DOI 10.7603/s40690-015-0001-0

JOURNAL OF AERONAUTICS AND SPACE TECHNOLOGIES
JANUARY 2015 VOLUME 8 NUMBER 1 (1-10)

IMPROVED PARTICLE SWARM OPTIMIZATION METHOD DIRECTED BY INDIRECT SURROGATE MODELING

Y. Volkan PEHLİVANOĞLU[*]
Technical Programs Chair
NCO College Gaziemir/İzmir
vpehlivanoglu@tekok.edu.tr

Serdar AY
Aerospace Engineering Dept.
TurAFA Yesilyurt/Istanbul
s.ay@hho.edu.tr

Faruk GÜL
Aerospace Engineering Dept.
TurAFA Yesilyurt/Istanbul
fgul@hho.edu.tr

Received: *06th February 2014,* ***Accepted:*** *25th April 2014*

ABSTRACT
An improved particle swarm optimization algorithm is proposed and tested for two different test cases: surface fitting of a wing shape and an inverse design of an airfoil in subsonic flow. The new algorithm emphasizes the use of an indirect design prediction based on a local surrogate modeling as a part of update equations in particle swarm optimization algorithm structure. For all the demonstration problems considered herein, remarkable reductions in the computational times have been accomplished.

Keywords: *PSO, Surrogate Modeling, Inverse Design.*

DOLAYLI VEKİL MODEL İLE YÖNLENDİRİLEN GELİŞTİRİLMİŞ PARÇACIK SÜRÜ ENİYİLEME ALGORITMASI

ÖZET
Bu çalışma kapsamında yeni bir Parçacık Sürü Optimizasyon (PSO) algoritması geliştirilmiş ve teklif edilen algoritma iki farklı test probleminde denenmiştir. Söz konusu test problemleri kanat yüzeyi modelleme ve sesaltı akış şartlarında kanat profilinin tersten tasarımıdır. Teklif edilen yeni algoritma dolaylı vekil model kullanımına dayalı olarak öngörülen aday çözümün PSO algoritmalarındaki temel güncelleme denklemlerine ilave edilmesini öngörmektedir. Çalışma dahilinde dikkate alınan problemlerin tümünde teklif edilen algoritmanın kayda değer hesaplama süresi azaltımları sağladığı görülmüştür.

Anahtar Kelimeler: *PSO, Vekil Model, Tersten Tasarım.*

1. INTRODUCTION

An inverse design problem is widely known in natural sciences. Most of the formulations of inverse problems may proceed to the setting of an optimization problem. In general, an inverse design problem can be expressed as follows:

$$find\ \{\boldsymbol{x} \in R^d\} \quad (1)$$

$$\min f(\boldsymbol{x}, \boldsymbol{y}) \quad (2)$$

Subject to

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \leq 0 \quad (3)$$

$$\boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U \quad (4)$$

where $\boldsymbol{x}$ is an input that is the design parameter vector whose values lie in the range given by upper and lower borders in (4). The objective function $f(\boldsymbol{x},\boldsymbol{y})$ is used to bring the computed response from the model as close as possible to the target output, $\boldsymbol{y}$.
In some problems, it may be necessary to satisfy certain inequality constraints given by $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})$. The objective function is usually a least-squares function given by

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} (y_i^c - y_i^t)^2 \quad (5)$$

where y_i^t is i^{th} value of the target response and y_i^c is i^{th} value of the computed response obtained from the simulation model.

[*] Corresponding Author

Published online: 06 April 2015

In most engineering problems, computational methods are gradually replacing empirical methods; and design engineers are spending more time in applying computational tools. Non-gradient based optimization techniques. These algorithms are population based, and they include a lot of design candidates waiting for the objective function computations in each generation. The major weakness of population based algorithms lies in their poor computational efficiency, because the evaluation of objective function is sometimes very expensive [1]. Despite the considerably improved computer power over the past few decades, computational simulation can still be prohibitive for a large number of executions in practical engineering design. Therefore, improving the efficiency of evolutionary search algorithms has become a key factor in their successful applications to real-world problems.

Two categories of techniques have been proposed to tackle the efficiency issue of evolutionary search methods; the first type is focused on devising more efficient variants of the canonical algorithms, the second type involves using a surrogate model which is a kind of approximation in lieu of the exact and often expensive function evaluations [2].

In literature, there are a lot of surrogate model-based optimization algorithms. The details of these algorithms can be found in Pehlivanoglu and Yagiz [3]. The key idea in these methods is to parameterize the space of possible solutions via a simple, computationally inexpensive model, and to use this model to generate inputs in terms of predicted objective function values for the optimization algorithm. Therefore, the whole optimization process is managed by surrogate model outputs. Such a model is often referred to as the response surface of the system to be optimized, leading to the definition of a so-called surrogate-model based optimization methodology [4]. Major issues in surrogate model-based design optimization are the approximation efficiency and accuracy. In case of the problem which has a high number of design variables, the construction of surrogate model may cause extremely high computational cost, which means computationally inefficient approximation. On the other hand, it is possible to miss the global optimum, because the approximation model includes uncertainty at the predicted point, and this uncertainty may mislead the optimization process in a wrong way.

The present paper introduces the application of an indirect surrogate modeling within velocity update formula to speed up the PSO algorithm and overcome problems such as inaccuracy and premature convergence during the optimization. To demonstrate the efficiency of the proposed PSO algorithm, it is applied to two different test cases, and the results were compared with four different PSOs, including constriction factor PSO (c-PSO), inertia weight PSO (w-PSO), vibrational PSO (v-PSO), and comprehensive learning PSO (cl-PSO). The test bed selected herein includes surface fitting of a wing shape and an inverse design of an airfoil in subsonic flow.

