Computer-Aided Civil and Infrastructure Engineering **26** (2011) 48–68

A Hybrid Particle Swarm—Gradient Algorithm for Global Structural Optimization

Vagelis Plevris & Manolis Papadrakakis*

Institute of Structural Analysis & Seismic Research, National Technical University of Athens, 9, Iroon Polytechniou Str., Zografou Campus, GR-15780 Athens, Greece

Abstract: *The particle swarm optimization (PSO) method is an instance of a successful application of the philosophy of bounded rationality and decentralized decision making for solving global optimization problems. A number of advantages with respect to other evolutionary algorithms are attributed to PSO making it a prospective candidate for optimum structural design. The PSO-based algorithm is robust and well suited to handle nonlinear, nonconvex design spaces with discontinuities, exhibiting fast convergence characteristics. Furthermore, hybrid algorithms can exploit the advantages of the PSO and gradient methods. This article presents in detail the basic concepts and implementation of an enhanced PSO algorithm combined with a gradient-based quasi-Newton sequential quadratic programming (SQP) method for handling structural optimization problems. The proposed PSO is shown to explore the design space thoroughly and to detect the neighborhood of the global optimum. Then the mathematical optimizer, starting from the best estimate of the PSO and using gradient information, accelerates convergence toward the global optimum. A nonlinear weight update rule for PSO and a simple, yet effective, constraint handling technique for structural optimization are also proposed. The performance, the functionality, and the effect of different setting parameters are studied. The effectiveness of the approach is illustrated in some benchmark structural optimization problems. The numerical results confirm the ability of the proposed methodology to find better optimal solutions for structural optimization problems than other optimization algorithms.*

*To whom correspondence should be addressed. E-mail: *mpapadra@central.ntua.gr*.

DOI: 10.1111/j.1467-8667.2010.00664.x

1 INTRODUCTION

In the past two decades, a number of optimization algorithms have been used in structural design optimization, ranging from gradient-based mathematical algorithms to nongradient probabilistic-based search algorithms, for addressing global nonconvex optimization problems. Many important probabilistic-based algorithms have been inspired by natural phenomena, such as evolutionary programming (EP), genetic algorithms (GA), evolution strategies (ES), among others. Recently, a family of optimization methods has been developed based on the simulation of social interactions among members of a specific species looking for food or resources in general. The term swarm intelligence (SI) describes the collective behavior of decentralized, self-organized natural or artificial systems. SI methods include particle swarm optimization (PSO), ant colony optimization (ACO) (Kaveh and Shojaee, 2007; Yang et al., 2007; Vitins and Axhausen, 2009) and other methods (Rodriguez and Reggia, 2009). PSO is based on the behavior reflected in flocks of birds, bees, and fish that adjust their physical movements to avoid predators and seek for food. The method has been given considerable attention in recent years among the optimization research community.

A swarm of birds or insects or a school of fish searches for food, resources, or protection in a very typical manner. If a member of the swarm discovers a desirable path to go, the rest of the swarm will follow quickly. Every member searches for the best in its locality, learns from its own experience as well as from the others typically from the best performer among them. Even human beings show a tendency to behave in this way as they learn from their own experience, their immediate neighbors, and the ideal performers in the society. The PSO method mimics the behavior described above. The algorithm was first proposed by Kennedy and Eberhart

(1995). It is a population-based optimization method built on the premise that social sharing of information among the individuals can provide an evolutionary advantage.

PSO shares many similarities with evolutionary computation techniques, such as GA, but the conceptual difference lies in its definition which is given in a social rather than a biological context. The common features of the two optimization approaches include the population concept of the design vectors, initialization with a population of random solutions, a fitness value to evaluate performance, searching for optima by updating iterations (generations) based on a stochastic process, no requirement for gradient information or user-defined initial estimates and no guaranteed final success. However, unlike GA, PSO has no genetic operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following a velocity update rule. Compared to GA, the information sharing mechanism in PSO is significantly different. In GA, chromosomes share information with each other, so the whole population moves like one group toward an optimal area. In PSO, only Gbest (the global best particle) communicates the information to the others, forming a one-way information sharing mechanism. According to Angeline (1998), two main distinctions can be made between PSO and an evolutionary algorithm (EA): (i) EAs rely on three mechanisms in their processing: parent representation, selection of individuals, and the fine tuning of their parameters. In contrast, PSO only relies on two mechanisms, because PSO does not adopt an explicit selection function. The absence of a selection mechanism in PSO is compensated by the use of leaders to guide the search. However, there is no notion of offspring generation in PSO as with EAs. (ii) The manipulation of the individuals is different in EAs and PSO. PSO uses an operator that sets the velocity of a particle to a particular direction. This can be seen as a directional mutation operator in which the direction is defined by both the particle's personal best and the global best (of the swarm). If the direction of the personal best is similar to the direction of the global best, the angle of potential directions will be small, whereas a larger angle will provide a larger range of exploration. In contrast, EAs use a mutation operator that can set an individual in any direction (although the relative probabilities for each direction may be different). In fact, the limitations exhibited by the directional mutation of PSO has led to the use of mutation operators similar to those adopted in EAs.

A number of advantages over other algorithms make PSO a prospective candidate to be used in structural optimization problems. It can handle nonlinear, nonconvex design spaces with discontinuities. Compared to other nondeterministic optimization methods it is considered efficient in terms of number of function evaluations as well as robust because it usually leads to better or the same quality of results. Its easiness of implementation makes it more attractive as it does not require specific domain knowledge information, while being a population-based algorithm, it can be straight forward implemented in parallel computing environments leading to a significant reduction of the total computational cost. Compared to GA, PSO is easier to implement and there are only a few parameters to adjust. According to the study of Hassan et al. (2005), PSO and GA can both obtain high quality solutions, yet the computational effort required by PSO to arrive to such high quality solutions is less than the corresponding effort required by GA. PSO has been successfully applied to many fields, such as mathematical function optimization, artificial neural network training, and fuzzy system control.

Particle swarms had not been used in the field of structural design optimization until recently, where limited studies have been performed. Promising results have been presented in the areas of structural shape optimization (Fourie and Groenwold, 2002; Venter and Sobieszczanski-Sobieski, 2004) as well as topology optimization (Fourie and Groenwold, 2001). Perez and Behdinan (2007b; 2007a) implemented the PSO algorithm for constrained structural optimization of plane and space truss structures while Li et al. (2007) tried a heuristic PSO scheme for the optimization of truss structures.

The numerical tests performed with the PSO algorithm have shown rapid convergence during the initial stages of a global search, but at the neighborhood of the global optimum, the search process becomes rather slow, a typical behavior of all evolutionary type optimization algorithms. On the contrary, recent studies revealed that gradient descending method can achieve faster convergence speed around global optimum and, at the same time, the convergence accuracy can be higher.

Various hybrid methods that combine EA with mathematical optimizers have been proposed in the past. Papadrakakis and Lagaros (2000) implemented a hybrid GA-MP scheme for structural sizing optimization, while Papadrakakis et al. (1999) proposed a hybrid ES-SQP scheme for structural shape optimization with very satisfactory results. Hybrid PSO methods have also been proposed recently. Kaveh and Talatahari (2008) implemented a hybrid PSO and ant colony optimization for the design of truss structures, while Dimopoulos (2007) proposed a hybrid GA-PSO scheme for the optimization of mathematical functions and the optimal design of a welded beam and a pressure vessel.

The PSO has been also combined with mathematical methods in various ways. Izui et al. (2005, 2007) combined a PSO scheme with gradients, where the members of the swarm were divided into sequential linear programming (SLP) and PSO individuals. Zhang et al. (2007) proposed a hybrid PSO – back-propagation algorithm for feed-forward neural network training. Chen et al. (2007) combined the PSO algorithm with a conjugate gradient-based local search method for the identification of nonlinear systems. Coelho and Mariani (2007) combined the PSO with a quasi-Newton local search method for solving an economic dispatch problem. Das et al. (2006) proposed adding a local search component to PSO to improve its convergence speed for estimating the parameters of a gene network model. Victoire and Jeyakumar (2004) combined the PSO with sequential quadratic programming (SQP) for solving the economic dispatch problem, where SQP was used to fine tune the solution of the PSO.

In this article, a hybrid algorithm combining the PSO algorithm with a gradient-based quasi-Newton SQP algorithm, referred to as PSO–SQP is proposed for the optimization of engineering structures. The hybrid algorithm can make use of not only the strong global searching ability of the PSO, but also the strong local searching ability of the SQP algorithm. Enhancements are also proposed for the PSO algorithm for constrained structural optimization with a new implementation of a nonlinear cubic weight update rule and a simple yet effective constraint handling technique. The numerical results show that the proposed hybrid PSO–SQP algorithm performs better than the standard PSO algorithm as well as other established EA algorithms in terms of convergence speed and final results achieved. The method is applied to structural engineering optimization problems where the aim is to find the optimum design of a structure under specific loads. The structures considered are plane or space trusses, the objective function is the weight of the structure, while the constraints refer to restrictions in the maximum values of stresses and displacements. The constraints are checked by a finite element analysis for every candidate optimum design.

With the improvement of solution algorithms and optimization methods and the increased efficiency of the computing power of high-performance computers, large-scale optimization has become a trend in structural design. Adeli and Cheng (1994) presented two concurrent augmented Lagrangian GAs for the optimization of large scale structures utilizing the multiprocessing capabilities of high-performance computers. Adeli and Kumar (1995a, 1995b) implemented a distributed GA for the optimization of large structures on a cluster of workstations and a mixed computational model for GA-based structural optimization on massively parallel supercomputers. Soegiarso and Adeli (1997, 1998) presented an efficient parallel-vector algorithm for the optimization of large scale frame steel structures subjected to realistic code-specified constraints, showing that both parallel processing and vectorization performance improve with the increase in the size of the structure. Kamal and Adeli (2000) presented a fuzzy discrete multicriteria cost optimization model for the design of space steel structures subjected to design codes constraints. Sarma and Adeli (2001) also investigated the optimization of very large steel structures subjected to the actual constraints of the AISC specifications in high-performance multiprocessor machines with GAs using a distributed memory Message Passing Interface with the processor farming scheme and the migration scheme.

