Simulated annealing-based particle swarm optimisation with adaptive jump strategy for modelling of dynamic cerebral pressure autoregulation mechanism

Shiru Sharma, Ranjana Patnaik and Neeraj Sharma*

Institute of Technology,
School of Biomedical Engineering,
Banaras Hindu University,
Varanasi – 221 005, UP, India
Fax: +91-542-2369841
E-mail: er_sharma11@rediffmail.com
E-mail: ranjanaswaraj@yahoo.mail.com
E-mail: er_neeraj29@indiatimes.com
*Corresponding author

J.P. Tiwari

Institute of Technology,
Department of Electrical Engineering,
Banaras Hindu University,
Varanasi – 221 005, UP, India
E-mail: jptiwari@bhu.ac.in

Abstract: This paper proposes a new particle swarm optimisation (PSO) algorithm based on simulated annealing (SA) with adaptive jump strategy to alleviate some of the limitations of the standard PSO algorithm. In this algorithm, swarm particles jump into the space to find new solutions. The jump radius is selected adaptively based on the particle velocity and its distance from the global best position. The designed algorithm has been tested on benchmark optimisation functions and on known autoregressive exogenous (ARX) model design problem. The results are superior as compared to the existing PSO methods. Finally, the designed algorithm has been applied for the analysis of the dynamic cerebral autoregulation mechanism.

Keywords: particle swarm optimisation; PSO; simulated annealing; SA; system identification; autoregressive exogenous; ARX; models; cerebral autoregulation.

Reference to this paper should be made as follows: Sharma, S., Patnaik, R., Sharma, N. and Tiwari, J.P. (2011) 'Simulated annealing-based particle swarm optimisation with adaptive jump strategy for modelling of dynamic cerebral pressure autoregulation mechanism', *Int. J. Bio-Inspired Computation*, Vol. 3, No. 4, pp.225–237.

Biographical notes: Shiru Sharma is an Assistant Professor at School of Biomedical Engineering, Institute of Technology, Banaras Hindu University. She had received her degree (Gold Medalist) in Electrical Engineering in 1995 from Regional Engineering College, Kurukshetra, India and received her PhD in Bio-Medical Engineering from IT, BHU. Her areas of interest are bio-system modelling, bio-control, and artificial intelligence.

Ranjana Patnaik is an Associate Professor at School of Biomedical Engineering, Institute of Technology, and Banaras Hindu University. She received her PhD in Neurophysiology in 1992 from Institute of Medical Sciences, Banaras Hindu University. Her area of interests are neurophysiology, electrophysiological signal analysis, and cerebral modelling. She has published 11 international papers, seven national papers, and contributed two book chapters in Academic Press, USA. Presently, she is also working as a Co-Investigator in an EOARD Sponsored Project.

Neeraj Sharma is an Assistant Professor at School of Biomedical Engineering, Institute of Technology, Banaras Hindu University. He received his BTech in Electrical Engineering in 1993 and MTech in Instrumentation in 1998 from Regional Engineering College, Kurukshetra, India. He received his PhD in Bio-Medical Engineering from IT, BHU. His research interests include bioinstrumentation, artificial intelligence and biomedical signal, and image processing.

J.P. Tiwari received his BE in Electrical Engineering; ME (Hons.) with specialisation in measurement and instrumentation from Department of Electrical Engineering, University of Roorkee (now IIT, Roorkee), Roorkee in 1968 and 1971, respectively; and PhD in Electrical Engineering from Banaras Hindu University in 1992. Since 1982, he has been in the Electrical Engineering Faculty of the Banaras Hindu University where he is currently a Professor. He is a Fellow Member of Institution of Engineers India. His research interests are systems modelling; digital and adaptive control systems, and motor drives.

1 Introduction

Particle swarm optimisation (PSO) has been recognised as a powerful tool for solving different optimisation problems. PSO is a population-based stochastic search method, inspired by the social behaviour of birds flocking and fishing school, and was developed by Kennedy and Eberhart (1995). Every particle in swarm finds its path on the basis of following information:

1 from its own best position

2 from the global best position in complete swarm.

Hence, through cooperation between individuals, the group is able to achieve its goal efficiently and effectively. The main advantages of PSO are:

1 it is derivative free global optimiser

2 fast and easy to parallelise

3 needs very less number of input parameters.

However, a standard PSO algorithm suffers from the following limitations:

1 Premature convergence, the reasons for premature convergence are:

   a the velocity of particle becomes zero after a few iterations

   b loss of diversity (Angeline, 1998).

2 Sensitivity to input parameters w, c_1, c_2, and population size (Eberhart and Shi, 2001).

Recently, simulated annealing (SA) has been effectively used as a tool to overcome the short comings of basic PSO. The aim of using SA is to avoid trapping of PSO into local minima and to increase the diversity (Sadati et al., 2009; Chaojun and Zulian, 2006; Niknam et al., 2009; Fang et al., 2007). Fang et al. (2007) have used PSO with SA for solving travelling salesman problem. Chaojun and Zulian (2006) have designed SA-based PSO algorithm. Through simulation analysis carried out on Griewank function, they have shown that disadvantages of standard PSO can be successfully conquered by combining PSO with SA, and the ability of global optimality is toned up. They have also introduced a mapping relation between inertial weights and annealing temperature, as a result searching speed and precision were improved. Sadati et al. (2009) have proposed the PSO-based SA algorithm by hybridising PSO with SA so that strong points of SA can be used in PSO. They have applied this optimisation algorithm for solving the under voltage load shedding problem.

Niknam et al. (2009) have also proposed hybridised PSO-SA. They have implemented SA-based PSO:

1 to search around the global solution

2 to increase the information exchange among particles using mutation operator to escape local optima.

The algorithm has been used for clustering the following datasets:

1 iris

2 Wisconsin breast cancer

3 Ripley's gloss datasets.

Their simulation results show that the SA-based PSO algorithm has better response and faster convergence as compared to K-means, PSO and SA algorithms.

However, in most of these SA PSO algorithms, SA has been implemented to search around global best solution, i.e., around g_{best} position, and the new locations around existing g_{best} position have been obtained either by mutation or by random jump in the search space. As a result, exploration is focused only around the g_{best} solution only, and there is every chance that, this g_{best} particle may be a particle trapped in local minima. Further, the new position generated may also fall within the same minima. Hence, the increase in diversity is not sufficient in case of existing SA-based PSO algorithms, which explores around the g_{best} position only, and thus, may not give the desired results. Now onward this type of SA-based PSO searching around global best (g_{best}) position is abbreviated as *SAPSO-G algorithm*.

The present work proposes SA-based PSO method with adaptive jump strategy. In present algorithm, SA-based search has been used to search around each particle in contrast to the SAPSO-G algorithms that search around the g_{best} position only. The main features of proposed algorithm are:

1 each particle adaptively jumps into the search space to find new position and new position so generated is accepted probabilistically

2 the jump radius is selected adaptively based on the velocity of particle and its distance from the global best position

3 with adaptive jump the particle become capable of coming out of the trap of deepest minima.

As a result, there is much wider increase in diversity. Hence, better exploration of search space is achieved that increases the probability of achieving global optima.

In the recent years, PSO technique has been extensively used for parameter estimation problems (Chen et al., 2009; Shinzawa et al., 2007; Meiying and Xiaodong, 2007; Kang et al., 2007). Hence, the designed algorithm has been applied to determine the ARX model of dynamic cerebral autoregulation mechanism.

2 Realisation of algorithm

2.1 Principles of standard PSO

In PSO, a number of particles in the form of swarms fly throughout the search space to search optimal or near optimal solutions. Each particle coordinate represents a possible solution to optimisation problem. The movement of particle is influenced by its own best position and position of the best particle in neighbourhood of total swarm (Clerc, 2007). The mathematical form of PSO is described as follows:

Suppose $x_i = (x_{i1}\ x_{i2}\ \ldots\ x_{id})$ is the current position of ith particle in d dimension search space.

