

# Performance of Differential Evolution Method in Least Squares Fitting of Some Typical Nonlinear Curves

SK Mishra  
Department of Economics  
North-Eastern Hill University  
Shillong (India)

**I. Introduction:** Curve fitting or fitting a statistical/mathematical model to data finds its application in almost all empirical sciences - viz. physics, chemistry, zoology, botany, environmental sciences, economics, etc. It has four objectives: the first, to describe the observed (or experimentally obtained) dataset by a statistical/mathematical formula; the second, to estimate the parameters of the formula so obtained and interpret them so that the interpretation is consistent with the generally accepted principles of the discipline concerned; the third, to predict, interpolate or extrapolate the expected values of the dependent variable with the estimated formula; and the last, to use the formula for designing, controlling or planning. There are many principles of curve fitting: the Least Squares (of errors), the Least Absolute Errors, the Maximum Likelihood, the Generalized Method of Moments and so on.

The principle of Least Squares (method of curve fitting) lies in minimizing the sum of squared errors,  $s^2 = \sum_{i=1}^n [y_i - g(x_i, b)]^2$ , where  $y_i (i=1, 2, \dots, n)$  is the observed value of dependent variable and  $x_i = (x_{i1}, x_{i2}, \dots, x_{im})$ ;  $i=1, 2, \dots, n$  is a vector of values of independent (explanatory or predictor) variables. As a problem the dataset,  $(y, x)$ , is given and the parameters  $(b_k; k=1, 2, \dots, p)$  are unknown. Note that  $m$  (the number of independent variables,  $x_j; j=1, 2, \dots, m$ ) and  $p$  (the number of parameters) need not be equal. However, the number of observations ( $n$ ) almost always exceeds the number of parameters ( $p$ ). The system of equations so presented is inconsistent such as not to permit  $s^2$  to be zero; it must always be a positive value. In case  $s^2$  may take on a zero value, the problem no longer belongs to the realm of statistics; it is a purely mathematical problem of solving a system of equations. However, the method of the Least Squares continues to be applicable to this case too. It is also applicable to the cases where  $n$  does not exceed  $p$ .

Take for example two simple cases; the first of two (linear and consistent) equations in two unknowns; and the second of three (linear and consistent) equations in two unknowns, presented in the matrix form as  $y = Xb + u$ :

$$\begin{bmatrix} 10 \\ 26 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 10 \\ 26 \\ 14 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 5 \\ -1 & 4 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

Since  $y = Xb + u$ , it follows that  $X^{-s}(y - u) = b$ . Here  $X^{-s}$  the generalized inverse of  $X$  (Rao and Mitra, 1971). Further, since  $X^{-s} = (X'X)^{-1}X'$  (such that  $(X'X)^{-1}X'X = I$ , an identity matrix), it follows that  $b = (X'X)^{-1}X'y - (X'X)^{-1}X'u$ . Now, if  $X'u = 0$ , we have  $b = (X'X)^{-1}X'y$ . For the first system of equations given above, we have

$$X'X = \begin{bmatrix} 1 & 3 \\ 2 & 5 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix} = \begin{bmatrix} 10 & 17 \\ 17 & 29 \end{bmatrix}; (X'X)^{-1} = \frac{1}{290-289} \begin{bmatrix} 29 & -17 \\ -17 & 10 \end{bmatrix} = \begin{bmatrix} 29 & -17 \\ -17 & 10 \end{bmatrix}; (X'X)^{-1}X' = \begin{bmatrix} -5 & 2 \\ 3 & -1 \end{bmatrix}$$

$$(X'X)^{-1}X'y = \begin{bmatrix} -5 & 2 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} 10 \\ 26 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

This solution is identical to the one obtained if we would have solved the first system of equations by any algebraic method (assuming  $u_i = 0 \forall i$ ).

Similarly, for the second system of equations, we have

$$X'X = \begin{bmatrix} 1 & 3 & -1 \\ 2 & 5 & 4 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 5 \\ -1 & 4 \end{bmatrix} = \begin{bmatrix} 11 & 13 \\ 13 & 45 \end{bmatrix}; (X'X)^{-1} = \frac{1}{326} \begin{bmatrix} 45 & -13 \\ -13 & 11 \end{bmatrix}; (X'X)^{-1}X' = \frac{1}{326} \begin{bmatrix} 19 & 70 & -97 \\ 9 & 16 & 57 \end{bmatrix}$$

$$(X'X)^{-1}X'y = \frac{1}{326} \begin{bmatrix} 19 & 70 & -97 \\ 9 & 16 & 57 \end{bmatrix} \begin{bmatrix} 10 \\ 26 \\ 14 \end{bmatrix} = \frac{1}{326} \begin{bmatrix} 652 \\ 1304 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

This solution is identical to any solution that we would have obtained by solving any combination of two equations (taken from the three equations). This is so since the three equations are mutually consistent.

Now, let us look at the problem slightly differently. In the system of equations that we have at hand (i.e.  $y-u = Xb$ ), the Jacobian (J, or the matrix of the first partial derivatives of  $y_i$  with respect to  $b_j$ ) is X. Or,

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix} = \begin{bmatrix} \partial y_1 / \partial b_1 & \partial y_1 / \partial b_2 & \dots & \partial y_1 / \partial b_m \\ \partial y_2 / \partial b_1 & \partial y_2 / \partial b_2 & \dots & \partial y_2 / \partial b_m \\ \dots & \dots & \dots & \dots \\ \partial y_n / \partial b_1 & \partial y_n / \partial b_2 & \dots & \partial y_n / \partial b_m \end{bmatrix} = J$$

Thus,  $b = (X'X)^{-1}X'y$  may be considered as  $(J'J)^{-1}J'y$ . In a system of linear equations J (the Jacobian, or the matrix of  $\partial y_i / \partial b_j \forall i, j$ ) is constant. However, if the system is nonlinear (in parameters), the J matrix varies in accordance with the value of  $b_j$  at which  $y_i$  is evaluated. This fact immediately leads us to the Gauss-Newton method (of nonlinear Least Squares). This method is an iterative method and may be described as follows.

Take any arbitrary value of  $b$ ,  $b_{(0)} = (b_{(0)1}, b_{(0)2}, \dots, b_{(0)p})$  and find  $J_{(0)}$  at that. Also, evaluate the equations at  $b_{(0)}$  to obtain  $y_{(0)i}; \forall i$ . This  $y_{(0)}$  will (almost always) be different from the  $y$  given in the dataset. Now, find  $\Delta b = (J'_{(0)}J_{(0)})^{-1}J'_{(0)}(y - y_{(0)})$ . Obtain the next approximation of  $b$  as  $b_{(1)} = b_{(0)} + \Delta b$ . Evaluate the equations at  $b_{(1)}$  to obtain  $y_{(1)}$  and also find  $J_{(1)}$  at  $b_{(1)}$ . As before, find  $\Delta b = (J'_{(1)}J_{(1)})^{-1}J'_{(1)}(y - y_{(1)})$ . Then, obtain  $b_{(2)} = b_{(1)} + \Delta b$ . And continue until  $\Delta b$  is negligibly small. The fixed point theorem guarantees convergence. Thus we obtain the estimated parameters,  $\hat{b}$ . Note that an approximate value of the first derivative (elements of the Jacobian matrix) of a function  $\varphi(b)$  at any point  $b_a$  may be obtained numerically as

$\left[ \frac{\partial \varphi}{\partial b} \right]_{b_a} \approx \frac{[\varphi(b_{a+\Delta}) - \varphi(b_{a-\Delta})]}{(b_{a+\Delta} - b_{a-\Delta})}$ . For example, the first derivative of  $\phi(v) = 2v^2 + 5v + 3$  at  $v = 2$  may be obtained as  $[\phi(v = 2 + 1) - \phi(v = 2 - 1)] / [2 + 1 - (2 - 1)]$  which is  $[(18 + 15 + 3) - (2 + 5 + 3)] / (3 -$

1) =  $[36 - 10]/2 = 13$ , which is equal to  $\partial\phi/\partial v = 4v + 5$  evaluated at  $v = 2$ . Note that although in this example we obtain the exact value of the first derivative, we would obtain, in general, only an approximate value.

The Gauss-Newton method is very powerful, but it fails to work when the problem is ill conditioned or multi-modal. Hence, many methods have been developed to deal with difficult, ill conditioned or multimodal problems. Levenberg (1944) and later Marquardt (1963) replaced the fundamental Gauss-Newton's  $\Delta b = (J'_{(k)} J'_{(k)})^{-1} J'_{(k)} (y - y_{(k)})$  by  $\Delta b = (J'_{(k)} J'_{(k)} + \lambda I)^{-1} J'_{(k)} (y - y_{(k)})$ , where  $\lambda$  is the dampening factor. Similarly, Tikhonov (1943, 1963; Tikhonov and Arsenin, 1977) modified the Gaussian formula for ill conditioned linear regression, which finally led to the Ridge Regression by Hoerl (1962) and Hoerl and Kennard (1970) wherein  $\hat{b} = (X'X + \lambda I)^{-1} X'y$ .

It may be noted that a nonlinear least squares problem is fundamentally a problem in optimization of nonlinear functions. Initially optimization of nonlinear functions was methodologically based on the Lagrange-Leibniz-Newton principles and therefore could not easily escape local optima. Hence, its development to deal with nonconvex (multimodal) functions stagnated until the mid 1950's. Stanislaw Ulam, John von Neumann and Nicolas Metropolis had in the late 1940's proposed the Monte Carlo method of simulation (Metropolis, 1987; Metropolis et al. 1953) and it was gradually realized that the simulation approach could provide an alternative methodology to mathematical investigations in optimization. George Box (1957) was perhaps the first mathematician who exploited the idea and developed his evolutionary method of nonlinear optimization. Almost a decade later, John Nelder and Roger Mead (1964) developed their simplex method and incorporated in it the ability to learn from its earlier search experience and adapt itself to the topography of the surface of the optimand function. MJ Box (1965) developed his complex method, which strews random numbers over the entire domain of the decision variables and therefore has a great potentiality to escape local optima and locate the global optimum of a nonlinear function. These methods may be applied to nonlinear curve fitting problem (Mishra, 2006), but unfortunately such applications have been only few and far between.