2. SURROGATE MODELING

The stages of surrogate-based modeling approach include a sampling plan in design space, numerical simulations at these design points, construction of a surrogate model based on simulations, and model validation [5]. There are different alternatives to construct the surrogate model. After surrogate-based modeling is completed, the optimization problem is described as follows

$$\begin{aligned} &\text{Minimize } \hat{f}(\boldsymbol{x}) \\ &\text{Subject to} \\ &\qquad \hat{g}_i(\boldsymbol{x}) \le 0, i = 1,2,\dots,I \\ &\qquad \boldsymbol{x}^L \le \boldsymbol{x} \le \boldsymbol{x}^U \end{aligned} \tag{6}$$

This is where the functions are the approximation models. The main purpose of constructing approximate models in this framework is to predict the value of objective and constraints.

Many different surrogate-model based optimization algorithms were applied in engineering problems. Examples are commonly from GA applications [4, 6-11]. There are also a few applications from PSO studies. Praveen and Duvigneau [12] have constructed radial basis function approximations and used them in conjunction with particle swarm optimization in an inexact evaluation procedure for the objective function values of candidate aerodynamic designs. Khurana et al. [13] developed an artificial neural network and validated with a relationship between the mapped PARSEC (a kind of geometry parameterization method) solution space and the aerodynamic coefficients of lift and drag. The validated surrogate model was used for airfoil shape optimization by replacing the flow solver from the direct numeric optimization loop. Multi-fidelity simulation and surrogate models were employed by Singh and Grandhi [14] in mixed-variable optimization problem. To get benefits from surrogate models in multi-objective optimization problems, Carrese et al. [15-16] presented the Kriging-assisted user-preference multi-objective particle swarm heuristic method.

In addition to the classical surrogate modeling approach, another methodology was also used in some evolutionary computation studies. The main purpose of constructing approximate models in this framework is to predict the positions of new design points, rather than to make inexact computational evaluations as in the surrogate model. An example given by Ong et al. [17] presented an Evolutionary Algorithm (EA) that leverages surrogate models. The essential backbone of the framework is an EA coupled with a feasible Sequential Quadratic Programming (SQP) solver in the spirit of Lamarckian learning. Pehlivanoglu and Baysal [18] and Pehlivanoglu and Yagiz [3] have also suggested a novel usage of regression model and neural networks. They used a new technique to predict better solution candidates using local response surface approximation based on neural networks inside the population for the direct shape optimization of an airfoil in transonic flow conditions. Another novel example is given by Hacioglu [19]. A new hybridization technique has been proposed to employ NNs and EAs together to solve the inverse design of an airfoil problem. The essential backbone of the framework is GA coupled with NN. Similar to Hacioglu [19], Pehlivanoglu [20] has been proposed to employ NNs and PSO together to solve the inverse design of an airfoil and a wing problem. The essential backbone of the framework is PSO coupled with NN including an indirect usage of surrogate model and training approach.

In all studies mentioned in the previous paragraph, the predicted particle(s) are put into the next swarm by replacing with some particles in the current swarm. On the other hand, instead of replacing fashion, another approach can also be employed. The present study takes an additional step and introduces the implementation of adding an indirect design prediction to the velocity update formula in PSO architecture for inverse design problems.

3. PRESENT FRAMEWORK

As in other evolutionary algorithms, PSO method is a population-based stochastic optimization algorithm that originates from "nature". PSO algorithms search the optimum within a population called "swarm." It benefits from two types of learning, such as "cognitive learning" based on an individual's own history and "social learning" based on swarm's own history accumulated by sharing information among all particles in the swarm. Since its development in 1995 by Eberhart and Kennedy [21], it has attracted significant attention. Let s be the swarm size, d be the particle dimension space, and each particle of the swarm has a current position vector $\boldsymbol{x}_i$, current velocity vector $\boldsymbol{v}_i$, individual best position vector $\boldsymbol{p}_i$ found by particle itself. The swarm has also the global best position vector $\boldsymbol{p}_g$ found by any particle during all prior iterations in the search space. Assuming that the function f is to be minimized and describing the following notations in t^{th} iteration, then the definitions are as follows:

$$\begin{gathered}\boldsymbol{x}_i(t) = \left(x_{i,1}(t), x_{i,2}(t), \dots, x_{i,d}(t)\right) \\ x_{i,j}(t) \in R^d, i = 1,2, \dots, s\end{gathered} \tag{7}$$

where each dimension of a particle is updated using the following equations:

$$\begin{aligned}v_{i,j}(t) = v_{i,j}(t-1) &\\ &+ c_1 r_1 \left(p_{i,j}(t-1)\right. \\ &\left. - x_{i,j}(t-1)\right) \\ &+ c_2 r_2 \left(p_{g,j}(t-1)\right. \\ &\left. - x_{i,j}(t-1)\right)\end{aligned} \tag{8}$$

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t) \tag{9}$$

In (8), c_1 and c_2 denote constant coefficients, r_1 and r_2 are elements from random sequences in the range of (0, 1). The personal best position vector of each particle is computed using the following expression:

$$\boldsymbol{p}_i(t) = \begin{cases}\boldsymbol{p}_i(t-1) & if\ f(\boldsymbol{x}_i(t)) \geq f(\boldsymbol{p}_i(t-1)) \\ x(t) & if\ f(\boldsymbol{x}_i(t)) < f(\boldsymbol{p}_i(t-1))\end{cases} \tag{10}$$

Then, the global best position vector is found by

$$\boldsymbol{p}_g(t) = arg \min_{\boldsymbol{P}_i(t)} f(\boldsymbol{p}(t)) \tag{11}$$