- $p_{ibest} = (p_{i1}\ p_{i2}\ \ldots\ p_{id})$ best position searched by ith particle.
- $g_{best} = (g_1\ g_2\ \ldots\ g_d)$ the best position searched by the swarm.

At each iteration, the velocity and position (coordinates) of particle are updated according to the following equations:

$$v_i^{t+1} = wv_i^t + c_1 r_1 \left(p_{ibest} - x_i\right) + c_2 r_2 \left(g_{best} - x_i\right) \tag{1}$$

$$x_i^{t+1} = x_i^t + av_i^{t+1} \tag{2}$$

where i = 1 to n, corresponds to number of particles in swarm, t = corresponds to number of iteration, $v_i = (v_{i1}\ v_{i2}\ \ldots\ v_{id})$ particle velocity vector, w = inertia weight, c_1 = cognitive constant, c_2 = social coefficient, r_1, r_2 are random numbers between 0 to 1, and a = velocity scaling coefficient.

2.1.1 Algorithm for standard PSO

Step 1 Initialise the population, each particle is generated randomly.

Initial velocity for each particle is also initialised and is kept zero.
c_1, c_2 and w are also initialised.

Step 2 Fitness of each particle is calculated using objective function.

Step 3 Based on the Step 2, g_{best} and p_{best} are selected for the swarm and each particle respectively.

Step 4 The velocity of the particles for next iteration are computed using equation (1).

Step 5 Update the particle position using equation (2).

Step 6 Check for the stopping criteria (if the number of maximum iteration are reached)
if $t = iter_{\max}$ stop

Else go to Step 2, with the updated values of iteration count ($t = t + 1$).

Step 7 The final g_{best} is the solution.

2.1.2 The effect of variations of parameters on performance of PSO is described as follow

- *Impact of inertia weight (w)*. Inertia weight describes confidence of particle in its own movement. This factor was originally introduced by Shi and Eberhart (1998) and represents the impact of previous velocity. To measure the impact of inertia weight (w), the standard PSO has been applied for optimisation of Rastrigin's function. With different values of w in the range of (0 to 1) the plot of fitness value of function vs. inertia weight (w) is shown in Figure 1. Rastrigin's function used here with two independent variables is defined by following equation:

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10\left(\cos 2\pi x_1 + \cos 2\pi x_2\right) \tag{3}$$

Rastrigin's function is having its global minimum at (0, 0).

From Figure 1, it can be concluded that for large values of w (0.7–1.0), the performance of PSO is decreased as particles are engaged in exploration and the task of exploitation is hampered. At the same time, low values of w ($w < 0.4$) also results in poor performance as particles are engaged in exploitation while the task of exploration is suffered. By suitably selecting the inertia weight a balance between exploration and exploitation is achieved. In present study, it has been found experimentally that value of w for best performance can be kept in the range (0.4–0.7). Different researchers have kept this value initially high to support exploration but later decreased to have good exploitation (Sadati et al., 2009; Niknam et al., 2009). The same implements as follows:

$$w = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{iter_{\max}}\right) \times t \tag{4}$$

where

$w_{\max}$ maximum value of inertia weight = 0.9

$w_{\min}$ minimum value of inertia weight = 0.4

$iter_{\max}$ maximum number of iteration

t current number of iteration.

- *Impact of cognitive and social constants*. The terms c_1 and c_2 in equation (1) (also called acceleration constant) maintain balance between particle's individual and social behaviour and are usually set to be equal to 2 i.e., $c_1 = 2$, $c_2 = 2$ (Eberhart and Shi 2001). Further,

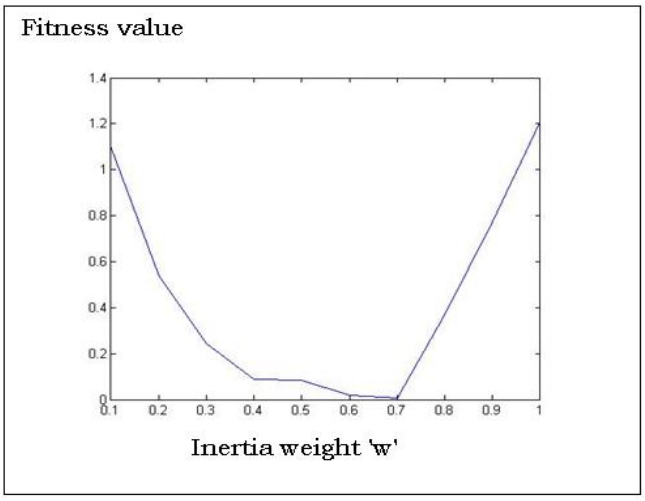
Ratnaweera et al. (2004) have proposed modified PSO with time-varying accelerator coefficients (MPSO-TVAC) to control the local search and improve the convergence to global optimum solution. In this algorithm, c_1 and c_2 are varied in a linear manner; equation for same is given below:

$$c_1 = 2.5 - 2.0 \times \frac{t}{iter_{\max}} \tag{5}$$

$$c_2 = 0.5 + 2.0 \times \frac{t}{iter_{\max}} \tag{6}$$

- *Size of population (swarm size)* with more number of particles explorations of search space is fast, while computation time (iteration time) is increased. With lesser number of particles iteration time is reduced but swarm takes longer time to yield a solution or may even fail to find acceptable solution. Hence, a compromise is made and empirically it has been found that a population size between 20 to 30 particles is sufficient to solve most of the problems (Clerc, 2007).

Figure 1 Plot of fitness value of Rastrigin's function vs. inertia weight (w) (see online version for colours)



2.2 Principles of SA

SA was independently developed by Kirkpatrick et al. (1983) and Cerny (1985). It is a stochastic method for global optimisation. SA has been hybridised with different algorithms to solve wide variety of problems (Sharma et al., 2009; Christober and Mohan, 2009; Niknam et al., 2009; Sadati et al., 2009; Chu et al., 2006). SA is a randomised gradient descent algorithm that permits uphill moves with some probability so that it can escape local minima. SA technique acts on the basis of physical analogy similar to annealing of metals, in which liquid metal if cooled at appropriate cooling rate will reach an absolute minimum energy state related to complete atomic ordering of metal, if the liquid is cooled at a faster rate the atoms will reach a suboptimal energy state (Sadati et al., 2009).

The two input parameters required for SA are initial temperature (T) and cooling rate (α). The initial temperature is usually kept high and is decreased slowly; the rate of decrement is decided by the cooling rate. The starting temperature, the rate of cooling and final temperature to be achieved for a system is called *annealing schedule*. The output solution for a given system is dependent on annealing schedule.

$$T \leftarrow \alpha T,$$

where α is cooling rate in the range (0.8 to 1.0).

With slow cooling the quality of solution is better.

2.2.1 Algorithm for SA

Step 1 Select an initial solution randomly, initialise the initial temperature (T) and cooling rate (α).

Step 2 Calculate the energy of system for this solution i.e., evaluate $f(x_0)$.

Step 3 Find another solution in the space (x_1) by random jump in the search space.

Step 4 Evaluate energy for this solution i.e., evaluate $f(x_1)$.

Step 5 Compute $\Delta E = f(x_1) - f(x_0)$.

Step 6 If $\Delta E < 0$ accept x_i as solution

Else if $f(x_1) > f(x_0)$ and $e^{-\left(\frac{\Delta E}{T}\right)} > R$ (random number)

x_1 is accept as solution i.e., $x_0 \leftarrow x_1$. Else x_0 is accepted as solution.

Step 7 Decrease the temperature using following equation

$T \leftarrow \alpha T$

Step 8 Compute the fitness of accepted solution.

Step 9 Check for stopping criteria

if desired level of fitness is achieved stop

Else go to Step 2.

Step 10 x_0 is the solution of the problem.

3 Application of SA to improve PSO

In the present work, we have proposed improvement in PSO using SA with following features:

1 SA-based search has been used to search around each particle in contrast to search around the g_{best} position only. As a result, there is much wider increase in diversity. Hence, better exploration of search space is achieved that increases the probability of achieving global optima.

