# Training RBF Neural Network by hybrid Adaptive Modified Particle Swarm Optimization – Tabu Search algorithm for Function Approximation

Mohammad Alipour Varmazabadi, Ali Solyemani Aiouri

**Abstract**—This days interests in metaheuristic optimization algorithms have been grown and this algorithms have been used in many fields . Because of this , many algorithms have born or mixed up together in order to get into better results . In this paper a new hybrid AMPSO-TS ( Adaptive Modified Particle Swarm Optimization – Tabu Search ) method is presented for training RBF (Radial Basis Function ) neural network that used to function approximation . This AMPSO-TS method uses the search ability of Tabu Search algorithm to optimize the Adaptive Modified Particle Swarm Optimization to result in more accurate . This hybrid method is used for learning RBF neural network and uses it for function approximation . Both PSO and TS method use the strategy of best solution of neighborhood in every iteration to get into best result , but every one of them has issues . By use of advantages of this methods we proposed the new method that uses TS to optimize the constants of PSO . numerical result show that the proposed method has improved results contrast to the simple PSO and gradient descent method to learn the RBF neural Network .

**Index Terms**— Function Approximation , metaheuristics , neural network , Particle Swarm Optimization , RBF , Tabu Search , gradient descent

## 1 INTRODUCTION

### 1.1 Function Approximation

Function approximation is needed in many fields , such as applied mathematics and computer science . It usually used to choose a function among the classes of functions or to estimate a function from a database that contain some inputs and their relative outputs .

As we said there are two major classes of function approximation : In first way we know the target function and we try to investigate how exact we could approximate the target function by a specific class of functions like polynomials or rational functions .

The other class contain a set point of input and output (x , g(x)) instead of target function named g that is unknown . Depends on the set point structure several techniques of approximating can be applied .

Some of famous techniques of function approximation are polynomial approximation that contain methods like Taylor expansion that when set points contain the function value and n derivatives at the same point had been used , Projection method that has been used when sat point contain n+1 arguments and n+1 function value at the same arguments , Interpolation techniques that contain methods like Lagrange interpolation that uses Lagrange coefficients to approximate the function , Hermite interpolation has been used when set points contain function value and derivatives value at the points [20].

Even there are artificial intelligence methods to approximate the functions that almost uses neural networks to do the job .

In this paper we have used this method by the RBF neural network to approximate the functions and tried to minimize the approximation error by the proposed algorithm .

### 1.2 RBF Neural Network

Neural network generally refer to the networks that simulate the neuron behavioral . This networks could be biological that are made of biological neurons that make the nervous system or even could be artificial that contain artificial neurons to simulate the nervous system and use it in many fields , such this paper that we used to approximation of functions .

This method don't need to know the whole model parameters even if the system be too complicated and could get to the proper result and that is a great advantage in contrast the other method .

- *Mohammad Alipour Varmazabadi is currently pursuing masters degree program in digital electronic engineering in SUT University. E-mail: m.alipour.v@gmail.com*
- *Ali Solyemani Aiouri is professor in electrical engineeringschool in SUT University. E-mail: soleamani_ali@yahoo.com*

The basic idea of neural network was proposed by Alexander Bain (1873) and William James (1890) . For Bain , every activity led to the firing of a certain set of neurons. When activities were repeated, the connections between those neurons strengthened , James's theory was similar to Bain's, however, he suggested that memories and actions resulted from electrical currents flowing among the neurons in the brain.

McCullouch and Pitts (1943) created a computational model for neural networks based on mathematics and algorithms. They called this model threshold logic.

In the late 1940s psychologist Donald Hebb created a hypothesis of learning based on the mechanism of neural plasticity that is now known as Hebbian learning .

Farley and Clark (1954) first used computational machines , then called calculators, to simulate a Hebbian network at MIT. Other neural network computational machines were created by Rochester , Holland, Habit , and Duda (1956) .

Rosenblatt (1958) created the perceptron, an algorithm for pattern recognition based on a two-layer learning computer network using simple addition and subtraction .

After this Neural Networks had a lot improvement and successfully applied to the artificial intelligence fields such as statistical estimation , classification , speech processing , image processing , adaptive control , software agents and so on .

