

Hybrid Particle Swarm Optimization and Group Method of Data Handling for Inductive Modeling

Godfrey C. Onwubolu¹, Anurag Sharma², Ashwin Dayal²,
Deepak Bhartu², Amal Shankar², and Kenneth Katafono²

¹Richmond Hill L4C, Canada

²School of Engineering & Physics, University of the South Pacific, Private Bag, Fiji

nwubolu@gmail.com, sharma_au@usp.ac.fj

Abstract. *This paper proposes a new design methodology which is based on hybrid of particle swarm optimization (PSO) and group method of data handling (GMDH). The PSO and GMDH are two well-known nonlinear methods of mathematical modeling. The proposed method constructs a GMDH network model of a population of promising PSO solutions. The new PSO-GMDH hybrid implementation is then applied to modeling and prediction of practical datasets and its results are compared with the results obtained by GMDH-related algorithms. Results presented show that the proposed algorithm appears to perform reasonably well and hence can be applied to real-life prediction and modeling problems.*

Keywords

Inductive modeling, PSO, GMDH, complex systems

1 Introduction

The GMDH is a heuristic self-organizing modeling method which Ivakhnenko [1] introduced as a rival to the method of stochastic approximations. The method is particularly useful in solving the problem of modeling multi-input to single-output data. Hence, GMDH can be utilized as a good data mining tool in which hidden knowledge is extracted from huge data sets for decision making where no *a priori* knowledge is available. The GMDH-type modeling algorithm is self-organizing because the number of neurons, the number of layers and the actual behavior of each created neuron are adjusting during the process of self-organization [4]. The following books are amongst the best in the field for readers who wish to have an understanding of fundamental concepts of GMDH [2—4]. Although GMDH provides for a systematic procedure of system modeling and prediction, it has also a number of shortcomings. Among the most problematic can be stated: (i) a tendency to generate quite complex polynomial (since the complexity of the network increases with each training and selection cycle through addition of new layers) for relatively simple systems (data input); (ii) an inclination to producing overly complex network when dealing with highly nonlinear systems owing to its limited generic structure; (iii) struggling to find good model solution where dimension is very large due to its combinatorial behavior to solve high dimensional problem. Since the introduction of GMDH, there have been variants devised from different perspectives to realize more competitive networks and to alleviate the problems inherent with the standard GMDH algorithm [5-7]. In this paper, we introduce a new hybrid modeling paradigm based on PSO and GMDH which are two well-known nonlinear methods of mathematical modeling.

2 The Group Method of Data Handling

The GMDH was first developed by Ivakhnenko [1] as a multivariate analysis method for complex system analysis modeling and identification. In this way, GMDH was used to circumvent the difficulty of knowing a priori knowledge of mathematical model of the process being considered. Therefore, GMDH can be used to model complex systems without having specific knowledge of the system. In a system identification problem, assume that a single valued output y , of an unknown system, behaves as a function of m input values, such that

$$y = f(x_1, x_2, \dots, x_m) \quad (1)$$

Given n training observations of these input-output data pairs $(x_{ij}; y_i)$, $i = 1, 2, \dots, n$; $j = 1, 2, \dots, m$ then the system identification problem is to approximate the function \bar{f} with an approximate function referred to as the complete form. The GMDH approach is one of the heuristic algorithms primarily designed to solve the system identification problem. According to the GMDH paradigm the relationship between the input vector x and its output can be represented by an infinite Volterra-Kolmogorov-Gabor (VKG) polynomial of the form [1]:

$$y = a_0 + \sum_{i=1}^m a_i x_i + \sum_{i=1}^m \sum_{j=1}^m a_{ij} x_i x_j + \sum_{i=1}^m \sum_{j=1}^m \sum_{k=1}^m a_{ijk} x_i x_j x_k \dots \quad (2)$$

where $a_0, a_i, a_{ij}, a_{ijk} \dots$ are coefficients or weights of this (so called) multiple inputs single output self-organizing network. This is the discrete-time analogue of a continuous time Volterra series and can be used to approximate any stationary random sequence of physical measurements. Ivakhnenko showed that the VKG series can be expressed as a cascade of second order polynomials using only pairs of variables [1][3]. The corresponding network can be constructed from simple polynomial and delay elements. As the learning procedure evolves, branches that do not contribute significantly to the specific output can be pruned, thereby allowing only the dominant causal relationship to evolve.

