Sādhanā Vol. 36, Part 6, December 2011, pp. 941–961. © Indian Academy of Sciences

A novel parameter estimation method for metal oxide surge arrester models

MEHDI NAFAR[1,*], GEVORK B GHAREHPETIAN[2] and TAHER NIKNAM[1]

[1]Department of Electrical Engineering, Marvdasht Branch, Islamic Azad University, Marvdasht, Iran
[2]Electrical Engineering Department, Amirkabir University of Technology, Tehran, Iran
e-mail: mnafar@miau.ac.ir

MS received 9 September 2010; revised 12 January 2011; accepted 12 May 2011

Abstract. Accurate modelling and exact determination of Metal Oxide (MO) surge arrester parameters are very important for arrester allocation, insulation coordination studies and systems reliability calculations. In this paper, a new technique, which is the combination of Adaptive Particle Swarm Optimization (APSO) and Ant Colony Optimization (ACO) algorithms and linking the MATLAB and EMTP, is proposed to estimate the parameters of MO surge arrester models. The proposed algorithm is named Modified Adaptive Particle Swarm Optimization (MAPSO). In the proposed algorithm, to overcome the drawback of the PSO algorithm (convergence to local optima), the inertia weight is tuned by using fuzzy rules and the cognitive and the social parameters are self-adaptively adjusted. Also, to improve the global search capability and prevent the convergence to local minima, ACO algorithm is combined to the proposed APSO algorithm. The transient models of MO surge arrester have been simulated by using ATP-EMTP. The results of simulations have been applied to the program, which is based on MAPSO algorithm and can determine the fitness and parameters of different models. The validity and the accuracy of estimated parameters of surge arrester models are assessed by comparing the predicted residual voltage with experimental results.

Keywords. Metal oxide surge arrester models; PSO; ACO; parameter estimation; EMTP.

1. Introduction

Metal oxide (MO) surge arresters are widely used as protective devices against switching and lightning over-voltages in power systems. The proper nonlinear voltage-current characteristics,

[*] For correspondence

ignorable power losses, high level reliability in the operation time, high speed response to the over-voltages and long life time are some advantages of MO surge arresters. Accurate modelling and simulation of their dynamic characteristics are very important for arrester allocation, systems reliability and insulation coordination studies (IEEE WG 1992; Pinceti & Giannettoni 1999; Martinez & Durbak 2005). For switching over voltages studies, the surge arresters can be represented by their nonlinear V-I characteristics. However, such a presentation would not be suitable for fast front transient and lightning surge studies. Because the MO surge arrester exhibits dynamic characteristics such that the voltage across the surge arrester increases as the time-to-crest of the arrester current decreases and the voltage of arrester reaches a peak before the arrester current peaks (IEEE WG 1992). Typically, the residual voltage for an impulse current having a front time equal to 1 μs is 8–12% higher than that of predicted for an impulse current having a front time equal to 8 μs. The residual voltage for longer time-to-crests between 45 and 60 μs, is 2–4% lower than that of a 8 μs current impulse (Martinez & Durbak 2005). In order to reproduce the MO surge arrester dynamic characteristics mentioned previously, a lot of researches have been done on modelling and simulation of MO surge arresters (Martinez & Durbak 2005). A dynamic model has been presented based on the data base of (IEEE WG 1992) for fast impulse currents (time-to-crest of 0.5–4.5 μs). To estimate the model parameters, an iterative trial and error procedure has been proposed, which matches the peak of discharge voltage obtained with 8/20 μs impulse current. The starting values for the parameters determination process, has been determined by considering the height and the number of column of the arrester, for five linear elements, and two V-I curves for the nonlinear elements. This method is usually time consuming and only applicable to the IEEE model (Martinez & Durbak 2005). The IEEE model was changed and simplified to other models by researchers (Pinceti & Giannettoni 1999; Fernandez & Diaz 2001; Popov *et al* 2002). The main problem of proposed models is essentially their parameters calculation and estimation (Li *et al* 2002; Zinik *et al* 2005). Numerical method has been proposed for estimating the parameters of the surge arrester models in (Li *et al* 2002). This method is based on comparing residual voltages of simulation and experimental results. The measurement of the residual voltage when a 8/20 μs impulse current is applied to the arrester is the start point of this method (Li *et al* 2002). In this paper, an adaptive parameter control is used for inertia weight by using a fuzzy logical controller and the cognitive and the social parameters are self-adaptively evaluated. Also, in order to avoid trapping in local optima, Ant Colony Optimization (ACO) algorithm is combined to Adaptive PSO (APSO) to explore the search space much more efficiently. Using the proposed algorithm and linking the MATLAB and EMTP programs, parameters of MO surge arrester models are identified. The estimated parameters of models are verified by comparing the results of simulations by EMTP (EMTP Role Book 1997) with the experimental results. The results show the ability of the proposed algorithm in estimating the surge arrester parameters.

2. Surge arrester models

In this paper, dynamic models of surge arresters are investigated in the following paragraphs. The IEEE WG 3-4–11 group proposed the model of figure 1, including the nonlinear resistances A_0 and A_1, which are separated by a RL low pass filter (IEEE WG 1992). The parameters calculation procedure of this model has been presented in (Martinez & Durbak 2005). It is based on the estimated height of the arrester, the number of columns of MO disks and the curves shown in figure 2.

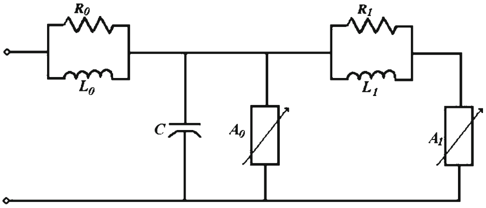

Figure 1. IEEE model.

The model, shown in figure 3, has been proposed by Pinceti–Gianettoni (Pinceti & Giannettoni 1999). This model is based on IEEE model with some differences. In this model the resistance R_0 stabilizes the numerical oscillations. The nonlinear resistors A_0 and A_1 can be estimated by using curves shown in figure 2. The calculation procedure of inductances for this model has been presented in (Pinceti & Giannettoni 1999).

Fernandez & Diaz (2001) have presented other model which is based on IEEE model too. This model is shown in figure 4. In Fernandez–Diaz model, the ratio I_0 to I_1 is assumed to be remained constant all over the voltage range of their protection characteristics where I_0 and I_1 are the currents following A_0 and A_1 respectively and the voltage percentage increasing of the input terminal depends only on the inductance L_1. The details of the calculation procedure of this model have been presented in (Fernandez & Diaz 2001).

The model, shown in figure 5, has been proposed by Popov for switching studies (Popov *et al* 2002). To estimate the parameters of this model, an iterative trial and error procedure has been proposed in (Popov *et al* 2002).

Above mentioned procedures do not always result in the best parameters, but these procedures can provide a good estimation (a starting point) (Li *et al* 2002). It should be mentioned that these procedures and their applications are limited to mentioned models.