In a neural network usually there is 3 layer : Input layer , hidden layer , output layer . Every layer consist of artificial neurons that are connected to the other neurons of other layers . this connection could be fully connected or partially connected . The coefficient of neuron connections is called weight that by a process we get to the proper weight for every connection that this process is called learning . In Fig 1 we see a schematice of a neural network .

RBF neural network is an artificial neural network that uses radial basis functions as activation functions in hidden layer . output of the network is calculated by (1), and the Gaussian function that is used in hidden layer presented in (2):

N : the number of neurons in hidden layer
ci : center vector of neuron i
αi : weights of linear output neuron

$$\varphi(x) = \sum_{i=1}^{N} \alpha_i \rho(||x - c_i||) \quad (1)$$

ρ : Gaussian basis function

$$\rho(||x - c_i||) = \exp(-\beta||x - c_i||^2) \quad (2)$$

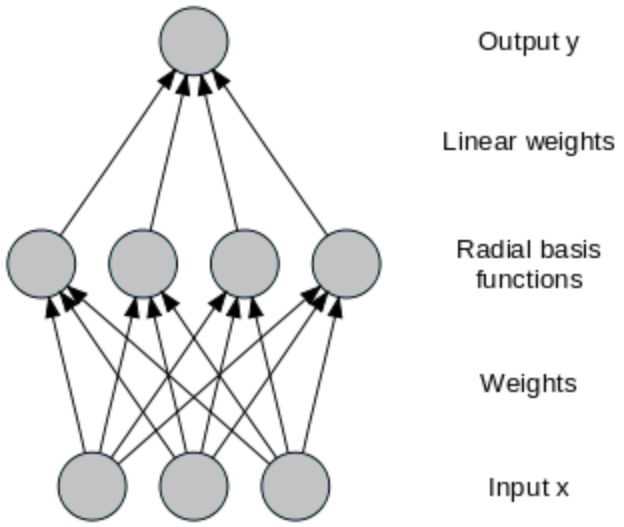


Fig 1 : Architecture of a radial basis function network

In RBF networks in Training phase three types of parameters became updated to get into the proper value . this parameters are center vectors ci , output weights ωi , and the RBF width parameters βi . there are several methods to train the network that are more general such as back propagation and gradient descent .

In this paper we have used Tabu Search algorithm to improve the result of the PSO in contrast with the simple PSO and gradient descent method for training the RBF neural network in order to Function Approximation .

The rest of this paper organized as follows ; section 2 presents basic PSO and TS and introduces the AMPSO – TS method. Section 3 presents and discusses the results . Section 4 is conclusions .

## 2 Adaptive Modified Particleswarm Optimization-TS

### 2.1 Particle Swarm Optimization

Particle swarm optimization is a meta heuristic algorithm that simulate the behavioral of birds flock or fish school . First it was proposed by Kenny and Eberhart as a optimization algorithm . In this algorithm every particles position is updated according to the swarms best position and the same particles best position ever around the search space .

PSO's particle in first round usually get random values and first value of velocity is set to zero , then the swarm value place in target function and best answer among the swarm chosen as global best , even for every particle if recent position be better than the previous one particle's best position have been updated . By use of new information new velocity will be updated by (3) and new position will be calculate by (4).

Where w is weight constant that relate the previous velocity to the new one and c1 and c2 are constant that follow relation in (5) and r1 and r2 are random values between 0 and 1 [1].

$$v_d^i(k+1) = w.v_d^i(k) + c_1.r_1.\left(p_d^i - x_d^i(k)\right) + c_2.r_2.\left(g_d^i - x_d^i(k)\right) \quad (3)$$

(6)

$$x_d^i(k+1) = x_d^i(k) + v_d^i(k+1) \quad (4)$$

$$c_1 + c_2 < 4 \quad (5)$$

PSO algorithm will continue till the iterations end or the fitness value get into the proper value .
The complete PSO algorithm is followed [2]:

- Initialize the particles
- For every iteration
- Calculate target function for every particles
- If the recent value of the particle is better than previous one update the $p_{best}$ and set it by this value
- Choose the best particle of swarm that has best fitness value as $g_{best}$
- Calculate the velocities by the equation
- Calculate the particles position by the equation
- End

PSO has many optimization applications in different fields , and we use it here for training the RBF neural network .