2 The new positions around each particle have been generated adaptively based upon its velocity and distance from g_{best} position. With this adaptive jump

technique, it is possible to prevent particles from becoming inactive.

Jump radius is selected adaptively, which is inversely proportional to the magnitude of particle velocity and its distance from global best location (g_{best}). The magnitude of velocity of *i*th particle is computed using equation (2) and is obtained as given below;

$$since \qquad x^{t+1} = x^t + v^{t+1}$$

$$therefore \qquad v^{t+1} = x^{t+1} - x^t$$

and magnitude of velocity is computed using

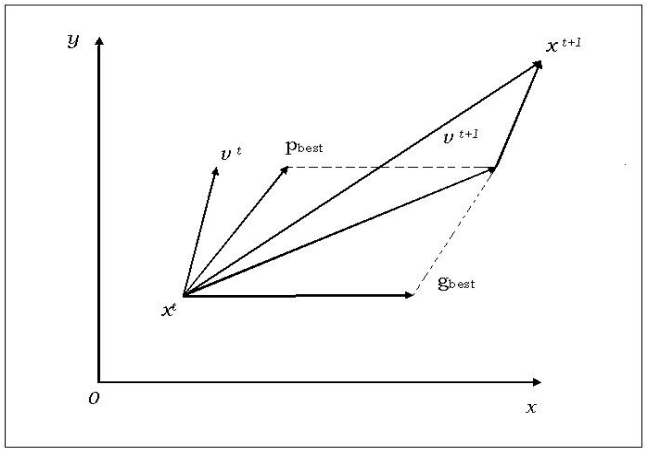
$$|| v ||= \left\| x_i^{t+1}, x_i^t \right\| \tag{7}$$

The vector representation of particle position update using equations (1) and (2) is shown in Figure 2 and thus, magnitude of velocity is the displacement distance between particle initial position(x^t) and new position (x_{t+1}). Distance between particle position and global best position is Euclidian distance between x_i and g_{best} positions and is given by:

$$d\left(x_i, g_{best}\right) = \left\| x_i, g_{best} \right\| \tag{8}$$

3 During the last iterations with the decrease in temperature algorithm worked as standard PSO and no change in position is allowed, hence, guaranteed convergence is achieved.

Figure 2 Vector representation of particle position update based on equations (1) and (2)



3.1 Proposed SAPSO-A algorithm

Basic parameters for SA-PSO algorithm are listed below.

The proposed SA PSO algorithm involves the following important steps:

Step 1 Initialisation of inputs and parameters

- $X = (x_1\ x_2\ \_\ \_\ \_\ x_k)$ the initial population with k number of particles is initialised randomly (k = 20 in present case).

  where x_i is the *i*th particle in d dimension space
- $V = (v_1\ v_2\ \_\ \_\ \_\ v_k)$ the 'd' dimensional velocity vectors corresponding to each particle are initialised randomly.
- $P_{best} = (p_{1best}\ p_{2best}\ \_\ \_\ \_\ p_{kbest})$ and g_{best} are initialised randomly.
- $iter_{\max}$ = maximum number of iterations (5,000)
- c_1 = cognitive coefficient (c_1 = 2 in present case)
- c_2 = 2 social coefficient (c_2 = 2 in present case)
- w = inertia weight (we have selected its value to be 0.5)
- T = initial temperature (T = 10000)
- α = cooling rate (α = 0.95)
- ε = stopping criterion ($\varepsilon = 10^{-8}$)

Step 2 The velocity and position of the particles are updated using the following equation:

$$v_i^{t+1} = v_i^t + c_1 r_1 \left(p_{ibest} - x_i\right) + c_2 r_2 \left(g_{best} - x_i\right)$$
$$\forall i = 1\ to\ k$$

where r_1 and r_2 are 'd' dimensional pseudo random vector with each element in the range of (0 1)

$$x_i^{t+1} = x_i^t + v_i^{t+1} \qquad \forall i = 1\ to\ k$$

Step 3 Compute the fitness of each particle, $f(x_i)$ using objective function

Step 4 Based on the fitness of particles computed in Step 2 select particle best (p_{best}) and global best (g_{best}) positions.

Step 5 Particles are allowed to jump adaptively. The new position of particle is computed by using following equation:

$$x_{i\_sa} = x_i + r_{ad} \times r_3 \tag{9}$$

where r_{ad} = adaptive jump radius given by following equation:

$$r_{ad} = \left(\frac{1}{\left\| x_i^{t+1}, x_i^t \right\| \times \left\| x_i, g_{best} \right\|} \right) \tag{10}$$

and r_3 = rand (1, d)

where function, rand (1, d) returns d dimensional vector with random values between 0 and 1

Step 6 The fitness of each particle at new position is computed, $f(x_{isa})$ are computed using mean square error (MSE) as objective function as defined in Step 2.

Step 7 Compute $\Delta E = f(x_{isa}) - f(x_i)$

Step 8 Decide whether to accept or reject the particle new position on the basis of change in its fitness (energy)

If $f(x_{isa}) < f(x_i)$ then allow the particle to accept new position i.e., $x_i \leftarrow x_{isa}$

Else if $f(x_{isa}) > f(x_i)$ and $e^{-\left(\frac{\Delta E}{T}\right)} > R$ (where R is pseudo random number)

then allows the particle to accept new position i.e.

Else the change in position is not allowed.

Step 9 Based on the accepted particle position calculate the fitness of each particle and based on this update the particles own best and global positions i.e., p_{ibest}, g_{best}.

Step 10 Check for the stopping criterion

Desired value of objective function is achieved or maximum number of iteration limit is reached, stop if criterion is met, else go to Step 2 with the value of temperature and iteration count updated as follows: $T = \alpha T$ and $t = t + 1$.

Step 11 The g_{best} particle coordinates is the optimal solution and represents the model parameters.

3.2 Application of proposed SAPSO-A algorithm for ARX modelling

The generalised equation of ARX model (Ljung, 1987) is given by following equation:

$$\begin{aligned} &y(t) + a_1 y(t-1) + a_2 y(t-2) + \ldots + a_{na} y\left(t - n_a\right) \\ &= b_1 u\left(t - n_k\right) + \ldots + b_{nb} u\left(t - n_k - n_b + 1\right) + e(t) \end{aligned} \tag{11}$$

In compact form, equation (11) can be rewritten as

$$y(t) = \varphi^T \theta_c + e(t) \tag{12}$$

where φ^T is lagged input output data matrix given by

$$\varphi T = \left[-y(t-1) - y(t-2) \ldots y\left(t - n_\alpha\right) u\left(t - n_k\right) \ldots u\left(t - n_k - n_b + 1\right)\right]$$

θ_c is the coefficient vector and is given by:

$$\theta_c = \left[a_1\ a_2\ \ldots a_{na}\ b_1\ b_2\ \ldots b_{nb}\right]^T$$

$a_1\ a_2 \_ \_ \_ a_{na}$ are autoregressive coefficient

$b_1\ b_2 \_ \_ \_ b_{nb}$ are exogenous coefficient

$y(t)$ output response of the system

$u(t)$ inputs to the system

$e(t)$ error or noise terms

n_a number of poles

n_b number of zeros

nk pure time delay (dead time).

The model order can be determined by minimising final prediction error criterion FPE with $\alpha = 2$ (Leontaritis and Billings, 1987). FPE is given by following equation:

$$FPE = V\left[\frac{(1 + ad/N)}{(1 - ad/N)}\right] \tag{13}$$

where

N = length of the data

$d = n_a + n_b + n_k$

a = constant ($a = 2$ in present case)

V = loss function.

The parameters for the given model structure are determined from its input output data using SAPSO-A algorithm. The objective function used in present case is MSE and is given by following equation:

$$MSE = \frac{1}{N}\sum_{l=1}^{N}\left[y(l) - \hat{y}(l)\right] \tag{14}$$

where

$y(l)$ actual output

$\hat{y}(l)$ computed output corresponding to the θ_c coefficient vector.