The layout of the article is as follows: Following the introduction, Section 2 presents the general formulation of a structural optimization problem. In Section 3, the main PSO algorithm for unconstrained optimization is described. Section 4 presents the proposed PSO method for structural constrained optimization using the nonlinear cubic inertia weight update rule and a penalty function. Section 5 describes the gradient-based SQP method implemented in this study, while Section 6 introduces the proposed PSO-SQP hybrid algorithm. In Section 7 the numerical results are presented, and finally Section 8 summarizes the concluding remarks.

2 FORMULATION OF THE STRUCTURAL OPTIMIZATION PROBLEM

A general continuous structural optimization problem can be stated as follows:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f(\boldsymbol{x}) \quad \boldsymbol{x} = [x_1, \ldots, x_n]^{\mathrm{T}} \\ &\text{Subject to} \\ &g_k(\boldsymbol{x}) \le 0, \quad k = 1, \ldots, m \\ &\boldsymbol{x}^{\mathrm{L}} \le \boldsymbol{x} \le \boldsymbol{x}^{\mathrm{U}} \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is a vector of length n containing the design variables, $f(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ is the objective function, which returns a scalar value to be minimized (usually the weight of the structure), the vector function $\boldsymbol{g}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^m$ returns a vector of length m containing the values of the inequality constraints evaluated at $\boldsymbol{x}$, and $\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$ are two vectors of length n containing the lower and upper bounds of the design variables, respectively. The above mathematical formulation contains only inequality constraints, as equality constraints are usually not the case in structural optimization.

Considering the weight of the structure, the objective function becomes

$$f(\boldsymbol{x}) = \rho \cdot \sum_{i=1}^{N_e} A_i \cdot L_i \tag{2}$$

where ρ is the material density, N_{e} is the number of elements of the model, and A_i, L_i are the cross sectional area and the length of each element, respectively.

A typical constraint k in structural optimization has the form

$$g_k(\boldsymbol{x}) = \left|q_k(\boldsymbol{x})\right| - q_{\mathrm{allow},k} \tag{3}$$

where $q_k(\boldsymbol{x})$ is a response measure (usually stress or displacement) for design $\boldsymbol{x}$ and $q_{\mathrm{allow},k}$ is its maximum allowable absolute value. It should be noted that $q_k(\boldsymbol{x})$ in this study is taken as the maximum (worst) value of the corresponding response measure among all nodes or elements of the model. For example, if the k-th constraint is a stress constraint of the type $|\sigma| \leq \sigma_{\mathrm{allow}}$ that applies for all N_{e} model elements, then for this constraint a single response measure is calculated as

$$q_k(\boldsymbol{x}) = \max_{i=1}^{N_{\mathrm{e}}} \{|\sigma_i|\} \tag{4}$$

3 THE PSO ALGORITHM FOR UNCONSTRAINED OPTIMIZATION

In a PSO formulation, multiple candidate solutions coexist and collaborate simultaneously. Each solution is called a "particle" that has a position and a velocity in the multidimensional design space. A particle "flies" in the problem search space looking for the optimal position. As "time" passes through its quest, a particle adjusts its velocity and position according to its own "experience" as well as the experience of other (neighboring) particles. A particle's experience is built by tracking and memorizing the best position encountered. As every particle remembers the best position it has visited during its "flight," the PSO possesses a memory. A PSO system combines local search method (through self experience) with global search method (through neighboring experience), attempting to balance exploration and exploitation.

3.1 Mathematical formulation of PSO

Each particle maintains two basic characteristics, velocity and position in the multi dimensional search space, that are updated in a stochastic way as follows:

$$\begin{aligned}\boldsymbol{v}^j(t+1) = w\boldsymbol{v}^j(t) + c_1\boldsymbol{r}_1 \circ \left(\boldsymbol{x}^{\mathrm{Pb},j} - \boldsymbol{x}^j(t)\right) \\ + c_2\boldsymbol{r}_2 \circ \left(\boldsymbol{x}^{\mathrm{Gb}} - \boldsymbol{x}^j(t)\right)\end{aligned} \tag{5}$$

$$\boldsymbol{x}^j(t+1) = \boldsymbol{x}^j(t) + \boldsymbol{v}^j(t+1) \tag{6}$$

where $\boldsymbol{v}^j(t)$ denotes the velocity vector of particle j at time t, $\boldsymbol{x}^j(t)$ represents the position vector of particle j at time t, vector $\boldsymbol{x}^{\mathrm{Pb},j}$ is the memory of particle j at current iteration (the personal best ever position of the particle, corresponding to the objective function value $Pbest_j$), and vector $\boldsymbol{x}^{\mathrm{Gb}}$ is the global best location found by the entire swarm up to the current iteration (corresponding to the objective function value *Gbest*, the same for all particles). The acceleration coefficients c_1 and c_2 represent "trust" settings which indicate the degree of confidence in the best solution found by the individual particle (c_1—cognitive parameter) and by the whole swarm (c_2—social parameter), respectively, while $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are two random vectors with numbers uniformly distributed in the interval [0, 1]. The symbol "$\circ$" of Equation (5) denotes the Hadamard product, i.e. the element-wise vector or matrix multiplication. The random vectors $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are used instead of random numbers given that for every particle and every iteration, n pairs of random numbers are required, as different random numbers have to be applied for every dimension of a particle.

In the above formulation, the global best location found by the entire swarm up to the current iteration ($\boldsymbol{x}^{\mathrm{Gb}}$) is used. This is called a fully connected topology (fully informed PSO), as all particles share information with each other about the best performer of the swarm. Other topologies have also been used in the past, where instead of the global best location found by the entire swarm, a local best location of each particle's "neighborhood" is used. The neighborhood of a particle includes a number of other particles with which a particle shares information. The neighborhood size n_{h} can vary, provided that it is smaller than the number of particles *NP*. In the fully connected topology, $n_{\mathrm{h}} = NP$.

The term w of Equation (5) is the inertia weight, a scaling factor employed to control the exploration abilities of the swarm, which scales the current velocity value affecting the updated velocity vector. The inertia weight was not part of the original PSO algorithm (Kennedy and Eberhart, 1995), as it was introduced later by Shi and Eberhart (1998) in a successful attempt to improve convergence. Large inertia weights will force larger velocity updates allowing the algorithm to explore the design space globally. Similarly, small inertia values will force the velocity updates to concentrate in the nearby regions of the design space. The inertia weight can also be updated during iterations, as will be described in detail in Section 4.

Particles' velocities in each dimension i ($i = 1, \ldots, n$) are restricted to a maximum velocity $v_i^{\max}$. The vector $\boldsymbol{v}^{\max}$ of dimension n holds the maximum absolute velocities for each dimension. It is more appropriate to use a vector rather than a scalar, as in the general case different velocity restrictions can be applied for different dimensions of the particle. If for a given particle j the sum of accelerations of Equation (5) causes the absolute velocity for dimension i to exceed $v_i^{\max}$, then the velocity on that dimension is limited to $\pm v_i^{\max}$. The vector parameter $\boldsymbol{v}^{\max}$ is employed to protect the cohesion of the system, in the process of amplification of the positive feedback.

3.2 Design variables bounds handling

As shown in Equation (1), constraints also apply in the available space for every design variable x_i, as in vector terms $\boldsymbol{x}^{\mathrm{L}} \leq \boldsymbol{x} \leq \boldsymbol{x}^{\mathrm{U}}$. If, after the velocity update rule of Equation (5), the position update of Equation (6) forces a particle to move outside the bounds for a dimension i ($x_i \leq x_i^{\mathrm{L}}$ or $x_i \geq x_i^{\mathrm{U}}$), then the design variable x_i is reset to the closest bound ($x_i = x_i^{\mathrm{L}}$ or $x_i = x_i^{\mathrm{U}}$). To avoid considering any points outside the specified design space, the corresponding coefficient $\boldsymbol{v}_i$ of the velocity vector $\boldsymbol{v}$ is reset to zero, to be used for the next iteration.

3.3 Main PSO parameters

The basic PSO has only a few parameters to adjust. Below is a list of the main parameters, their typical values and other details.

3.4 Convergence criteria

Due to the repeated process of the PSO search, convergence criteria have to be applied for the termination of the optimization procedure. Two widely adopted convergence criteria are the maximum number of iterations of the PSO algorithm and the minimum error requirement on the calculation of the optimum value of the objective function. The selection of the maximum number of iterations depends generally on the complexity of the optimization problem at hand. The second criterion presumes prior knowledge of the global optimum value, which is feasible for testing or fine-tuning the algorithm in mathematical problems when the optimum is known a priori, but this is certainly not the case in practical structural optimization problems where the optimum is not known *a priori*.

In our study, together with the maximum number of iterations, we have implemented the convergence criterion connected to the rate of improvement of the value of the objective function for a given number of iterations. If the relative improvement of the objective function over the last k_f iterations (including the current iteration) is less or equal to a threshold value f_m, convergence is supposed to have been achieved. In mathematical terms, denoting as $Gbest_t$ the best value of the objective function found by the PSO at iteration t, the relative improvement of the objective function can be written for the current iteration t as follows:

$$\frac{Gbest_{t-k_f+1} - Gbest_t}{Gbest_{t-k_f+1}} \leq f_m \tag{7}$$

In Table 1 there is a list of the main PSO parameters, while Table 2 shows the convergence parameters of the PSO used in this study, with description and details.

4 PSO FOR CONSTRAINED STRUCTURAL OPTIMIZATION

Two important features which require special attention when dealing with practical engineering optimization problems are the improvement of the convergence rate and the handling of the problem constraints. As described below, different modifications can be made to the original algorithm to address these features making the algorithm capable of dealing with more demanding constrained optimization problems such as those present in the optimum design of engineering structures.