In the classical GMDH algorithm, there is the need to construct $\alpha = \binom{m}{2} = m(m-1)/2$ new variables $y_1, y_2, y_3, \dots, y_\alpha$, in the *training dataset* for all independent variables (columns of X), two at a time so that there are n data triples of the form $((x_{ir}, x_{is}); y_i)$, $r, s \in (1, 2, \dots, m), s > r; i = 1, 2, \dots, n$. The mathematical description of these two new variables or neurons represented by quadratic polynomials is of the form

$$\hat{y}_i = a_0 + a_1 x_{ir} + a_2 x_{is} + a_3 x_{ir}^2 + a_4 x_{is}^2 + a_5 x_{ir} x_{is} \quad \text{at points } (x_{i,r}, x_{i,s}) \quad (3)$$

The matrix formulation of this system of equation is simply, $\mathbf{Xa} = \mathbf{Y}$ where $\mathbf{a} = [a_0, a_1, a_2, a_3, a_4, a_5]^T$ is the vector of coefficients. The least square technique from multiple-regression analysis provides the formula to obtain the coefficients in the following form: $\mathbf{a} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$.

3 Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) algorithm, inspired by the social behavior of bird flocking or fish schooling was originally designed by Eberhart and Kennedy [8] for solving many kinds of continuous and binary problems of large domain. It is certainly not as powerful as some specific algorithms, but, on the other hand, it can easily be modified for any discrete/combinatorial problem for which good specialized algorithms do not exist [9]. To grasp the abstract phenomenon behind this powerful algorithm, a simple analogy is given. The analogy of PSO involves simulating social behavior among individuals (particles) "flying" through a multidimensional search space, each particle representing a single intersection of all search dimensions. The particles evaluate their positions relative to a goal (fitness) at every iteration; particles in a local neighborhood share memories of their "best" positions, then use those memories to adjust their own velocities, and thus subsequent positions [10]. The original PSO formulae define each particle as potential solution to a problem in D-dimensional space. The position of particle i is represented as

$$X_i = (x_{i1}, x_{i2}, \dots, x_{iD}) \quad (4)$$

Each particle also maintains a memory of its previous best position, represented as

$$P_i = (p_{i1}, p_{i2}, \dots, p_{iD}) \quad (5)$$

A particle in a swarm is moving; hence, it has a velocity, which can be represented as

$$V_i = (v_{i1}, v_{i2}, \dots, v_{iD}) \quad (6)$$

At each iteration, the velocity of each particle is adjusted so that it can move towards the neighborhood's best position known as $lbest(P_i)$ and global best position known as $gbest(P_g)$ attained by any particle present in the swarm [10]. After finding the two best values, each particle updates its velocity and positions according to (7) and (8) weighted by a random number c_1 and c_2 whose upper limit is a constant parameter of the system, usually set to value of 2.0 [11]. The cognitive influence factor is known as c_1 and c_2 is social influence factor.

$$V_i(t+1) = V_i(t) + c_1 \cdot (P_i - X(t)) + c_2 \cdot (P_g - X_i(t)) \quad (7)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (8)$$

All swarm particles tend to move towards better positions; hence, the best position (i.e. optimum solution) can eventually be obtained through the combined effort of the whole population. The methodology of obtaining the velocity vector $v_i(t+1)$ is illustrated in Fig. 1, and it has been observed that particle i has moved towards the neighborhood's best position and its own best position. Fig. 1 illustrates the operators' modification until the finest form is not obtained. A few points can be noted during the modification: (1) the new position is converging towards the best position region; (2) the velocity could explode towards infinity. Hence new position on that situation is undefined.