The simulation-based optimization became a hotbed of research due to the invention of the 'genetic algorithm' by John Holland (1975). A number of other methods of global optimization were soon developed. Among them, the 'Clustering Method' of Aimo Törn (1978, Törn & Viitanen, 1994), the "Simulated Annealing Method" of Kirkpatrick and others (1983) and Cerny (1985), "Tabu Search Method" of Fred Glover (1986), the "Particle Swarm Method" of Kennedy and Eberhart (1995) and the "Differential Evolution Method" of Storn and Price (1995) are quite effective. All these methods use the one or the other stochastic process to search the global optima. On account of the ability of these methods to search optimal solutions of quite difficult nonlinear functions, they provide a great scope to deal with the nonlinear curve fitting problems. These methods supplement other mathematical methods used to this end.

**II. The Differential Evolution Method of Optimization:** The method of Differential Evolution (DE) was developed by Price and Storn in an attempt to solve the Chebychev

polynomial fitting problem. The crucial idea behind DE is a scheme for generating trial parameter vectors. Initially, a population of points ( $p$  in  $m$ -dimensional space) is generated and evaluated (i.e.  $f(p)$  is obtained) for their fitness. Then for each point ( $p_i$ ) three different points ( $p_a$ ,  $p_b$  and  $p_c$ ) are randomly chosen from the population. A new point ( $p_z$ ) is constructed from those three points by adding the weighted difference between two points ( $w(p_b-p_c)$ ) to the third point ( $p_a$ ). Then this new point ( $p_z$ ) is subjected to a crossover with the current point ( $p_i$ ) with a probability of crossover ( $c_r$ ), yielding a candidate point, say  $p_u$ . This point,  $p_u$ , is evaluated and if found better than  $p_i$  then it replaces  $p_i$  else  $p_i$  remains. Thus we obtain a new vector in which all points are either better than or as good as the current points. This new vector is used for the next iteration. This process makes the differential evolution scheme completely self-organizing.

**III. Objectives of the Present Work:** The objective of the present work is to evaluate the performance of the Differential Evolution at nonlinear curve fitting. For this purpose, we have collected problems - models and datasets - mostly from two main sources; the first from the website of NIST [National Institute of Standards and Technology, US Department of Commerce, USA at [http://www.itl.nist.gov/div898/strd/nls/nls\\_main.shtml](http://www.itl.nist.gov/div898/strd/nls/nls_main.shtml)] and the second, the website of the CPC-X Software (makers of the AUTO2FIT Software at <http://www.geocities.com/neuralpower> now new website at [www.7d-soft.com](http://www.7d-soft.com)). *In this paper we will use 'CPC-X' and 'AUTO2FIT' interchangeably.* Some models (and datasets) have been obtained from other sources also.

According to the level of difficulty, the problems may be classified into four categories: (1) Lower, (2) Average, (3) Higher, and (4) Extra Hard. The list of problems (dealt with in the present study) so categorized is given below:

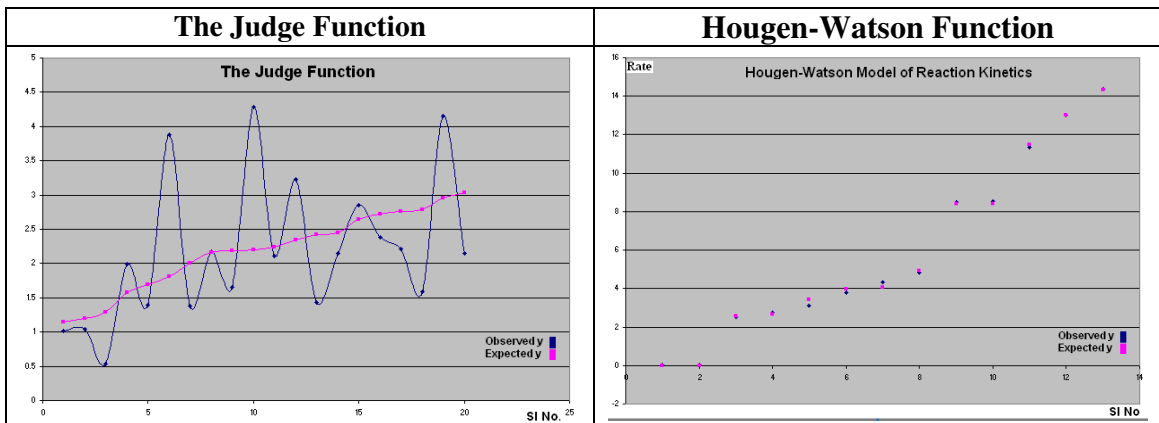
<b>Difficulty level</b>	<b>Problem Names</b>	<b>Source of Problem</b>	<b>Classified by</b>
Lower	Chwirut, Gauss-1, Gauss-2, Lanczos-3	NIST	NIST
	Judge	Goffe	Author
	Mount, Sin-Cos, Cos-Sin	CPC-X	Author
Average	ENSO, Gauss-3, Hahn, Kirby, Lanczos-1, Lanczos-2, MGH-17, Misra-1(c), Misra-1(d), Nelson, Roszman	NIST	NIST
Higher	Bennett, BoxBOD, Eckerle, MGH-09, MGH-10, Ratkowsky-42, Ratkowsky-43, Thurber	NIST	NIST
	Hougen	Mathworks.com	Author
	Multi-output	CPC-X	Author
Extra Hard	CPC-X problems (all 9 challenge functions)	CPC-X	CPC-X

It may be noted that the difficulty level of a Least Squares curve fitting problem depends on: (i) the (statistical) model, (ii) the dataset, (iii) the algorithm used for optimization, and (iv) the guessed range (or the starting points of search) of parameters. For the same model and the optimization algorithm starting at the same point, two different datasets may present different levels of difficulty. Similarly, a particular problem might be simple for the one algorithm but very difficult for the others and so on.

Again, different algorithms have different abilities to combine their explorative and exploitative functions while searching for an optimum solution. Those with better exploitative abilities converge faster but are easily caught into the local optimum trap. They are also very sensitive to the (guessed) starting points. The algorithms that have excellent explorative power often do not converge fast. Therefore, in fitting a nonlinear function to a dataset, there's many a slip between cup and lip.

**IV. The Findings:** In what follows, we present our findings on the performance of the Differential Evolution method at optimization of the Least Squares problems. The datasets and the models are available at the source (NIST, CPC-X Software, Mathworks, Goffe's SIMANN). In case of any model, the function has been fitted to the related data and the estimated values,  $\hat{y}$ , of the predicted variable (y or the dependent variable) has been obtained. The expected values ( $\hat{y}$ ) have been arranged in an ascending order and against the serial number so obtained the expected  $\hat{y}$  and observed y have been plotted. The purpose is to highlight the discrepancies between the observed and the expected values of y. The goodness of fit of a function to a dataset may be summarily judged by  $R^2$  (that always lies between 0 and 1),  $s^2$  or RMS. These values (along with the certified values) have been presented to compare the performance of the Differential Evolution.

**1. The Judge's Function:** This function is given in Judge et al (1990). Along with the associated data it is a rather simple example of nonlinear least squares curve fitting (and parameter estimation) where  $s^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = f(b_0, b_1)$  is bimodal. It has the global minimum for  $s^2 = f(0.864787293, 1.2357485) = 16.0817301$  and a local minimum (as pointed out by Wild, 2001)  $f(2.498571, -0.9826092) = 20.48234$  (not  $f(2.35, -0.319) = 20.9805$  as mentioned by Goffe, 1994 as well as in the computer program *simann.f*). It is an easy task for the Differential Evolution method to minimize this function.

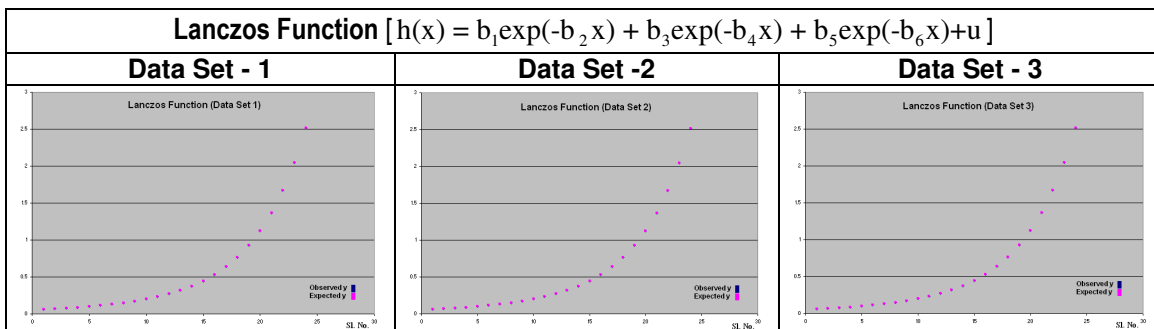
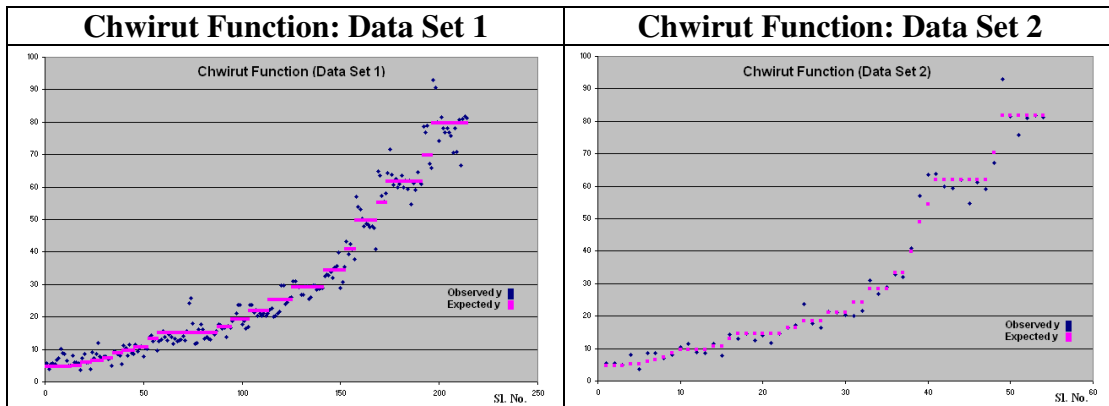


**2. The Hougen-Watson Function:** The Hougen-Watson model (Bates and Watts, 1988; see at Mathworks.com) for reaction kinetics is a typical example of nonlinear regression model. The rate of kinetic reaction (y) is dependent on the quantities of three inputs: hydrogen ( $x_1$ ), n-pentane ( $x_2$ ) and isopentane ( $x_3$ ). The model is specified as:

$$y = \text{rate} = \frac{b_1 x_2 - \frac{x_3}{b_5}}{1 + b_2 x_1 + b_3 x_2 + b_4 x_3} + u$$

For the given dataset the minimum  $s^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = f(b_1, b_2, b_3, b_4, b_5) = f(1.25258511, 0.0627757706, 0.0400477234, 0.112414719, 1.19137809) = 0.298900981$ . The graphical presentation of the observed values against the expected values of y suggests that the model fits to the data very well.