3.1. Comparative PSO Algorithms

Four well known PSO algorithms are selected as comparative optimization algorithms. These are c-PSO, w-PSO, v-PSO, and cl-PSO. In c-PSO algorithm the particle swarm with a constriction factor is introduced by Clerc and Kennedy [22], which investigated the use of a parameter called the constriction factor. With the constriction factor K, the particle velocity and position dimensions are updated via:

$$\begin{gathered}v_{i,j}(t) = K((v_{i,j}(t-1) \\ +c_1 r_1 \left(p_{I,j}(t-1) - x_{i,j}(t-1)\right) \\ +c_2 r_2 (p_{g,j}(t-1) - x_{i,j}(t-1))) \\ K = \frac{2}{\left|2 - \psi - \sqrt{\psi^2 - 4\psi}\right|}, \\ \psi = c_1 + c_2, \psi > 4 \\ x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t)\end{gathered} \tag{12}$$

Typically, values of *2.05* are used for c_1 and c_2, making ψ is equal to *4.1* and K is equal to *0.729*. In the second algorithm called w-PSO, Shi and Eberhart [23] introduced the idea of a time-varying inertia

weight. The velocity is updated in accordance with the following expressions:

$$\begin{aligned} v_{i,j}(t) &= w(t)v_{i,j}(t-1) \\ &+c_1 r_1\left(p_{I,j}(t-1) - x_{i,j}(t-1)\right) \\ &+c_2 r_2(p_{g,j}(t-1) - x_{i,j}(t-1)) \\ x_{i,j}(t) &= x_{i,j}(t-1) + v_{i,j}(t) \end{aligned} \quad (13)$$

The inertia weight,w is decreased linearly starting from initial point, w_{ini}, and ending to last point, w_{end}, related to maximum iteration number, T. Normally, the starting value of the inertia weight is set to *0.9* and the final to *0.4*. However, we tuned them to [0.6, 0.2] range for better performance. In v-PSO Pehlivanoglu [24] proposed periodic mutation activation based on the wavelet analysis of diversity in the swarm. Right after updating applications, in every fr^{-1} period of the generations applying the mutation operator to all particle dimensions of the whole swarm, particles in the swarm spread throughout the design space. This operator is called global mutation operator and given by

$$\begin{aligned} x_{i,j}(t) &= x_{i,j}(t)[1 + A(0.5 - rand)\delta], \\ & i = 1,2,\dots,s, j = 1,2,\dots,d \\ \delta &= \begin{Bmatrix} 1 & if\ t = nfr, n = 1,2,\dots \\ 0 & if\ t \neq nfr \end{Bmatrix} \end{aligned} \quad (14)$$

where A is an amplitude factor defined by the user; $rand$ is a random number specified by random number generator in accordance with $N[0, 1]$. In the applications, Gaussian probability density function is used. The velocity and the positions are updated via (13) except the generations corresponding to the mutation period. The comprehensive learning particle swarm optimizer (cl-PSO) is proposed by Liang et al. [25], which uses all other particles' historical best information to update a particle's velocity. A particle's velocity and its position are updated by the following equations:

$$\begin{aligned} v_{i,j}(t) &= w(t)v_{i,j}(t-1) \\ &+ cr\left(p_{I(f_i(j)),j}(t-1)\right. \\ &\left. - x_{i,j}(t-1)\right) \\ x_{i,j}(t) &= x_{i,j}(t-1) + v_{i,j}(t) \end{aligned} \quad (15)$$

where f_i defines which particles' best position vector the particle i should follow, c is the constant value, and r is a random number drawn from a random sequence in the range of (0,1). The decision about f_i depends on the learning probability value, Pc_i which is defined as the following:

$$Pc_i = 0.05 + 0.45\frac{(\exp\left(\frac{10(i-1)}{s-1}\right)-1)}{(\exp(10)-1)} \quad (16)$$

For each dimension of particle i, a random number is generated and compared with the value of Pc_i. If a random number is larger than the learning probability value, the related dimension will learn from its own best position vector; otherwise, it will learn from another particle's $\boldsymbol{p}_i$. A tournament selection procedure is taken into consideration to determine the particle i. The inertia weight w is decreased linearly starting from initial point w_{ini}, and ending to last point w_{end}, related to maximum iteration number T.

3.2. Proposed PSO Algorithm

The proposed algorithm is named s-PSO. The backbone of the new algorithm is PSO coupled with single or multiple surrogate models. The basic steps of the proposed algorithm are outlined here:

┌ Initialization ┐
S^I | Determination of initial swarm using random number operator
└ 1^{st} Swarm ┘
┌ Design cycle ┐
f |Computation by high-fidelity solver
ε |Convergence check
RSM |Response Surface Model fitting
$\boldsymbol{p}_i$ |Updating of particle best position
$\boldsymbol{p}_g$ |Updating of global best position
$\boldsymbol{p}_p$ |New particle predicted by RSM
S^U |Updating of swarm by updating equations including $\boldsymbol{p}_p$
└ t^{th} Swarm ┘

At first, we generate the initial swarm of designs including the particles, S^I, computed by using random number operator. After initiation, all particles in the swarm are evaluated by using high-fidelity objective function solver. By the way, the convergence check is done whether the determined criteria such as the tolerance, $f(\boldsymbol{x}, t)$-$f(\boldsymbol{x}, t\text{-}1) < \varepsilon$, is satisfied or not. After that, all of the design points and the associated exact values of the objective function are archived in the database.

In the next step, the input-output couples are used to construct Response Surface Model (RSM). For a local response surface, Radial Basis Neural Network (RBNN) approximates the response values as a weighted sum of radial basis functions. Matlab routine of *newrb* was used to construct RBNN [26]. Then, particle best position vectors and the global best position vector are determined.