4.1 Inertia weight update

The PSO global convergence is affected by the degree of local/global exploration provided by the c_1 and c_2 parameters, while the relative rate of convergence is affected by the inertia weight parameter. Studies have shown that for a fixed inertia value there is a significant reduction in the algorithm convergence rate as iterations progress. This is the consequence of excessive momentum in the particles, which results in large step sizes that overshoot the best design areas. During the initial optimization stages, a large inertia weight is needed in order for the design space to be searched thoroughly. Once the most promising areas of the design space have been discovered and the convergence rate starts to slow down, the inertia weight should be reduced, in order for the particles' momentum to decrease allowing them to concentrate in the best design areas. To accomplish the above strategy, Shi and Eberhart (1998) proposed a time-dependent value of the inertia weight. A commonly used inertia update rule is the linearly decreasing, calculated by the formula:

$$w_{t+1} = w_{\max} - \frac{w_{\max} - w_{\min}}{t_{\max}} \cdot t \tag{8}$$

where t is the iteration number (starting from iteration 0), $w_{\max}$ and $w_{\min}$ are the maximum and minimum values, respectively, of the inertia weight. In general, the

Table 1
Main PSO parameters

Symbol	*Description*	*Details*
NP	Number of particles	A typical range is 10–40. For most problems 10 particles is sufficient enough to get acceptable results. For some difficult or special problems the number can be increased to 50–100
n	Dimension of particles	It is determined by the problem to be optimized
w	Inertia weight	Usually is set to a value less than 1, i.e. 0.95. It can also be updated during iterations
$\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$	Vectors containing the lower and upper bounds of the n design variables, respectively	They are determined by the problem to be optimized. Different ranges for different dimensions of particles can be applied in general
$\boldsymbol{v}^{\max}$	Vector containing the maximum allowable velocity for each dimension during one iteration	Usually is set half the length of the allowable interval for the given dimension: $v_i^{\max} = (x_i^{\mathrm{U}} - x_i^{\mathrm{L}})/2$. Different values for different dimensions of particles can be applied in general
c_1, c_2	Cognitive and social parameters	Usually $c_1 = c_2 = 2$. Other values can also be used, provided that $0 < c_1 + c_2 < 4$ (Perez and Behdinan, 2007a)

Table 2
PSO convergence parameters

Symbol	*Description*	*Details*
$t_{\max}$	Maximum number of iterations for the termination criterion	Determined by the complexity of the problem to be optimized, in conjunction with other PSO parameters (n, NP)
k_f	Number of iterations for which the relative improvement of the objective function satisfies the convergence check	If the relative improvement of the objective function over the last k_f iterations (including the current iteration) is less or equal to f_m, convergence has been achieved
f_m	Minimum relative improvement of the value of the objective function	

linearly decreasing inertia weight has shown better performance than the fixed one.

In this article, we adopted a new nonlinear weight update strategy: The total allowed iterations $t_{\max}$ are divided into three stages. At the end of each stage, the change (reduction) of w compared to the one at the end of the previous stage has to be a_w times its value. Given that, we can define the value of w at $t_{\max}/3$ and $2{\cdot}t_{\max}/3$ iterations. A cubic polynomial is then calculated that interpolates the four points (starting point $(0, w_{\max})$, ending point $(t_{\max}, w_{\min})$ and two intermediate points, $(1/3{\cdot}t_{\max}, w_{\max} - a_w^2 \cdot b)$ and $(2/3{\cdot}t_{\max}, w_{\min}+b)$), where b is the reduction of w for the third stage, as shown in Figure 1a. The parameter b is not a new parameter as it is dependent on $w_{\max}$, $w_{\min}$ and a_w and can be easily calculated as

$$b + a_w \cdot b + a_w^2 \cdot b = w_{\max} - w_{\min} \Leftrightarrow \tag{9}$$

$$b = \frac{w_{\max} - w_{\min}}{a_w^2 + a_w + 1} \tag{10}$$

Compared to the linear update rule, the proposed nonlinear 3rd-order formulation has the advantage of a fast reduction of the inertia weight in the first stage of the optimization, while in the vicinity of the optimum, the reduction becomes slower, as shown in Figure 1b. This type of behavior is in most cases favorable in PSO optimization, as will be shown in the numerical examples section. The linear update rule can be obtained by setting $a_w = 1$. Typical values for a_w are in the interval [1.0, 2.0]. Values smaller than 1 should not be considered as they would lead to the opposite undesirable result; a small reduction of the inertia weight in the first stages and a fast reduction near the optimum.

4.2 Constraint handling techniques

Although the PSO has been applied for the solution of a number of problems recently, its applications are mainly focused on unconstrained optimization. Various methods have been proposed for handling nonlinear constraints by EAs in general. Koziel and Michalewicz (1999) grouped them into four categories: (i) methods based on preserving feasibility of the solutions; (ii) methods based on penalty functions; (iii) methods that search for feasibility; (iv) other hybrid methods. Very

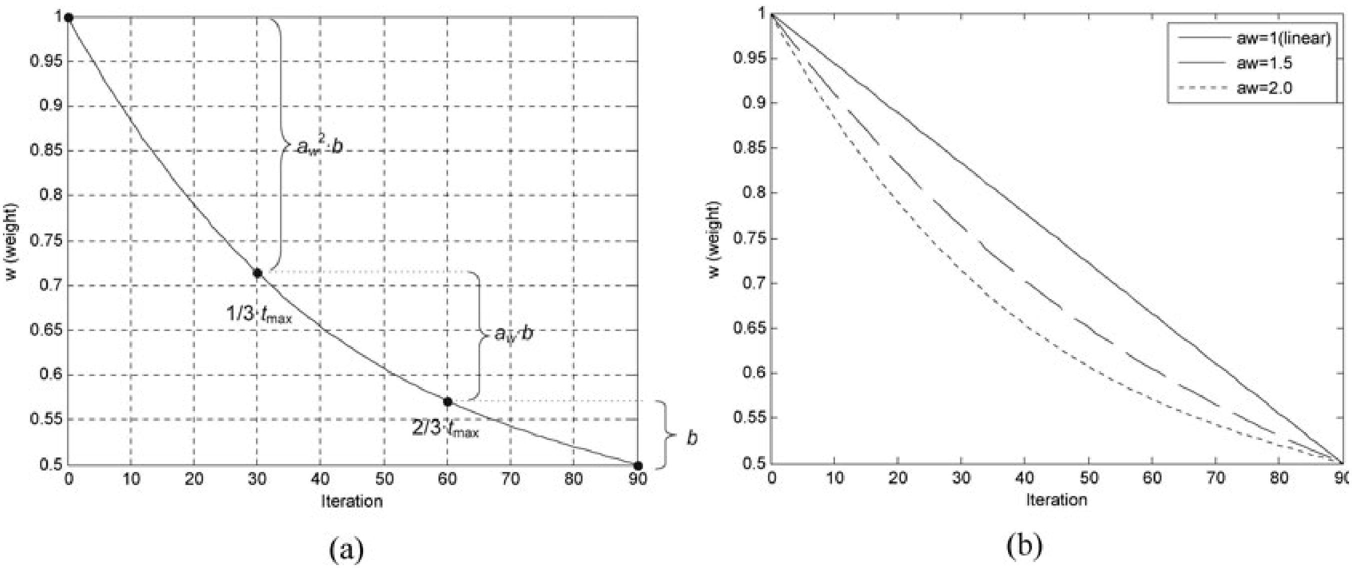

Fig. 1. The proposed nonlinear weight update rule: (a) For $t_{max} = 90$, $w_{min} = 0.5$, $w_{max} = 1$ and $a_w = 2$, (b) For different values of a_w (1, 1.5, 2).

few studies have extended the application of PSO to constrained optimization problems (Hu and Eberhart, 2002; Parsopoulos and Vrahatis, 2002).

A simple approach for PSO would be to recalculate the velocity vector for an infeasible individual (a particle with at least one violated constraint) using new random numbers r_1 and r_2, until the new position of the particle becomes feasible (all constraints are satisfied). This simplistic approach guarantees the feasibility of the final optimum design, yet it has a strong disadvantage as it needs too many calculations of the constraint functions and subsequently of finite element analyses, especially in cases where the feasible region is small compared to the entire design space, making it impractical for structural engineering applications.

Another approach is to avoid taking into account the infeasible designs in the calculation of *Pbest* or *Gbest* for a particle, given that the swarm is initialized in the feasible region (Hu et al., 2003). This "death penalty" approach that guarantees the feasibility of the final optimum has the disadvantage that it does not take into account the degree of the violation of the constraints. Moreover, a search over the feasible region only is usually less efficient than over both the feasible and infeasible region, as the latter makes it possible to approach the optimum from any direction (Michalewicz, 1995).

Venter and Sobieszczanski-Sobieski (2004) proposed a constraint handling mechanism for PSO that redirects the violated designs back to the feasible region. After a particle j has moved to an infeasible position at iteration t, the method modifies the velocity vector $\boldsymbol{v}^j(t)$, by resetting it to zero. Then, the velocity vector $\boldsymbol{v}^j(t+1)$ for next iteration $t+1$ is obtained from Equation (5) omitting the inertia coefficient $w \cdot \boldsymbol{v}^j(t)$ that equals zero. The new velocity of particle j at iteration t+1 is thus only influenced by the best point found so far by the particle ($\boldsymbol{x}^{\mathrm{Pb},j}$) and the current best point found by the entire swarm ($\boldsymbol{x}^{\mathrm{Gb}}$). Given that both these best points are feasible, the new velocity vector will point back to a feasible region of the design space, ensuring in most cases that the particle is directed back to the feasible space, or at least closer to the feasibility boundary. This method is also simple, but has the disadvantage that it does not guarantee feasibility of the particles and as a result, there is no guarantee that for the optimum solution all constraints will be satisfied.

Some researchers attempted to solve the constrained problem indirectly by transforming it to an unconstrained problem using the traditional penalty function strategy (Parsopoulos and Vrahatis, 2002; Perez and Behdinan, 2007a). The penalty function is an effective auxiliary tool to deal with constrained problems in general and has been a popular approach because of its simplicity and ease of implementation. Yeniay (2005) examined various penalty function methods for GA, highlighting the strengths and weaknesses of each method.