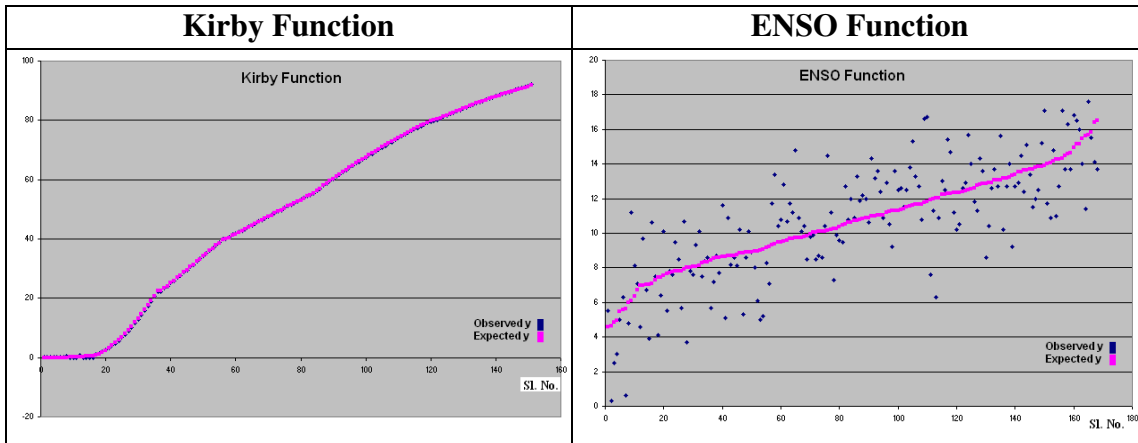
**3. The Chwirut Function:** This function (specified as  $y = \exp(-b_1 x) / (b_2 + b_3 x) + u$ ) describes ultrasonic response (y) to metal distance (x). This function has been fitted to two sets of data (data-1 and data-2). In case of the first set of data the Differential Evolution method has found the minimum value of  $s^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = f(b_1, b_2, b_3)$  which is  $f(0.190278183, 0.00613140045, 0.0105309084) = 2384.47714$ . However, for the second set of data the results are marginally sub-optimal. For the second set of data, the certified value of  $s^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = f(b_1, b_2, b_3)$  is 513.04802941, but we have obtained  $f(0.167327315, 0.00517431911, 0.0121159344) = 515.15955$ .



**4. Lanczos Function:** Lanczos (1956) presented several data sets (at different accuracy levels) generated by an exponential function  $g(x) = 0.0951 \exp(-x) + 0.8607 \exp(-3x) + 1.5576 \exp(-5x)$ . Using the given dataset of this problem one may estimate the parameters of  $h(x) = b_1 \exp(-b_2 x) + b_3 \exp(-b_4 x) + b_5 \exp(-b_6 x) + u$  and check if the values of

$(b_1, b_2, b_3, b_4, b_5, b_6) = (0.0951, 1, 0.8607, 3, 1.5576, 5)$  are obtained. We have obtained  $s^2 = f(0.0951014297, 1.00000728, 0.860703939, 3.00000927, 1.55759463, 5.00000322) = 9.07870717E-18$  for the first data set, while the certified value is  $1.4307867721E-25$ . The estimated parameters are very close to the true parameters. For the second data set we obtained  $s^2 = f(0.0962570522, 1.00576317, 0.864265983, 3.00786966, 1.55287658, 5.00289537) = 2.22999349E-11$  against the certified value of  $2.2299428125E-11$ . The estimated parameters are once again very close to the true ones. For the third data set we have obtained  $s^2 = f(1.58215875, 4.98659893, 0.844297096, 2.95235111, 0.0869370574, 0.955661374)$  to be  $1.61172482E-008$ . The certified value is  $1.6117193594E-008$ .

**5. The Kirby Function:** Kirby (NIST, 1979) measured response values (y) against input values (x) to scanning electron microscope line width standards. The Kirby function is the ratio of two quadratic polynomials,  $y = g(x) = (b_1 + b_2x_1 + b_3x_1^2)/(1 + b_4x_1 + b_5x_1^2) + u$ . We have obtained  $s^2 = f(1.67450632, -0.13927398, 0.00259611813, -0.00172418116, 2.16648026E-005) = 3.90507396$  against the certified value of  $3.9050739624$ .



**6. The ENSO Function:** This function (Kahaner, et al., 1989) relates y, monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia and time (x). The difference in the atmospheric pressure (y) drives the trade winds in the southern hemisphere (NIST, USA). The function is specified as

$$y = b_1 + b_2\cos(2\pi x/12) + b_3\sin(2\pi x/12) + b_5\cos(2\pi x/b_4) + b_6\sin(2\pi x/b_4) + b_8\cos(2\pi x/b_7) + b_9\sin(2\pi x/b_7) + u$$

Arguments to the  $\sin(\cdot)$  and  $\cos(\cdot)$  functions are in radians.

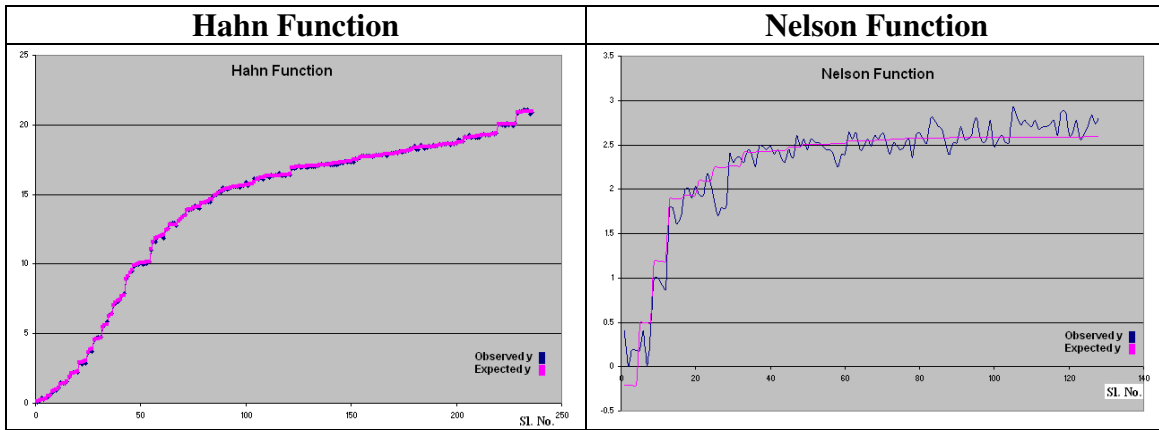
We have obtained  $s^2 = f(10.5107492, 3.0762128, 0.532801425, 26.8876144, 0.212322867, 1.49668704, 44.3110885, -1.62314288, 0.525544858) = 788.539787$  against the certified value of  $788.53978668$ .

**7. The Hahn Function:** Hahn (197?) studied thermal expansion of copper and fitted to data a model in which the coefficient of thermal expansion of copper (y) is explained by a

ratio of two cubic polynomials of temperature (x) measured in the Kelvin scale. The model was:  $y = (b_1 + b_2x + b_3x^2 + b_4x^3)/(1 + b_5x + b_6x^2 + b_7x^3) + u$ . We have obtained

$$s^2 = f(1.07763262, -0.122692829, 0.00408637261, -1.42626427E-006, -0.0057609942, 0.000240537241, -1.23144401E-007) = 1.53243829 \text{ against the certified value} = 1.5324382854.$$

If in place of specifying the cubic in the denominator as  $(1 + b_5x + b_6x^2 + b_7x^3)$ , we permit the specifications as  $(b_8 + b_5x + b_6x^2 + b_7x^3)$  such that the model specification is  $y = (b_1 + b_2x + b_3x^2 + b_4x^3)/(b_8 + b_5x + b_6x^2 + b_7x^3) + u$  and fit it to Hahn's data, we have:  $s^2 = f(-1.89391801, 0.215629874, -0.00718170192, 2.50662711E-006, 0.0101248026, -0.000422738373, 2.16423365E-007, -1.75747467) = 1.532438285361130$  that meets the certified value given by NIST (1.5324382854) for entirely different set of parameters. The value of  $b_8$  is remarkably different from unity. Of course, Hahn's specification is parsimonious.



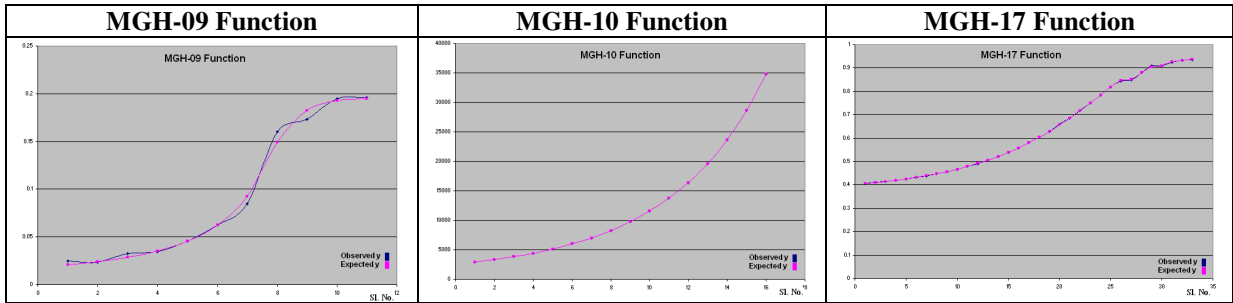
**8. The Nelson Function:** Nelson (1981) studied performance degradation data from accelerated tests and explained the response variable dielectric breakdown strength (y, in kilo-volts) by two explanatory variables - time ( $x_1$ , in weeks) and temperature ( $x_2$ , in degrees centigrade). He specified the model as  $y = b_1 - b_2x_1 \exp(-b_3x_2) + u$ . We have obtained  $s^2 = f(2.5906836, 5.61777188E-009, -0.0577010131) = 3.797683317645143$  against the NIST-certified value, 3.7976833176. Another minimum of  $S^2 = f(b_1, b_2, b_3)$  is found to be  $s^2 = f(-7.4093164, 5.61777132E-009, -0.0577010134) = 3.797683317645138$ .