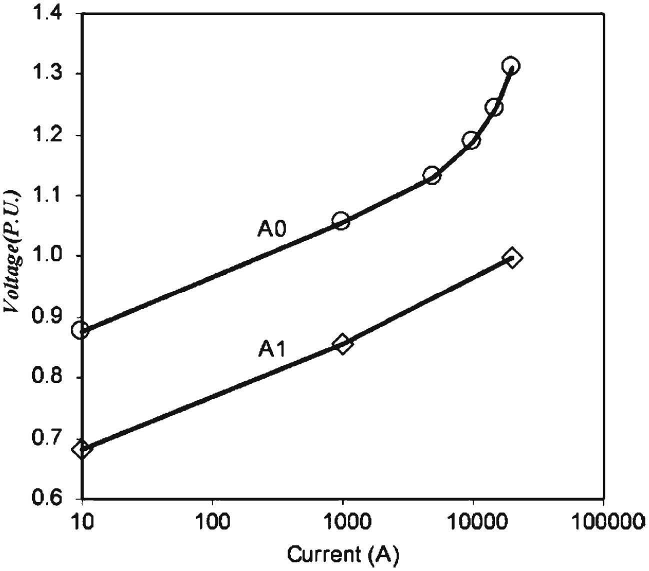

Figure 2. Nonlinear characteristics of A_0 and A_1.

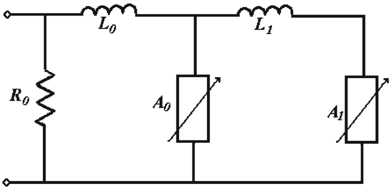

Figure 3. Pinceti–Gianettoni model.

In recent years, different researches have been presented for determining the parameters of all arrester models (Li *et al* 2002; Zinik *et al* 2005). A numerical method was proposed for estimating the parameters of the three suggested models in (Li *et al* 2002). This technique is based on comparison of the simulation results of residual voltages and the results derived from 8/20 μs experimental measurements (Li *et al* 2002). In this method, surge arrester parameters are estimated by minimizing the following objective function:

$$E = \int_0^T W(t)[V(t,\bar{x}) - Vm(t)]^2 dt, \tag{1}$$

where E is the sum of the quadratic error, T is the duration of injected impulse current signal, $V(t,\bar{x})$ is the predicted residual voltage obtained from simulation results, $V_m(t)$ is the measured residual voltage, $\bar{x}$ is the state variable vector (parameters of surge arrester model) and $W(t)$ is the weighting function, derived from numerical experimentation. It can take the following form for different models:

$$\begin{aligned} W(t) &= \cos\left(\frac{\pi t}{3T} + \frac{\pi}{6}\right) \\ W(t) &= \cos\left(\frac{\pi t}{2T}\right) \\ W(t) &= 1.0. \end{aligned} \tag{2}$$

In this method, the non-linear resistances have presented by piece-wise functions and consequently a linearization has been adopted. The problem of optimization has been solved in two stages with an aim of avoiding possible numerical oscillations of the simulated voltage.

In this paper, a new technique based on heuristic algorithms is proposed to estimate the best parameters of surge arresters models. Proposed method is general and can be applied to all surge arrester models. Unlike existing methods, formulation or equation and information on the MO surge arrester dimensions are not necessary. Also, the application of the weighting function is not necessary in the proposed method and the non-linear resistances can be presented by an

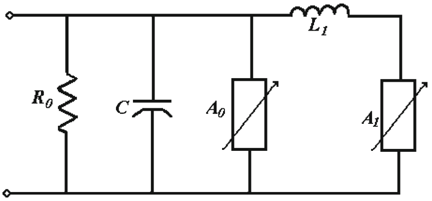

Figure 4. Fernandez–Diaz model.

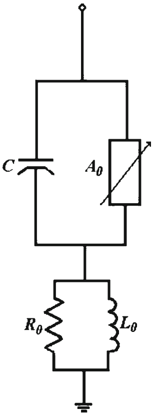

Figure 5. Popov model.

exponential voltage–current characteristic, as expressed by the equation (3) (Zinik *et al* 2005)

$$I = p\left(\frac{V}{V_{ref}}\right)^{q}, \tag{3}$$

where p and q are constant values; V and I are voltage and current of surge arrester, respectively and V_{ref} is an arbitrary reference voltage.

3. Objective function

In this paper, the ATP-EMTP software is used as a simulation tool. The 10 kA, 8/20 μs current is applied to the simulated models of surge arresters. The residual voltage of each model is compared to the experimental results obtained from (Schmidt *et al* 1989; Kim *et al* 1996; Hinrichsen 2001) and the parameters of surge arrester models can be estimated by minimizing the following equation:

$$E = \int_0^T [V(t,\bar{x}) - Vm(t)]^2 dt. \tag{4}$$

This objective function is similar to the objective function proposed in (Li *et al* 2002). Only in (Li *et al* 2002) an additional term (weighting function) has been considered to increase the convergence speed. In proposed MAPSO method, the application of the weighting function is not necessary. Usually, the measured residual voltage is a discrete function. Accordingly, the objective function (4) can be rewritten as follows:

$$E = \sum_{j=1}^{N} [V(j\Delta t,\bar{x}) - V_m(j\Delta t)]^2 \Delta t, \tag{5}$$

where N is the number of discrete points and the $\Delta t = T/N$ is the time interval.

4. Optimization procedure

4.1 *ACO algorithm*

Ants are insects which live together. Being blind, they find the shortest path from nest to food using the pheromone. Pheromone, a chemical material deposited by the ants, serves as a critical communication facility among ants which help them in their path recognition. Density of pheromone deposited by ants, determines the shortest path of their ways to food (Niknam *et al* 2005).

Generally, the state transition probability to select the next path could be expressed, as follows:

$$P_{ij} = \frac{(\tau_{ij})^{\gamma_2}(1/L_{ij})^{\gamma_1}}{\sum\limits_{j=1, j\neq i}^{NA} (\tau_{ij})^{\gamma_2}(1/L_{ij})^{\gamma_1}}. \tag{6}$$

After choosing the next path, updating the trail density of pheromone is by the following equation:

$$\tau_{ij}(k+1) = \rho\tau_{ij}(k) + \Delta\tau_{ij}, \tag{7}$$

where τ_{ij} is the intensity of pheromone between the nodes j and i, L_{ij} is the length of path between the nodes$j$ and i, γ_1 is the control parameter for determining the weight of trail intensity, γ_2 is the control parameter for determining the weight of the length of path, *NA* is the number of ants, ρ is a coefficient such that $(1-\rho)$ represents evaporation of trail between time k and $k+1$ and $\Delta\tau_{ij}$ is the amount of pheromone trail added to τ_{ij} by ants.

4.2 *PSO algorithm*

PSO is a population based stochastic optimization algorithm developed by Eberhart and Kennedy, inspired by social behaviour of bird flocking or fish schooling. It is a useful technique to solve many optimization problems (Kennedy & Eberhart 1995; Eberhart & Shi 2001; Gaing 2003; Park 2005). PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO algorithm, the potential solutions, called particles, fly through the problem space by following the current optimum particles. Equation (8) could describe the content of this concept.

$$\begin{aligned} Vel_i^{(t+1)} &= \omega.Vel_i^{(t)} + c_1.rand_1(.).(Pbest_i - X_i^{(t)} + c_2.rand_2(.).(Gbest - X_i^{(t)}) \\ \bar{X}_i^{(t+1)} &= \bar{X}_i^{(t)} + Vel_i^{(t+1)}, \end{aligned} \tag{8}$$

where Vel_i^t is velocity of the i th particle, $rand_1(.)$ and $rand_2(.)$ are random numbers between 0 and 1, *Pbest* is best previous experience of the i th particle that recorded and *Gbest* is best particle among the entire population.