The present indirect prediction strategy is applied right after the updating of $\boldsymbol{p}_i$ and $\boldsymbol{p}_g$ phases. In classical surrogate modeling approach, $\boldsymbol{x}_i$ particle position vectors in each swarm are used as input values and f_i or $\boldsymbol{y}_i^c$ values computed by high-fidelity model are used as output values. These couples are sample points and used to train RBNN. During the

optimization process, some particle's objective function ($\hat{f}_i$) or response values ($\hat{y}_i^c$) are predicted by trained neural net(s) to shorten the computation time. On the contrary, it is possible to use the computed response values ($\boldsymbol{y}_i^c$) as input values and particle position vectors ($\boldsymbol{x}_i$) as output values in neural network training process. Furthermore, we may predict a new design vector by using the target value(s) in inverse design problem as input for the trained neural network. Following the prediction process, the updating equations are applied for the new particles. At this stage, a new particle predicted by indirect surrogate model can be added to velocity update equation as follows:

$$\begin{aligned} v_{i,j}(t) &= w(t)v_{i,j}(t-1) \\ &+c_1r_1(p_{I,j}(t-1)-x_{i,j}(t-1)) \\ &+c_2r_2(p_{g,j}(t-1)-x_{i,j}(t-1)) \\ &+c_3r_3(p_{p,j}(t-1)-x_{i,j}(t-1)) \end{aligned} \tag{17}$$

or

$$\begin{aligned} v_{i,j}(t) &= K((v_{i,j}(t-1) \\ &+c_1r_1(p_{I,j}(t-1)-x_{i,j}(t-1)) \\ &+c_2r_2(p_{g,j}(t-1)-x_{i,j}(t-1)) \\ &+c_3r_3(p_{p,j}(t-1)-x_{i,j}(t-1))) \end{aligned} \tag{18}$$

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t)$$

where c_3 denote another constant coefficient, r_3 is elements from random sequences in the range of (0, 1), and $\boldsymbol{p}_p$ is the predicted particle. This application provides a local but controlled diversity within the population. At the next design cycle, all particles in the new swarm are evaluated by using high-fidelity objective function solver. All the design points and the associated exact values of the objective function are added to the database. This cycle is repeated until the convergence criterion is satisfied.

4. NUMERICAL STUDIES

4.1. Surface Fitting of a Wing

One of the important issues in computer graphics is a surface reconstruction and it consists of obtaining a smooth surface that approximates a set of points given in three-dimensional (3D) space. It has a significant role in real engineering problems such as the design of ground, naval, or air vehicle surfaces. A typical application is a reverse engineering where free-form parametric surfaces are constructed from a set of points obtained from surface scanning process. This issue is not a trivial problem and several optimization algorithms including PSO were used to solve the surface reconstruction issue [27-28]. A set of surface points belong to wing in 3D can be modeled by using Bezier surface functions. Example wing surface is depicted in Fig. 1. This wing surface has different airfoil sections in each station through the x_2 axis. The root airfoil is selected as *RAE2822* airfoil and the tip airfoil is chosen as *NACA0012* symmetric airfoil. The wing is a rectangular wing and there is no any swept or dihedral angle. The length of chord is fixed to 1 unit. A general form of Bezier surface [29] is given below:

$$\begin{aligned} x_1(u,v) &= \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\,x_{1,i,j}, \\ x_2(u,v) &= \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\,x_{2,i,j} \\ x_3(u,v) &= \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\,x_{3,i,j} \\ B_i^n(u) &= \frac{n!}{i!\,(n-i)!}u^i(1-u)^{n-i}, \\ B_j^m(v) &= \frac{m!}{j!\,(m-j)!}v^j(1-v)^{m-j}, \\ 0 \le u \le 1, &\; 0 \le v \le 1 \end{aligned} \tag{19}$$

where $\boldsymbol{x_1(u,v)}$, $\boldsymbol{x_2(u,v)}$, and $\boldsymbol{x_3(u,v)}$ are surface coordinates, u and v are parametric coordinates, n and m are the degrees of Bezier surface and they are fixed to 1 by 12, respectively. $x_{1,i,j}$, $x_{2,i,j}$, and $x_{3,i,j}$ are the control points of Bezier surface and only $x_{3,i,j}$ the third coordinates of control points are selected as the design parameters. The number of design parameters is fixed to 44. A half of them are used to parameterize the upper surface of the wing and the remaining 22 parameters are used to parameterize the lower surface of the wing. The control points are placed only on the root and tip sections of the wing. Phenotype of an example initial swarm and a particle from an example initial swarm are depicted in Fig. 2 and 3, respectively.

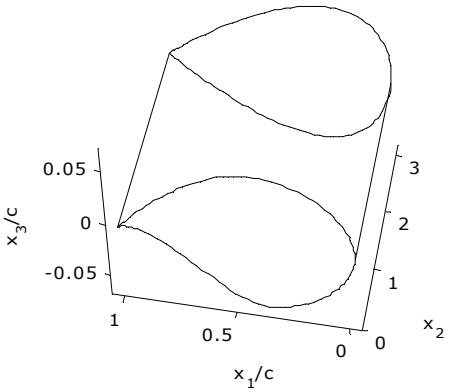

Figure 1. Target wing surface in 3D environment.

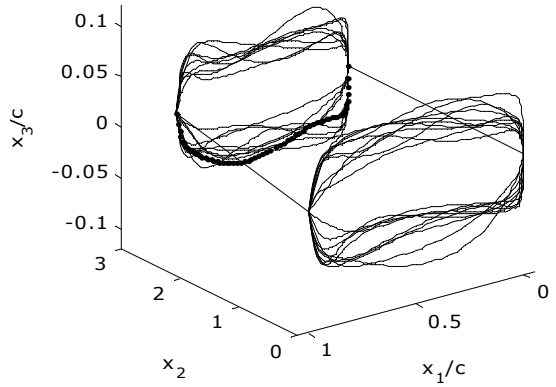

Figure 2. Phenotype of an initial swarm.