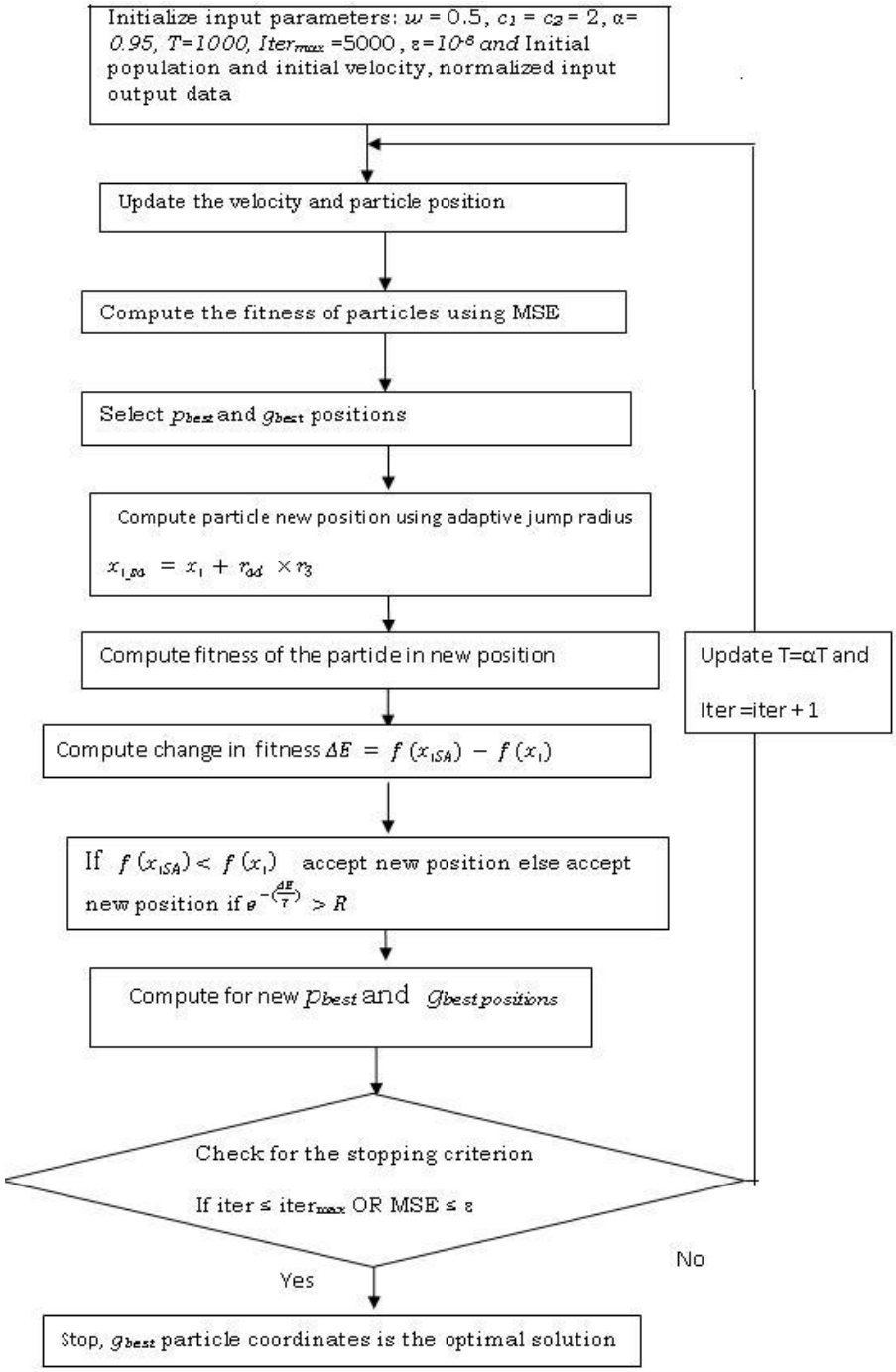
The objective is to optimally determine the coefficient vector θ_c that minimises the MSE. The programming of this algorithm has been done in MATLAB. The flow chart of SAPSO-A is shown in Figure 3.

4 Results

4.1 Testing of proposed SAPSO-A algorithm on Rastrigin's function a standard optimisation problem

Rastrigin's function is often used to test the optimisation algorithms, because its many local minima make it difficult for optimisation algorithms to find the global minimum (Ioannisc et al., 2004). Initial testing of algorithm has been done on the standard *Rastrigin's function*. Rastrigin's function for two independent variables, parameters is given by equation (3). Rastrigin's function is having multiple local minima and one global minimum, which occurs at the point (0, 0), where the value of the function is 0. At any minima other than global minima, the value of Rastrigin's function is greater than 0. The farther the local minimum is from global minima, the larger the value of the function is at that point. A comparison of performance of the proposed SAPSO-A algorithm, with standard PSO, MPSO-TVAC, SAPSO-G has been done on the basis of fitness value of the Rastrigin's function.

Figure 3 Flow chart of proposed SAPSO-A for ARX modelling



The evaluation of the performance of the four algorithms has been done on the basis of averaged values of the function for ten runs. Table 1 compiles the results of the four algorithms, i.e., PSO, SAPSO-G, SAPSO-A, and MPSO-TVAC applied to Rastrigin's function. The performance of these algorithms can be ordered as follows:

1 SAPSO-A

2 SAPSO-G

3 MPSO-TVAC

4 PSO.

Table 1 Fitness value and solution obtained for Rastrigin's function using SAPSO-A, PSO, SAPSO-G and GA algorithms respectively

S. no.	*Algorithm*	*Fitness value*	*Parameters: [x_1 x_2]*
1	PSO	0.0055	–0.0038–0.0037
2	MPSO-TVAC	0.0015	0.00012–0.0028
3	SAPSO-G	0.0014	0.0025–0.0009
4	*SAPSO-A*	*9.4215e-005*	*0.0004241–0.0005432*

4.1.1 Parameter setting

The values of parameters have been selected as follows: The inertia weight w = 0.5, and two accelerator coefficients c_1, c_2 are both set 2.0 for standard PSO, SAPSO-G, and SAPSO-A, while for MPSO-TVAC c_1 decreased from 2.5 to 0.5, while c_2 increased from 0.5 to 2.5 and w is kept 0.5. Number of particles N = 20, maximum number of iteration $iter_{max}$ = 500 and $\varepsilon = 10^{-5}$. And for the two SA-based PSO algorithms initial temperature, T = 10,000 and cooling rate, α = 0.95 has been selected. The present algorithm has not been optimised in term of operating speed. The SA cooling procedure is kept slow, with slow cooling more number of iteration are available for particles to change their state, i.e., to jump to higher energy state, which increases their chances to come out of the trap of local minima. Further, optimal values of initial temperature and cooling rate can be determined experimentally for the specific problem. The annealing schedule for a given problem depends upon the dimensionality of search space and complexity of the objective function to be optimised.

Performance analysis of the proposed SAPSO-A algorithm and its comparison with standard PSO, SAPSO-G and MPSO-TVAC has also been carried out on the following benchmark optimisation functions:

- Sphere function

$$f_1(x)\sum_{i=1}^{n} x_i^2$$

- Schaffer's function

$$f_2(x) = 0.5 - \frac{\left(\sin\sqrt{x^2+y^2}\right)^2 - 0.5}{\left(1.0 + 0.001\left(x^2+y^2\right)\right)^2}$$

- Alpine function

$$f_3(x) = \sum_{i=1}^{n} \left| x_i \sin\left(x_i\right) + 0.1x_i \right|$$

- Ackley function

$$f_4(x) = -20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right) - \exp\left(\frac{1}{n}\sum_{j=1}^{n}\cos 2\pi x_i\right) + 20 + e$$

In present analysis, the dimensions of the functions has been kept as follows f_1(30 D), f_2(2 D), f_3(10 D), f_4(30 D). The parameter setting has been kept same as reported above except for the:

a maximum number of iteration, same has been changed to 2,000

b stopping criterion (ε), this condition is removed.