The constants c_1 and c_2 are positive weighting coefficients of the stochastic acceleration terms which stimulate each particle towards *Pbest* and *Gbest* positions. Low values allow particles to go far from the target region and high values result in abrupt movements toward, or backward the target region. The inertia weight ω presents the degree of the particles momentum. The appropriate selection of inertia weight ω in (8) provides a proper global and local search as it is essential to minimize iteration average to achieve a sufficient optimal solution. Approximately the coefficient ω often decreases linearly from 0.9 to 0.4 during a run.

4.3 *Adaptive PSO*

As given in equation (8), three parameters ω, c_1 and c_2 have great influence on the PSO algorithm performance and control the behaviour and efficacy of the PSO algorithm. The inertia weight ω is used to control the impact of the previous history of velocities on the current velocity. Proper choice of the ω provides a balance between global and local optimum points (Niknam *et al* 2010). The factors c_1 and c_2 determine the effect of the personal best $Pbest_{,j}$ and global best *Gbest*.

Since c_1 represents how much the particle trusts its own historical experience, it is called cognitive parameter. On the other hand, c_2 represents the 'social influence' that pushes the swarm to converge to the current globally best region, and is called social parameter. Also, on each dimension, particle velocities are limited to minimum and maximum velocities, which are user-defined parameters as follows:

$$Vel_{j,\min} \leq Vel_j^{t+1} \leq Vel_{j,\max}. \tag{9}$$

To control excessive roaming of particles outside the search space, usually, $Vel_{j,\min}$ is assumed as $Vel_{j,\max}$. If $Vel_{j,\max}$ is too high, particles may fly past good solutions. If $Vel_{j,\max}$ is too small, particles may not explore sufficiently beyond local solutions. In many experiences with PSO, $Vel_{j,\max}$ was often set at 10–20% of the dynamic range on each dimension.

The appropriate choice of control parameters is very important for the success in this type of evolutionary algorithms. In Arumugam & Rao (2008) have described two ways of modifying the parameter control; adaptive parameter control and self-adaptive parameter control.

Adaptive parameter control, takes place when there is some form of feedback from the search that is used to determine the direction and/or magnitude of the change to the strategy parameter. In this method, the parameter change (according to a heuristic rule) takes feedback from the current search state. The information of current state is usually the current iteration of the search, the operator's performance, and/or the diversity of the population. For example, in (Eberhart & Kennedy 1995; Arumugam & Rao 2008), linearly decreasing inertia over the generations is used as adaptive inertia weight control and in (Niknam *et al* 2010) a fuzzy logical controller has been used. While in self-adaptive parameter control, the parameters of the meta-heuristic are incorporated into the representation of the solution. Thus, the values of the parameters evolve together with the solutions of the population. In this paper, a fuzzy-adaptive parameter control for the inertia weight is used. Also, the other parameters of the evolution variables, are investigated, in order to self adaptively control them.

4.3a *Fuzzy adaptive inertia weight factor*: In PSO, The search process is a nonlinear and dynamic procedure. Therefore, when the environment itself is dynamically changed over the time, the optimization algorithm should be able to adapt dynamically to the changing environment. The change of the particle's situation is directly correlated to the inertia weight. Proper choice of the inertia weight ω provides a balance between global and local optimum points.

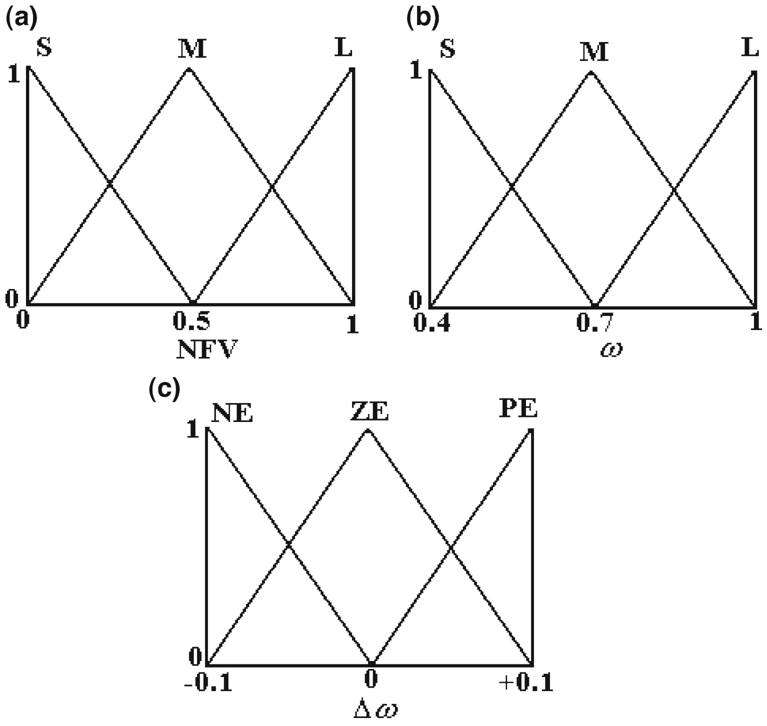

Figure 6. Membership functions of inputs and outputs; (**a**) NFV, (**b**) ω and (**c**) $\Delta\omega$.

Several methods have been applied to handle the inertia weight during the progression of the optimization process. Constant inertia weight, linearly decreasing inertia weight and random inertia weight are some examples (Niknam *et al* 2010). In this paper, a fuzzy IF/THEN rules are used to adaptively control the inertia weight of PSO. Four steps are taken to create the fuzzy system: fuzzification, fuzzy rules, fuzzy reasoning, and defuzzification. These steps are described in the following subsections.

4.3b *Fuzzification*: The fuzzification comprises the process of transforming crisp values into grades of membership for linguistic terms of fuzzy sets. For each input and output variable selected, two or more membership functions are described. Normally, they are three but can be more. In this paper, among a set of membership functions, left-triangle, triangle, and right triangle membership functions are used for every input and output as shown in figure 6. All the memberships of input are presented in three linguistic levels; S, M, and L for small, medium,

Table 1. Fuzzy rules for variations of inertia weight.

$\Delta\omega$		ω		
		S	M	L
NFV	S	ZE	NE	NE
	M	PE	ZE	NE
	L	PE	ZE	NE

and large, respectively in table 1. The output variable has been presented in three fuzzy sets of linguistic values; NE (negative), ZE (zero), and PE (positive) with associated membership functions, as shown in figure 6 (Niknam *et al* 2010).

4.3c *Fuzzy rules*: The fuzzy rules are a series of IF/THEN statements. These statements are usually derived by an expert to achieve optimum results. In this paper, the Mamdani-type fuzzy rules have been used to evaluate the conditional statements that comprise fuzzy logic. For example: if (NFV is L) and (ω is M) THEN ($\Delta\omega$ is ZE), where NFV is normalized fitness value and NFV is an input variable between 0 and 1. The fuzzy rules of table 1 are used to select the inertia weight correction ($\Delta\omega$). Each rule represents a mapping from the input space to the output space.

4.3d *Fuzzy reasoning*: In this paper, Mamdani's fuzzy inference method is used to map the inputs to the outputs. The AND operator is used for the combination of membership values for each fired rule to generate the membership values for the fuzzy sets of output variables in the consequent part of the rule.

Since there may be several rules fired in the rule sets, for some fuzzy sets of the output variables there may be different membership values obtained from different fired rules.

To obtain a better inertia weight under the fuzzy system, the current best performance evaluation and current inertia weight are selected as inputs variable, where as the output variable is the change in the inertia weight. The NFV is used as an input variable between 0 and 1, and is defined as:

$$NFV = \frac{FV - FV_{\min}}{FV_{\max} - FV_{\min}}. \tag{10}$$

In the first iteration, the calculated value of FV may be as $FV_{\min}$ for the next iterations.