In our study, we propose a simple yet effective multiple linear segment penalty function to deal with constraints. Consider a structural optimization problem where displacement and stress constraints are imposed. For a given design $\boldsymbol{x}$, the corresponding objective function value is computed and a finite element analysis is performed for the constraints check where each

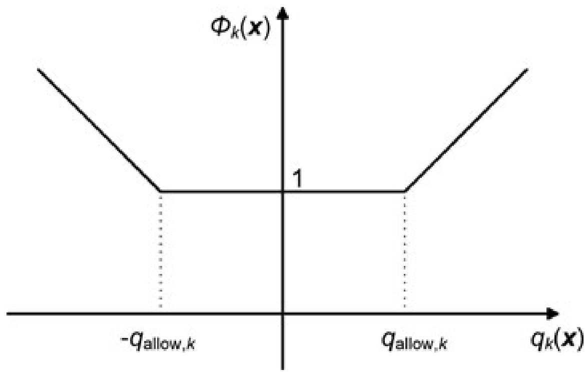

Fig. 2. A multiple linear segment penalty function.

structural element is checked for stress violation, and each model node is checked for displacement violation. If no violation is detected, then no penalty is imposed on the objective function $f(\boldsymbol{x})$. If any of the constraints are violated, a penalty is applied to the objective function and the value of the penalty is related to the degree to which the constraints are violated. The penalty function $\Phi_k(\boldsymbol{x})$ for the typical constraint k of Equation (3) is defined as

$$\Phi_k(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} \leq 1 \\ \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} & \text{if } \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} > 1 \end{cases} \tag{11}$$

Figure 2 gives a graphical representation of the above formula.

Having obtained the penalty function factors for all violated constraints, the penalized fitness value of a design $\boldsymbol{x}$ is obtained by multiplying the objective function (structural weight or structural material volume) to be minimized by the maximum penalty factor among all m constraints:

$$f_{\text{p}}(\boldsymbol{x}) = f(\boldsymbol{x}) \cdot \max\{\Phi_k(\boldsymbol{x})\}, \quad k = 1, \ldots, m \tag{12}$$

where f_{p} is the new fitness (penalized objective function). The resulting penalized objective function quantitatively represents the extent of the violation of constraints and provides a relatively meaningful measurement of the performance of each solution.

Using the above formulation, there is a case where the penalized objective function $f_{\text{p}}(\boldsymbol{x})$ can obtain a better value compared to the global optimum $Gbest_t$ found by the entire swarm until iteration t. This will result in resetting $Gbest_t$ to an infeasible design. Indeed, this can happen for an infeasible design if $\max\{\Phi_k(\boldsymbol{x})\} < Gbest_t/f(\boldsymbol{x})$. To avoid this undesirable case, $Gbest_t$ is used instead of $f(\boldsymbol{x})$ in Equation (12), when $f(\boldsymbol{x}) < Gbest_t$ and $\max\{\Phi_k(\boldsymbol{x})\} > 1$ (infeasible design). In this sense, $Gbest_t$ is penalized instead of $f(\boldsymbol{x})$, for infeasible designs with objective functions $f(\boldsymbol{x})$ better than $Gbest_t$. This ensures that the best design found by the swarm will always stay in the feasible region, as will be shown in the numerical examples section.

5 GRADIENT-BASED SQP METHOD

Mathematical (gradient-based) optimization methods are generally considered as local methods. They exhibit fast convergence by exploiting gradient information but they cannot guarantee the estimation of the global optimum, as they can be easily trapped in local minima. These methods require user-defined initial estimates of the solution. The mathematical optimizer used in this study is a SQP method. SQP methods are the standard general purpose mathematical programming algorithms for solving nonlinear programming (NLP) optimization problems. They are also considered to be the most suitable methods for solving structural optimization problems with the mathematical programming approach. Such methods make use of local curvature information derived from linearization of the original functions, by using their derivatives with respect to the design variables at points obtained in the process of optimization.

Given the problem description of Equation (1), SQP method proceeds with the conversion of the NLP problem into a sequence of Quadratic Programming (QP) subproblems based on a quadratic approximation of the Lagrangian function:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{k=1}^{m} \lambda_k g_k(\boldsymbol{x}) \tag{13}$$

where λ_k are the Lagrange multipliers under the non-negativity restriction for the inequality constraints. The QP subproblem can be obtained by linearizing the nonlinear constraints. Each QP subproblem has the following form:

$$\begin{aligned} &\min_{p \in ¡^n} \frac{1}{2} \boldsymbol{p}^T \boldsymbol{H}_\ell \boldsymbol{p} + \nabla f(\boldsymbol{x}_\ell)^T \boldsymbol{p} \\ &\text{Subject to} \\ &\nabla g_k(\boldsymbol{x}_\ell)^T \boldsymbol{p} + g_k(\boldsymbol{x}_\ell) \leq 0, \quad k = 1, \ldots, m \end{aligned} \tag{14}$$

Where $\boldsymbol{p}$ is the search direction and $\boldsymbol{H}_\ell$ a positive definite approximation of the Hessian matrix of the Lagrangian function of Equation (13). To construct the Jacobian and the Hessian matrices of the QP subproblem, the derivatives of the objective and constraint functions are required. These derivatives can be calculated either analytically, using a closed form if available, semi-analytically or with a global finite difference method, which is used in this study.

An estimate of the Hessian matrix of the Lagrangian function is updated at each iteration using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) quasi-Newton formula and a line search is performed using a merit function to determine the step length parameter a_ℓ. The new design point is then calculated as

$$\boldsymbol{x}_{\ell+1} = \boldsymbol{x}_\ell + a_\ell \boldsymbol{p}_\ell \tag{15}$$

where ℓ denotes the current SQP iteration. The QP subproblem is solved using an active set strategy. Constrained quasi-Newton methods guarantee superlinear convergence by accumulating second-order information regarding the Karush—Kuhn–Tucker (KKT) equations using a quasi-Newton updating procedure. The KKT equations are necessary conditions for optimality for a constrained optimization problem.

6 HYBRID PSO-SQP ALGORITHM

An important feature of the SQP optimizer is that it captures relatively fast the right path to the nearest optimum. Yet, unless a good model initialization is provided, the algorithm can converge to a local suboptimum. Therefore, global algorithms less vulnerable to local optima attractors and therefore more reliable in obtaining the global optimum for nonconvex optimization problems are frequently proposed, but have often exhibited unacceptable slow convergence rates due to their random search, especially near the area of the global optimum.

In an effort to increase the robustness and the computational efficiency of the optimization procedure, hybrid algorithms can benefit from the advantages of both methodologies and alleviate their particular drawbacks. The proposed hybrid optimization strategy is divided into two separate phases. During the first phase, the PSO explores the design space thoroughly and detects the neighborhood of the global optimum. When the PSO process terminates using a rather relaxed termination criterion, the second phase starts by applying the SQP method starting from the best estimate of the PSO and using gradient information to accelerate convergence to the global optimum. The combined algorithm is denoted as PSO-SQP.

7 NUMERICAL EXAMPLES

The performance of the proposed optimization algorithm is examined in three benchmark test examples. The first one is a ten bar truss example with 10 design variables, the second is a 25 member space truss with 8 design variables, and the third is a 72 member space truss with 16 design variables.

The PSO scheme used in the study is the fully informed PSO described in Section 3, equipped with the nonlinear inertia update rule described in Section 4.1 and the constraint handling technique of Section 4.2, unless otherwise stated.

7.1 Example 1. The 10 bar plane truss

This is the standard benchmark 10 bar plane truss shown in Figure 3 with the following structural characteristics: Modulus of Elasticity $E = 10{,}000$ ksi, material weight $\rho = 0.1$ lb/in^3, length $L = 360$ in, load $P = 100$ kip. The structural members are divided into 10 groups. The design variables are the cross section areas of each member group in the interval [0.1, 35] (in^2). The constraints are imposed on stresses and displacements. The maximum allowable displacement in the $\pm x$ and $\pm y$ directions for each node is $d_{\max} = 2$ in, while the maximum allowable stress (absolute value) is $\sigma_{\text{allow}} = 25$ ksi in tension or compression and the objective is to minimize the weight of the structure under the specified constraints.

The influence of the inertia update rule on the optimization process of the PSO schemes will be first investigated. Three PSO schemes are considered, with a fixed inertia value $w = w_{\max}$ during all iterations, the linear update rule of Equation (8) and the proposed nonlinear update rule described in Section 4.1. The basic PSO parameters used are shown in Table 3.

Ten PSO optimization runs are performed for each of the three cases. The results obtained for 200 PSO iterations are reported in Table 4 which shows the objective function values obtained for the best run, the worst run, and the average of the 10 runs for each case.

It can be observed that the linear rule performs much better than the fixed rule, while the nonlinear cubic rule improves further the result of the linear rule. Figure 4

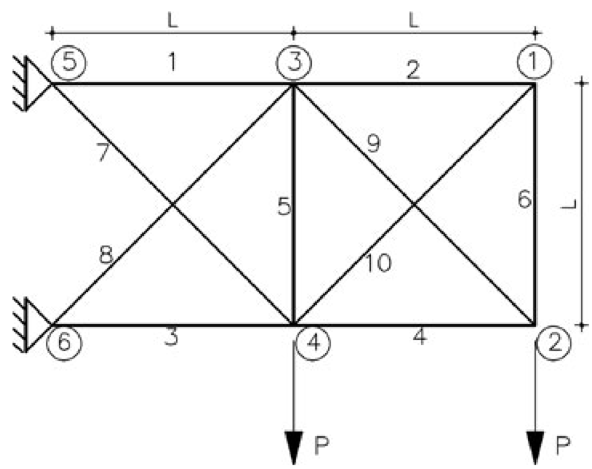

Fig. 3. Example 1. The 10 bar plane truss.