### 2.1 Tabu Search

Tabu search is an iterative optimization algorithm like PSO that first introduced by Glover in 1986 [] . unlike PSO that may fall in local optimum , TS uses memory structure and tabu list strategy that don't let algorithm stock in local . In TS we use neighborhood strategy to select the candidate list in every iteration . After making the solution list , every solution will check in tabu list , if it wasn't tabu active it could be one of the answers then we put it in target function and if the result had proper fitness value we update the fitness condition and then we put this solution on tabu list to prevent falling in local or loop . Best answer in the neighborhood will be chosen as center of neighborhood in the next iteration and this will continue until stopping criteria has been satisfied . To prevent of loop or local optimum every member of tabu list will be active for constant period of iteration after that it will be free to select .
The following pseudo code shows the TS algorithm [10,11] :

- Input: Tabu List size
- Output: Sbest
- Construct Initial Solution and put in Sbest
- Empty the Tabu List
- While ( Stop Condition())
- Empty Candidate List
- For (Candidate swarm ϵ Sbest neighborhood )
- If any of candidate swarm isnot in the Tabu List
- Put Candidate swarm in the Candidate List
- End
- End
- Generate Candidate List
- Find best answer in the swarm and get it as new Candidate
- If value of Candidate is better than the best answer so far
- put Candidate as Sbest
- Put the Candidate List in the Tabu List
- While Tabu List is greater than Tabu list size
- Delete extra feature in the Tabu List
- End
- End
- End
- Return Sbest

### 2.2 Proposed method

Simple PSO has some issues that could cause the divergence in the solution . One of them is velocity , that if we couldn't handle it might get very large values and take away the solutions from optimum value . The other factor that effect on the final solution are the constants w , c1 and c2 . By choosing the proper value for this constant the convergence could be faster and more accurate . So in this paper we used the Modified PSO by modified velocity [] and by considering it's constant as adaptive values and entering the TS algorithm to do it proposed a new method that called Adaptive Modified Particle Swarm Optimization - TS or AMPSO – TS .
First we discuss about modifying the velocity .

Because of random generation of $r_1$ and $r_2$, this values could be both too large or too small . If both be too large then both of personal and social experience of particle (respectively $p_{best}$ and $g_{best}$ ) become overused and it could drive the particle far from the local optimum .
And if both of random values be too small use of personal and social experience is not enough to update the proper velocity and convergence would not be low . So we define two values for maximum and minimum velocity and update the velocity as follows (6,7,8) [21] :

$$v_d^i(k+1) = r_2.v_d^i(k) + c_1.r_1.(1-r_2).\left(p_d^i - x_d^i(k)\right) + c_2.(1-r_2).(1-r_1).\left(g_d^i - x_d^i(k)\right) + c_2.r_2.\left(g_d^i - x_d^i(k)\right) \quad (6)$$

$$v_d^i(k+1) = \frac{v_d^i(k+1).v_d^{max}}{|v_d^i(k+1)|}, \quad if\ |v_d^i(k+1)| > v_d^{max} \quad (7)$$

$$v_d^i(k+1) = \frac{v_d^i(k+1).v_d^{min}}{|v_d^i(k+1)|}, \quad if\ |v_d^i(k+1)| < v_d^{min} \quad (8)$$

Now after modifying the PSO we enter the TS algorithm to make the PSO as an adaptive method .

By choosing constant values for the c1 and c2 we define the effect of previous velocity , distance from pbest and gbest on the new velocity . As we get more close to the fitness value it might be needed to reduce or increase the effect of distance from pbest and gbest . So we have used Tabu Search algorithm as a stage of PSO .