In equation (10), $FV_{\max}$ is the worst solution to the minimization process. Typical inertia weight value is $0.4 \leq \omega \leq 0.9$. Both positive and negative corrections limits are required for the inertia weight. Therefore, a range of [−0.1 0.1] has been chosen for the inertia weight correction.

$$\omega^{t+1} = \omega^{t} + \Delta\omega. \tag{11}$$

In order to choose an appropriate representative value as the final output (crisp values), defuzzification must be done. It will be illustrated at a later point.

4.3e *Defuzzification*: In order to obtain a crisp value, the output must be defuzzyfied. For defuzzification of every input and output, the method of centroid (center-of-sums) has been used for the membership functions shown in figure 6.

4.4 *Self-adaptive parameter control*

In order to obtain a self-adaptive parameter control, the parameters must be encoded within the solution of the problem. A self-adaptive control of c_1, c_2 and $V_{\max}$ is considered to avoid the cumbersome task of first localizing and then fine-tuning these three parameters. Appropriate tuning of c_1 and c_2 in equation (8) may improve efficiency, accelerate the search process and reduce the risk of settling in one of the local minima. A study of these acceleration parameters is given for PSO in (Kennedy 1999). As default values, $c_1 = c_2 = 2$ were proposed, but next researches indicated that alternative configurations, depending on the application, may produce superior performance. Recent work has been shown that it might be even better to select a larger cognitive parameter, c_1, than a social parameter, c_2, with constraint $c_1 + c_2 \leq 4$. In (Arumugam & Rao 2008), it is suggested that the acceleration coefficients should neither set to a constant value nor set as a linearly decreasing time varying function. Instead, these parameters are defined as a function of local best and global best values of the fitness function of a minimization problem. In this paper, the acceleration coefficients and the clamping velocity are neither set to a constant value (like in standard PSO) nor set as a time varying function (like in adaptive PSO variants). Instead they are incorporated into the optimization problem, as explained below. The parameters of particle will be allowed to self-adapt by using the same process used by PSO given by equations (8). To this end, these three parameters are considered as three new variables that are added with position vectors Xj.

In general, if N_g is the dimension of the problem and p is the number of self-adapting parameters, the new position vector for particle j will be, as fallows:

$$X_j^{new} = [x_{j,1}, x_{j,2}, ..., x_{j,Ng}, x_{j,Ng+1}, ..., x_{j,Ng+p}]. \tag{12}$$

It is obvious that the first N_g variables correspond to the real position vector of the particle in the search space, while the last p variables for its personal acceleration constants and velocity limit. Obviously, these self-adapting parameters do not enter the fitness function but are manipulated using the same mixed individual–social learning paradigm as the one used in PSO.

Also, dimension of $Velj$ and $Pbest_{,j}$, which represent the velocity and best position so far for particle j, respectively, increase as follows:

$$V_j^{new} = [v_{j,1}, v_{j,2}, ..., v_{j,Ng}, v_{j,Ng+1}, ..., x_{j,Ng+p}], \tag{13}$$

$$P_{best,j}^{new} = [P_{best,1}, P_{best,2}, ..., P_{best,Ng}, P_{best,Ng+1}, ..., P_{best,Ng+p}]. \tag{14}$$

Using equation (8), each particle would be additionally endowed with the ability of adapting its parameters by aiming at both the parameters it had when it got its best position in the past and the parameters of the leader, which managed to bring this best particle to its advantaged position.

5. Proposed MAPSO algorithm

The PSO method should be considered as a useful method, which is powerful enough to handle various kinds of nonlinear optimization problems. Nevertheless, it may be trapped into local optima, if over a number of iterations, global best and local best positions are equal to the position of the particle. Recently, numerous ideas have been used to overcome this drawback using other global optimization algorithms such as Evolutionary programming (EP), Genetic Algorithm (GA) or Simulated Annealing (SA) along with the PSO (Gaing 2003; Park 2005). The performance of the standard PSO greatly depends on its parameters, such as inertia weight, cognitive and the social parameters, and it often suffers from the problem of being trapped in the local optima. In this paper, an adaptive parameter control is used for inertia weight by using a fuzzy logical controller and the cognitive and the social parameters are self-adaptively evaluated. Also, in order to avoid trapping in local optima, ACO algorithm is combined to APSO to explore the search space much more efficiently. This new algorithm proposes the application of the intelligent decision-making structure of ACO algorithm to the APSO algorithm such that a unique global best position is obtained for each particle. However, it uses random selection procedure of ACO algorithm to determine different global best positions of each distinct agent. This algorithm called Modified Adaptive Particle Swarm Optimization (MAPSO) is used to minimize the cost function of the surge arrester parameters estimation problem. The proposed MAPSO algorithm has the following steps:

Step 1: *Generate the initial population and initial velocity*

The initial population and initial velocity of each particle are randomly generated as follows:

$$
\begin{aligned}
Population &= \begin{bmatrix} \overline{X}_1 \\ \overline{X}_2 \\ \cdots \\ \overline{X}_{N_{Swarm}} \end{bmatrix} \\
\overline{X}_i &= [x_{i,1}, x_{i,2}, ..., x_{i,Ng}, x_{i,Ng+1}, ..., x_{i,Ng+p}] \\
x_{i,j} &= rand(.) \times (x_j^{\max} - x_j^{\min}) + x_j^{\min} \\
j &= 1, 2, ..., (Ng + p);\ i = 1, 2, ..., N_{swarm};\ p = 3,
\end{aligned} \tag{15}
$$

$$
\begin{aligned}
Velocity &= \begin{bmatrix} Vel_1 \\ Vel_2 \\ \cdots \\ Vel_{N_{Swarm}} \end{bmatrix} \\
Vel_i &= [vel_{i,1}, vel_{i,2}, ..., vel_{i,Ng}, vel_{i,Ng+1}, ..., vel_{i,Ng+p}] \\
vel_{i,j} &= rand(.) \times (vel_j^{\max} - vel_j^{\min}) + vel_j^{\min} \\
j &= 1, 2, ..., (Ng + p);\ i = 1, 2, ..., N_{swarm};\ p = 3,
\end{aligned} \tag{16}
$$

where N_{swarm} is the number of the swarms, Ng is the number of the state variable, $x_i^{\max}$ is the maximum of i th state variable, $x_i^{\min}$ is the minimum of i th state variable, $v_i^{\max}$ is the maximum velocity of ith state variable and $v_i^{\min}$ is the minimum velocity of i th state variable.

Step 2: *Generate the initial trail intensity*

In this initialization phase, it is assumed that the trail intensity between each pair of swarms is the same and is generated, as follows:

$$
\begin{aligned}
Trail\_Intensity &= [\tau_{ij}]_{N_{Swarm} \times N_{Swarm}} \\
\tau_{ij} &= \tau_0,
\end{aligned} \tag{17}
$$

where, τ_0 is the initial trial intensity.

Step 3: *Coupling to EMTP*

The surge arrester model is simulated by EMTP using the given data (parameters). Then, the simulation results are transferred to the MAPSO-based developed program to calculate the objective function.

Step 4: *Calculate the objective function*

The objective function (i.e., equation (5)) is calculated for each individual by using the simulation results obtained in the step 3. (Only the first Ng variables enter the fitness function.)

Step 5: *Sort the initial population*

In this step, the initial population is sorted in ascending order considering the value of the objective function of each individual.