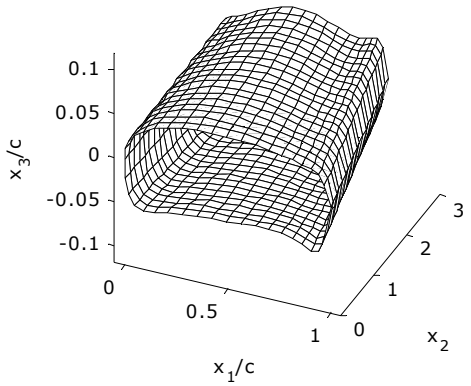

Figure 3. An example surface of a particle from an initial swarm.

The objective function value is based on the difference between the target surface points and the particle surface points. However, to facilitate the computation of the objective function only the surface points on the root and tip sections are considered. The objective function *f* is given below:

$$f = \sum_{j=1}^{k} (x_{3_j}^{c} - x_{3_j}^{t})^2 \tag{20}$$

where *k* is the number of target points and fixed to 256. We need to point out that this number contains both upper and lower surface pints of the root and tip sections.

4.1.1. The Effect of *c* Values

To observe the effect of c values in (18), the swarm particles are optimized in accordance with the given objective function by using s-PSO algorithm. The swarm size is selected as 15; the maximum generation number is selected as 1000. *N* is equal to 20 which means the last 2 generations are used to train RSM in s-PSO algorithm. In a comparative study, all algorithms are run 30 times and the averaged global best particle values versus generations are taken into consideration for a fair comparison. In Fig. 4, the convergence histories of the averaged global best particle's objective function values for different c value combinations are depicted.

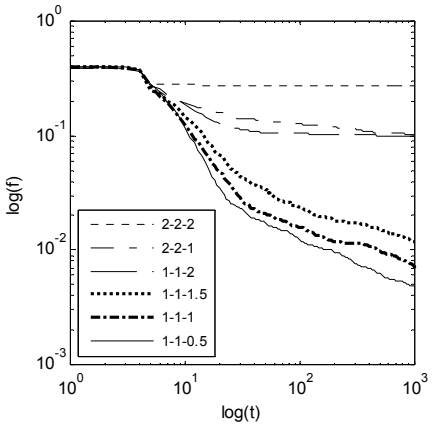

Figure 4. Convergence histories for different c value combinations.

The simulations for different c value combinations show that the most efficient and accurate result is provided by the following c value combination: $c_1 = 1,\ c_2 = 1$, and $c_3 = 0.5$.

4.1.2. Optimization Results

To compare the proposed algorithm with the comparative algorithms, the swarm particles are optimized in accordance with given objective function. For all simulations, the swarm size is selected as 10; the maximum generation number is selected as 500. Peculiar settings are the following: c_1 and c_2 are equal to 2.05 for c-PSO; c_1 and c_2 are equal to 2.05; w_{ini} and w_{end} are equal to 0.6, 0.2, respectively for w-PSO; c is equal to 1.49445; w_{ini} and w_{end} are equal to 0.9, 0.4, respectively for cl-PSO; c_1 and c_2 are equal to 2.05, w_{ini} and w_{end} are equal to 0.6, 0.2, respectively; *fr* is equal to 50, *A* is equal to 0.5 for v-PSO. In a comparative study, all algorithms are run 40 times and the averaged global best particle values versus generations are taken into consideration for a fair comparison. The optimization results including convergence histories and an example surface optimized by s-PSO are depicted in Fig. 5 and Fig. 6, respectively.

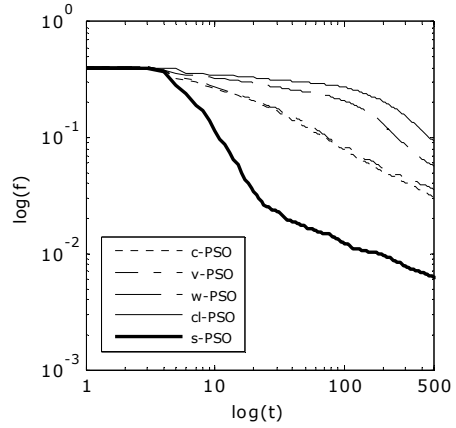

Figure 5. Convergence histories for Bezier surface fitting problem.

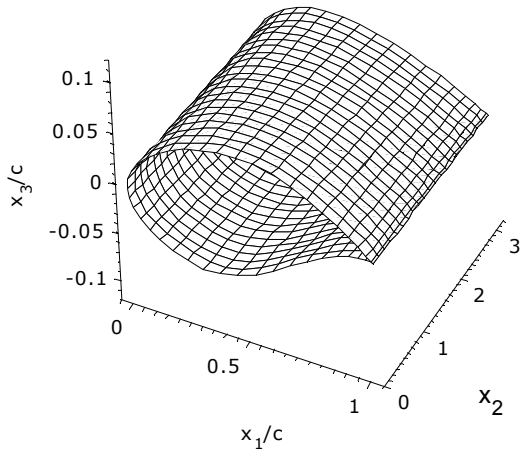

Figure 6. Optimized wing surface model.

Among the classical PSO algorithms the best performance belongs to c-PSO algorithm. It reaches the value of 0.0312 at *500th* generation. On the other hand, s-PSO does outperform the regular algorithms. It reaches the value of 0.0312 at *21st* generation and 0.0063 at *500th* generation. This result means an approximately 96% decrease in the required generations as compared with c-PSO.