We see the pseudo code of proposed AMPSO-TS algorithm as follows :

- Initialize the particles
- Initialize the w, $c_1$ and $c_2$
- Initialize the Tabu List
- For every iteration
- Calculate target function for every particles
- If the recent value of the particle is better than previous one update the $p_{best}$ and set it by this value
- Choose the best particle of swarm that has best fitness value as $g_{best}$
- Origin the neighborhood of w , $c_1$ and $c_2$
- For every solution of neighborhood
- If the solution is not Tabu calculate the fitness value
- Choose the best value
- Update the Tabu List
- Calculate the velocities by the equation
- Calculate the particles position by the equation
- End

As we said the proposed method has been used to train the RBF neural network for Function Approximation . So for every function we use the weights and width and center of Gaussian radial basis function as swarm of PSO and we train the network to get to the fitness value . In this method we put the algorithm in exchange of weight updating , Gaussian function center updating and the Gaussian function width updating formula and use the $g_{best}$ value as new updated value of weight , center and width . So in every iteration of RBF neural network we run the algorithm for all the inputs . The input of proposed training method will be networks input data and target function is followed by the input function value . At first iteration of neural network weights , centers positions and width will be generated by random but at the next iterations last weights , centers positions and width of previous iteration would be beginning weights . Training process will continue until stopping criteria has been satisfied .

Pseudo code of Training phase of RBF neural network is followed :

- Initialize the parameters
- For every neural network iteration
- For every input
- Calculate the radial basis function
- Run the AMPSO-TS algorithm
- End
- Update the parameters
- End.

## 3 NUMERICAL RESULTS

To test the proposed algorithm , we compared this method with Gradient descent method and simple PSO for training the RBF neural network . The Gradient descent method exposed to the RBF with 500 iteration . The simple PSO that we used to train has swarm size equal to 64 , with w=0.75 , c1=c2=1.9 . The PSO include 30 iteration and stopping criteria of neural network is 500 iteration . The proposed AMPSO – TS method parameters include , swarm size equal to 64 , the PSO iterations is equal to 20 , TS iterations is 10 , TS neighborhood aspect size is equal to 5 , Tabu period = 5 , and c1and c2 are chosen with Tabu Search part of algorithm .

We have used 20 different function to test the algorithm that are listed in appendix .

Numerical result after training the network has shown in table 1. As we see simple PSO has better answers than Gradient descent and proposed AMPSO – TS method has better answers than simple PSO and it's obvious that convergence rate in simple PSO and proposed method AMPSO – TS is so better than Gradient descent because this methods get to the answer in much less iterations . we compare the convergence rate of simple PSO and AMPSO – TS methods for some of selected functions in table 2 . The comparison is based on sum of error percentage on whole test data . Finally we see the pictorial comparision of some selected functions in fig 2.

TABLE 1
MEAN OF ERROR PERCENTAGE FOR DIFFERENT ALGORITHMS

| $f$ | GD | PSO | AMPSO-TS |
|---|---|---|---|
| AK | 4.9619 | 3.7880 | 3.4576 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| BL | 1.9160 | 0.5872 | 0.4677 | CV | 10.0810 | 9.6288 | 9.5487 | 9.8236 | 9.5014 | 9.5014 |
| L | 2.7569 | 1.1342 | 0.9120 | DJ | 4.2255 | 2.7786 | 2.6608 | 2.9303 | 2.4737 | 2.2504 |
| MT | 87.5128 | 15.0667 | 10.0092 | DP | 6.5757 | 6.4659 | 5.2926 | 4.3114 | 3.1829 | 2.2937 |
| BO | 0.9779 | 0.6793 | 0.4516 | Z | 0.8654 | 0.6185 | 0.4204 | 0.3005 | 0.1977 | 0.1428 |
| RT | 11.6295 | 4.7561 | 2.6103 | | | | | | | |
| RC | 4.0158 | 0.4138 | 0.3810 | | | | | | | |
| R | 4.0816 | 1.0023 | 0.7781 | | | | | | | |
| CV | 10.6003 | 9.5381 | 9.5014 | | | | | | | |
| SC | 0.5687 | 0.5534 | 0.1088 | | | | | | | |
| DJ | 3.3655 | 2.1313 | 2.0504 | | | | | | | |
| SR | 13.6399 | 4.4190 | 3.2671 | | | | | | | |
| DP | 13.7654 | 5.2163 | 2.2937 | | | | | | | |
| SS | 11.3544 | 6.6851 | 5.6404 | | | | | | | |
| GP | 29.4366 | 7.4211 | 2.8106 | | | | | | | |
| T | 12.0993 | 8.7235 | 5.9420 | | | | | | | |
| G | 47.0722 | 4.9003 | 0.8426 | | | | | | | |
| Z | 0.3127 | 0.2925 | 0.0913 | | | | | | | |
| HM | 38.2086 | 32.0129 | 29.4768 | | | | | | | |
| B | 117.0404 | 78.4538 | 62.1267 | | | | | | | |