Table 3
Example 1. PSO parameters used for the inertia update rule check

Symbol	*Value*	*Symbol*	*Value*
NP	20	$\boldsymbol{v}^{\max}$	$v_i^{\max} = 17.5$ for all dimensions (in^2)
n	10	c_1, c_2	$c_1 = c_2 = 2$
w	$w_{\max} = 0.95$ $w_{\min} = 0.5$	$t_{\max}$	200
$\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$	$x_i^{\mathrm{L}} = 0.1,\ x_i^{\mathrm{U}} = 35$ for all dimensions (in^2)		

Table 4
Example 1. Statistical results for 10 PSO runs after 200 iterations

Result	*Fixed rule* ($w = w_{\max}$)	*Linear rule*	*Nonlinear cubic rule* ($a_w = 1.3$)
Best (lb)	5,212.69	5,111.72	5,062.30
Worst (lb)	5,573.70	5,225.75	5,121.46
Average (lb)	5,396.36	5,162.84	5,098.77

depicts the convergence history for the three cases in terms of the average results over 10 optimization runs.

For this test example, the best design and the values of the constraints obtained by the nonlinear rule are given in the 8th column of Table 7. It can be seen that for the above optimum design, all constraints have been

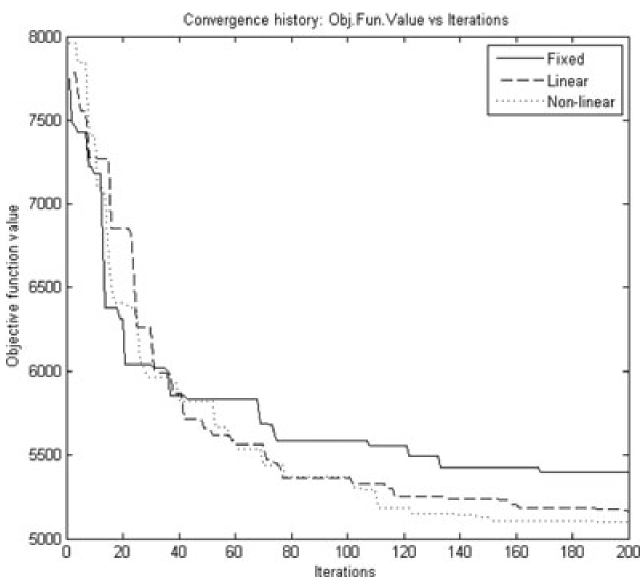

Fig. 4. Example 1. Convergence history for the three PSO schemes.

Table 5
Example 1. Statistical results for 10 PSO runs

Result	*Fixed rule* ($w = w_{\max}$)	*Linear rule*	*Nonlinear rule* ($a_w = 1.3$)
Average number of iterations	122	172	185
Best (lb)	5,356.71	5,102.53	5,065.24
Worst (lb)	5,720.01	5,443.08	5,227.05
Average (lb)	5,548.38	5,259.88	5,106.04

Table 6
Example 1. Convergence behavior of SQP

Starting point (in^2)	*Iterations*	*Obj. function evaluations*	*Obj. function value*
"35"	17	210	5,473.62
"25"	14	181	5,473.62
"15"	15	184	5,179.48
"5"	21	250	5,179.48

met and are active, as the proposed constraint handling technique for the PSO always guarantees feasibility of the optimum design achieved.

The performance of the optimization algorithms is also studied with a convergence criterion connected to the improvement of the value of the objective function for a given number of iterations. If the relative improvement of the objective function over the last $k_f = 30$ iterations is less or equal to $f_m = 10^{-6}$, convergence is supposed to have been achieved.

The results reported in Table 5 include the average number of iterations needed for convergence and the objective function values obtained for the best run, the worst run, and the average of 10 runs.

The nonlinear rule performs much better than the other two PSO schemes in terms of the best result, the worst result and the average result. This is due to the fact that using the nonlinear rule, convergence toward the optimum is smoother, and as a result the optimizer is more likely to find a better optimum solution before the termination criterion is satisfied. This is very important in practical structural optimization where the number of iterations needed for convergence is not known *a priori.*

7.1.1 Constraint handling technique investigation. In this study, the proposed linear segment penalty function constraint handling technique is compared with the death penalty approach (Hu et al., 2003) and the redirection approach (Venter and Sobieszczanski-Sobieski, 2004), described in detail in Section 4.2. With the proposed technique, a penalty function proportional to the

Table 7
Example 1. Optimum designs from the literature (a)

Design variable (in^2)	*(Rizzi, 1976)*	*(Gellatly and Berke, 1971)*	*(Schmit and Miura, 1976)*	*(Ghasemi et al., 1997)*	*(Schmit and Farshi, 1974)*	*(Dobbs and Nelson, 1976)*	*The proposed PSO*
A1	30.7300	31.3500	30.5700	25.7300	33.4300	30.5000	30.9810
A2	0.1000	0.1000	0.3690	0.1090	0.1000	0.1000	0.1000
A3	23.9340	20.0300	23.9700	24.8500	24.2600	23.2900	23.1714
A4	14.7330	15.6000	14.7300	16.3500	14.2600	15.4300	15.6935
A5	0.1000	0.1400	0.1000	0.1060	0.1000	0.1000	0.1000
A6	0.1000	0.2400	0.3640	0.1090	0.1000	0.2100	0.5848
A7	8.5420	8.3500	8.5470	8.7000	8.3880	7.6490	7.4298
A8	20.9540	22.2100	21.1100	21.4100	20.7400	20.9800	20.6310
A9	21.8360	22.0600	20.7700	22.3000	19.6900	21.8200	21.3287
A10	0.1000	0.1000	0.3200	0.1220	0.1000	0.1000	0.1000
Weight (lb)	5,127.58	5,112.62	5,107.32	5,095.64	5,091.50	5,080.21	5,062.30
Max stress (ksi)	20.3549	22.9369	20.3959	18.5255	21.1915	24.0675	24.9745
Max displacement (in)	1.9823	1.99999	1.99998	**2.0137**	1.9998	1.9999	1.999998

degree of the maximum violation of the constraints is applied to the objective function. In the death penalty approach, infeasible designs are ignored for the calculation of *Pbest* or *Gbest*, which is equivalent to applying a very severe penalty to every infeasible design. In the redirection approach, infeasible designs are redirected closer to the feasibility boundary by resetting to zero the velocity vector $\boldsymbol{v}^j(t)$ for a particle j with violated constraints at iteration t.

The PSO parameters of the previous study are used, with the nonlinear weight update rule. The redirection constraint handling technique failed to produce good quality of results, because it converged to infeasible solutions far from the feasibility boundary for a number of runs, while for other runs it converged to feasible solutions far from the optimum. For this reason the two other constraint handling techniques are compared which produce always feasible optimum designs and good quality of results. The convergence history of these two methods is depicted in Figure 5.

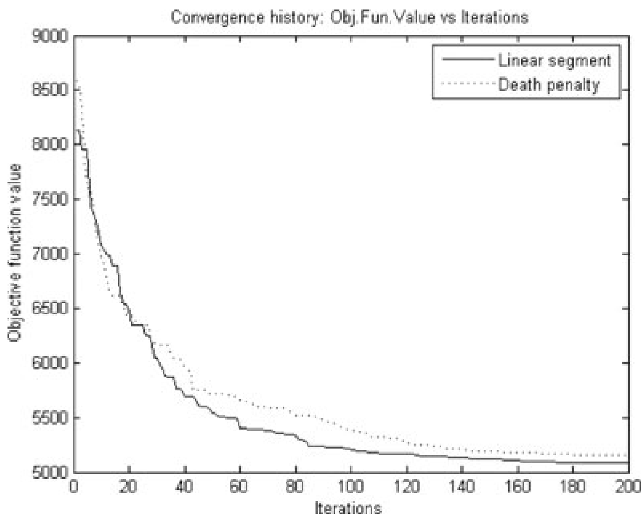

Fig. 5. Example 1. Convergence history for the two PSO constraint handling techniques.

It can be seen that both methods converge to the optimum, while the convergence rate of the proposed constraint handling method is better. This is due to the fact that the proposed method takes into account the infeasible designs as well, which is beneficial for the convergence behavior of the optimization algorithm (Michalewicz, 1995). Figure 6 depicts the ratio of the feasible particles in the population, as a percentage of *NP*, throughout the PSO iterations, for the linear segment penalty function approach. It can be seen that the ratio varies widely with the iterations and that the algorithm always tries to improve the ratio once it reaches lower values of 10%–30%. At the end of the optimization process, the value of the ratio improves, reaching 60%.

7.1.2 The hybrid PSO-SQP method. For the hybrid PSO-SQP scheme a relaxed termination criterion is applied for the PSO before SQP takes over the search for the optimum. The PSO is mainly used to explore the design space, detect the neighborhood of the global optimum and provide a good starting design point for the SQP phase.

First we apply the SQP optimizer, for various initial designs. Four initial designs have been selected, corresponding to design variables values 35, 25, 15, and 5 for

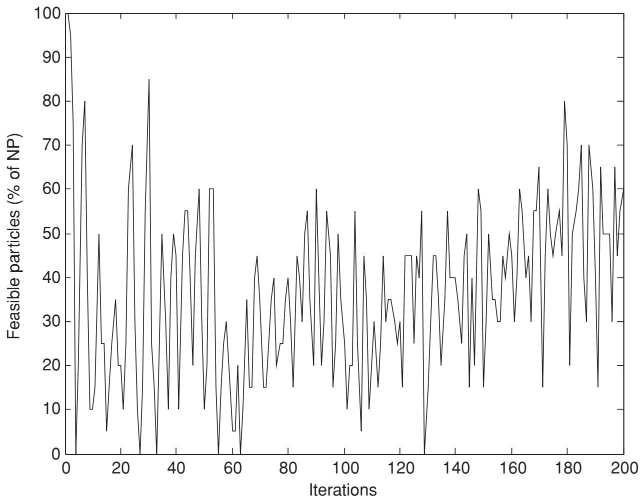

Fig. 6. Example 1. Ratio of feasible particles in the population.