**9. The MGH Functions:** More, Garbow and Hillstrom (1981) presented some nonlinear least squares problems for testing unconstrained optimization software. These problems were found to be difficult for some very good algorithms. Of these functions, MGH-09 (Kowalik and Osborne, 1978; NIST, USA) is specified as  $y = b_1(x^2 + b_2x)/(x^2 + b_3x + b_4) + u$  that fits to MGH-09 data with NIST certified  $s^2 = 3.0750560385E-04$  against which have obtained  $s^2 = f(0.192806935, 0.191282322, 0.123056508, 0.136062327) = 3.075056038492363E-04$ .

Another problem (MGH-10; NIST, USA) is the model (Meyer, 1970) specified as  $y = b_1 \exp(b_2/(x + b_3)) + u$  whose parameters are to be estimated on MGH-10 data. We

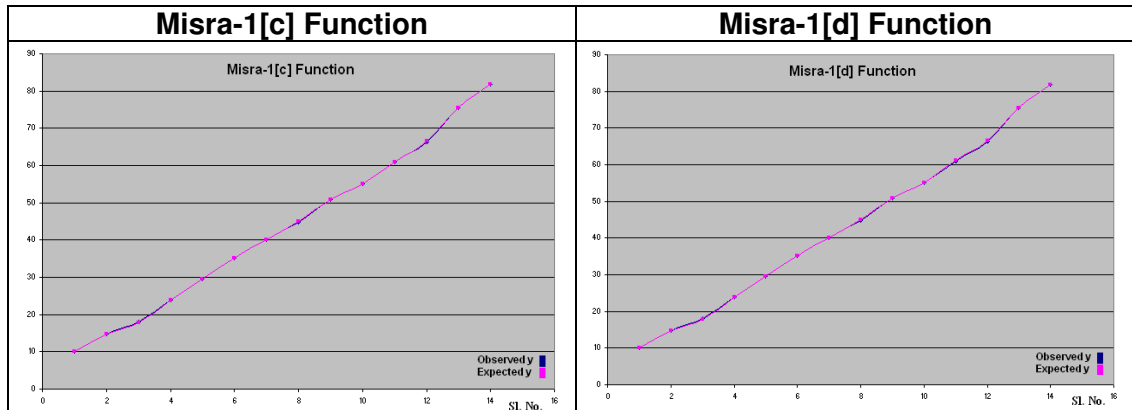
have obtained  $s^2 = f(0.00560963647, 6181.34635, 345.223635) = 87.94585517018605$  against the NIST certified value of  $s^2 = 87.945855171$ .

Yet another problem (MGH-17; NIST, USA) is the model (Osborne, 1972) specified as  $y = b_1 + b_2 \exp(-b_4 x) + b_3 \exp(-b_5 x) + u$  whose parameters are to be estimated on MGH-17 data. We have obtained  $s^2 = f(0.375410053, 1.93584702, -1.46468725, 0.0128675349, 0.0221226992) = 5.464894697482394E-05$  against  $s^2 = 5.4648946975E-05$ , the NIST certified value of  $s^2$ .



**10. The Misra Functions:** In his dental research monomolecular adsorption study, Misra (1978) recorded a number of datasets and formulated a model that describes volume ( $y$ ) as a function of pressure ( $x$ ). His model Misra-1[c] is:  $y = b_1(1-(1+2b_2x)^{-0.5}) + u$ . We have fitted this function to data (Misra-1[c]) and against the NIST certified value of 0.040966836971 obtained  $s^2 = f(636.427256, 0.000208136273) = 0.04096683697065384$ .

Another model,  $y = b_1 b_2 x((1+b_2x)^{-1}) + u$  was fitted to Misra-1[d] data set and we obtained  $s^2 = f(437.369708, 0.000302273244) = 0.05641929528263857$  against the NIST certified value, 0.056419295283.



**11. The Thurber Function:** Thurber (NIST, 197?) studied electron mobility ( $y$ ) as a function of density ( $x$ , measured in natural log) by a model  $y = \frac{(b_1 + b_2x + b_3x^2 + b_4x^3)}{(1 + b_5x + b_6x^2 + b_7x^3)} + u$ .

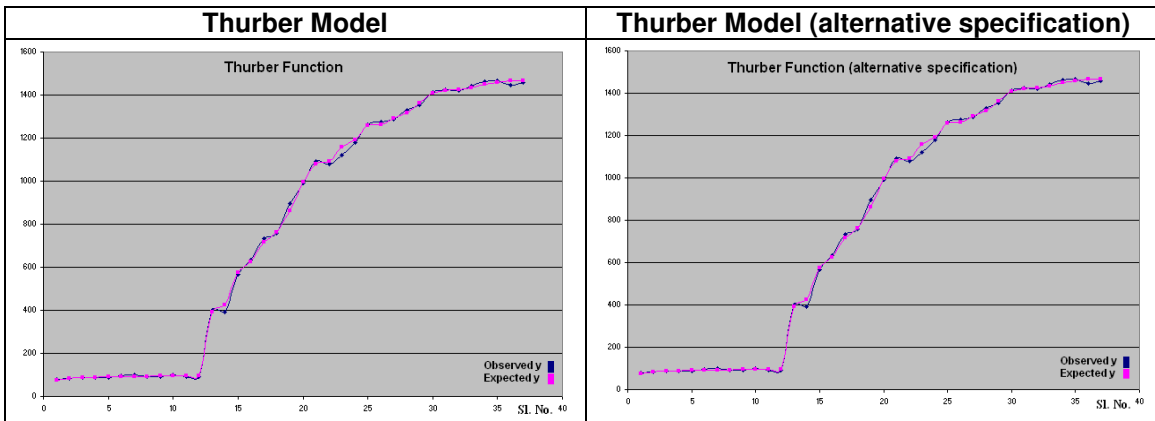
We fitted this model to the given data and obtained minimum  $s^2 = 5.642708239666791E+03$  against the NIST-certified value =  $5.6427082397E+03$ . The estimated model is obtained as:

$$\hat{y} = \frac{1288.13968 + 1491.07925x + 583.238368x^2 + 75.4166441x^3}{1 + 0.96629503x + 0.397972858x^2 + 0.0497272963x^3}$$

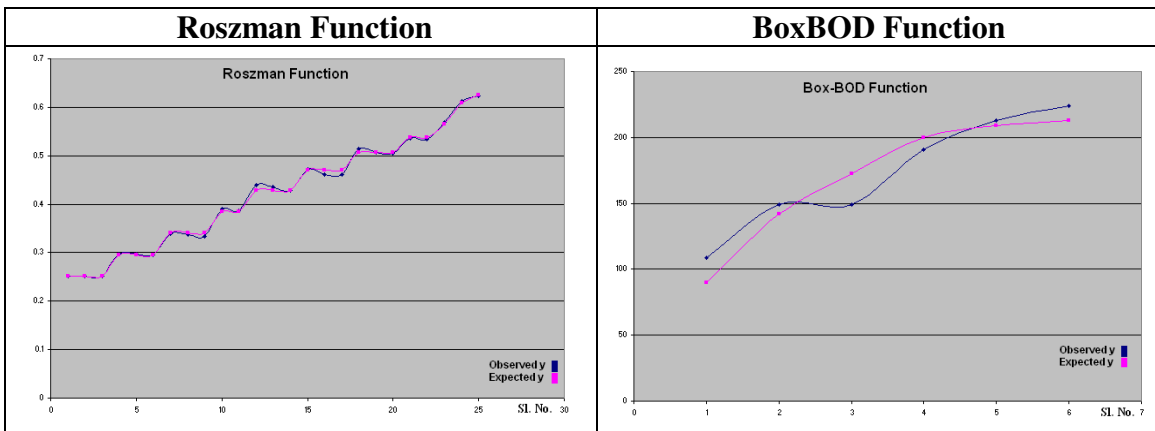
Alternatively, if we specify the model as  $y = \frac{(b_1 + b_2x + b_3x^2 + b_4x^3)}{(b_8 + b_5x + b_6x^2 + b_7x^3)} + u$ , we obtain

$$\hat{y} = \frac{1646.30744 + 1905.67444x + 745.408029x^2 + 96.386272x^3}{1.27805041 + 1.23497375x + 0.508629371x^2 + 0.0635539913x^3}; s^2 = 5.642708239666863E+03$$

It appears that replacing of 1 by  $b_8 = 1.27805041$  in the model serves no purpose except demonstrating that the parameters of the model are not unique. Note that on uniformly dividing all the parameters of the (estimated) alternative model by  $b_8 (=1.27805041)$  we do not get the estimated parameters of the original model.

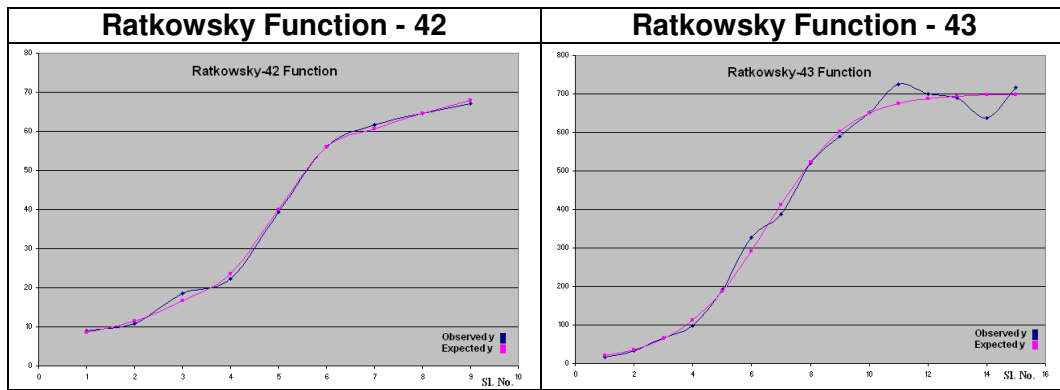


**12. The Roszman Function:** In a NIST study Roszman (19??) investigated the number of quantum defects ( $y$ ) in iodine atoms and explained them by the excited energy state ( $x$  in radians) involving quantum defects in iodine atoms (NIST, USA). The model was specified as  $y = b_1 - b_2x - \arctan(b_3/(x-b_4))/\pi + e$ . We estimated it on the given data and obtained  $s^2 = f(0.201968657, -6.1953505E-006, 1204.4557, -181.34271) = 4.948484733096893E-04$  against NIST certified value  $4.9484847331E-04$ .