Average values of the functions for ten runs have been reported, the results are compiled in Table 2.

4.2 Testing of algorithm on known ARX model

The algorithm developed has been tested on known ARX model (m_{sim}) described by equation:

$$y(t) + 1.5y(t-1) - 0.7y(t-2) = 2.5u(t-2) + 0.9u(t-3) \quad (15)$$

The same equation in transfer function in polynomial format is as follows:

$$T.F. = \left(\frac{1 - 1.5z^{-1} + 0.7z^{-2}}{2.5z^{-2} + 0.9z^{-3}}\right) \quad (16)$$

Table 2 Comparison results on benchmark functions

S. no.	Algorithm	Fitness value of function			
		f_1(30 D)	f_2(2 D)	f_3(10 D)	f_4(30 D)
1	PSO	7.6412e-08	0.0035	1.7143e-10	2.0902e-05
2	MPSO-TVAC	2.5663e-15	0.0029	8.8182e-16	3.6690e-11
3	SAPSO-G	9.0688e-12	0.0031	6.1930e-16	9.7066e-10
4	*SAPSO-A*	*1.5783e-20*	*0.0026*	*6.9389e-18*	*3.1353e-12*

Table 3 Model parameters obtained using the ARX, PSO, MPSO-TVAC, SAPSO-G, and SAPSO-A algorithms

S. no.	Modelling method	Curve fitness (%)	MSE	$\theta_c = [a_1\ a_2\ b_3\ b_4]$ estimated model parameters
1	ARX	67.95	0.0099	θ_c = [–1.5 0.7 0.0 0.9]
2	PSO	97.89	0.0020	θ_c = [–1.488 0.692 2.337 1.162]
3	SAPSO-G	97.45	0.0022	θ_c = [–1.473 0.6808 2.171 1.37]
4	*SAPSO-A*	*99.99*	*3.11×10^{-8}*	*θ_c = [–1.5 0.7 2.501 0.8999]*

Note: With model structure: n_a = 2, n_b = 2, and n_k = 2

And the coefficient vector θc is represented as follows:

$$\theta_c = \begin{bmatrix} a_1 & a_2 & b_3 & b_4 \end{bmatrix}$$

$$\theta_c = \begin{bmatrix} -1.5 & 0.7 & 2.5 & 0.9 \end{bmatrix}$$

The order structure for the model described by above equation is

n_a number of poles, 2

n_b number of zeros, 2

n_k pure time delay (dead time), 2.

We have remodelled the system using unit impulse (of length 58 points) as input data and the corresponding output response of m_{sim} as output data. For present study, MSE has been taken as objective function and the model structure has been selected as [n_a = 2 n_b = 2 n_k = 2]. For a comparison model, parameters have been estimated using following techniques:

1 ARX

2 standard PSO

3 SAPSO-G

4 proposed SAPSO-A algorithm.

The performance of each technique for this modelling problem has been evaluated on the basis of fitness of model generated and MSE of the model, results of same are compiled in Table 3. The output curves corresponding to actual m_{sim} model, ARX model, PSO model and SAPSO-A model are shown in Figure 4 for comparison. For this modelling problem $iter_{\max}$ and ε have been changed to 5,000 and 10^{-8}, while rest of the parameters are kept same as mentioned above.

To analyse and compare the performance of SAPSO-A with SAPSO-G and standard PSO for modelling problem we have carried out following analysis:

- *Analysis of velocity*: Magnitude of velocity i.e., the displacement distance between particle initial position (x^t) and new position (x^{t+1}) is plotted against iterations for the three algorithms: standard PSO, SAPSO-G, and SAPSO-A, the plots are shown in Figure 5, Figure 6, and Figure 7 respectively. Velocity of a particle is an indicator for its activeness and hence, its contribution in determination of global solution. The normalised jump radius of a particle in case of SAPSO-A is shown in Figure 8.
- *Analysis of inertia weight (w) on the performance*: The performance of the three algorithm (standard PSO, SAPSO-G, and SAPSO-A) has been evaluated for the different values of w = [0.1, 0.2 … 1.0]. The analysis of velocity and inertia weight was carried out on known ARX model (m_{sim}) given by equation (15). The parameter setting has been kept same as mentioned in Section 4.1. Each algorithm has been executed for ten times and the average of output (MSE and percentage of curve fitness) of these runs has been done. The results are compiled in Table 4 and their plots are shown in Figure 9. A typical plot of the fitness vs. iteration for standard PSO, SAPSO-A and SAPSO-G corresponding to w = 0.2 is shown in Figure 10.

Figure 4 Plot of output curves corresponding to actual model (m_{sim}), ARX model, PSO model and SAPSO-A model

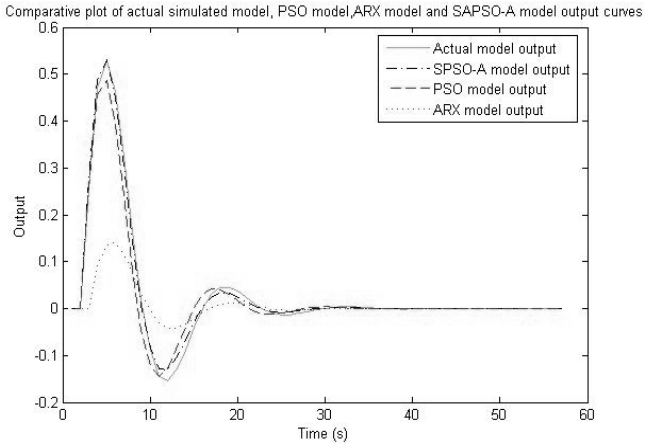


Figure 5 Plot of velocity magnitude vs. iteration number for standard PSO algorithm