Step 6: *Select the best global position*

The surge arrester parameters in additional self-adaptive parameters are represented the position of particle in the swarm. The individual that has the minimum value of the objective function is selected as the best global position (i.e., *Gbest*).

Step 7: *Select the best local position*

The best local position (*Pbest*) is selected for each individual.

Step 8: *Update the parameters*

In this algorithm, the proper choice of inertia weight, ω, is updated by the fuzzy rules and the other parameters are tuning by self-adaptive parameters.

Step 9: *Select the ith individual*

The ith individual is selected and neighbours of this particle should be dynamically defined as follows:

$$S_i = \left\{ \overline{x}_j \middle| \left\| \overline{x}_i - \overline{x}_j \right\| \leq 2D_0 \left(\frac{1}{1 - \exp\left(\frac{-at}{t_{\max}}\right)} \right), i \neq j \right\}, \tag{18}$$

where, D_0 is the initial neighbourhood radius and a is the parameter used to tune the neighbourhood radius.

Step 10: *Calculate the next position for the ith individual*

There are two approaches to calculate the next position, as follows:

- *Approach A)* if $S_i \neq \{\}$, where $\{\}$ stands for the null set.

In this case, the transition probabilities between $\overline{x}_i$ and each individual in S_i are calculated by the following equation:

$$\begin{aligned} \left[Probability\right]_i &= [P_{i1}, P_{i2}, ..., P_{i,M}]_{1\times M} \\ P_{ij} &= \frac{(\tau_{ij})^{\gamma_2}(1/L_{ij})^{\gamma_1}}{\sum\limits_{j=1}^{M} (\tau_{ij})^{\gamma_2}(1/L_{ij})^{\gamma_1}} \\ L_{ij} &= \frac{1}{\left|F(\overline{x}_i) - F(\overline{x}_j)\right|}. \end{aligned} \tag{19}$$

Then the cumulative probabilities are calculated as follows:

$$[Cumulative\ probability]_i = [C_1, C_2, ..., C_M]_{1\times M},$$

where

$$\begin{aligned} C_1 &= P_{i1} \\ C_2 &= C_1 + P_{i2} \\ &\ldots. \\ C_j &= C_{j-1} + P_{ij} \\ &\ldots. \\ C_M &= C_{M-1} + P_{iM}, \end{aligned} \tag{20}$$

where, *M* is the number of members in *Si* and *Cj* is the cumulative probability for the jth individual in S_i.

The roulette wheel is used for the stochastic selection of the best global position, as follows:

A number between 0 and 1 is randomly generated and compared with the calculated cumulative probabilities. The first term of the cumulative probabilities (*Cj*), which is greater than the generated number, is selected and the associated position is considered as the best global position.

Then the ith particle is moved according to following rules, if $\overline{x}_j$ is selected as the best:

$$\begin{cases} Vel_i^{(t+1)} = \omega.Vel_i^{(t)} + c_1.rand_1(\circ).(Pbest_i - \overline{x}_i^{(t)}) \\ \qquad\qquad + c_2.rand_2(\circ).(\overline{x}_j - \overline{x}_i^{(t)}) \\ \overline{x}_i^{(t+1)} = \overline{x}_i^{(t)} + Vel_i^{(t+1)}. \end{cases} \tag{21}$$

The presumed pheromone level between *Xi* and *Xj* is updated in the next stage:

$$\tau_{ij}(t+1) = \rho.\tau_{ij}(t) + P_{ij}. \tag{22}$$

- *Approach B)* if $S_i = \{\}$, which means there is not any individual in particle's neighbourhood.

In this case, the ith particle is moved according to the following rules:

$$\begin{cases} Vel_i^{(t+1)} = \omega.Vel_i^{(t)} + c_1.rand_1(\circ).(Pbest_i - \overline{x}_i^{(t)}) \\ \qquad\qquad + c_2.rand_2(\circ).(Gbest - \overline{x}_i^{(t)}) \\ \overline{x}_i^{(t+1)} = \overline{x}_i^{(t)} + Vel_i^{(t+1)}. \end{cases} \tag{23}$$

Then, the trail intensity is updated, as follows:

$$\tau_{ij}(t+1) = \rho.\tau_{ij}(t) + r; \quad 0.1 \le r \le 0.5, \tag{24}$$

where index j represents the best particle index in the group.

The modified position for the ith individual is checked with its limit.

Step 11: If all individuals have been selected, go to the next step, otherwise $i = i + 1$ and go back to step 5.

Step 12: Check the termination criteria.
If the current iteration number reaches the predetermined maximum iteration number, the search procedures should be stopped; otherwise the initial population is replaced with the new population of swarms and then the algorithm goes back to step 3.

Step 13: The last *Gbest* is the solution of the problem.

6. Linking EMTP with MAPSO algorithm in MATLAB

Both EMTP and MATLAB are currently available on popular computer for electrical engineering applications. It can be said that the best optimization methods can be easily developed in

MATLAB and transient models of power system equipments can be simulated by EMTP. To use the ability of both software, a link between these programs is necessary. In Mahseredjian *et al* (1998), several techniques for linking EMTP with MATLAB are presented, where MATLAB functions can be called as EMTP. But, in this paper, ATP file is called as an input file of MATLAB. This approach is much easier than the other one. The parameters of surge arrester models have been determined by using MAPSO algorithm developed in MATLAB and the surge arrester models have been simulated by EMTP. A FORTRAN code file (ATP file) is developed for each EMTP Simulation file. By using input/output functions of MATLAB, ATP file can be called in

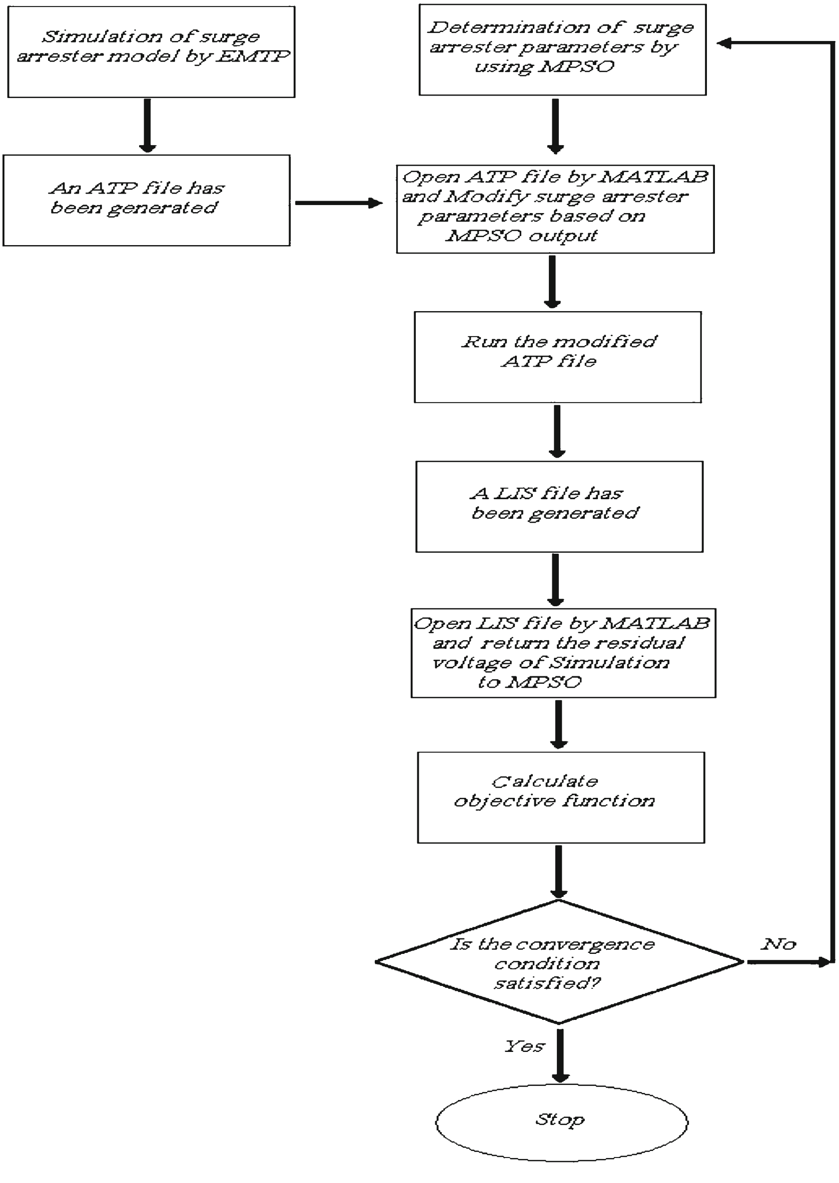
Simulation of surge arrester model by EMTP
An ATP file has been generated
Determination of surge arrester parameters by using MPSO
Open ATP file by MATLAB and Modify surge arrester parameters based on MPSO output
Run the modified ATP file
A LIS file has been generated
Open LIS file by MATLAB and return the residual voltage of Simulation to MPSO
Calculate objective function
Is the convergence condition satisfied?
No
Yes
Stop