TABLE 2
COMPARISON OF CONVERGENCE OF PSO AND AMPSO-TS

| | PSO (Iterations) | | | AMPSO-TS (Iterations) | | |
|---|---|---|---|---|---|---|
| *f* | 10 | 20 | 30 | 5 | 10 | 15 |
| AK | 4.8876 | 4.4241 | 4.1503 | 3.4576 | 3.4576 | 3.4576 |
| BL | 0.6954 | 0.6763 | 0.5872 | 0.4677 | 0.4677 | 0.4677 |
| BO | 0.8084 | 0.7678 | 0.6793 | 0.5047 | 0.4633 | 0.4572 |
| RC | 0.8910 | 0.6740 | 0.4203 | 0.8546 | 0.7727 | 0.5810 |

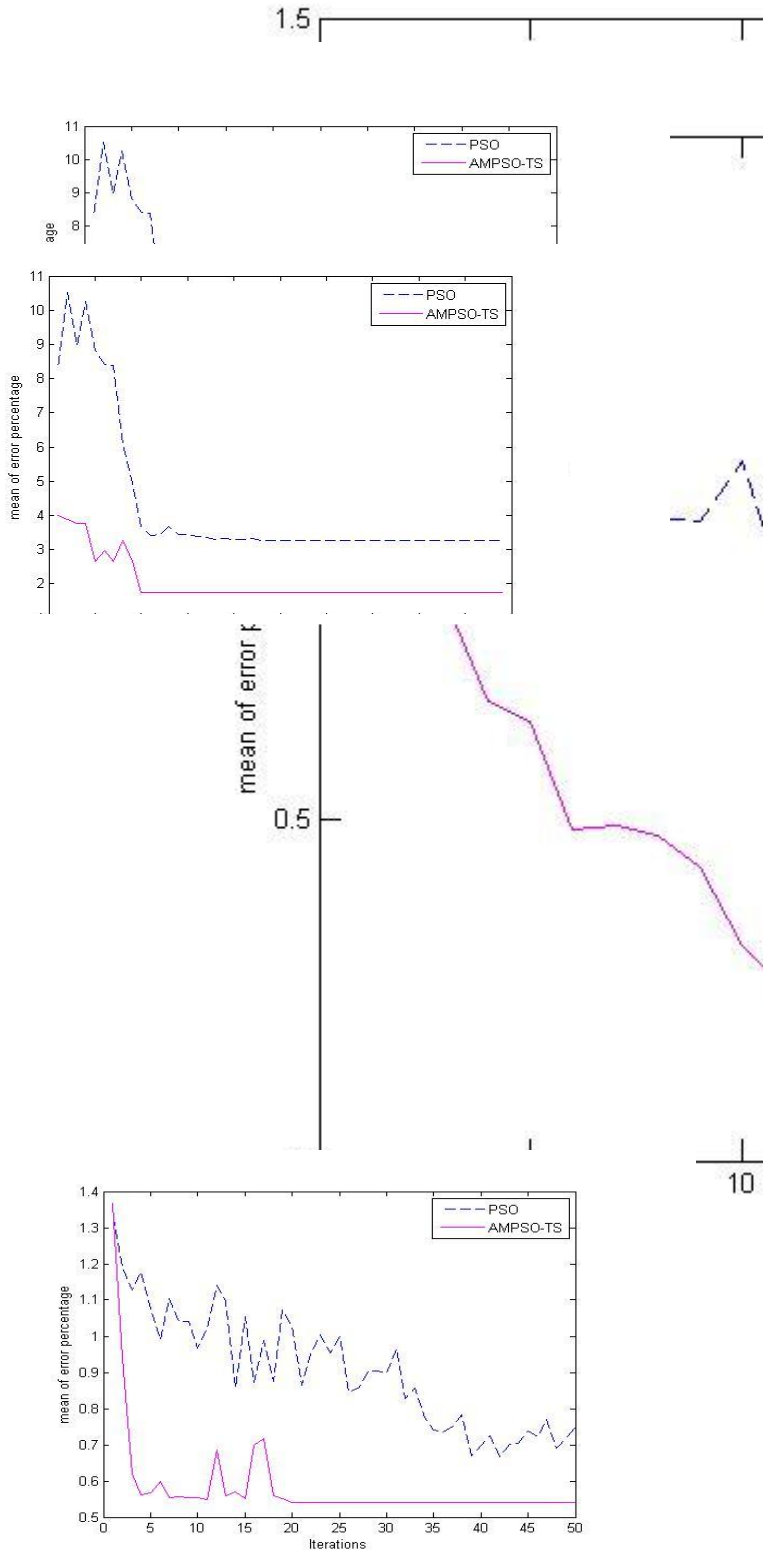


Fig.2.(c) Convergence rate for Brain RCOS function

simple PSO and Gradient descent method for training the RBF neural network . This method showed lower error and more convergence rate , that this shows modifying the velocity and choosing proper constant of PSO by using TS algorithm has effectively improve the performance .

## 4 CONCLUSIONS

In this paper we proposed AMPSO – TS method , that is a successful combination of simple PSO and TS algorithm with velocity modification . This method has shown better result than

## 5 APPENDIX

The functions formula that we used to check the proposed methods results .

| $f$ | Function Name | Formula |
|---|---|---|
| AK | Ackley | $AK_n(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{-\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$ |
| RC | Brain Rcos | $RC(x) = (x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$ |
| B | Bohachevsky | $B_2(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$ |
| BL | Beale | $BL(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_1^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$ |
| BO | Booth | $BO(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$ |