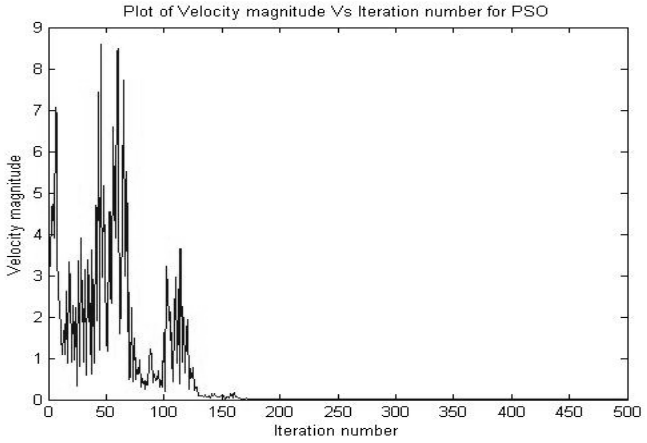


Figure 6 Plot of velocity magnitude vs. iteration number for SAPSO-G algorithm

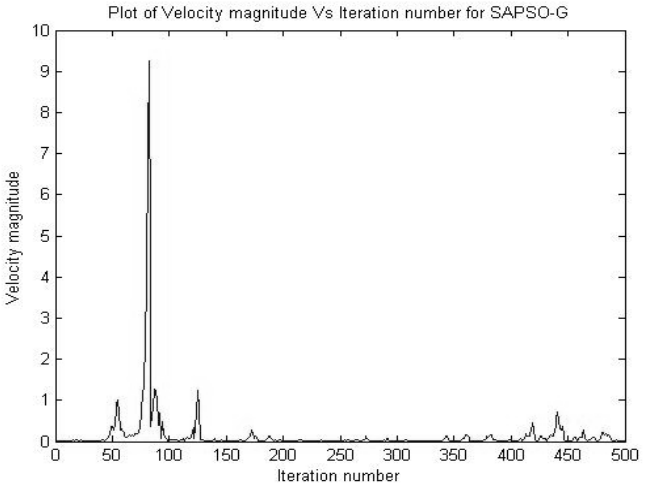


Figure 7 Plot of velocity magnitude vs. iteration number for SAPSO-A algorithm

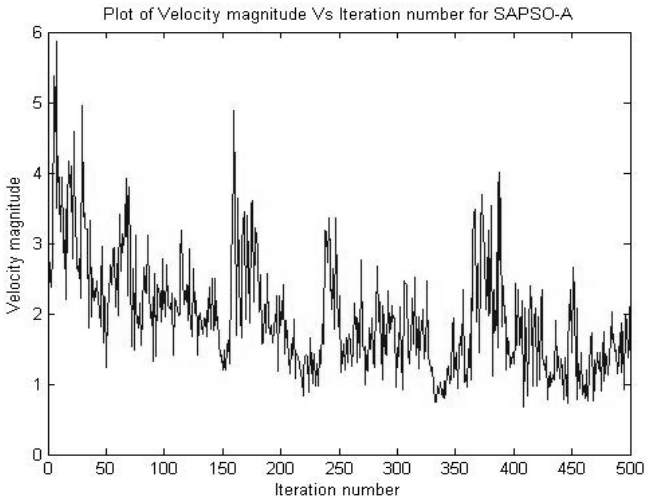


Figure 8 Plot of normalised jump radius magnitude vs. iteration number for proposed algorithm (SAPSO-A)

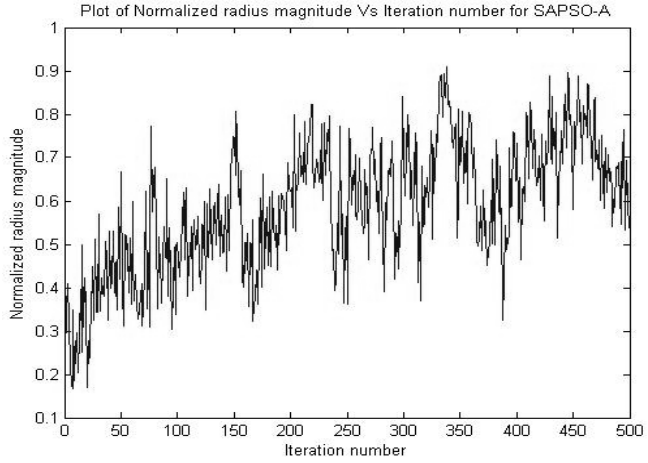


Figure 9 Plot of the MSE Vs inertia weight for SAPSO-A, SAPSO-G, and standard PSO algorithms

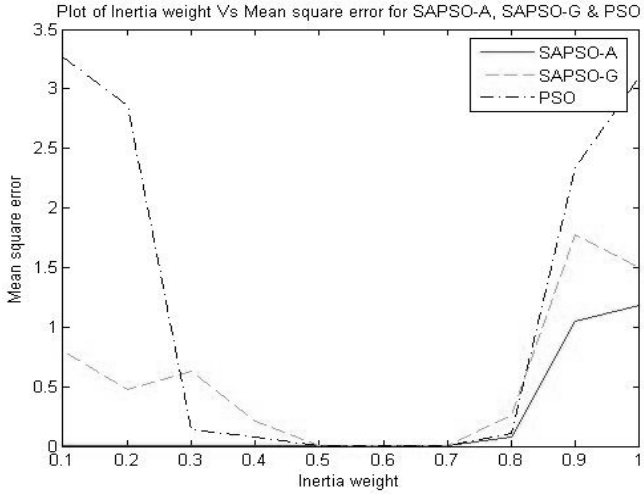
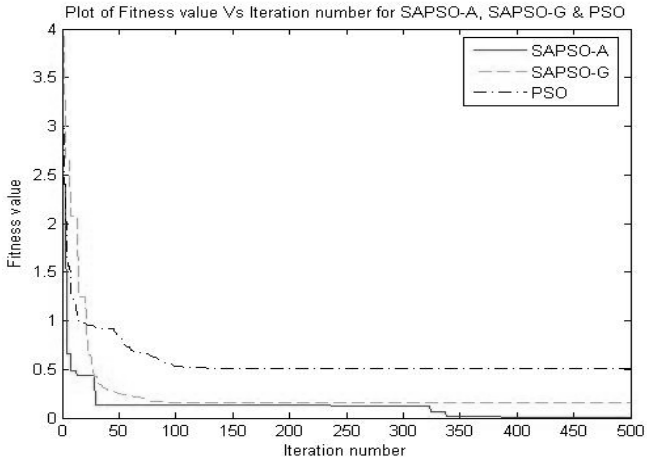


Figure 10 Comparative plot of the fitness values vs. iteration number for standard PSO, SAPSO-A and SAPSO-G



Note: Value of inertia weight '$w = 0.2$', for present case.

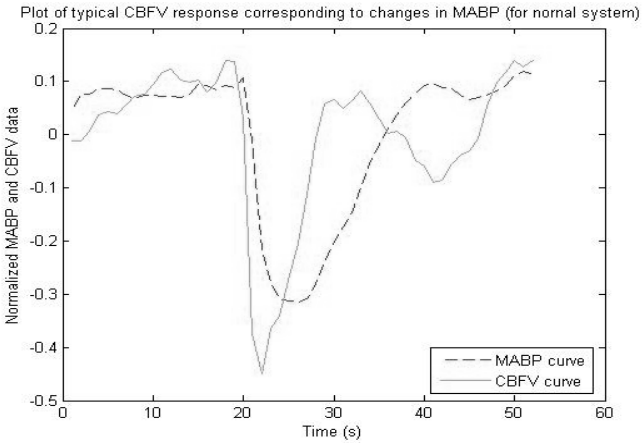
Table 4 Shows the impact of w on the performance of three algorithms

S. no.	Inertia weight (w)	*SAPSO-G*		*PSO*		*SAPSO-A (proposed algorithm)*	
		Curve fitness (%)	*MSE*	*Curve fitness (%)*	*MSE*	*Curve fitness (%)*	*MSE*
1	0.1	48.51	0.8050	3.82	3.2654	*97.37*	*0.0021*
2	0.2	61.23	0.4799	22.74	2.8617	*97.57*	*0.0022*
3	0.3	54.53	0.6261	78.91	0.1349	*96.43*	*0.0034*
4	0.4	74.37	0.2050	84.85	0.0762	*98.33*	*0.0018*
5	0.5	97.78	0.0023	96.89	0.0030	*97.80*	*0.0020*
6	0.6	96.19	0.0032	97.40	0.0022	*98.02*	*0.0019*
7	0.7	97.45	0.0022	96.75	0.0032	*98.78*	*0.0017*
8	0.8	70.43	0.2651	82.4	0.1038	*83.85*	*0.0792*
9	0.9	24.44	1.7746	17.12	2.3431	*45.71*	*1.0452*
10	1.0	31.67	1.4978	1.291	3.1081	*40.17*	*1.1820*

4.3 Application of SAPSO-A to model dynamic cerebral pressure autoregulation mechanism.

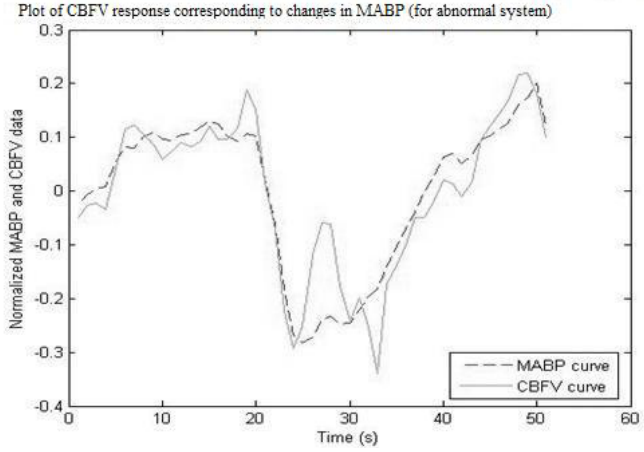
Model structures for dynamic cerebral pressure autoregulation mechanism have been obtained by minimising FPE criterion. The model structures obtained are further used to find the model parameters, i.e., coefficient vector. Mean arterial blood pressure (MABP) and cerebral blood flow velocity (CBFV) have been taken as input and output data. The modelling has been done corresponding to normal and abnormal dynamic autoregulation mechanisms. Figure 11 and Figure 12 shows the plots of typical and atypical CBFV response curves corresponding to changes induced in MABP during the thigh cuff tests for dynamic cerebral autoregulation mechanism respectively, as reported by Mahony et al. (2000).