Figure 7. Link between MATLAB and EMTP in the proposed method.

Table 2. Estimated parameters based on experimental results of (Schmidt *et al* 1989).

Parameter	IEEE Model	Pinceti Model	Fernandez Model	Popov Model
$R_0(\Omega)$	0.5022	29.738	1600	0.0305
$R_1(\Omega)$	0.1815	—	—	—
$L_0(\mu H)$	0.2505	0.0698	—	0.261
$L_1(\mu H)$	9.8010	0.6803	0.9193	—
$C(nF)$	1.4468	—	2.9547	0.325
p_0	3036.32	1746.44	5377.88	2982
q_0	10.67	10.22	16.69	10.75
p_1	2540.14	2508.79	5401.51	—
q_1	10.95	9.32	9.28	—
$Vreff_0[V]$	7570.49	7392.20	8125.15	7001
$Vreff_1[V]$	7177.07	7457.62	8097.14	—

MATLAB and can be modified. So, the surge arrester parameters can be modified according to MAPSO outputs. Using SYSTEM command, the modified ATP file can be run in MATLAB. After running the ATP file, a LIS file is generated. The simulation outputs of surge arrester models (for example, residual voltage) are in this file. Then, using input/output function, LIS file can be opened in MATLAB and the residual voltage of simulation can be returned to MAPSO algorithm in MATLAB. This procedure has been shown in the figure 7.

7. Parameter determination of surge arrester models

In this section, the proposed method is used to estimate parameters of the transient models of MO surge arresters. The surge arrester models have been simulated by EMTP. The surge arrester parameters, determined by MAPSO algorithm in MATLAB, are imported to the EMTP. The simulation is performed by using the (10 kA, 8/20 μs) impulse current. During optimization, the residual voltage of the model is determined by EMTP and then it is transferred to MAPSO algorithm in MATLAB to evaluate the fitness function. This procedure continues until the optimal solution has been determined for parameters. The residual voltage of simulations has been compared with the experimental data of from references (Schmidt *et al* 1989; Kim *et al* 1996;

Table 3. Estimated parameters based on experimental results of Kim *et al* (1996).

Parameter	IEEE Model	Pinceti Model	Fernandez Model	Popov Model
$R_0(\Omega)$	0.0990	15.00	7.22	0.0721
$R_1(\Omega)$	0.0980	—	—	—
$L_0(\mu H)$	0.4135	0.042	—	0.3423
$L_1(\mu H)$	1.99	0.407	0.8969	—
$C(nF)$	1.9999	—	9.01	0.705
p_0	2752.79	2174.26	3094.26	2479.9
q_0	20.00	8.50	12.32	8.57
p_1	3642.65	2849.41	6199.99	—
q_1	5.443	9.67	19.19	—
$Vreff_0[V]$	7546.225	8020.06	7909.69	6181.3
$Vreff_1[V]$	7099.503	7221.58	7811.48	—

Table 4. Estimated parameters based on experimental results of Hinrichsen (2001).

Parameter	IEEE Model	Pinceti Model	Fernandez Model	Popov Model
$R_0(\Omega)$	0.6127	33.49	85	0.0502
$R_1(\Omega)$	0.6114	—	—	—
$L_0(\mu H)$	0.1388	0.010	—	0.4243
$L_1(\mu H)$	2.445	0.998	1.850	—
$C(nF)$	2.028	—	0.227	0.546
p_0	2553.3	1530.13	2582.09	2164.42
q_0	10.54	9.81	10.59	9.24
p_1	4741.4	3100.98	4298.76	—
q_1	9.24	8.21	18.22	—
$Vreff_0[V]$	11412.89	11141.23	11222.62	9812
$Vreff_1[V]$	11222.95	10817.45	11411.71	—

Hinrichsen 2001). The experimental data of the arresters used in this paper have been listed in table 8. The estimated parameters of different surge arrester models, which have been determined by using the experimental data of (Schmidt *et al* 1989; Kim *et al* 1996; Hinrichsen 2001), are listed in tables 2–4. A comparison among experimental and the simulated residual voltage obtained by estimated parameters of the models of (IEEE WG 1992; Pinceti & Giannettoni 1999; Fernandez & Diaz 2001; Popov *et al* 2002) is presented in figures 8–10.

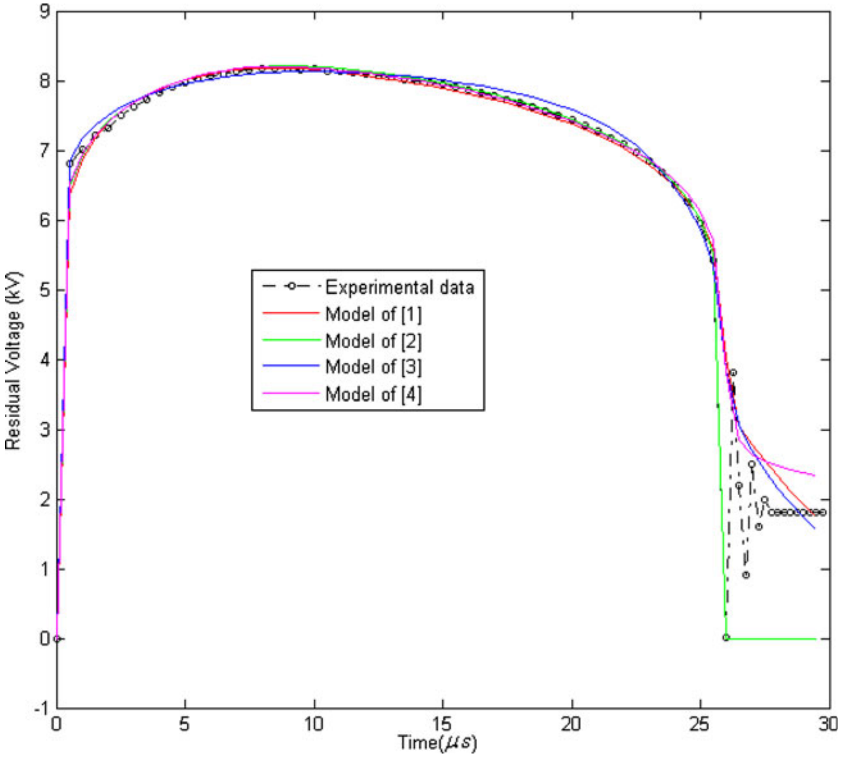

Figure 8. Comparison of residual voltage determined by estimated models and measured by Schmidt *et al* (1989).