| CV | Colville | $CV(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2$ $+90(x_3^2 - x_4)^2 + \ 10.1((x_2 - 1)^2 + (x_4 - 1)^2)$ $+19.8(x_2 - 1)(x_4 - 1)$ |
| DJ | De Joung | $DJ(x) = x_1^2 + x_2^2 + x_3^2$ |
| DP | Dixon & Price | $DP_n(x) = (x_1 - 1)^2 + \sum_{i=1}^{n} i(2x_i^2 - x_{i-1})^2$ |
| GP | Goldstein & Price | $GP(x) = \left(1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + \ 3x_2^2)\right)$ $* (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2))$ |
| G | Griewank | $G_n(x) = \sum_{i=1}^{n} \frac{x_i}{4000} - \prod_{i=1}^{n} \cos\left(x_i/\sqrt{i}\right) + 1$ |

| | | |
|---|---|---|
| HM | Hump | $HM(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$ |
| L | Levy | $L_n(x) = sin^2(\pi y_1) + \sum_{i=1}^{n-1}[(y_i - 1)^2(1 + 10sin^2(\pi y_i + 1))]$ $+(y_n - 1)^2(1 + 10sin^2(2\pi y_n)), y_i = 1 + \frac{x_i - 1}{4}$ |
| MT | Matyas | $MT(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$ |
| RT | Rastrigin | $RT_n(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$ |
| R | Rosenbrock | $R_n(x) = \sum_{i=1}^{n-1}[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$ |
| SC | Schwefel | $SC_n(x) = 418.9829n - \sum_{i=1}^{n}(x_i sin\sqrt{\lvert x_i \rvert})$ |
| SR | Sphere | $SR_n(x) = \sum_{i=1}^{n}x_i^2$ |
| SS | Sum Square | $SS_n(x) = \sum_{i=1}^{n}ix_i^2$ |
| T | Trid | $T_n(x) = \sum_{i=1}^{n}(x_i - 1)^2 - \sum_{i=2}^{n}x_ix_{i-1}$ |
| Z | Zakharov | $Z_n(x) = \sum_{i=1}^{n}x_i^2 + (\sum_{i=1}^{n}0.5ix_i)^2 + (\sum_{i=1}^{n}0.5x_i)^4$ |

## REFERENCES


[1] Kennedy, J.; Eberhart, R. (1995). Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942–1948.

