2,1.6^3,1.6^4,1.6^5,1.6^6,1.6^7,1.6^8,1.6^9,1.6^10, 1,1.7,1.7^2,1.7^3,1.7^4,1.7^5,1.7^6,1.7^7,1.7^8,1.7^9,1.7^10, 1,1.8,1.8^2,1.8^3,1.8^4,1.8^5,1.8^6,1.8^7,1.8^8,1.8^9,1.8^10, 1,1.9,1.9^2,1.9^3,1.9^4,1.9^5,1.9^6,1.9^7,1.9^8,1.9^9,1.9^10, 1,2.0,2.0^2,2.0^3,2.0^4,2.0^5,2.0^6,2.0^7,2.0^8,2.0^9,2.0^10];

	Constant p(11,1)=[0,0,0,0,0,pi,0,0,0,0,0]; x=Matrix(c^Inv*p);
优化方法-1	Algorithm = SM2; Constant c(0:10,0:10)=[1,1,1,1,1,1,1,1,1,1,1, 1,1.1,1.1^2,1.1^3,1.1^4,1.1^5,1.1^6,1.1^7,1.1^8,1.1^9,1.1^10, 1,1.2,1.2^2,1.2^3,1.2^4,1.2^5,1.2^6,1.2^7,1.2^8,1.2^9,1.2^10, 1,1.3,1.3^2,1.3^3,1.3^4,1.3^5,1.3^6,1.3^7,1.3^8,1.3^9,1.3^10, 1,1.4,1.4^2,1.4^3,1.4^4,1.4^5,1.4^6,1.4^7,1.4^8,1.4^9,1.4^10, 1,1.5,1.5^2,1.5^3,1.5^4,1.5^5,1.5^6,1.5^7,1.5^8,1.5^9,1.5^10, 1,1.6,1.6^2,1.6^3,1.6^4,1.6^5,1.6^6,1.6^7,1.6^8,1.6^9,1.6^10, 1,1.7,1.7^2,1.7^3,1.7^4,1.7^5,1.7^6,1.7^7,1.7^8,1.7^9,1.7^10, 1,1.8,1.8^2,1.8^3,1.8^4,1.8^5,1.8^6,1.8^7,1.8^8,1.8^9,1.8^10, 1,1.9,1.9^2,1.9^3,1.9^4,1.9^5,1.9^6,1.9^7,1.9^8,1.9^9,1.9^10, 1,2.0,2.0^2,2.0^3,2.0^4,2.0^5,2.0^6,2.0^7,2.0^8,2.0^9,2.0^10]; Constant p(0:10)=[0,0,0,0,0,pi,0,0,0,0,0]; Parameter x(0:10); Function For(i=0:10)(Sum(j=0:10)(x[j]*c[i,j])=p[i]);
优化方法-2	Algorithm = SM2; Function For(i=0:10)(Sum(j=0:10)(x[j]*(1+i*0.1)^j)=if(i=5,pi,0));

上述四种方法均可得到相同的结果，矩阵解法-2 最方便快捷。

12. 复数拟合功能的改进和增强；

13. “MDataSet”功能改进；

14. “ParVariable”可以定义多个缺失变量；

15. 众多细节改进及 Bug 修复