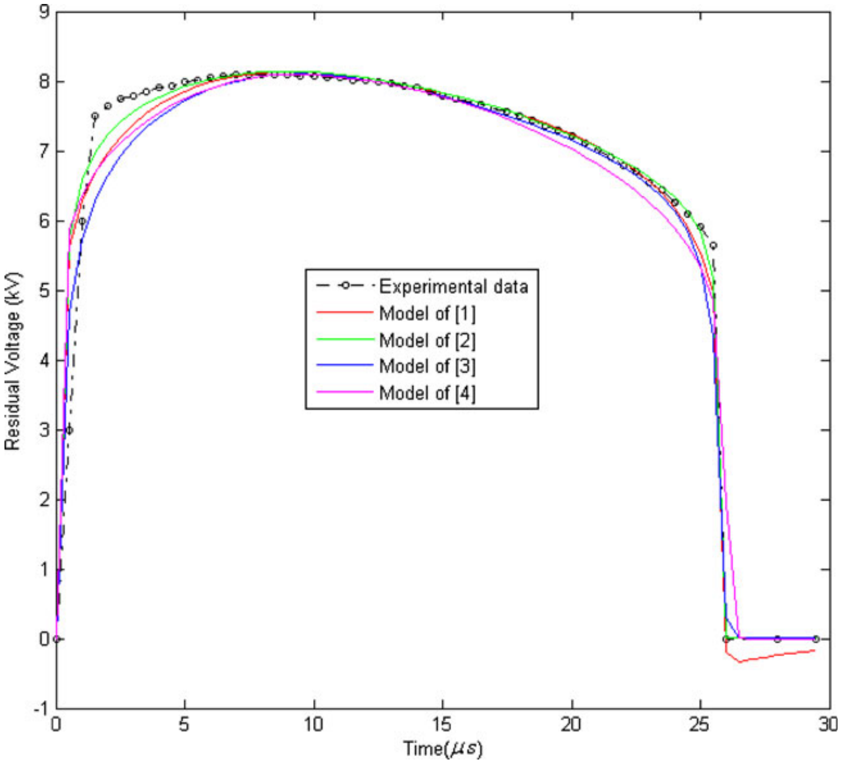

Figure 9. Comparison of residual voltage determined by estimated models and measured by Kim *et al* (1996).

It is obvious from these figures that the simulation of models using the estimated parameters results in residual voltages which have a good agreement with the experimental data. It should be noted that, using proposed algorithm, parameters of all models can be properly estimated.

8. Error analysis

In this section, by using the experimental data obtained from (Kim *et al* 1996), the ability of the proposed method to identify the parameters of surge arrester models and the ability of models to simulate the arrester dynamic behaviour are presented. Surge arresters have dynamic characteristics in which rising rate of residual voltage depend on the time to crest and peak value of current. These characteristics become important in considering the insulation coordination studies. In this section, the models of (IEEE WG 1992; Pinceti & Giannettoni 1999; Fernandez & Diaz 2001; Popov *et al* 2002), have been compared based on the estimated parameters of table 3. The impulse current peak values are 2.5 kA, 5 kA and 10 kA and the selected rise and fall times are 1/2 μs, 4/10 μs and 8/20 μs. The 1/2 μs impulse current has been applied to the surge arrester models and the simulation results have been used to determine the error by the following equation:

$$Error\% = \frac{VR_{sim} - VR_{meas}}{VR_{meas}} \times 100, \tag{25}$$

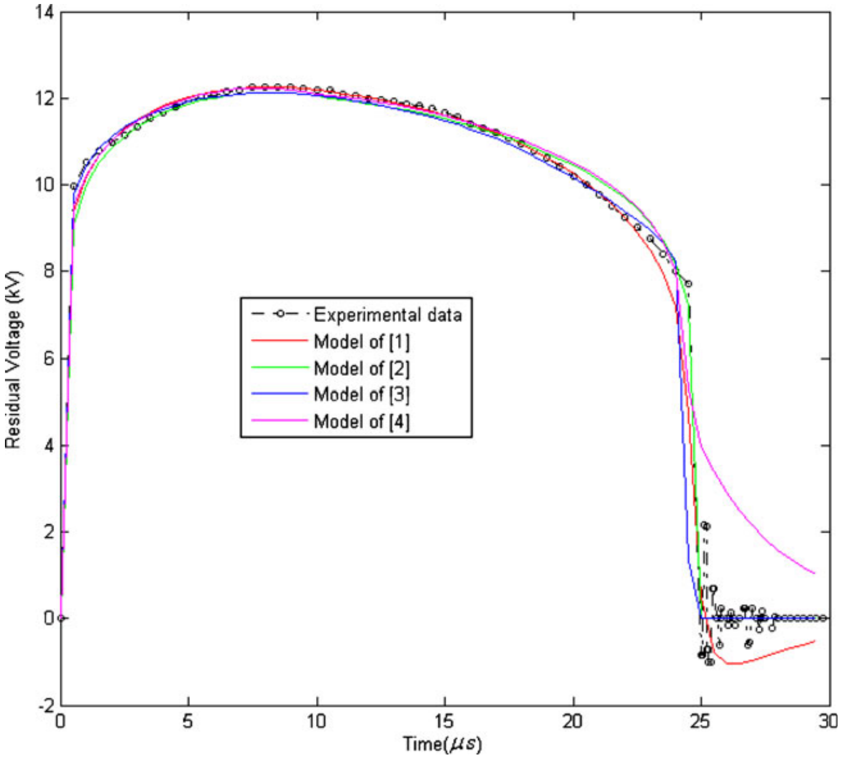

Figure 10. Comparison of residual voltage determined by estimated models and measured by Hinrichsen (2001).

Table 5. Comparison between simulated results and experimental results of Kim *et al* (1996).

	1/2 μs impulse current		
	Models	Peak of residual voltage (kV)	Error%

P.C. = 2.58 kA
P.V. = 7.46 kV
	IEEE	6.9467	−6.88

	Fernandez	7.2571	−2.72
	Pinceti	7.1006	−4.82
	Popov	6.485	−13

P.C. = 5.04 kA
P.V. = 7.97 kV
	IEEE	7.7548	−2.7

	Fernandez	7.8807	−1.12
	Pinceti	8.0130	0.54
	Popov	8.504	6.7

P.C. = 10.48 kA
P.V. = 8.62 kV
	IEEE	8.6978	0.90

	Fernandez	8.5170	−1.19
	Pinceti	8.707	1.01
	Popov	10.80	25

Table 6. Comparison between simulated results and experimental results of Kim *et al* (1996).