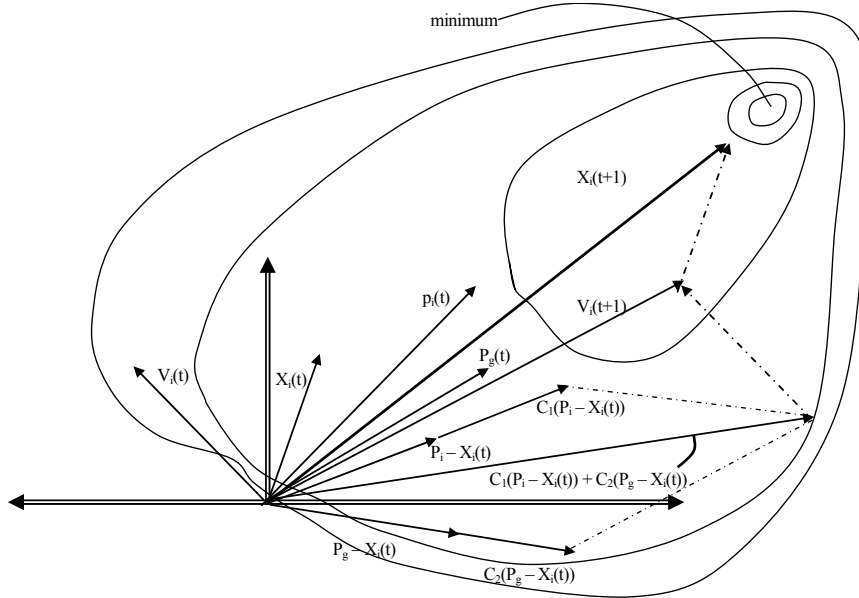


Fig. 1 Contour Lines and the Process for Generating New Locations in PSO Scheme

Constriction parameters have a great impact on the operations. Varying these parameters has the effect of varying the strength of the pull "towards" the two best position, which could be verified by observing Fig. 1. If accidentally the coefficients c_1 and c_2 exceeds the value 4.0, both the velocities and positions explode towards infinity. Thus almost all implementation of the particle Swarm limit each of the two coefficient c_1 and c_2 to 2.0 [11]. To control the explosion of the system a new constriction coefficient is used, which is called inertia weight. A large inertial weight facilitates global exploration while a small inertial weight tends to facilitate local exploration to fine tune the current search area [13]. Hence, equation 7 can be modified to have a place for new constriction constant, inertial weight (α) [14]:

$$V_i(t+1) = \alpha \cdot V_i(t) + c_1 \cdot (P_i - X(t)) + c_2 \cdot (P_g - X_i(t)) \quad (9)$$

where inertial weight (α) = $\frac{k}{abs\left[\frac{1 - \frac{a}{2} - \sqrt{|a^2 - 4a|}}{2}\right]}$; $k \in (0, 1]$ and $a = c_1 + c_2$ such that $a > 4$.

Equation 9 is generic operator for all types of objective functions. This is to tune up these coefficients for any particular kind of problem domain. The contents of this Section are mostly from our work [17].

4 The proposed hybrid PSO-GMDH Algorithm

The new PSO-GMDH algorithm that is proposed in this paper consists of two components: (i) the PSO structural optimization module; (ii) the GMDH parametric optimization module.

4.1 Structural Optimization with PSO

As mentioned above in section 3 PSO uses swarm particles as its unit search agent. The position (sequence of numbers) of each particle is used to determine which salient combinations of input variables of previous layer to be moved to next layer. Each particle must contain 3 system parameters (P_1, P_2, P_3). $P_1 \in [1, 3]$ is randomly generated and represents the order of polynomial that will be generated from previous layer, which can be either 2 or 3 but for simplicity we normally take 2 in each layer. $P_2 \in [1, r], r = \min(D, 5)$ is again randomly generated number which determines number of input variables to be taken from previous layer where D is the width of the input dataset; the default lower bound is $r = 2$. $P_3 = \{a \in Z^+ | 1 \leq a \leq D\}$ is a sequence of integers stored as position of a particle represents the entire candidates in the current layer of the network. The determination of combinations of nodes to be moved to next layer is determined by all three parameters. Firstly P_1 determines the order of polynomial to be formed then P_2 determines the number of combinations of nodes forming the polynomial which in turn will be selected from P_3 as first P_2 nodes from the whole sequence. Fig. 2 depicts the usage of all 3 system parameters to determine the polynomial, and shows PSO-GMDH process of each layer. Each particle consists of separate set of system parameters which are used to generate the polynomial. These polynomials are actually used as objective function for PSO. It is a candidate for model function which determines how promising the model is.