4.2. Inverse Design Based on C_p Distribution

In inverse design problem from aerodynamics, the pressure distribution around the shape is known or predicted and the geometry of the shape is investigated. This approach recognizes that the designer usually has an idea of the kind of pressure distribution that will lead to the desired performance. Thus, it is useful to consider the inverse problem of calculating the shape that will lead to a given pressure distribution [30]. Within the second inverse design test case, *RAE2822* airfoil is selected as the test airfoil and the pressure coefficient (C_p) distribution of this airfoil under subsonic flow conditions is chosen as the target C_p distribution. An airfoil shape can be represented using the Bezier curves with a set of control points as follows:

$$x_1(t) = \sum_{i=0}^{q} B_i^q t^i (1-t)^{q-i} x_{1,i}$$
$$x_2(t) = \sum_{i=0}^{q} B_i^q t^i (1-t)^{q-i} x_{2,i} \quad (21)$$
$$B_i^m = q!/i!\,(q-1)!$$

where t is the step size parameter of the curve whose values vary uniformly between [0,1], q is the number of control points, $(x_{1,i},\ x_{2,i})$ are the coordinates of the i^{th} control point which define the airfoil coordinates $(x_1(t),x_2(t))$.

The initial swarm is generated by using random number operator. The objective function value is based on the difference between the target C_p points and the computed particle C_p points. The angle of attack is assumed to be zero during the optimization process. The fitness function *f is* defined as,

$$f = \sum_{j=1}^{k} (C_{p_j}^{\ c} - C_{p_j}^{\ t})^2 \quad (22)$$

where k is the number of panels and it is fixed to 128. The pressure coefficient is computed by using panel solver [31]. The reference C_p distribution and the initial swarm are depicted in Fig. 7 and 8, respectively.

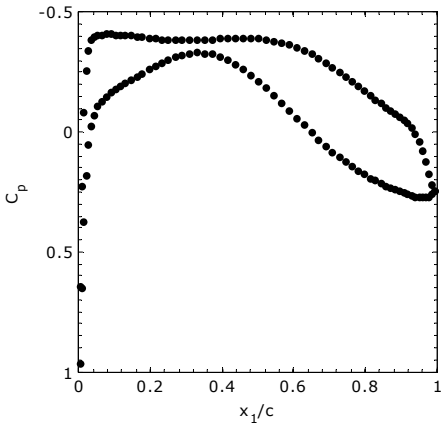

Figure 7. *RAE2822* airfoil C_p distribution in subsonic flow.

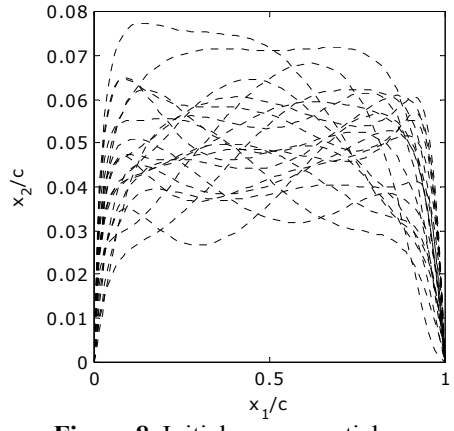

Figure 8. Initial swarm particles.

4.2.1. The Effect of c Values

To observe the effect of c values in (18), the swarm particles are optimized in accordance with the given objective function by using s-PSO algorithm. The swarm size is selected as 15; the maximum generation number is selected as 1000. N is equal to 20 which means the last 2 generations are used to train RSM in s-PSO algorithm. In a comparative study, all algorithms are run 30 times and the averaged global best particle values versus generations are taken into consideration for a fair comparison. In Fig. 9, the convergence histories of the averaged global best particle's objective function values for different c value combinations are depicted.

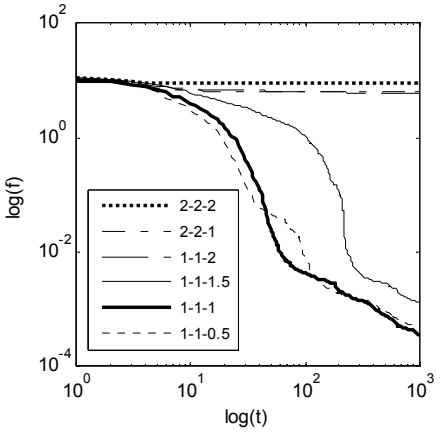

Figure 9. Convergence histories for different c value combinations.

The simulations for different c value combinations show that the most efficient and accurate result is provided by the following c value settings: $c_1 = 1$, $c_2 = 1$, and $c_3 = 1$.

4.2.2. Optimization Results

The swarm particles are optimized in accordance with given objective function by using five PSO algorithms including c-PSO, w-PSO, cl-PSO, v-PSO, and s-PSO. The swarm size is selected as 15; the maximum generation number is selected as 1000. The problem dimension is fixed to 22 as the control points of Bezier curves. Peculiar settings are the following: c_1 and c_2 are equal to 2.05 for c-PSO; c_1 and c_2 are equal to 2.05; w_{ini} and w_{end} are equal to 0.6, 0.2, respectively for w-PSO; c is equal to 1.49445; w_{ini} and w_{end} are equal to 0.9, 0.4, respectively for cl-PSO; c_1 and c_2 are equal to 2.05, w_{ini} and w_{end} are equal to 0.6, 0.2, respectively; fr is equal to 20, A is equal to 0.5 for v-PSO; N is equal to 50 which means the last five generations for s-PSO. In a comparative study, all algorithms are run 30 times and the averaged global best particle values versus generations are taken into consideration for a fair comparison.