Figure 11 Plot of typical CBFV response to changes in MABP for normal autoregulatory system during thigh cuff test



Source: Mahony et al. (2000)

Figure 12 Plot of atypical CBFV response to changes in MABP for abnormal autoregulatory system during thigh cuff test



Source: Mahony et al. (2000)

Table 5 Results of models obtained using SAPSO-A for the model structures [4 4 0], for normal and impaired dynamic cerebral autoregulation mechanism, the parameter values have been kept as follows: $w = 0.5$, $c_1 = 2$, $c_2 = 2$, $T = 10{,}000$, $\alpha = 0.95$, *iteration* = 5,000 and $\varepsilon = 10^{-8}$

S. no.	*Type of dCA*	*Model structure* $[n_a\ n_b\ n_k]$	*MSE*	*Curve fitness (%)*
1	Normal dCA	*[4 4 0]*	*0.0028*	*82.16*
2	Impaired dCA	*[4 4 0]*	*0.0031.*	*74.22*

Figure 13 Comparative plot of actual CBFV output and SAPSO-A model response curves for normal autoregulatory system for model structure [4 4 0]

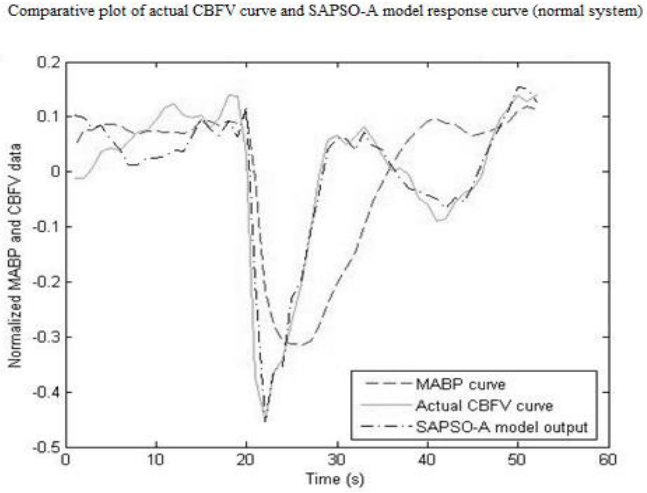
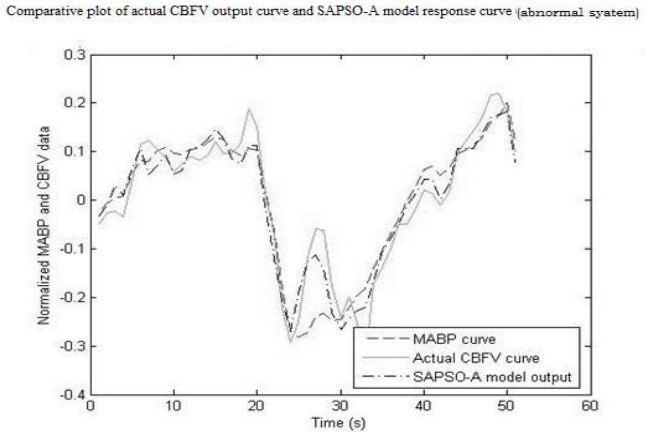


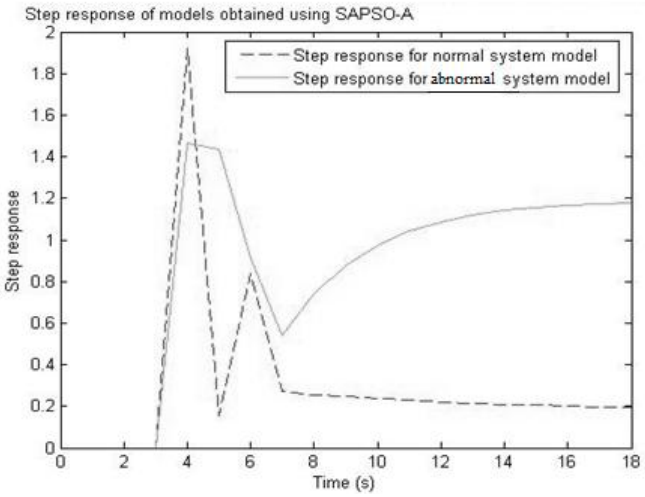
Figure 14 Comparative plot of actual CBFV output and SAPSO-A model response curves for abnormal autoregulatory system for model structure [4 4 0]



The results of models fitness and MSE, obtained using SAPSO-A are compiled in Table 5. These models of cerebral autoregulation system have been computed using SAPSO-A algorithm with the following setting of input parameters: $w = 0.5$, $c_1 = 2$, $c_2 = 2$, $T = 10{,}000$, $\alpha = 0.95$, $iter_{\max} = 5{,}000$ and $\varepsilon = 10^{-8}$. The comparative plots of the output curves corresponding to actual output and SAPSO-A model output for normal and abnormal autoregulatory

system corresponding to model structure of [n_a = 4; n_b = 4; n_k = 0] are shown in Figure 13 and Figure 14 respectively. The step response of selected model of dynamic CA system is shown in Figure 15, which distinguish the autoregulatory capacity of the cerebral system under normal and abnormal conditions. The step response is an indicator of cerebral autoregulatory capacity. A fast decrease in step response predicts the active cerebral autoregulation as compared to sluggish response in case of abnormal CA system (Panerai 2004).

Figure 15 Step response of models for normal and abnormal autoregulatory system obtained using SAPSO–A



5 Discussion

In the present study SA-based PSO algorithm with adaptive jump strategy (SAPSO-A) has been designed and developed. The developed SAPSO-A algorithm is then applied to ARX modelling of cerebral autoregulation mechanism.

The important aspects of SAPSO-A with respect to earlier SA-based PSO, searching around global best (g_{best}) position i.e., SAPSO-G and standard PSO is that its performance is not affected by the input parameters w and same can be directly observed from Table 4 and Figure 8 (plot of comparative performance of SAPSO-A, standard PSO and SAPSO-G). Further, in case of SAPSO-G, only g_{best} position is changed, which is having impact only on the III term ($g_{best} - x_i$) of velocity equation given by equation (1), while there is no influence on I term $\left(w \times v_i^t\right)$ and II term ($p_{ibest} - x_i$) of this velocity equation. The jump is random in case of SAPSO-G and there is every possibility that it may not be high enough to come out of the local minima. Hence, there is exploration only about the g_{best} position, which may not incorporate sufficient increase in the diversity of swarm.

The performance of SAPSO-A has also been compared with MPSO-TVAC algorithm using Rastrigin's function and other benchmark functions, the results of SAPSO-A are better. The reason for lower performance of MPSO-TVAC may be that in case of MPSO-TVAC, the social and cognitive parameters are adjusted linearly, without any relationship with the particle velocity.

To improve the performance of standard PSO several other techniques have been proposed such as *multi swarm PSO* (Zhao and Yang, 2008; Lovberg et al., 2001), *attractive and repulsive PSO* (Gazi and Passino, 2003), *multistart PSO* (Van den Bergh and Engelbrecht, 2002; Litinetski and Abrahamzon, 1998), and adaptive PSO (Dong et al., 2008; Cai et al., 2007; Chu et al., 2006; Xie et al., 2002). The main idea in implementing most of these techniques is to increase the diversity and to avoid the velocities of the particles to go zero. In comparison to these techniques, the present algorithm differs by the way of being SA-based PSO with adaptive jump strategy, implemented to avoid the trapping of PSO into local minima and to increase the diversity.