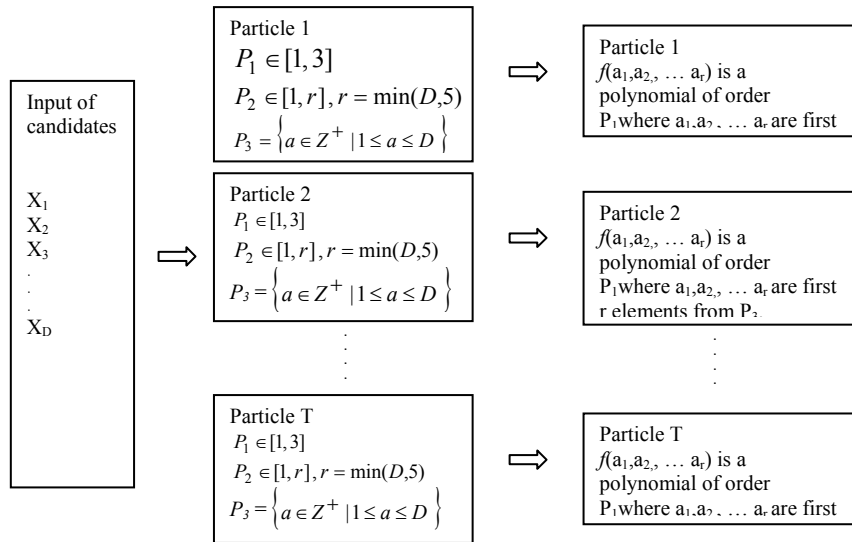


Fig. 2. Formation of polynomials by Swarm Particles using 3 System Parameters P_1, P_2, P_3 .

To understand network realization let us take an example. Suppose we have input dataset of 5 dimensional vectors (record). Each vector has only and only one output. PSO-GMDH process would require number of particles to be specified by the user. We can take 10, 15, 20 or any other appropriate integer as the size of the swarm particles. Here we can take just 5. Take a scenario where a particle determines the type of polynomial using its system parameters. Suppose value of P_1 is 2 which means polynomial will be of order 2. Value of P_2 is also 2 which will tell the particle to choose first 2 integers from the sequence of integers stored in parameter P_3 . Value of P_3 is given in Table 1.

Tab. 1. Values of P_3

Particles	Parameter P_3 (or position of corresponding particles)	Parameter P_2 (2 nodes to make a polynomial)
Particle 1	1, 3, 5, 4, 2	1, 3
Particle 2	5, 4, 3, 2, 1	5, 4
Particle 3	2, 1, 3, 5, 4	2, 1
Particle 4	4, 3, 1, 5, 2	4, 3
Particle 5	1, 5, 2, 4, 3	1, 5

4.2 Parametric Optimization with GMDH

The objective function (performance index) is a basic instrument guiding the evolutionary search in the solution space [26]. For particle 3 (see Table 1) the generated polynomial would be:

$$f(x_2, x_1) = c_1 + c_2 x_2 + c_3 x_1 + c_4 x_2 x_1 + c_5 x_2^2 + c_6 x_1^2$$

where c_1, c_2, \dots, c_6 are the constants evaluated using training dataset. As discussed in Section 2, the least square technique from multiple-regression analysis provides the formula to obtain the coefficients in the following form: $c = (X^T X)^{-1} X^T Y$.

4.3 Procedure of the General Learning PSO- GMDH Algorithm

The general learning procedure for constructing the PSO-GMDH model can be described as follows:

- (1) Create an initial population randomly (PSO structures and its corresponding learning parameters c_1 and c_2).
- (2) Structural optimization is achieved by the PSO variation operators described in Section 3.
- (3) If better structure is found, then go to step 4; otherwise go to step 2.
- (4) Parametric optimization is found using the least square technique from multiple-regression analysis.
- (5) If the maximum number of local is reached or no better parameter vector is found for a significantly long time, then go to step 6; otherwise go to step 4.
- (6) If a satisfactory solution is found, then the algorithm is stopped; otherwise go to step 2.