The optimization results including convergence histories and an example airfoil optimized by s-PSO are depicted in Fig. 10 and Fig. 11, respectively. Among the classical PSO algorithms including c-PSO, w-PSO, cl-PSO, and v-PSO the best performance belongs to c-PSO algorithm. It reaches the value of 0.0027 at 500^{th} generation. On the other hand, s-PSO does again outperform the regular algorithms. It reaches the value of 0.0027 at 168^{th} generation and the value of 0.00033 at 500^{th} generation. This result means an approximately 83% decrease in the required generations as compared with c-PSO.

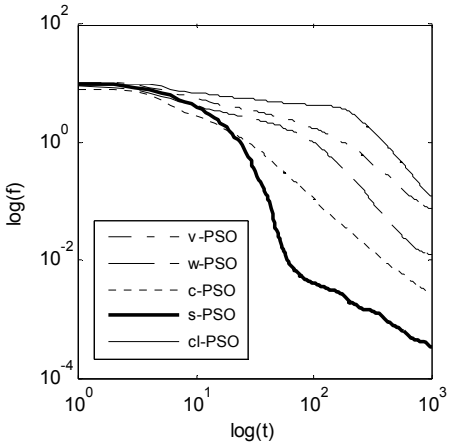

Figure 10. Convergence histories for the inverse design of an airfoil problem.

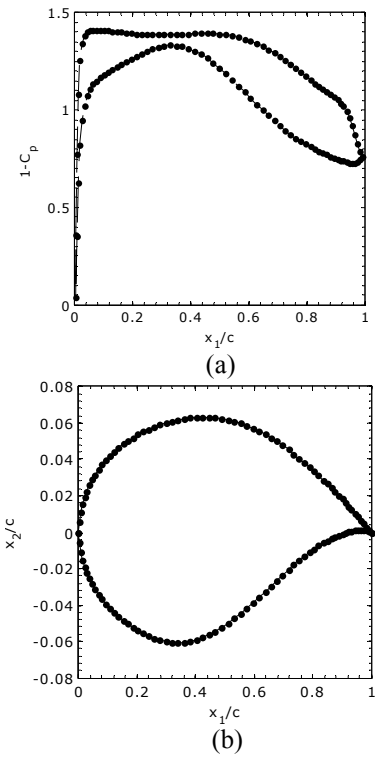

Figure 11. (a) Target C_p points (• markers) and optimized particle`s $C_p$ points (solid line), and (b) Target airfoil points (• markers) and optimized particle`s curve points (solid line).

5. CONCLUSIONS

The present paper introduced a new use of surrogate modeling in PSO algorithm structure to speed up the optimization algorithm and overcome problems such as low efficiency and premature convergence. Then, depending on the nature of the problem at hand, the present approach employed a local response surface approximation constructed by using neural networks to provide a local but controlled diversity within the population. The average best-individual-fitness values of the algorithms were recorded for a fair comparison among them. To demonstrate their merits, a new approach and four comparative algorithms such as c-PSO, w-PSO, cl-PSO, and v-PSO were applied to two different test scenarios. The principal role of the use of a surrogate model was to direct particles to the optimal solution. Due to still being a PSO based technique, this method was as robust as the plain PSO algorithms. Based on the results obtained, it was concluded that the proposed PSO algorithm approach is an efficient and fast algorithm in inverse design problems.

6. REFERENCES

[1] Peigin, S. and Epstein, B., "Robust optimization of 2D airfoils driven by full Navier–Stokes computations", *Computers & Fluids* vol. 33, no. 9, pp. 1175–1200, 2004.

[2] Song, W. and Keane, A.J., "A new hybrid updating scheme for an evolutionary search strategy using genetic algorithms and Kriging", *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference,* AIAA paper 2005-1901, 2005.

[3] Pehlivanoglu, Y.V. and Yagiz, B., "Aerodynamic design prediction using surrogate-based modeling in genetic algorithm architecture," *Aerospace Science and Technology*, 23 (2012) 479–491, 2011.

[4] Jouhaud, J.C., Sagaut, P., Montagnac, M., and Laurenceau, J., "A surrogate-model based multidisciplinary shape optimization method with application to a 2D subsonic airfoil", *Computers & Fluids,* vol. 36, no. 3, pp. 520–529, 2007.

[5] N.V. Qoeipo, R.T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P.K. Tucker, "Surrogate-based analysis and optimization", *Progress in Aerospace Sciences,* vol. 41, no. 1, pp. 1–28, 2005.

[6] A. Vavalle and N. Qin, "Iterative response surface based optimization scheme for transonic airfoil design", *Journal of Aircraft,* vol. 44, no. 2, pp. 365-376, 2007.

[7] Keane, A.J., "Statistical improvement criteria for use in multi objective design optimization", *AIAA Journal,* vol. 44, no. 4, pp. 879-891, 2006.

[8] Glaz, B., Goel, T., Liu, L., Friedmann, P.P., and Haftka, R.T., "Multiple-surrogate approach to helicopter rotor blade vibration reduction", *AIAA Journal,* vol. 47, no. 1, pp. 271-282, 2009.

[9] Papila, N., Shyy, W., Griffin, L., and Dorney, D.J., "Shape optimization of supersonic turbines using global approximation methods", *Journal of Propulsion and Power,* vol. 18, no. 3, pp. 509-518, 2002.

[10] Xiong, C.Y. and Chen, W., "Multi-response and multistage meta-modeling approach for design optimization", *AIAA Journal,* vol. 47, no. 1, pp. 206-218, 2009.

[11] Duchaine, F., Morel, T., and Gicquel, L.Y.M., "Computational fluid dynamics based Kriging optimization tool for aeronautical combustion chambers", *AIAA Journal,* vol. 47, no. 3, pp. 631-645, 2009.