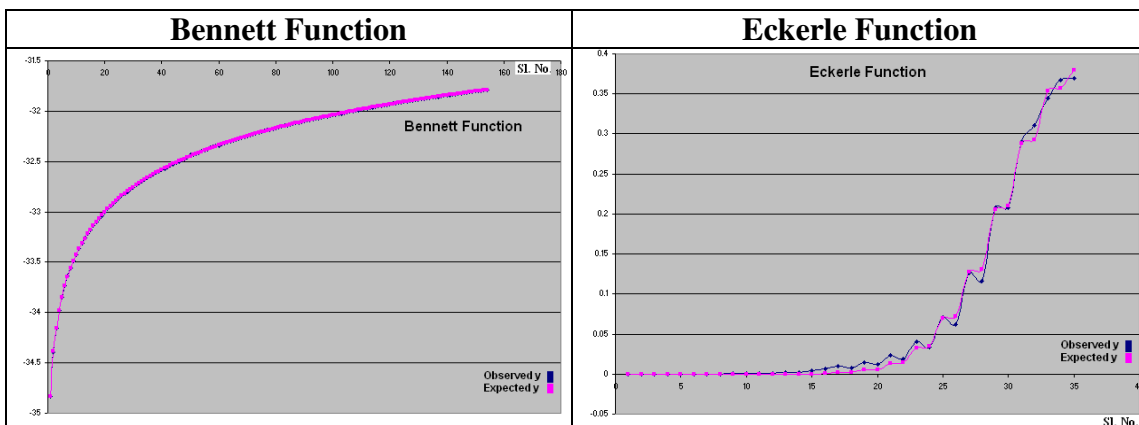


**13. The BoxBOD Function:** Box et al. (1978) explained the biochemical oxygen demand (y, in mg/l) by incubation time (x, in days) by the model  $y = b_1(1-\exp(-b_2x)) + u$ . We have obtained the minimum  $s^2 = f(213.809409, 0.547237484) = 1.16800887655550E+03$  against the NIST certified value, 1.1680088766E+03.

**14. The Ratkowsky Functions:** Two least squares curve-fitting problems presented by Ratkowsky (1983) are considered relatively hard. The first (RAT-43, NIST, USA), specified as  $y = b_1 / (1+\exp(b_2-b_3x)) + u$  with the dataset RAT-42, has been estimated by us to yield  $s^2 = f(72.4622375, 2.61807684, 0.0673592002) = 8.056522933811241$  against the NIST certified value, 8.0565229338. The second model (RAT-43, NIST, USA), specified as  $y = b_1 / ((1+\exp(b_2-b_3x))^{(1/b_4)} + u$  with the dataset RAT-43, has been estimated by us to yield  $s^2 = f(699.641513, 5.27712526, 0.75962938, 1.27924837) = 8.786404907963108E+03$  against the NIST certified value, 8.7864049080E+03.

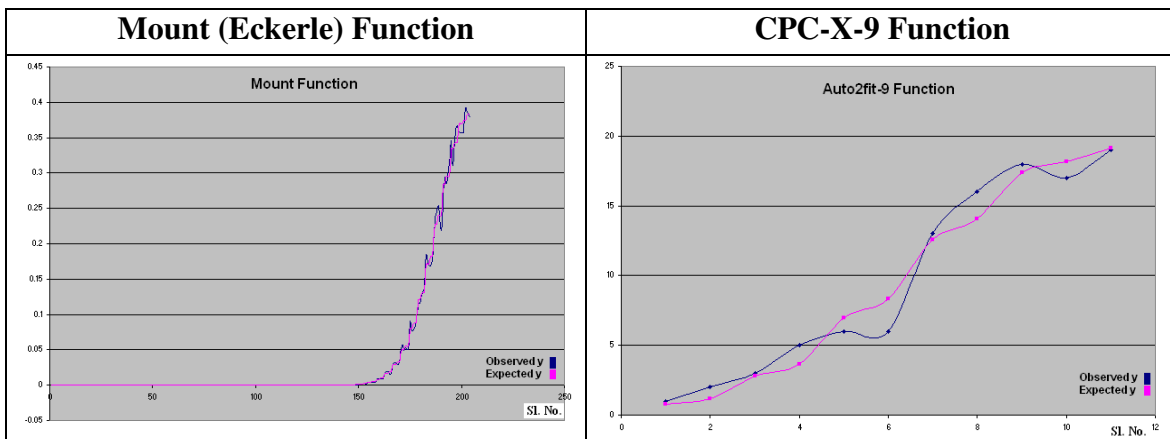


**15. The Bennett Function:** Bennett et al. (NIST, 1994) conducted superconductivity magnetization modeling and explained magnetism (y) by duration (x, log of time in minutes) by the model  $y = b_1(b_2 + x)^{-1/b_3} + u$ . Against the NIST certified value of minimum  $s^2 = 5.2404744073E-04$ , we have obtained  $s^2 = f(-2523.80508, 46.7378212, 0.932164428) = 5.241207571054023E-04$ . The rate of convergence of the DE solution towards the minimum has been rather slow.



**16. The Eckerle Function:** In a NIST study Eckerle (197?, NIST, USA) fitted the model specified as  $y = (b_1/b_2) \exp(-0.5((x-b_3)/b_2)^2) + u$  where  $y$  is transmittance and  $x$  is wavelength. We have obtained  $s^2 = f(-1.55438272, -4.08883218, 451.541218) = 1.463588748727469E-03$  against the NIST certified value,  $1.4635887487E-03$ .

**17. The Mount Function:** Although the specification of this function is identical to the Eckerle function, the CPC-X Software have fitted it to a different dataset. Against the reference value of  $5.159008779E-03$  of CPC-X, we have obtained the value of  $s^2 = f(1.5412806, 4.01728442, 450.892013) = 5.159008010368E-03$ . Further, against the reference values of RMS and  $R^2$  ( $5.028842682e-03$  and  $0.9971484642$ ) we have obtained  $5.028842307409E-03$  and  $0.997148464588044$  respectively.



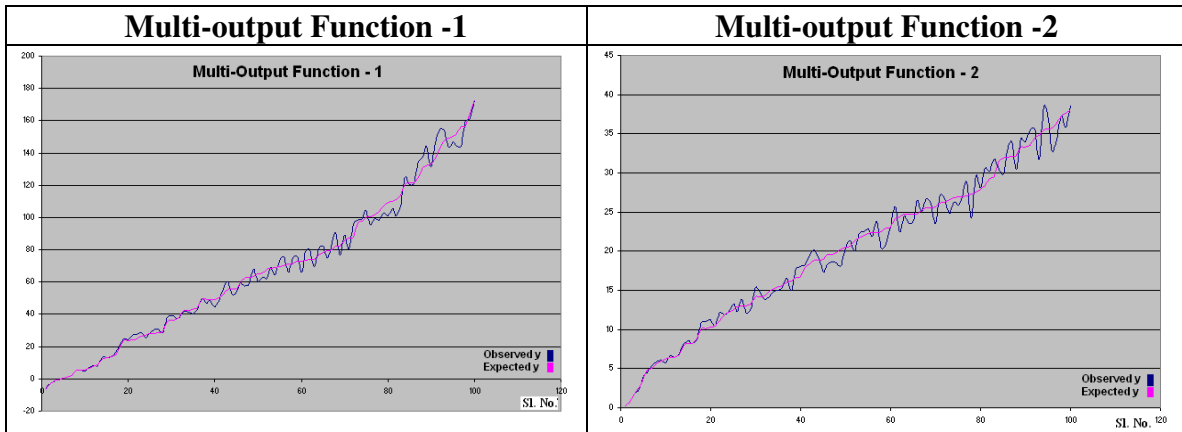
**18. The CPC-X-9 Function:** This function is specified as  $y = b_1 \exp(b_2 | (x + b_3) |^{b_4}) + u$ . We fitted this function to the given data. We obtained  $R^2 = 0.9699794119704664$  (against  $0.9704752$ ) and  $RMS = 1.154690900182629$  (against  $1.1546909$ ) obtained by AUTO2FIT.  $S^2 = f(19.1581777, -0.362592746, -29.8159227, 2.29795109) = 14.66642182461028$ .

**19. The Multi-output Function:** The CPC-X has given an example of a multi-output function in which two dependent variables ( $y_1$  and  $y_2$ ) are determined by the common independent variables ( $x_1, x_2, x_3$ ) and they have some common parameters ( $b$ ) such that:

$$y_1 = x_1^{b_1} + b_2 \ln(x_2) \exp(x_3^{b_3}) + u_1$$

$$y_2 = x_1^{b_1} + b_2 \exp(x_2^{b_4}) \ln(x_3^{b_3}) + u_2$$

Most of the software/methods do not provide any scope to deal with multiple dependent variables having some common parameters and/or independent variables, although such problems are real (viz. production functions of a multi-product firm, etc). We have fitted these functions to the dataset, provided by the CPC-X, in two ways; first when (i) we have not constrained the sum of errors  $\sum u_1$  and  $\sum u_2$  individually to be near zero, (ii) we have constrained each of them to be less than  $1.0E-06$  in magnitude. The two fits differ marginally as shown in the table below:



Estimated Parameters of Multi-output function: (i). Unconstrained and (ii). Constrained										
	$b_1$	$b_2$	$b_3$	$b_4$	$R_1^2$	$R_2^2$	$s_1^2$	$s_2^2$	RMS <sub>1</sub>	RMS <sub>2</sub>
i	0.255886425	2.96331268	0.100767089	0.0929527106	0.990723	0.98468	1962.219	146.5238	4.429694	1.21047
ii	0.256305001	2.97236378	0.100648248	0.0926076197	0.99072	0.984674	1962.89	146.5788	4.430452	1.210697

The reference values of  $R_1^2$  and  $R_2^2$  are 0.990522 and 0.984717 respectively. It may be noted that we have no information as to how the CPC-X has defined the minimand function. Yet, our results are not quite different from theirs.