4.4 User Interface of the General Learning PSO- GMDH Algorithm

The PSO-GMDH software developed is flexible and user-friendly. Development environment was based on the following tools: C++ programming language, Borland C++ version 5.02 compiler, Borland Object Windows Library 5.0 (OWL) for graphical user interface, Visual Basis 6.0, and Microsoft Office suite for system requirements documentation.

5 Experimentation

The problem solved is the "Box and Jenkins furnace data" [28] which is a benchmark problem often reported in the literature. There are originally 296 data points $\{y(t), u(t)\}$, from $t=1$ to $t=296$. In this problem, $y(t)$ is the output CO2 concentration and $u(t)$ is the input gas flow rate. Here we are trying to predict $y(t)$ based on $\{y(t-1), y(t-2), y(t-3), y(t-4), u(t-1), u(t-2), u(t-3), u(t-4), u(t-5), u(t-6)\}$. This reduces the number of effective data points to 290. Most methods find that the best set of input variables for predicting $y(t)$ is $\{y(t-1), u(t-4)\}$. Sugeno and Yasukawa has found that the best set of input variables for predicting $y(t)$ is $\{y(t-1), u(t-4), u(t-3)\}$. The first column below is the output variable $y(t)$, the remaining columns are the input variables $\{y(t-1), y(t-2), \dots, u(t-6)\}$. Consequently, we have the inputs and output as: output $y(t)$; input $y(t-1) y(t-2) y(t-3) y(t-4) u(t-1) u(t-2) u(t-3) u(t-4) u(t-5) u(t-6)$

5.1 PSO-GMDH modeling

The parameters used for the modeling of this problem are:

Swarm size	0
Maximum best column	0
Even row selection	1
Maximum tour	9000000
Intensity 'n' = no; 'y' = yes	n
Maximum evaluation	4
Convergence case	5

The number of swarm size is given by the user but when PSO-GMDH is required to determine this automatically the value of '0' is given. This is the case for the maximum best column also. For the selection for the test data, three cases exist: random (0), even (1), and odd (2). When random selection is made, PSO-GMDH randomly selects data for testing but when 1 or 2 is selection even or odd rows of data are selected for testing. When a brute-force search is required, then intensity is 'yes' or 'y' otherwise, the choice is 'no' or 'n'. The maximum evaluation is set by the user or left to PSO-GMDH to determine if '0' is chosen. There are five convergences cases to choose from. The results generated by the PSO-GMDH after each iteration are reported as:

After iteration 1 the best particle is particle [1]
position: 7 2 1 9 6 5 3 8 4 10 7

velocity: (size = 1) : magnitude [1 , 7]
 objective functional value: 0.0801373
 distance: 0
 After iteration 2 the best particle is particle [7]
 position: 1 6 3 4 2 5 7 8 9 10 1
 velocity: (size = 6) : magnitude [9 , 5] [5 , 4] [2 , 6] [4 , 9] [4 , 2] [9 , 5]
 objective functional value: 0.0714395
 distance: 0
 After iteration 3 the best particle is particle [7]
 position: 1 6 3 4 2 9 7 8 5 10 1
 velocity: (size = 6) : magnitude [9 , 5] [5 , 4] [2 , 6] [4 , 9] [4 , 2] [9 , 5]
 objective functional value: 0.0687775
 distance: 0
 After iteration 4 the best particle is particle [7]
 position: 1 6 3 4 2 9 7 8 5 10 1
 velocity: (size = 6) : magnitude [9 , 5] [5 , 4] [2 , 6] [4 , 9] [4 , 2] [9 , 5]
 objective functional value: 0.0687775

Figure 3 shows the measured and estimated outputs for the gas furnace problem. The training error of 0.126 and the testing error of 0.068 are displayed. These errors are based on mean square error (MSE). The testing error is often used in the literature to determine the efficacy of a modeling approach; this gives more information than the graph. Figure 4 shows the GMDH network for the gas furnace problem. There are 10 inputs and one output, but we have only three inputs shown in the network. The explanation is simple. Only parameters 2, 3 and 10 are connected to the three best nodes in layer 1. This means that the objective functions of pair-wise combinations of parameters 1, 4, 5, 6, 7, 8, and 9 do not give competitive nodes, so those nodes are relegated to the background; in other words they are dropped.