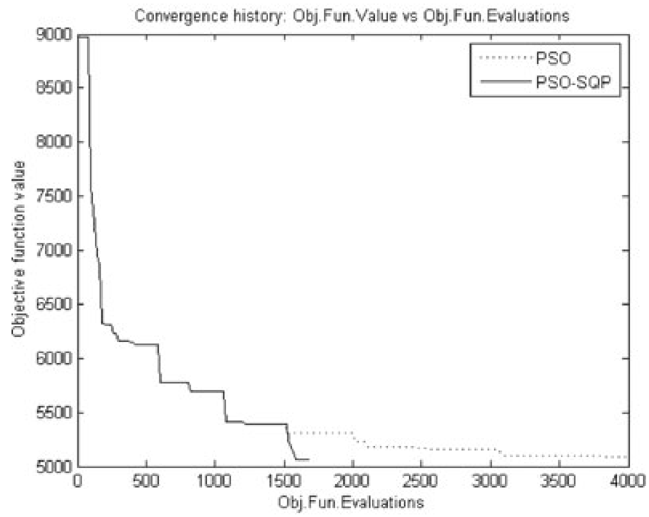

Fig. 7. Example 1. Convergence history for the hybrid PSO-SQP scheme.

every dimension of the problem, resulting to the objective function values given in Table 6.

It can be seen that the SQP method converges to suboptimal design points, as a good model initialization is always required for SQP to produce good results.

Next, we implement the PSO-SQP scheme. If the relative improvement of the objective function over the last $k_f = 15$ iterations of the PSO optimizer is less or equal to $f_m = 10^{-4}$, the PSO phase is terminated and subsequently the SQP starts from the best estimate of the PSO. The SQP termination criterion is connected to the first-order optimality measure for constrained optimization, in terms of the infinite norm. If the magnitude of directional derivative in the search direction is less than a tolerance value of 10^{-5} and there is no constraint violation, convergence has been achieved.

The convergence history of the hybrid PSO-SQP scheme, compared to the basic PSO scheme is depicted in Figure 7 in terms of objective function value and objective function evaluations. In the hybrid scheme, the PSO needs, 1,520 objective function evaluations to reach an objective function value of 5,395.43 and SQP needs another 169 function evaluations to converge to an optimum value of 5,060.85. The total number of objective function evaluations for the hybrid scheme is 1,689. The best design and the values of the constraints obtained by the PSO-SQP method are given in the 4th column of Table 8. It can be seen that for the optimum design, all constraints have been met and are active.

7.1.3 Comparison with results from the literature. For this specific benchmark problem, various results from the literature can be found (Perez and Behdinan, 2007a). In Tables 7 and 8, the objective function value and the constraints values are calculated for every proposed optimum design. Nonfeasible, violated constraints, in mathematical terms, are marked in bold, even where there is a very small violation.

It is clear from Tables 7 and 8 that the best feasible optimum designs are the ones found with the proposed hybrid PSO-SQP and PSO algorithms, because any better design in terms of objective function value violated at least one of the problem constraints.

7.2 Example 2. The 25 bar space truss

The second test example is a 25-member space truss. The structure is depicted in Figure 8. Variations of this test example can be found in the literature (Zhou and Rozvany, 1993; Perez and Behdinan, 2007a). The problem described below is the one described in Zhou and Rozvany (1993), as in Perez and Behdinan (2007a) the load cases and the stress constraints are different, leading to different, noncomparable results. The structural dimensions and nodal coordinates can be found in Zhou and Rozvany (1993).

The structural characteristics are the following: Modulus of Elasticity $E = 10{,}000$ ksi, material weight $\rho = 0.1$ lb/in^3. The structural members are divided into eight groups. The design variables are the cross section areas of each member group in the range [0.01, 5] (in^2). The 8 design variable groups together with the constraints imposed on stresses for each group are presented in Table 9. The two load cases can be seen in Tables 10 and 11.

The maximum allowable displacement in the $\pm x$, $\pm y$, and $\pm z$ directions for each node is $d_{\max} = 0.35$ in. Two

Table 8
Example 1. Optimum designs from the literature (b)

Design variable (in^2)	*(Haug and Arora, 1979)*	*(Haftka and Gürdal, 1992)*	*The proposed PSO-SQP*	*(Adeli and Kamal, 1991)*	*(Perez and Behdinan, 2007b)*	*(El-Sayed and Jang, 1994)*	*(Galante, 1992)*	*(Memari and Fuladgar, 1994)*
A1	30.0300	30.5200	30.5218	31.2800	33.5000	32.9700	30.4400	30.5610
A2	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
A3	23.2740	23.2000	23.1999	24.6500	22.7660	22.7990	21.7900	27.9460
A4	15.2860	15.2200	15.2229	15.3900	14.4170	14.1460	14.2600	13.6190
A5	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
A6	0.5570	0.5510	0.5514	0.1000	0.1000	0.7390	0.4510	0.1000
A7	7.4680	7.4570	7.4572	7.9000	7.5340	6.3810	7.6280	7.9070
A8	21.1980	21.0400	21.0364	21.5300	20.4670	20.9120	21.6300	19.3450
A9	21.6180	21.5300	21.5285	19.0700	20.3920	20.9780	21.3600	19.2730
A10	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
Weight (lb)	5,061.63	5,060.93	5,060.85	5,052.63	5,024.25	5,013.39	4,999.22	4,981.06
Max stress (ksi)	24.9206	**25.0027**	25.0000^{-}	23.0690	**25.0171**	**31.2885**	**25.0867**	20.5999
Max displacement (in)	**2.000004**	1.99996	2.0000^{-}	**2.0195**	**2.0389**	**2.0131**	**2.0280**	**2.0605**

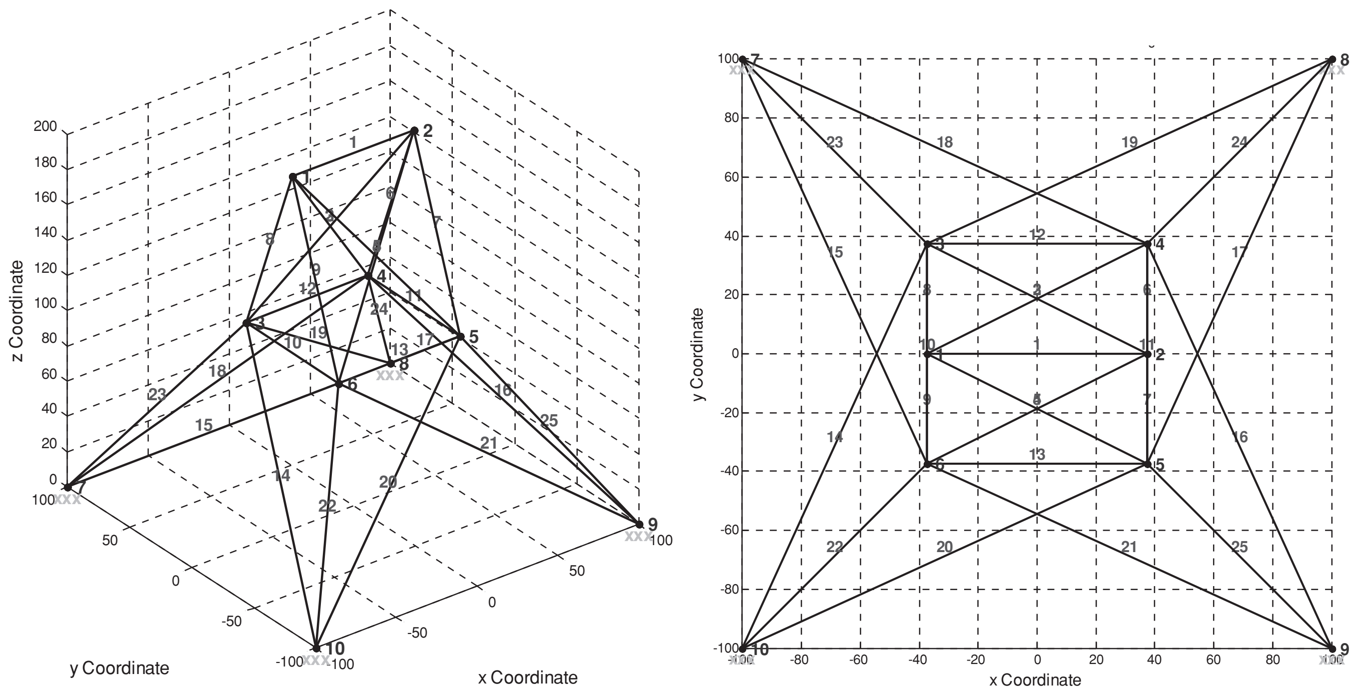

Fig. 8. Example 2. 3-D view and top view of the 25 bar space truss.

load cases have been considered. The nodal loads for each load case are presented in Tables 15 and 16. The objective to minimize is the weight of the structure under the constraints described above for both load cases simultaneously.

The influence of the inertia update rule on the optimization process of the PSO schemes is investigated first. Three PSO schemes are considered, as in the previous example. The basic PSO used is shown in Table 12.