The major features of present SAPSO-A are:

- Every particle is involved.
- With adaptive jump the particles become capable of coming out of the trap of deepest minima.
- Particles are rhythmically involved in exploration and exploitation of search space, hence, results are much better and there is excellent increase in diversity of swarm.
- It is less sensitive to inertia weight (w), and is tolerant enough to parameter variations; hence, there is least burden on the operator.
- During last iteration SAPSO working as standard PSO is bound to converge.

It is worth mentioning here that SAPSO_A is a stochastic method of achieving global optimisation and is computationally expensive. However, the expensive nature of the algorithm is compensated by its excellence of achieving global minima.

6 Conclusions

In this work, SA-based PSO algorithm with adaptive jump strategy (SAPSO-A) has been designed and developed. With present approach some of the limitations of the standard PSO algorithm have been successfully alleviated. The main advantages of the present algorithm are: it is less sensitive to input parameters, SA-based adaptive jump strategy impart it capability of coming out of the trap of local minima, particles are systematically involved in exploration and exploitation of search space. Hence, proposed SAPSO-A algorithm is having excellent capability in searching global minima. Finally, the proposed algorithm has been successfully applied to model the dynamic cerebral autoregulation mechanism.

References

Angeline, P.J. (1998) 'Evolutionary optimisation versus particle swarm optimisation: philosophy and performance difference', *Proc. of 7th Annual Conf. on Evolutionary Programming*, pp.601–610.

Cai, X., Cui, Z., Zeng, J. and Tan, Y. (2007) 'Performance dependent adaptive particle swarm optimization', *International Journal Innov. Comput. Inf. Control*, Vol. 3, No. 6, pp.1697–1706.

Cerny, V. (1985) 'A thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm', *Journal of Optimisation Theory and Applications*, Vol. 45, pp.41–45.

Chaojun, D. and Zulian, Q. (2006) 'Particle swarm optimisation algorithm based on the idea of simulated annealing', *International journal of Computer Science and Network Security*, Vol. 6, No. 10, pp.152–157.

Chen, S., Hong, X., Luk, B.L. and Harris, C.J. (2009) 'Non-linear system identification using particle swarm optimisation tuned radial basis function models', *International Journal of Bio-Inspired Computation*, Vol. 1, No.4, pp.246–258.

Christober, A.R.C. and Mohan, M.R. (2007) 'An evolutionary programming based simulated annealing method for solving the unit commitment problem', *International J. Elect. Power Energy Syst*., Vol. 29, No. 7, pp.540–550.

Chu, S.C., Tsai, P. and Pan, J.S. (2006) 'Parallel particle swarm optimisation algorithms with adaptive simulated annealing', *Studies in Computational Intelligence Book Series*, Vol. 31, pp.261–279, Springer Berlin Heidelberg.

Clerc, M. (2007) 'Particle swarm optimisation', *ISTE*, pp.87–138 USA.

Dong, C., Wang, G., Chen, Z. and Yu, Z. (2008) 'A method of self adaptive inertia weight for PSO', *Proc. of Int. conference on Computer Science and Software Engineering*, Vol. 1, pp.1195–1198.

Eberhart, R. and Shi, Y. (2001) 'Particle swarm optimisation: development, application and resources', *IEEE Congress on Evolutionary Computation*, pp.81–86.

Fang, L., Chen, P. and Liu, S. (2007) 'Particle swarm optimisation with simulated annealing for TSP', *Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases*, pp.206–210, Corfu Island, Greece.

Gazi, V. and Passino, K.M. (2003) 'Stability analysis of swarms', *IEEE Transaction on Automatic Control*, Vol. 48, pp.692–697.

Ioannisc, C.K., Efthymlas, I.K. and Kyriakos, C.G. (2004) 'Gradient assisted radial basis function network; theory and application', *Applied Mathematical Modelling*, Vol. 28, pp.197–209.

Kang, D., Lee, B. and Won, S. (2007) 'Nonlinear system identification using ARX and SVM with advanced PSO', *33rd Annual Conference of the IEEE Industrial Electronics Society*, pp.598–603, Taipei, Taiwan.

Kennedy, J. and Eberhart, R. (1995) 'Particle swarm optimisation', *International Conference on Neural Networks*, Vol. 4, pp.1942–1948, Piscataway, NJ.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983) 'Optimisation by simulated annealing', *Science*, Vol. 220, pp.671–680.

Leontaritis, I.J. and Billings, S.A. (1987) 'Model selection and validation methods for nonlinear systems', *International Journal of Control*, Vol. 45, pp.311–341.

Litinetski, V.V. and Abrahamzon, B.M. (1998) 'A multistart adaptive random search method for total constrained optimisation in engineering applications', *Engineering Optimisation*, Vol. 30, pp.125–154.

Ljung, L. (1987) *System Identification – Theory for the User*, pp.197–235, Prentice-Hall, Englewood Cliffs, NJ.

Lovberg, M., Rasmussen, T. and Krink, T. (2001) 'Hybrid particle swarm optimizer with breeding and subpopulation', *Proceedings of the Third Genetic and Evolutionary Computation Conference*, Vol. 1, pp.469–476.

Mahony, P.J., Panerai, R.B., Deverson, S.T., Hayes P.D. and Evans D.H. (2000) 'Assessment of thigh cuff technique for measurement of dynamic cerebral autoregulation', *Stroke*, Vol. 31, p.476.

Meiying, Y. and Xiaodong, W. (2007) 'PSO-based parameter estimation of nonlinear systems', *Proceedings of the 26th Chinese Control Conference*, pp.533–536, Zhangjiajie, Hunan, PR China.

Niknam, T., Amiri, B., Olamaei, J. and Arefi, A. (2009) 'An efficient hybrid evolutionary optimisation algorithm based on PSO and SA for clustering', *Journal of Zhejiang University Science A*, Vol. 10, No. 4, pp.512–519.

Panerai, R.B. (2004) 'System identification of human cerebral blood flow regulatory mechanisms', *Cardiovascular Engineering: An International Journal*, Vol. 4, No. 1, pp.59–71.

Ratnaweera, A., Halgamuge, S.K. and Watson, H.C. (2004) 'Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients', *IEEE Transactions on Evolutionary Computation*, Vol. 8, pp.240–255.

Sadati, N., Amraee, T. and Ranjbar, A.M. (2009) 'A global particle swarm based simulated annealing optimisation technique for under voltage load shedding problem', *Applied Soft Computing*, Vol. 9, pp.652–657, Elsevier,.

Sharma, N., Ray, A.K., Sharma, S., Shukla, K.K., Pradhan, S. and Aggarwal, L.M. (2009) 'Segmentation of medical images using simulated annealing based fuzzy C means algorithm', *International Journal of Biomedical Engineering and Technology*, Vol. 2, No. 3, pp.260–278.

Shi, Y. and Eberhart, R. (1998) 'A modified particle swarm optimizer', *Proc. IEEE World Congress on Computational Intelligence*, pp.69–73.

Shinzawa, H., Jiang, J.H., Iwahashi, M. and Ozaki, Y. (2007) 'Robust curve fitting method for optical spectra by least median squares (LMedS) estimator with particle swarm optimisation', *Analytical Sciences*, Vol. 23, pp.781–785.

Van den Bergh, F. and Engelbrecht, A.P. (2002) 'A new locally convergent particle swarm optimizer', *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, Hammamet, Tunisia.

Xie, X.F., Zhang, W.J. and Yang, Z.L. (2002) 'Adaptive particle swarm optimisation on individual level', *International Conference on Signal Processing*, pp.1215–1218, Beijing, China.

Zhao, L. and Yang, Y. (2008) 'PSO-based single multiplicative neuron model for time series prediction', *Expert Systems with Applications*, Vol. 36, No. 2, pp.2805–2812.