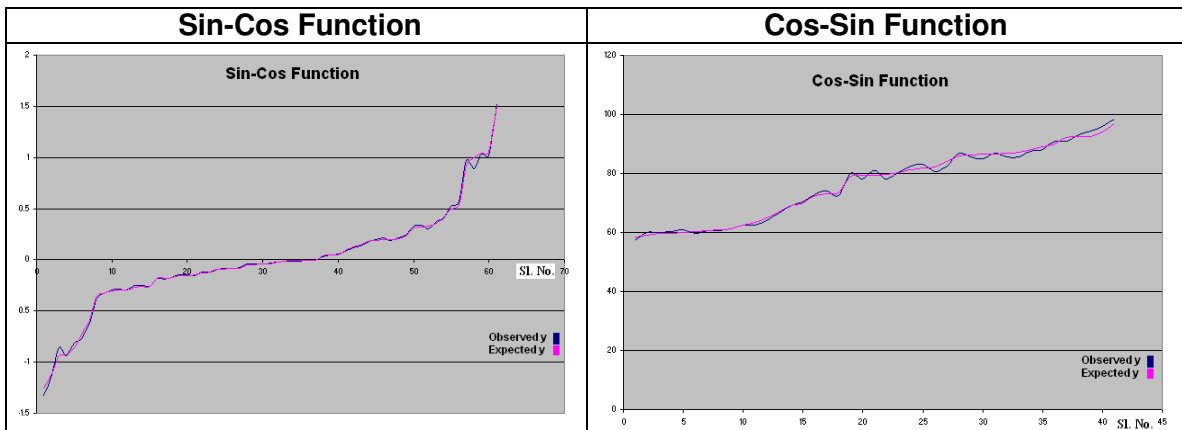
**20. The Sin-Cos Function:** This function (given by the CPC-X Software) is specified as

$$y = (b_1 + x_1/b_2 + \cos(b_3 x_2/x_3))/(b_4 \sin(x_1 + x_2 + x_3)) + u$$

We have obtained  $\hat{b} = (0.493461213, 2.93908006, 10.9999618, 5.83684187)$ ;  $R^2 = 0.99740460$  and  $RMS = 0.025161467$  against reference values 0.9974045694 and 0.02516162826 respectively.

**21. The Cos-Sin Function:** This function (given by the CPC-X Software) is specified as

$$y = ((b_1/x_1) - \cos(b_2 x_2)) x_3 b_3/x_1 + u$$



We have obtained  $\hat{b} = (2.49225824, -49.9980138, 2.13226556)$ ;  $R^2 = 0.9930915320764427$  and  $RMS = 1.011115788318$  against reference values 0.9930915321 and 1.011115788 respectively. This function is more difficult than the Sin-Cos function to fit.

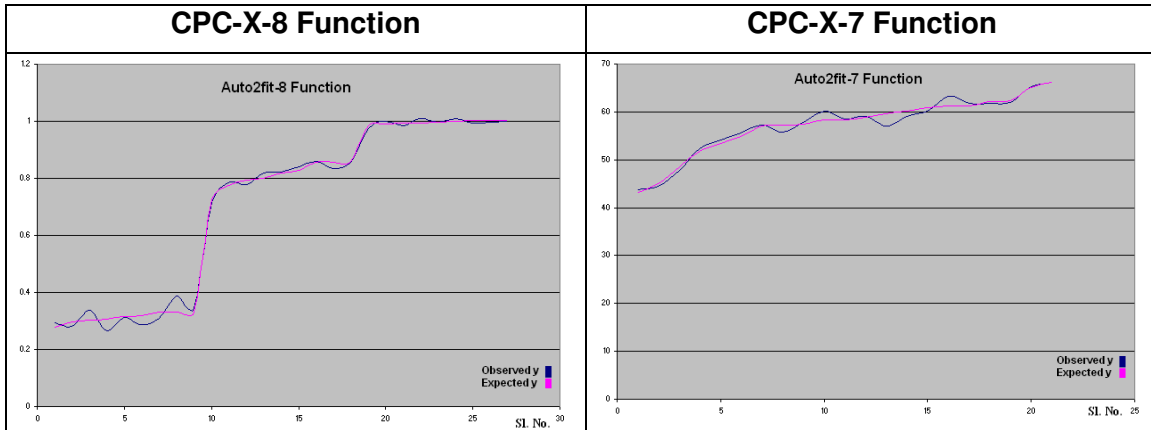
**22. The CPC-X-8 Function:** This is a composite multivariate sigmoid function given as

$$y = \frac{b_1}{(b_2 + x_1)(1 + b_3 x_2)(x_3 - b_4)^2} + b_5 x_3^{b_6} + u$$

We have fitted this function to AUTO2FIT data and obtained  $R^2 = 0.9953726879097797$  slightly larger than the  $R^2 (= 0.995372)$  obtained by AUTO2FIT. The estimated function is

$$\hat{y} = \frac{174808.701}{(3615.41672 + x_1)(1 + 0.536364662 x_2)(x_3 - 27.8118343)^2} + 160.016475 x_3^{-2.5}$$

The value of  $s^2$  is 0.01056060934407798 and  $RMS = 0.0197770998736$  against 0.01977698 obtained by AUTO2FIT. Further, there is some inconsistency in the figures of  $R^2$  and  $RMS$  (of errors) reported by CPC-X. If their  $R^2$  is smaller than our  $R^2$  then their  $RMS(E)$  cannot be smaller than our  $RMS(E)$ .

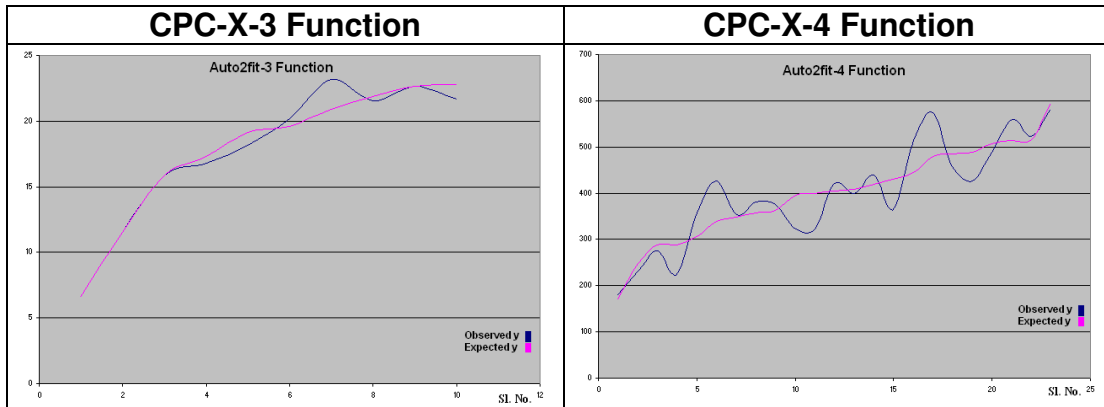


**23. The CPC-X-7 Function:** This function is specified as  $y = \frac{b_1 + b_2 x_1 + b_3 x_2 + b_4 x_1 x_2}{1 + b_5 x_1 + b_6 x_2 + b_7 x_1 x_2} + u$ .

We have fitted it to CPC-X data and obtained  $R^2 = 0.9715471304250647$  against the  $R^2 = 0.9715471$  of AUTO2FIT. The value of  $RMS(E)$  is 1.006260685261970 against the AUTO2FIT value 1.00626078. Our  $s^2$  is 21.263771900781600. The estimated function is

$$\hat{y} = \frac{92.0738767 - 0.0267347156 x_1 - 2.72078474 x_2 + 0.000744446437 x_1 x_2}{1 - 0.000384550462 x_1 - 0.0303920084 x_2 + (1.07039964E-005) x_1 x_2}$$

**24. The CPC-X-3 Function:** The function specified as  $y = b_1 / (1 + b_2/x + x/b_3) + u$  has been fitted to the test dataset provided by the CPC-X. We obtain  $R^2 = 0.969923509396039$  (against reference value, 0.969929562),  $RMS = 0.87672786941874$  (against 0.8767278) and  $s^2 = f(-101.078841, -1258.50244, -170.113552) = 7.68651757015526$ .



**25. The CPC-X-4 Function:** This function is a ratio of two linear functions, both in four predictor variables. Its specification is:  $y = \frac{b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4}{1 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4} + u$ . We have fitted this function to the data (given by CPC-X) and obtained  $R^2 = 0.8051428644699052$  against the reference value, 0.80514286. The estimated function is:

$$\hat{y} = \frac{674.67934 + 227.745644x_1 + 2120.32578x_2 + 1.64254986x_3 - 176.051025x_4}{1 + 0.572582178x_1 + 5.55641932x_2 + 0.0334385585x_3 - 0.560015248x_4}$$

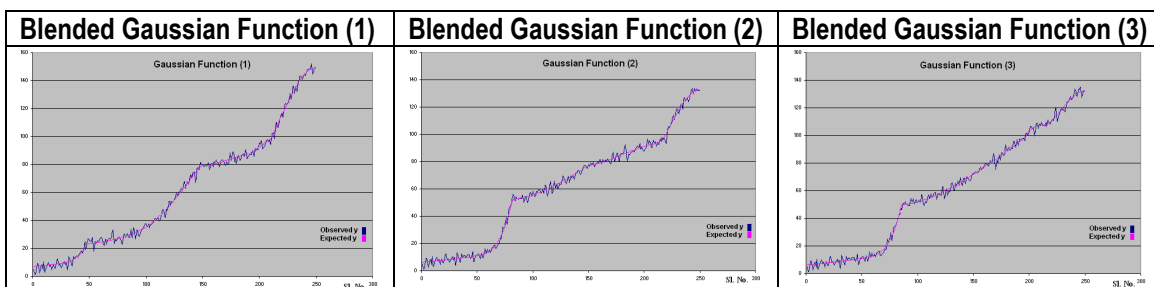
The  $s^2 = 53118.2415305900$  and  $RMS = 48.0571405953$  (against reference value 48.05714).