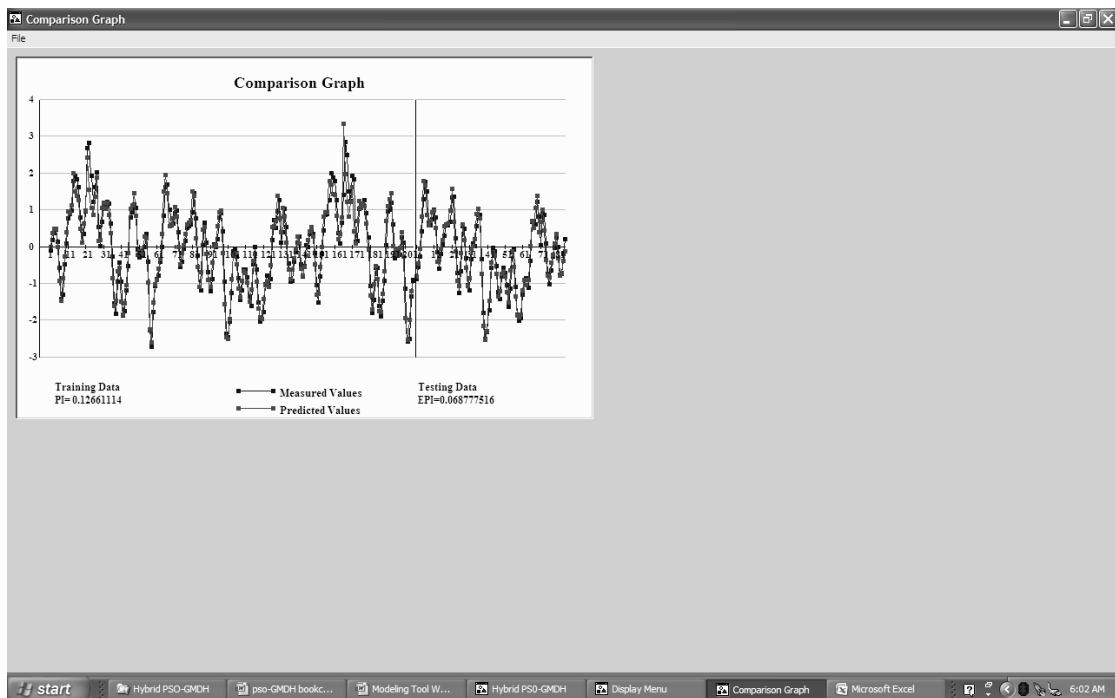


Fig. 3. Measured and estimated outputs for the gas furnace problem

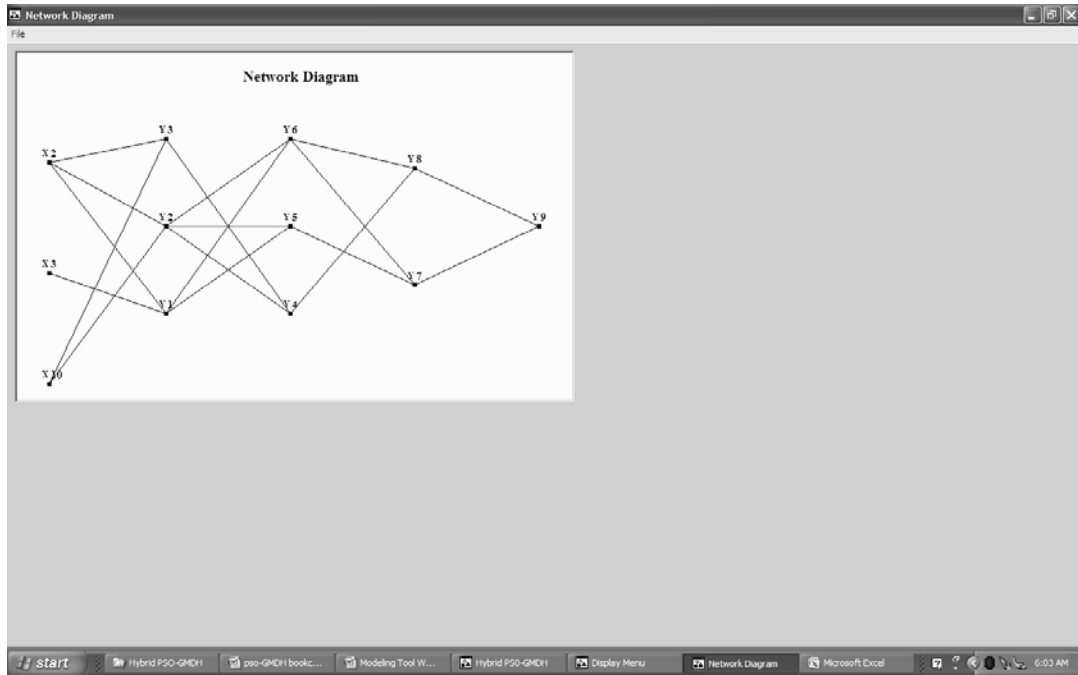


Fig. 4. GMDH network for the gas furnace problem

5.2 PSO-GMDH database

Table 2 shows the database of output results for the gas furnace problem. This database is rich in information because it gives the network topology and the equations for each node. A bit of interpretation of the output data is necessary. Whereas the network shows that y_1 is connected to x_2 and x_3 , y_2 is connected x_2 and x_{10} while y_3 is connected x_2 and x_{10} while the database gives the true interpretation that y_7 is connected to x_2 and x_3 , y_2 is connected x_2 and x_{10} while y_1 is connected x_2 and x_{10} . The differences are merely due to the ways of numbering on the network and internally in the database. The database gives the exact connections but the network labels are sequential for the active neurons. Non-active neurons are not numbered on the network but internally the numbering is maintained for the database archiving.

Tab. 2. Database of output results for the gas furnace problem

PARTICLE SWARM OPTIMIZATION									
Optimum value	0.068778								
Best Layer at	3								
PI (training):	0.068778								
EPI (testing):	0.126611								
Network of Layers					Coefficients				
7	2	3	-0.99279	0.315822	0.365218	-0.00781	0.001328	6.38E-11	
2	10	2	-0.99713	-0.0828	0.018472	0.018933	0.055631	1.32E-06	
1	10	2	-0.99713	-0.0828	0.018472	0.018933	0.055631	1.32E-06	
1	7	2	-0.9924	-1.67656	0.083294	0.063762	0.103198	0.002982	
6	1	2	-2.6647	-3.86E-06	1.13451	1.03578	1.89E-05	-3.96E-08	