As two load cases are considered, the number of finite element analyses needed for each iteration is $2 \cdot NP = 30$, while the number of objective function evaluations for each iteration is NP. Ten PSO optimization runs are performed for each of the three cases. The results obtained for 200 PSO iterations are reported in Table 13

Table 9
Example 2. Design variable groups and allowable stresses

Design variable	*Member*	*Allowable tension stress (ksi)*	*Allowable compression stress (ksi)*
1	1	40	−35.092
2	2–5	40	−11.590
3	6–9	40	−17.305
4	10,11	40	−35.092
5	12,13	40	−35.092
6	14–17	40	−6.759
7	18–21	40	−6.759
8	22–25	40	−11.082

Table 10
Example 2. Nodal loads—first load case

Node	F_x *(kip)*	F_y *(kip)*	F_z *(kip)*
1	1	10	−5
2	0	10	−5
3	0.5	0	0
6	0.5	0	0

Table 11
Example 2. Nodal loads—second load case

Node	F_x *(kip)*	F_y *(kip)*	F_z *(kip)*
1	0	20	−5
2	0	−20	−5

Table 12
Example 2. PSO parameters used for the inertia update rule check

Symbol	*Value*	*Symbol*	*Value*
NP	15	$\boldsymbol{v}^{\max}$	$v_i^{\max} = 2.5$ for all dimensions (in^2)
n	8	c_1, c_2	$c_1 = c_2 = 2$
w	$w_{\max} = 0.95$ $w_{\min} = 0.5$	$t_{\max}$	200
$\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$	$x_i^{\mathrm{L}} = 0.01,\ x_i^{\mathrm{U}} = 5$ for all dimensions (in^2)		

Table 13
Example 2. Statistical results for 10 PSO runs after 200 iterations

Result	*Fixed rule* ($w = w_{\max}$)	*Linear rule*	*Nonlinear cubic rule* ($a_w = 1.3$)
Best (lb)	599.79	547.78	545.45
Worst (lb)	679.46	604.92	558.97
Average (lb)	627.35	557.14	549.08

Table 14
Example 2. Statistical results for 10 PSO runs

Result	*Fixed rule* ($w = w_{\max}$)	*Linear rule*	*Nonlinear rule* ($a_w = 1.3$)
Average number of iterations	107	144	175
Best (lb)	581.72	576.80	546.12
Worst (lb)	697.88	692.44	694.15
Average (lb)	659.60	633.97	576.16

Table 15
Example 2. Convergence behavior of SQP

Starting point (in^2)	*Iterations*	*Obj. function evaluations*	*Obj. function value (lb)*
“5”	34	396	825.991
“3.5”	41	514	825.991
“2”	18	308	653.357
“0.5”	100	1,204	No convergence

Table 16
Example 2. PSO results for the second test example

Design variable (in^2)	*(Zhou and Rozvany, 1993)*	*The proposed PSO*	*The proposed PSO-SQP*
A1	0.0100	0.01000	0.01000
A2	1.9870	2.0363	2.04300
A3	2.9935	3.1216	3.00239
A4	0.0100	0.01000	0.01000
A5	0.0100	0.01000	0.01000
A6	0.6840	0.6740	0.68337
A7	1.6769	1.5771	1.62296
A8	2.6621	2.6657	2.67194
Weight (lb)	**545.163**	**545.45**	**545.037**

which shows the objective function values obtained for the best run, the worst run, and the average of the 10 runs for each case.

Figure 9 depicts the convergence history for the three cases, in terms of the average result over 10 optimization runs. Furthermore, the performance of the algorithm is studied with the convergence criterion $f_m = 10^{-6}$ connected to the relative improvement of the objective function over the last $k_f = 30$ iterations. The results reported in Table 14 include the average number of iterations needed for convergence and the objective function values obtained for the best run, the worst run, and the average of the 10 runs. The nonlinear rule outperforms the other two rules in terms of the best result, the worst result, and the average result for 10 runs, due to its smoother convergence characteristics.

Table 17
Example 2. Feasibility of the optimum design

Constraints	*Allowable value*	*Zhou and Rozvany (1993)* *Value*	*Active*	*The proposed PSO* *Value*	*Active*	*The proposed PSO-SQP* *Value*	*Active*
Max stress (tensile) (ksi)	40	6.9846	○	7.3735	○	7.1580	○
Max nodal displacement (in)	0.35	**0.3500012**	•	0.3499	•	0.3500^{-}	•
Min stress for groups 1, 4, 5 (max. compressive) (ksi)	−35.092	−5.2989	○	−5.3741	○	−5.4084	○
Min stress for group 2 (max. compressive) (ksi)	−11.590	−6.9382	○	6.8239	○	−6.8192	○
Min stress for group 3 (max. compressive) (ksi)	−17.305	−4.8531	○	−4.6253	○	−4.7991	○
Min stress for groups 6, 7 (max. compressive) (ksi)	−6.759	−5.3200	○	−5.6338	○	−5.4665	○
Min stress for group 8 (max. compressive) (ksi)	−11.082	−4.0980	○	−4.1084	○	−4.1037	○

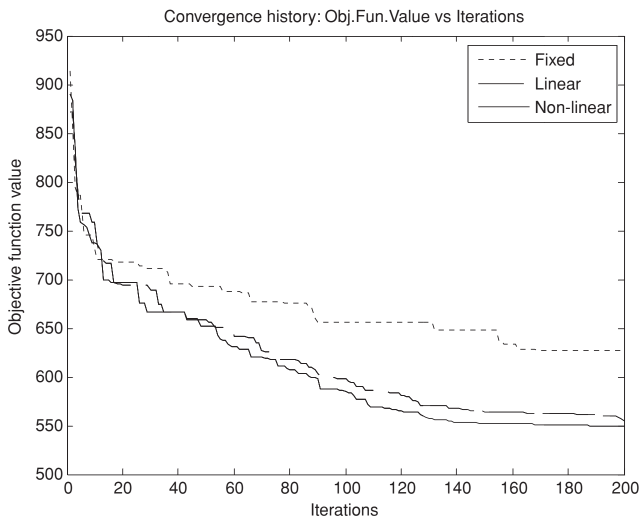

Fig. 9. Example 2. Convergence history for the three PSO schemes.

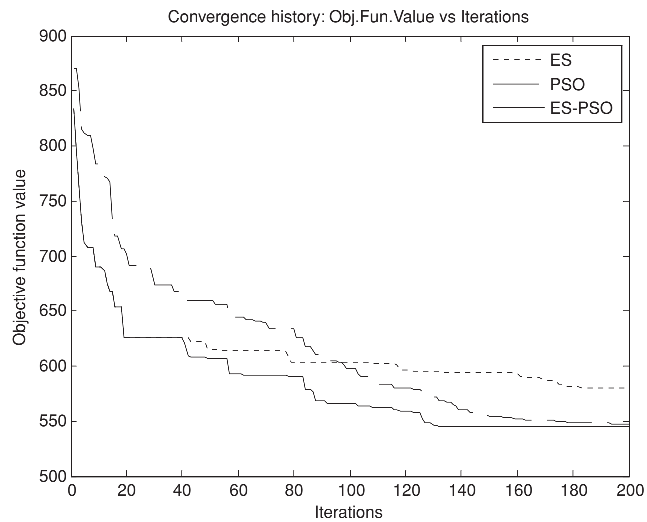

Fig. 11. Example 2. Convergence history for the combined ES-PSO (b).

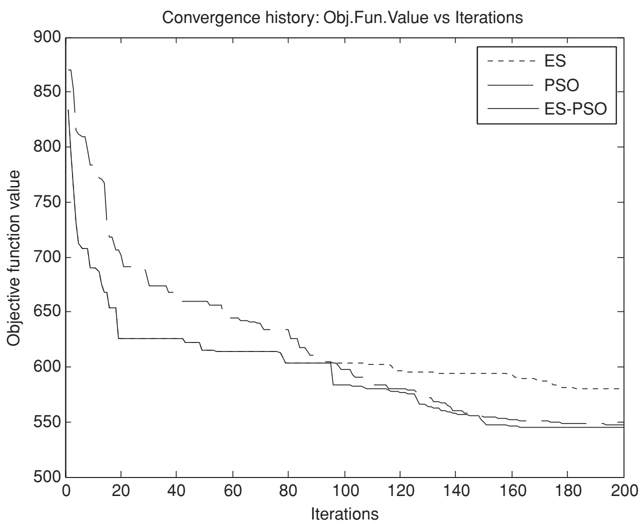

Fig. 10. Example 2. Convergence history for the combined ES-PSO (a).

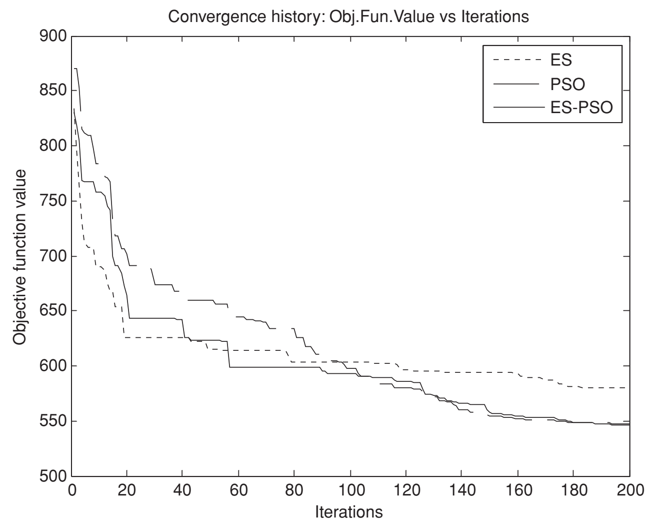

Fig. 12. Example 2. Convergence history for the combined ES-PSO (c).

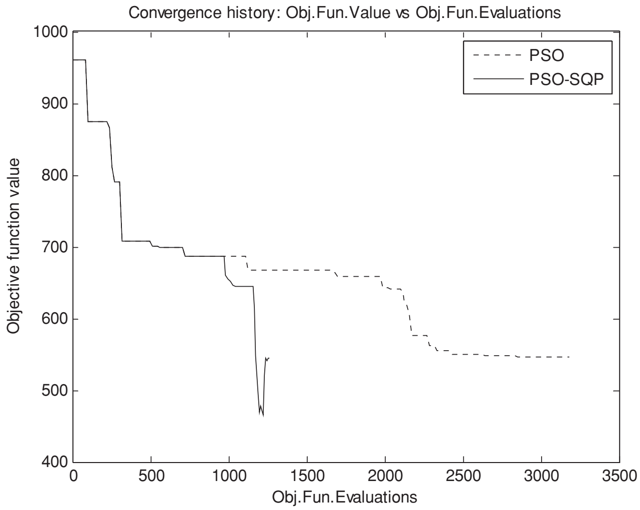

Fig. 13. Example 2. Convergence history for the hybrid PSO-SQP.