**26. The Blended Gaussian Function:** NIST has given three datasets (with different difficulty levels) to fit a blended Gaussian function. The function is specified as

$$y = b_1 \exp(-b_2x) + b_3 \exp(-(x-b_4)^2/b_5^2) + b_6 \exp(-(x-b_7)^2/b_8^2) + u$$

We have fitted this function to the three sets of data and obtained the following results.

Estimated Parameters of Blended Gaussian Function with Different Datasets						
Function	$b_1$	$b_2$	$b_3$	$b_4$	$b_5$	$b_6$
Gauss1	98.7782107	0.0104972764	100.489906	67.4811113	23.1297733	71.9945029
Gauss2	99.0183284	0.0109949454	101.880225	107.030955	23.578584	72.0455895
Gauss3	98.9403689	0.0109458794	73.7050314	147.761643	19.6682212	100.695531
	$b_7$	$b_8$	NIST certified $s^2$	Ours $s^2$	Ours RMS	Ours $R^2$
Gauss1	178.998050	18.3893889	1315.8222432	1315.822206428	2.294186	0.996962322
Gauss2	153.270102	19.5259727	1247.5282092	1247.528209231	2.233856	0.996486539
Gauss3	111.636195	23.3005001	1244.4846360	1244.484636013	2.231129	0.996899074

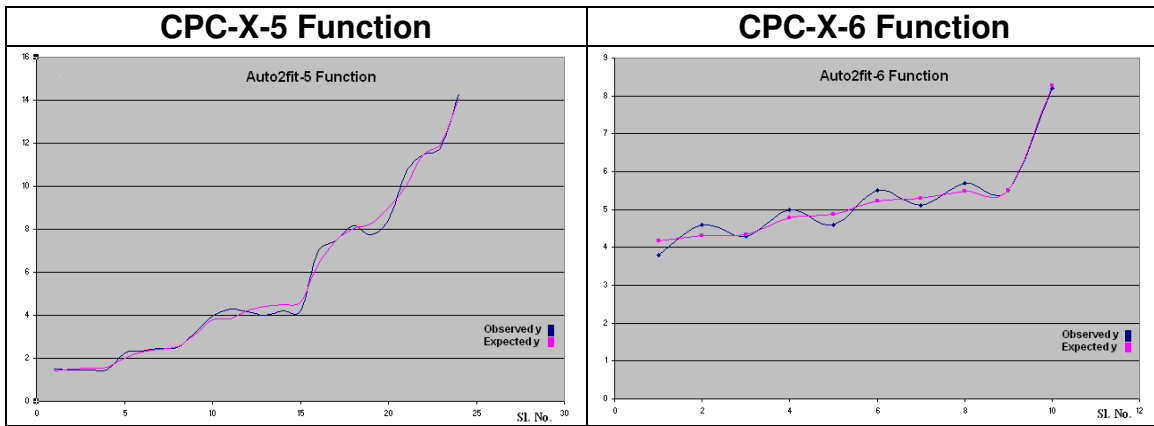


It is worth reporting that the function fitting to dataset-1 is easier as it is robust to a choice of  $b_2$  than the other two datasets. A range ( $0 < b_2 < 10$ ) yields the results. However, the other two datasets need ( $0 < b_2 < 0.1$ ) else the algorithm is caught in the local optimum trap. All the three datasets are problematic if  $b_5$  or  $b_8$  is given a range much beyond (0, 50).

**27. The CPC-X-5 Function:** The function  $y = b_1 + b_2x_1^{b_3} + b_4x_2^{b_5} + b_6x_1^{b_7}x_2^{b_8} + u$  has been fitted to the data provided by CPC-X. We have obtained  $R^2 = 0.9932818431032495$  against 0.994632848 and  $RMS = 0.3024453470938$  against 0.2703296 reported by the makers of AUTO2FIT. Ours  $s^2 = 2.1953565114881$ . The estimated model is

$$\hat{y} = 0.833300621 + 0.0894933939x_1^{-0.312443915} + 0.634308339x_2^{1.42617267} - 0.631664635x_1^{-0.00228812301}x_2^{1.42909022}$$

We would also like to mention that the present solution needed several trials to get at these values. The problem is extremely ill conditioned and very sensitive to the choice of the domain or the initial (starting) values of parameters.



**28. The CPC-X-6 Function:** The function  $y = b_1 + b_2x^{b_3} + b_4x^{b_5} + b_6x^{b_7} + u$  has been fitted to CPC-X-6 data. We obtain  $R^2 = 0.9614190785232305$  and  $RMS = 0.2236173793023$  against the reference values 0.999644261 and 0.0214726136 as reported. We obtained  $s^2 = 0.5000473232604$  for the following estimated model.

$$\hat{y} = -13104.0498 + 1042.09568x^{0.199999589} - 114.02134x^{0.499998867} + 12184.2476x^{-0.0132847799}$$

The problem is extremely unstable.

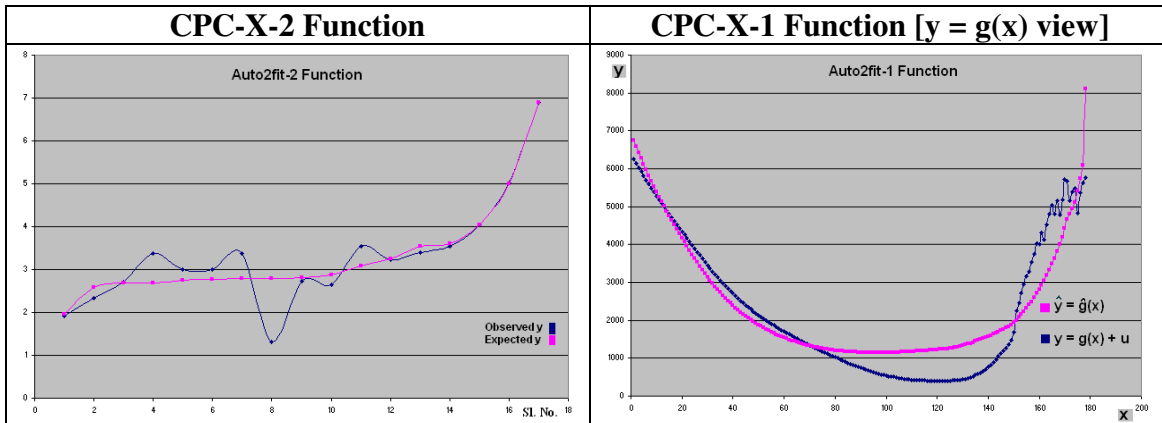
**29. The CPC-X-2 Function:** This function is a ratio of two other linear functions given as

$$y = \frac{b_1 + b_2x_1 + b_3x_2 + b_4x_3 + b_5x_4}{1 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4} + u$$

We have obtained  $R^2 = 0.8622291597468909$  and  $RMS = 0.439647321698$  against the reference values of 0.9346422 and 0.3028129 respectively. The estimated function is

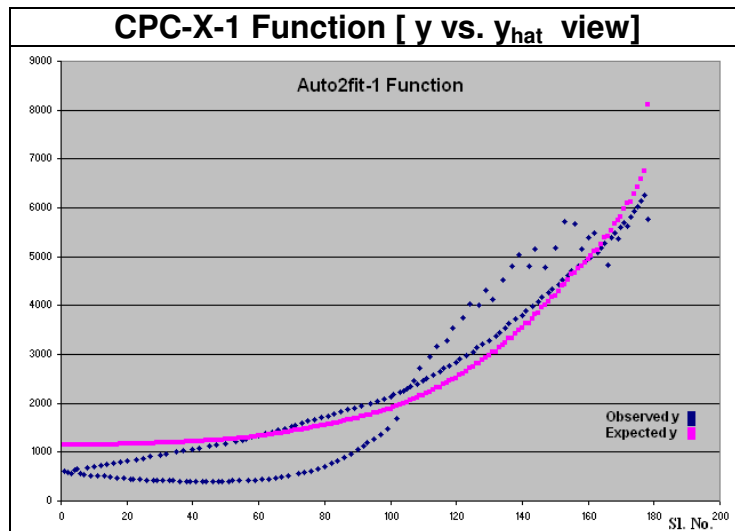
$$\hat{y} = \frac{4.58342731 + 0.000262177177x_1 - 7.95307951E-006x_2 - 0.0270514576x_3 + 0.0331768444x_4}{1 + 9.54335611E-005x_1 - 3.04612509E-006x_2 - 0.0066977514x_3 + 0.00668129827x_4}$$

For this estimated model the value of  $s^2$  is 3.479215814564.



**30. The CPC-X-1 Function:** This function is given as:  $y = \frac{1}{b_1 + b_2 x^{b_3}} + b_4 x^{b_5} + u$ . We have fitted this function to CPC-X-1 data to obtain  $R^2 = 0.882331799699548$  against the reference value 0.99678004 and RMS = 622.7034 against 104.376667. The value of  $s^2$  obtained by us is 69021203.98784. Our estimates are far off the mark. We obtain

$$\hat{y} = \frac{1}{0.450036183 - 0.450036183x^{-0.00346072131}} + 1.46250826E-006x^{4.3671976}$$



**V. Concluding Remarks:** The Differential Evaluation (DE) method applied to fit functions to datasets given by NIST and others has exhibited a mixed performance. It has been successful at the job for all problems, of various degrees of difficulty, given by NIST, although, the Blended Gauss functions have been relatively difficult and sensitive to the choice of initial values or range of parameters. It may be noted that *unless otherwise stated or discussed, the DE has been successful to obtain the optimum results even if the domains of parameters were too wide*. Oftentimes, the DE does not require the domain (of parameters) to be specified in a narrow range as do the other software/methods to solve the nonlinear least squares problem. However, in a few cases

when a too wide domain made the program unstable, wayward or haywire, narrower domains were specified. Such cases have been duly reported.