2	7	2	-0.9924	-1.67656	0.083294	0.063762	0.103198	0.002982	
1	1	6	-1.00367	1.35655	-0.34384	0.048684	0.282728	-0.00344	
6	1	2	-2.4166	8.10E-06	1.26712	0.730777	2.12E-05	-5.85E-08	
1	1	6	-1.00337	1.51282	-0.43421	-0.04207	0.375178	0.000249	
Processing time: 8.188 seconds; Iterations: 4; Evaluations: 872									

6 Conclusions

In this paper, both methods of PSO and GMDH are explained and a new design methodology which is a hybrid of *PSO* and *GMDH*, which we refer to as *PSO-GMDH*, is proposed. The architecture of model is not predefined, but can be self-organized automatically during the design process. We have applied the *PSO-GMDH* approach to the problem of developing predictive model for time-series problem; it has been applied to other practical problems. The predictive models based on *PSO-GMDH* network give reasonably good solutions. In our approach, we first present a methodology for modeling, and then develop predictive model(s) of the problem being solved in form of second-order equations based on the input data and coefficients realized. It was previously concluded from previous study [16] that population-based optimization techniques (genetic programming [GP], genetic algorithm [GA], differential evolution [DE], scatter search [SS], ant colony system [ACS], particle swarm optimization [PSO], etc.) are all candidates of hybridization with GMDH. In the past, only the use of GA, GP, and DE has been studied for hybridization with GMDH; this paper extends the catalog to include PSO. Further research activities include incorporating better parameter optimization techniques such as singular value decomposition (SVD) in order to improve the modeling solutions.