	1/2 μs impulse current		
	Models	Peak of residual voltage (kV)	Error%
P.C. = 2.95 kA	IEEE	7.0148	−3.38
P.V. = 7.26 kV	Fernandez	7.2393	−0.28
	Pinceti	6.9796	−3.86
	Popov	6.410	−11.7
P.C. = 4.98 kA	IEEE	7.5971	−0.30
P.V. = 7.62 kV	Fernandez	7.6785	0.7677
	Pinceti	7.5244	−1.25
	Popov	6.935	8.9
P.C. = 10.42 kA	IEEE	8.4106	2.32
P.V. = 8.22 kV	Fernandez	8.3245	1.27
	Pinceti	8.4513	2.79
	Popov	7.964	3.1

where VR_{sim} and VR_{meas} are the simulated and measured values, respectively. The result of this calculation has been presented in table 5. The same simulations and calculations have been repeated for 4/10 μs and 8/20 μs currents. The results are presented in tables 6 and 7, respectively. In these tables P.C. and P.V. stand for 'peak of impulse current' and 'peak of measured residual voltage', respectively.

According to the results of these tables, the following points can be drawn:

- The Popov model can not simulate dynamics behaviour of MO surge arresters properly. Because this model is a very simplified version of the IEEE model, and also, this model has been proposed for switching studies (Popov *et al* 2002).

Table 7. Comparison between simulated results and experimental results of Kim *et al* (1996).

	1/2 μs impulse current		
	Models	Peak of residual voltage (kV)	Error%
P.C. = 2.50 kA	IEEE	6.7254	−4.05
P.V. = 7.01 kV	Fernandez	7.0433	0.4750
	Pinceti	6.7794	−3.29
	Popov	6.375	9.05
P.C. = 5.01 kA	IEEE	7.4399	−0.67
P.V. = 7.49 kV	Fernandez	7.5426	0.70
	Pinceti	7.3984	−1.22
	Popov	7.0947	5.2
P.C. = 10.23 kA	IEEE	8.1014	0.26
P.V. = 8.08 kV	Fernandez	8.0983	0.22
	Pinceti	8.1029	0.28
	Popov	8.1004	0.24

Table 8. Experimental data of (Schmidt *et al* 1989; Kim *et al* 1996; Hinrichsen 2001).

Time (μs)	Experimental data of (Schmidt *et al* 1989) (kV)	Experimental data of (Kim *et al* 1996) (kV)	Experimental data of (Hinrichsen 2001) (kV)
0.0	0.0	0.0	0.0
0.5	6.802	3.000	9.974
1.0	7.001	6.001	10.52
1.5	7.207	7.500	10.78
2.5	7.508	7.740	11.15
4.5	7.905	7.937	11.80
7.0	8.138	8.060	12.20
9.0	8.179	8.080	12.24
9.9	8.200	8.076	12.19
11	8.130	8.045	12.10
14	8.000	7.918	11.83
16	7.868	7.706	11.42
20	7.437	7.207	10.21
22	7.084	6.780	9.257
24	6.499	6.250	8.024
25	5.951	5.900	−0.85
25.5	5.423	5.650	0.70
26	0.020	0.010	−0.15
26.5	2.2	0.0	0.0

- The Error in IEEE model and Pinceti model increases when the peak of the impulse current decreases.
- Models of (IEEE WG 1992; Pinceti & Giannettoni 1999; Fernandez & Diaz 2001) simulate the dynamic behaviour of surge arresters properly.
- The proposed algorithm (MAPSO) can be applied to all models.
- The proposed algorithm is a powerful tool to estimate parameters of MO surge arrester models.

9. Conclusion

In this paper a new method, based on APSO and ACO algorithms has been proposed by linking the MATLAB and EMTP. Using this method it is possible to determine the parameters of different surge arrester models. In this method, the proposed optimization algorithm known as MAPSO can be implemented easily and the parameters of surge arrester models are estimated based on the MO surge arrester residual voltage measurements. The proposed method has been applied to different models of surge arresters. It is shown that the estimated parameters of all surge arrester models lead to simulation results which are in agreement with the residual voltage measurements. It should be mentioned that the previous studies were limited to special models but the proposed method is general and can be applied to all existing models.

References

Arumugam M S, Rao M V C 2008 On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems. *Applied Soft Comp.* 8(1): 324–336

Canadian/American EMTP User Group 1987 *Alternative Transients Program Rule Book*, Leuven, Belgium

Eberhart R, Kennedy J 1995 A new optimizer using particle swarm theory. *Proc. of the Sixth International Symposium on Micro Machine and Human Science*, IEEE Service Center, Piscataway, NJ, pp. 39–43

Eberhart R, Shi Y 2001 Particle swarm optimization: development, application and resources. *IEEE Cong. on Evol. Comp.* 1: 81–86

Fernandez F, Diaz R 2001 Metal oxide surge arrester model for fast transient simulations. *The Int. Conf. on Power System Transients* IPAT'01, Rio De Janeiro, Brazil

Gaing Z L 2003 Particle swarm optimization to solving the economic dispatch considering the generator constraints. *IEEE Trans. Power Syst.* 18(3): 1187–1195

Hinrichsen V 2001 *Metal Oxide Surge Arrester Fundamentals*. *Handbook on High Voltage Metal Oxide Surge Arrester*, SIEMENS AG, Berlin, July

IEEE Working Group 3.4.11 1992 Modeling of metal oxide surge arresters. *IEEE Trans. Power Deliv.* 7(1): 302–309

Kennedy J 1999 Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. *Proc. of Cong. Evol. Comput. IEEE Press* 3: 1931–1938

Kennedy J, Eberhart R 1995 Particle swarm optimization. *IEEE Int. Conf. on Neural Networks*. Perth, Australia, pp. 1942–1948

Kim I, Funabashi T, Sasaki H, Hagiwara T, Kobayashi M 1996 Study of ZnO arrester model for steep front wave. *IEEE Trans. Power Deliv.* 11(2): 834–841

Li H J, Birlasekaran S, Choi S S 2002 A parameter identification technique for metal-oxide surge arrester models. *IEEE Trans. Power Deliv.* 17(3): 736–741

Mahseredjian J, Benmouyal G, Lombard X 1998 A Link between EMTP and MATLAB for user-defined modeling. *IEEE Trans. Power Deliv.* 13(2): 667–674

Martinez J A, Durbak D W 2005 Parameter determination for modeling systems transients-part V: Surge arrester. *IEEE Trans. Power Deliv.* 20(3): 2073–2078

Niknam T, Doagou Mojarrad H, Nayeripour M 2010 A new fuzzy adaptive particle swarm optimization for non-smooth economic dispatch. *Energy* 35(4): 1764–1778

Niknam T, Ranjbar A M, Shirani A R 2005 A new approach for distribution state estimation based on ant colony algorithm with regard to distributed generation. *Journal of Intelligent Fuzzy System* 16(2): 119–131

Park J B 2005 A particle swarm optimization for economic dispatch with nonsmooth cost function. *IEEE Trans. Power Syst.* 20(1): 34–42

Pinceti P, Giannettoni M 1999 A simplified model for zinc oxide surge arresters. *IEEE Trans. Power Deliv.* 14(2): 393–398

Popov M, Van der Sluis L, Paap G C 2002 Application of a new surge arrester model in protection studies concerning switching surges. *IEEE Power Eng. Rev.* 22(9): 19–19

Schmidt W, Meppelink J, Richter B, Fester K 1989 Behavior of metal oxide surge arrester blocks to fast transient. *IEEE Trans. Power Deliv.* 4(1): 292–300

Zinik B, Babuder M, Muhr M, Zitnik M, Thottappillil R 2005 Numerical modeling of metal oxide varistor. *Proc. of the XIVth Int. Symposium on High Voltage Engineering*, Tsinghua University, Beijing, China, pp. 25–29