[2] Shi, Y.; Eberhart, R.C. (1998). A modified particle swarm optimizer. Proceedings of IEEE International Conference on Evolutionary Computation. pp. 69–73.

[3] F. Glover and C. McMillan . The general employee scheduling problem: an integration of MS and AI. Computers and Operations Research,1986.

[4] J. J. HOPFIELD Neural networks and physical systems with emergent collective computational abilities. Proc. NatL Acad. Sci. USA Vol. 79, pp. 2554-2558, April 1982 Biophysics

[5] R.C. Eberhart and J. Kennedy A New Optimizer using Particle Swarm Theory. In Proceedings of the Sixth International Symposium on Micromachine and Human Science, 1995, pp 39–43.

[6] J. Kennedy. Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. In Proceedings of the IEEE Congress on Evolutionary Computation, volume 3,July 1999,pages 1931–1938.

[7] J. Kennedy and R. Mendes Population Structure and Particle Performance. In Proceedings of the IEEE Congress on Evolutionary Computation, 2002. pages 1671– 1676. IEEE Press.

[8] J. Kennedy, R.C. Eberhart, and Y. Shi. Swarm Intelligence. Morgan Kaufmann, 2001.

[9] W.-K. Chen, Linear Networks and Systems. Belmont, Calif.: Wadsworth, pp. 123-135, 1993. (Book style)

[10] F.Glover, M.Laguna, Tabu Search, Univ. Boulder, 1997.

[11] A.hertz, E.Taillard, D.De werra, A Tutorial on Tabu Search, OR Spektrum 11, 1989, 131-141.

[12] A.Sebastian and P.Kumar and M.P.Schoen.AStudy on Hybridization of Particle Swarm and Tabu Search Algorithms for unconstrained Optimization and estimation Problems, 14th WSEAS international conference on systems 2010.pp.458-463.

[13] M.Pant, R.Thangaraj, A.Abraham. A New PSO Algorithm with Crossover Operator for Global Optimization Problems. Springer-Verlag Berlin Heidelberg 2007.

[14] N.Alamelumangai, J.D.Shree. PSO Aided Neuro Fuzzy Inference System for Ultrasound Image Segmentation. International Journal of Computer Applications, volume 7-No.14,October 2010.

[15] A.G.Bros. Introduction of the Radial Basis Function (RBF) Networks. Online Symposium of Electronics Engineering. issue 1, vol. 1, DSP Algorithms: Multimedia, Feb. 13 2001, pp. 1-7.

[16] N.Holden,A.A.Frietas. A Hybrid PSO/ACO Algorithm for classification GECCO (Companion) 2007: 2745-2750

[17] M.Vakil-Baghmisheh. N.Pasevic. Training RBF networks with selective back propagation . Elsevier, neurocomputing 62, 2004, pp.39-64.

[18] J.Q.Li, Q.K.Pan, S.X.Xie, B.X.Jia, Y.T.Wang. A hybrid particle swarm optimization and tabu search algorithm for flexible job-shop scheduling problem. International Journal of Computer Theory and Engineering, vol 2, No.2 April, 2010.

[19] Y.Zhang. L.Wu. A Hybrid TS-PSO Optimization Algorithms. Journal of Convergence Information Technology, volume 6, No.5, May 2011.

[20] S.Ferrari, R.Robbert, F.Stengel. Smooth FunctionApproximation Using Neural Network. IEEE transaction on neural network, vol 16, No.1, Jan.2005.

[21] Z.Zainuddin, O.Pauline. Function Approximation Using Artificial Neural Network. WSEAS transaction on Mathematics, issue 6, Vol 7, June 2008.

[22] J.Brownlee. Clever Algorithms: Nature-Inspired Programming Recipes. First Edition, Lulu Enterprises, January 2011. ISBN: 978-1-4467-8506-5.

[23] S.L.Ho, S.Yang, G.ni, H.C.Wong. A Particle Swarm Optimization Method With Enhanced Global Search Ability for Design Optimizations of Electromagnetic Devices. IEEE transaction on magnetic. Vol. 42, No.4, April 2006.