References

- [1] Ivakhnenko A. G.: The Group Method of Data Handling-A rival of the Method of Stochastic Approximation. *Soviet Automatic Control*, vol 13 c/c of *avtomatika*, 1, 3, 43-55, 1968.
- [2] Farlow, S. J. (ed.): *Self-organizing Methods in Modeling. GMDH Type Algorithms*. Marcel Dekker. New York, Basel, 1984
- [3] Madala, H.R.; Ivakhnenko, A.G.: *Inductive Learning Algorithms for Complex Systems Modeling*. CRC Press Inc., Boca Raton, Ann Arbor, London, Tokyo, 1994
- [4] Mueller, J-A., and Lemke, F.: *Self-Organizing Data Mining: An Intelligent Approach to Extract Knowledge From Data*, Dresden, Berlin, 1999.
- [5] Iba, H., de Garis, H., Sato, T.: Genetic programming using a minimum description length principle, In *Advances in Genetic Programming*, Kinnear, K. E. Jr (ed), Cambridge: MIT, 1994, pp 265-284.
- [6] Nariman-Zadeh, N., Darvizeh, A., and Ahmad-Zadeh, G. R.: Hybrid genetic design of GMDH-type neural networks using singular value decomposition for modeling and predicting of the explosive cutting process, Nariman-Zadeh, *Proc. Instn Mech. Engrs Vol 217 Part B: Nariman-Zadeh*, 779-790.
- [7] Onwubolu, G. C.: Design of hybrid differential evolution and group method in data handling networks for modeling and prediction, *Information Sciences*, 178, 3618-3634, 2008, doi:10.1016/j.ins.2008.05.013.
- [8] Eberhart, R. C. and Kennedy, J.: A new optimizer using particle swarm theory, in *Proc. Sixth International Symposium on Micro Machine and human science*, Nagoya, Japan, IEEE Service Center, Piscataway, 1995
- [9] Clerc, M.: Discrete particle swarm optimization illustrated by the traveling salesman problem," *New Optimization techniques in Engineering*. Springer-Verlag, Berlin, Germany, 2004.
- [10] A. Carlisle, and G. Dozier, *Adapting Particle Swarm Optimization to Dynamic Environments*. Available at <http://www.CartistleA.edu>, 1998.
- [11] Kennedy, J. and Eberhart, R. C.: The particle swarm: social adaptation in information processing systems," In *Corne, D., Dorigo, M., and Glover, F., Eds., New Ideas in Optimization*. London: McGraw-Hill, pp. 379-387, 1999.
- [12] Kennedy, J. and Eberhart, R. C.: A discrete binary version of the particle swarm algorithm," *International Conference on Systems, Man, and Cybernetics*, 1997.
- [13] Kennedy, J.: The particle swarm: social adaptation of knowledge, *IEEE international conference on Evolutionary computation*, indianapolis, Indiana, IEEE Service Center, Piscataway, NJ, 1997.
- [14] Clerc, M. and Kennedy, J.: "The particle swarm: explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, Vol. 6, 58-73, 2002.
- [15] Box, G. E. P., and Jenkins, G. M., *Time Series Analysis, Forecasting and Control*
San Francisco, Holden Day, 1970, pp. 532-533.
- [16] Onwubolu, G. C.: Design of hybrid differential evolution and group method in data handling for modeling, *International Workshop on Inductive Modeling, IWIM 2007*, Prague, Czech, September 23-26, 2007, pp.87-95.
- [17] Onwubolu, G. C. and Sharma A.: *Particle Swarm Optimization for the assignment of facilities to locations*, *New Optimization Techniques in Engineering*, Springer-Verlag, 2004.