Among the CPC-X functions (including the Mount, the Sin-Cos, the Cos-Sin and the Multi-output functions) - ten of them posed by the CPC-X Software as the challenge problems - the DE has been able to deal with nine (challenge functions # 9, 8, 7, 3, 4; and other functions namely the Mount, the Sin-Cos, the Cos-Sin and the Multi-output functions) either comfortably or with some trial and error in setting the ranges of parameters to be estimated. In particular, the Mount, the Sin-Cos, the Cos-Sin and the Multi-output functions have been very easy to fit. The function # 5 has been quite difficult to optimize and although the DE took the solution very close to the one reported by CPC-X, but it remained, after all, sub-optimal. The DE solution to the CPC-X-6 function remained appreciably far from the optimal fit.

The DE performed miserably in dealing with two CPC-X functions: #1 and #2. In spite of several trials, the DE failed to reach any closer to the optimal solution (the reference  $R^2$  provided by the CPC-X).

The Differential Evolution optimizer is a (stochastic) population-based method. It may be noted that all population-based methods of optimization partake of the probabilistic nature inherent to them. As a result, one cannot obtain certainty in their results, unless they are permitted to go on for indefinitely large search attempts. Larger is the number of attempts greater is the probability that they would find out the optimum. Secondly, all of them adapt themselves to the surface on which they find the optimum. The scheme of adaptation is largely based on some guesswork since nobody knows as to the true nature of the problem (environment or surface) and the most suitable scheme of adaptation to fit the given environment. Surfaces may be varied and different for different functions. Further, like any other population-based method of optimization, the DE method operates with a number of parameters that may be changed at choice to make it more effective. This choice is often problem oriented and that for obvious reasons. A particular choice may be extremely effective in a few cases, but it might be ineffective (or counterproductive) in certain other cases. Additionally, there is a relation of trade-off among those parameters.

The CPC-X problems are the challenge problems for any nonlinear Least Squares algorithm. About these problems, the CPC-X Software themselves remark: "*Some of those test data are very hard, and may never get right answers without using Auto2Fit. Even for Auto2Fit, it does not ensure every run will be successful. ... In some cases, you may try to change the control parameter of 'Population Size' ...*". They have suggested that to solve these problems one should use Global Levenberg-Marquardt or Global BFGS method. The CPC-X has also fitted the multi-output function by the DE method - not by the Global Levenberg-Marquardt or the Global BFGS method. If the DE has performed well at more than half the number of such challenge problems (*and done better than the AUTO2FIT in some cases*), we may conclude that its success rate is appreciably high and it may be used for solving nonlinear curve fitting problem with some good degree of reliability and dependability (for performance of other software on

NIST functions see Lilien, 2000). It may be noted that there cannot be any ‘sure success method’ to solve all the problem of nonlinear least squares curve fitting.

Additionally, the DE oftentimes allows for a large and wide domain for the parameters to start the search. The most of other algorithms for solving a nonlinear least squares problem are too (impracticably) demanding on the initial guess of parameters.

## References

- 7d-soft High Technology Inc (--) AUTO2FIT Software The New Website of CPC-X Software <http://www.geocities.com/neuralpower> now at [www.7d-soft.com](http://www.7d-soft.com)
- Bates, D and Watts, D (1988) *Nonlinear Regression Analysis and Its Applications*, John Wiley and Sons, New York.
- Bennett, L, Swartzendruber, L and Brown, H (1994) Superconductivity Magnetization Modeling, National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Box, GEP (1957) “Evolutionary Operation: A Method for Increasing Industrial Productivity”, *Applied Statistics*, 6 , pp. 81-101.
- Box, GP, Hunter, WG and Hunter, JS (1978) *Statistics for Experimenters*. Wiley, New York, pp. 483-487.
- Box, MJ (1965) “A New Method of Constrained Optimization and a Comparison with Other Methods”. *Comp. J.* 8, pp. 42-52.
- Cerny, V (1985) "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Calculations by Fast Computing Machines", *J. Chem. Phys.*,21, 6, 1087-1092.
- CPC-X Software (--) At [http://www.geocities.com/neuralpower/Regression\\_Test.htm#3.%20One%20More%20Test%20Data](http://www.geocities.com/neuralpower/Regression_Test.htm#3.%20One%20More%20Test%20Data) of <http://www.geocities.com/neuralpower>
- Eberhart RC and Kennedy J (1995) “A New Optimizer using Particle Swarm Theory”, *Proceedings Sixth Symposium on Micro Machine and Human Science*, pp. 39–43. IEEE Service Center, Piscataway, NJ.
- Eckerle, K (197?) Circular Interference Transmittance Study, National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Glover F (1986) " Future Paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, 5:533-549.
- Goffe, Ferrier and Rogers (1994) "Global Optimization of Statistical Functions with Simulated Annealing," *Journal of Econometrics*, 60 (1/2), pp. 65-100.
- Hahn, T (197?) *Copper Thermal Expansion Study*. National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Hoerl, AE (1962) “Application of Ridge Analysis to Regression Problems”, *Chemical Engineering Progress*, 58, pp. 54-59.
- Hoerl, AE and Kennard, RW (1970) “Ridge Regression: Biased Estimation for Nonorthogonal Problems”, *Technometrics*, 12 (3), pp. 55-67.
- Holland, J (1975) *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor.
- Judge, GG, Griffith, WE, Hill, RC, Lee, CH and Lotkepohl, H (1990) *The Theory and Practice of Econometrics*, John Wiley, New York.
- Kahaner, D, Moler, C and Nash, S (1989) *Numerical Methods and Software*. Prentice Hall, Englewood Cliffs, NJ: pp. 441-445.

- Kirby, R. (1979) *Scanning Electron Microscope Line Width Standards*. National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Kirkpatrick, S, Gelatt, CD Jr., and Vecchi, MP (1983) "Optimization by Simulated Annealing", *Science*, 220, 4598, 671-680.
- Kowalik, JS and Osborne, MR (1978) *Methods for Unconstrained Optimization Problems*. Elsevier North-Holland, New York.
- Lanczos, C. (1956). *Applied Analysis*. Prentice Hall, Englewood Cliffs, NJ, pp. 272-280.
- Levenberg, K (1944) "A Method for the Solution of Certain Non-Linear Problems in Least Squares", *Quart. Appl. Math.* 2, pp. 164-168.
- Lilien, DM (2000) "Review: Econometric Software Reliability and Nonlinear Estimation **In Views: Comment**" *Journal of Applied Econometrics*, 15(1), pp. 107-110.
- Marquardt, D (1963) "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", *SIAM J. Appl. Math.* 11, pp. 431-441.
- Mathworks.com (.) Statistical Toolbox - Example: Nonlinear Modeling Hougen-Watson Model [http://www.mathworks.com/access/helpdesk\\_r13/help/toolbox/stats/nonlin\\_3.html](http://www.mathworks.com/access/helpdesk_r13/help/toolbox/stats/nonlin_3.html)
- Metropolis, N (1987) The Beginning of the Monte Carlo Method. *Los Alamos Science*, No. 15, Special Issue, pp. 125-130.
- Metropolis, N, Rosenbluth, A, Rosenbluth, M, Teller, A, and Teller, E (1953) "Equation of State Simulation Algorithm", *J. Opt. Theory Appl.*, 45, 1, 41-51.
- Meyer, RR (1970) Theoretical and Computational Aspects of Nonlinear Regression. in *Nonlinear Programming*, Rosen, JB, Mangasarian, OL and Ritter, K (Eds). Academic Press, New York, pp. 465-486.
- Mishra, SK (2006) "Fitting a Logarithmic Spiral to Empirical Data with Displaced Origin", SSRN <http://ssrn.com/abstract=897863>
- Misra, D (1978) Dental Research Monomolecular Adsorption Study, National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- More, JJ, Garbow, BS, and Hillstom, KE (1981) Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*. 7(1), pp. 17-41.
- Nelder, JA and Mead, R (1964) "A Simplex Method for Function Minimization" *Computer Journal*, 7: pp. 308-313.
- Nelson, W (1981) Analysis of Performance-Degradation Data, *IEEE Transactions on Reliability*, 2-R-30(2), pp. 149-155.
- NIST (--) Nonlinear Regression [http://www.itl.nist.gov/div898/strd/nls/nls\\_main.shtml](http://www.itl.nist.gov/div898/strd/nls/nls_main.shtml)
- Osborne, MR (1972) Some Aspects of Nonlinear Least Squares Calculations. in *Numerical Methods for Nonlinear Optimization*, Lootsma (Ed). Academic Press, New York, pp. 171-189.
- Rao, CR and Mitra, SK (1971) *Generalized Inverse of Matrices and its Applications*, Wiley, New York.
- Ratkowsky, DA (1983) *Nonlinear Regression Modeling*. Marcel Dekker, New York.
- Roszman, L (19??) Quantum Defects for Sulfur I Atom, National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Storn, R and Price, K (1995) "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces": *Technical Report, International Computer Science Institute, Berkley*.
- Thurber, R (197?) Semiconductor electron mobility modeling. National Institute of Standards and Technology (NIST), US Department of Commerce, USA.
- Tikhonov, AN (1943) "On the Stability of Inverse Problems", *Dokl. Akad. Nauk SSSR*, 39 (5), pp. 195-198.

- Tikhonov, AN (1963) “Solution of Incorrectly Formulated Problems and the Regularization Method”, *Soviet Math Dokl*, 4, pp. 1035-1038 English translation of *Dokl Akad Nauk SSSR*, 151, pp. 501-504.
- Tikhonov, AN and Arsenin, VA (1977) *Solution of Ill-posed Problems*, Winston & Sons, Washington.
- Törn, AA (1978) “A search Clustering Approach to Global Optimization” , in Dixon, LCW and Szegö, G.P. (Eds) *Towards Global Optimization – 2*, North Holland, Amsterdam.
- Törn, AA and Viitanen, S (1994) “Topographical Global Optimization using Presampled Points”, *J. of Global Optimization*, 5, pp. 267-276.
- Wild, J (2001) “Simann.f - Bug in Description of Judge's Function” letter to [netlib\\_maintainers@netlib.org](mailto:netlib_maintainers@netlib.org) and [bgoffe@whale.st.usm.edu](mailto:bgoffe@whale.st.usm.edu), in the Simulated Annealing based Fortran Computer program for nonlinear optimization Simann.f available at <http://netlib2.cs.utk.edu/opt/simann.f>

Note: The author has written his own program (FORTRAN 77). The source codes are available on request to [mishrasknehu@yahoo.com](mailto:mishrasknehu@yahoo.com)