[12] Praveen, C. and Duvigneau, R., "Low cost PSO using metamodels and inexact pre-evaluation: application to aerodynamic shape design", *Comput. Methods Appl. Mech. Engrg.* 198 1087–1096, 2009.

[13] Khurana, M.S., Winarto, H. and Sinha, A.K., "Airfoil optimization by swarm algorithm with mutation and artificial neural networks", *47th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition* Orlando, Florida AIAA 2009-1278, 2009.

[14] Singh, G. and Grandhi, R.V., "Mixed-variable optimization strategy employing multi-fidelity simulation and surrogate models", *AIAA Journal,* vol. 48, no. 1, pp. 215-223, 2010.

[15] Carrese, R., Winarto, H. and Li, X., "Integrating user-preference swarm algorithm and surrogate modeling for airfoil design", *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition* Orlando Florida AIAA 2011-1246, 2011.

[16] Carrese, R., Sobester, A., Winarto, H. and Li, X., "Swarm heuristic for identifying preferred solutions in surrogate-based multi-objective engineering design", *AIAA Journal,* vol. 49, no. 7, pp. 1437-1449, 2011.

[17] Ong, Y. S., Nair, P. B., and Keane, A. J., "Evolutionary optimization of computationally

expensive problems via surrogate modeling", *AIAA Journal*, vol. 41, no. 4, pp. 687-696, 2003.

[18] Pehlivanoglu, Y.V. and Baysal, O., "Vibrational genetic algorithm enhanced with fuzzy logic and neural networks", *Aerospace Science and Technology,* vol. 14, no. 1, pp. 56-64, 2010.

[19] Hacioglu, A., "Fast evolutionary algorithm for airfoil design via neural network", *AIAA Journal,* vol. 45, no. 9, pp. 2196-2203, 2007.

[20] Pehlivanoglu, Y. V., "Improved particle swarm optimization method in inverse design problems", In International Work Conference on Artificial Neural Networks (IWANN). LNCS 7902, Rojas, I., Joya, G., and Cabestany, J. (Eds.), pages 218–231, 2013.

[21] Eberhart, R.C. and Kennedy, J., "A new optimizer using particle swarm theory", *Proc. 6th Int. Symp. Micromachine Human Sci.* Nagoya Japan, pp. 39–43, 1995.

[22] Clerc, M. and Kennedy, J., "The particle swarm-explosion, stability, and convergence in a multidimensional complex space", *IEEE Trans. Evol. Comput.,* vol. 6, no. 1, pp. 58–73, 2002.

[23] Shi, Y. and Eberhart, R., "A modified particle swarm optimizer", *Proc. of the World Congr. Comput. Intell,* pp. 69–73, 1998.

[24] Pehlivanoglu, Y.V., "Hybrid Intelligent Optimization Methods for Engineering Problems", Ph.D. Dissertation, Dept. of Aerospace Engineering, Old Dominion Univ., Norfolk, VA, 2010.

[25] Liang, J.J., Qin, A.K., Suganthan, P.N., and Baskar, S., "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", *IEEE Trans. Evol. Comput.,* vol. 10, no. 3, pp. 281-295, 2006.

[26] Neural Network Toolbox, *Matlab the language of technical computing* Version R2007b The MathWorks, Inc., 2007.

[27] Gálvez, A., Cobo, A., Puig-Pey, J. and Iglesias, A., "Particle swarm optimization for Bézier surface reconstruction", *Lecture Notes in Computer Science,* vol. 5102, pp. 116-125, 2008.

[28] Gálvez, A., Iglesias, A., Cobo, A., Puig-Pey, J. and Espinola, J., "Bézier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least-squares approximation", *Lecture Notes in Computer Science,* vol. 4706, pp.680-693, 2007.

[29] Farin, G., Curves and surfaces for computer aided geometric design; a practical guide, Academic Press Inc. p. 41-42, 1993.

[30] Jameson, A., Essential Elements of Computational Algorithms for Aerodynamic Analysis and Design, NASA/CR-97-206268 ICASE Report No. 97-68, p. 34-35, 1997.

[31] Anderson, J.D., Fundamentals of Aerodynamics, Mc-Graw Hill, Inc. pp. 217-222, 1984.

VITAE

Assoc.Prof.Dr.Col. Y. Volkan PEHLİVANOĞLU
He was born in 1973 in Erzurum. He graduated from the Technical University of Istanbul/Turkiye in 1993 as an aeronautical engineer. He has worked in different sections of Turkish Air Force until 2004. He had M.Sc. degree in 2006 and PhD degree in 2010 from the department of aerospace engineering in Old Dominion University/USA. He is associate professor since 2014. His research interests are optimization methods, aerodynamics, and path planning problems.

Dr.Col. Serdar AY
Serdar AY received the B.Eng. degree in mechanical engineering from Technical University of Istanbul in 1992, Istanbul,Turkey, a Master's Degree in mechanical engineering from Eskisehir Osmangazi University in 2003, Eskisehir, Turkey, and the PhD degree from aerospace engineering The Aeronautics and Space Technologies Institute (ASTIN) of Turkish Air Force Academy (TuAFA), Istanbul,Turkey in 2013. He has been working as a military lecturer at the Turkish Air Force Academy (TuAFA), Istanbul,Turkey since 2007. His research interests are kinematics and workspaces of parallel mechanisms.

Lt.Col. Faruk GÜL
He was born in 1973 in YOZGAT. He graduated from the Technical University of Istanbul/Turkey in 1996 as a civil engineer. He worked in different sections of the Turkish Air Force until 2006. He had an M.Sc. degree in 2004. Currently, he is working in the Turkish Air Force Academy and a PhD candidate at the department of civil engineering in Osman Gazi University. His research interests are impact loading, and material science.