7.2.1 Comparison of PSO with Evolution Strategies (ES). To investigate the performance of the proposed PSO algorithm with respect to established Evolutionary Programming algorithms, we consider one of the most efficient optimization algorithms, namely the Evolution Strategies (ES), for comparison. The ES version implemented is a $(10 + 15)$ ES version with 10 parents and 15 offspring and the latest version of a series of improvements developed by the senior author and his associates (Papadrakakis et al., 2001). The PSO scheme is implemented with 15 individuals. Ten independent ES runs are also performed. For both methods, 30 finite element analyses are needed for every iteration (or generation). A given number of iterations (200) is adopted as the termination criterion. Figure 10 shows the convergence history of the two optimization methods, where the horizontal axis represents the PSO iterations or the ES generations. It can be seen that the ES performs better than the PSO during the first 95 iterations, however its convergence rate deteriorates substantially after that point. On the contrary, PSO continues to converge in a uniform rate until it reaches the lowest value of 548.30 (in average) for the objective function.

7.2.2 A combination of ES and PSO. Taking into consideration the results of the previous study, where the ES was shown to perform better than the PSO during the first iterations while the PSO performed better at the end of the optimization process, we investigated a combined ES-PSO scheme, where ES is implemented at the beginning and PSO is implemented after the ES has reached a plateau of no substantial reduction of the objective function for a certain number of generations. The PSO takes over after 95 ES generations with initial conditions the final generation of the ES. All members of the swarm are initialized at the position of the best estimate of the ES and they are given random velocities. Figure 10 shows also the convergence history for the hybrid ES-PSO scheme compared to the ES and PSO schemes. A similar test was conducted with the PSO taking over after 40 ES generations and the results are depicted in Figure 11.

Next we will examine a combined ES-PSO scheme using a termination criterion for the ES based on the improvement of the objective function. If the relative improvement of the objective function over the last $k_f = 30$ ES generations is less or equal to $f_m = 10^{-6}$, then the PSO starts from the best estimate of the ES, given random velocities. Figure 12 shows the history of the combined scheme compared to the previous ES and PSO schemes, where the horizontal axis represents the PSO iterations or the ES generations. It can be seen that the combined method produces the same quality of the final result as the PSO method, while its convergence rate is better over the iterations, while the difference between the ES and ES-PSO schemes until generation 86 is due to the stochastic nature of the ES method.

7.2.3 The hybrid PSO-SQP method. First we apply the SQP alone, for various initial designs. Four initial designs have been selected, namely the ones corresponding to design variables values 5, 3.5, 2, and 0.5 (in^2) for every dimension of the problem. As can be seen in Table 15, the SQP method converges to suboptimal design points, or does not converge at all, as a good model initialization is always required for SQP to produce good results. Next, we implement the PSO-SQP scheme. The termination criterion and other settings are the same as those used for the previous PSO-SQP test example.

The convergence history of the hybrid PSO-SQP scheme, compared to the simple PSO scheme is depicted in Figure 13 in terms of objective function value and objective function evaluations. In the hybrid scheme, the PSO needs 960 function evaluations to reach an objective function value of 687.097, while from that point on, SQP needs 306 objective function evaluations to converge to an optimum value of 545.037. The total number of objective function evaluations for the hybrid scheme is 1,266.

The best design obtained by the hybrid PSO-SQP method, together with the best result of the PSO employing the nonlinear rule and the results of Zhou & Rozvany (1993) are presented in Table 16. In Table 17, the constraints are checked for every optimum design. A constraint is supposed to be active when it is *almost* equal to the threshold value.

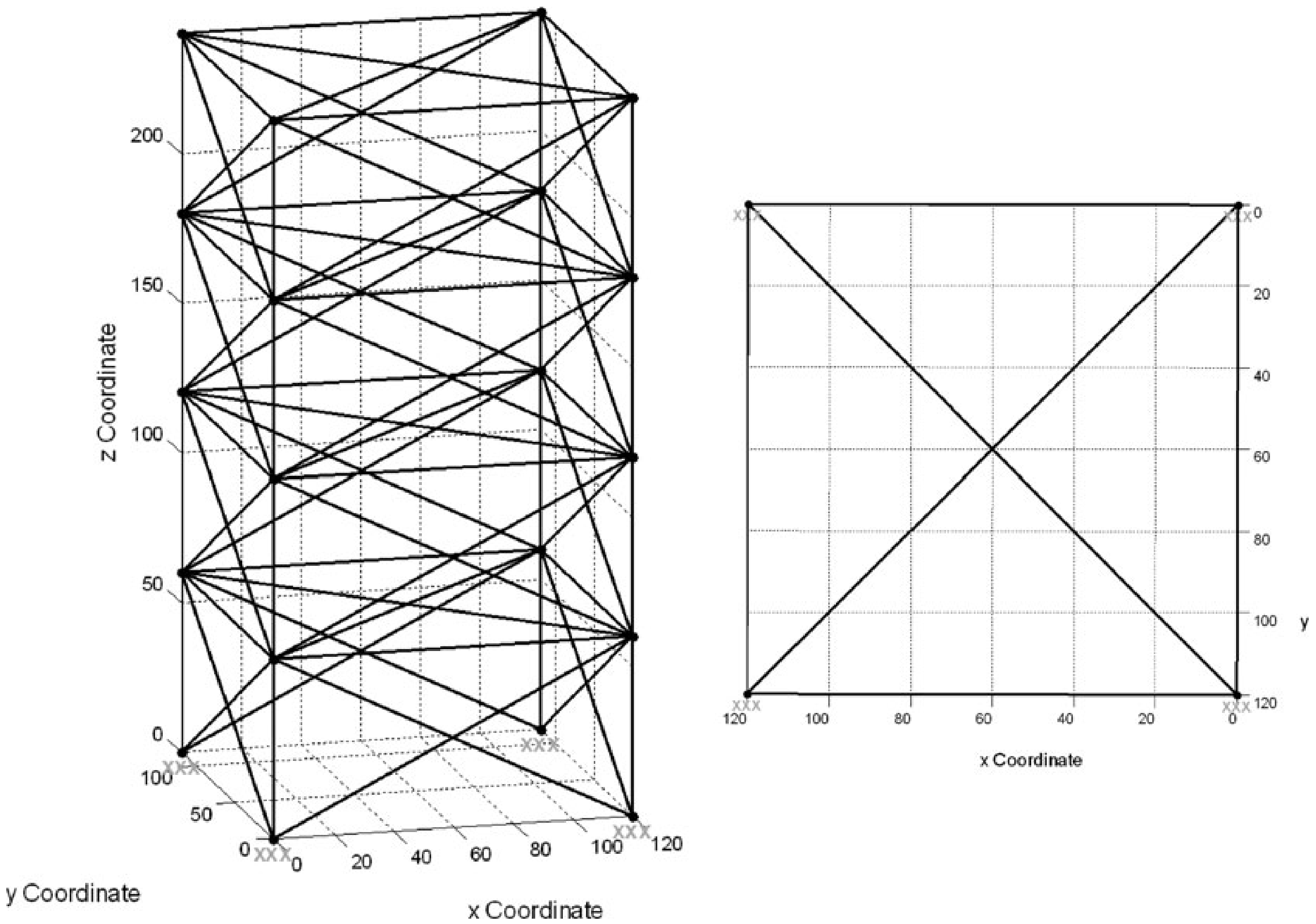

Fig. 14. Example 3. 3-D view and top view of the model.

Table 18
Example 3. PSO parameters used

Symbol	*Value*	*Symbol*	*Value*
NP	40	$\boldsymbol{v}^{\max}$	$v_i^{\max} = 1.5$ for all dimensions (in^2)
n	16	c_1, c_2	$c_1 = c_2 = 2$
w	$w_{\max} = 0.95$ $w_{\min} = 0.5$	$t_{\max}$	2000
$\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}$	$x_i^{\mathrm{L}} = 0.01$ for all dimensions (in^2)	a_w	1.3

It can be seen that the best design, in terms of objective function value, is the one achieved by PSO-SQP. It can also be seen that although the result of Zhou and Rozvany is slightly better than the one obtained with the proposed PSO, there is a slight violation of the maximum nodal displacement constraint, possibly due to rounding errors. For the optimum designs achieved by PSO and PSO-SQP, all constraints have been met while the maximum nodal displacement constraint is active.

7.3 Example 3. The 72 bar space truss

The third test example is a space truss with 72 members, shown in Figure 14. It can be found in the work of Adeli and Kamal (1986), Adeli and Park (1998), Sarma and Adeli (2000), among others. The modulus of Elasticity is $E = 10{,}000$ ksi and the material weight $\rho = 0.1$ lb/in^3. The basis of the structure is a rectangle with a side of 120 in, while the total height is 4×60 in $= 240$ in. The structural members are divided into 16 groups. Two load cases are considered. The first load case consists of a loading $[F_x, F_y, F_z] = [5, 5, -5]$ (kip) applied at node 17. The second load case consists of a loading $[F_x, F_y, F_z] = [0, 0, -5]$ (kip) applied at nodes 17, 18, 19, 20. The design variables are the cross section areas of each member group with a lower limit of 0.01 in^2 and no upper limit. The constraints are imposed on stresses and

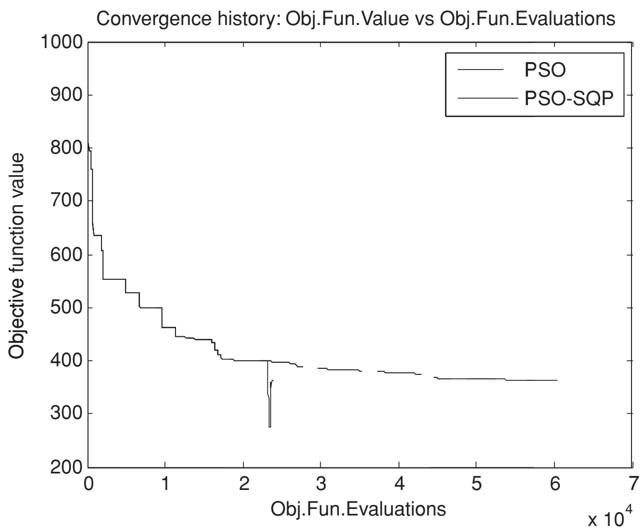

Fig. 15. Example 3. Convergence history for the hybrid PSO-SQP.

displacements. The maximum allowable displacement in the $\pm x$ and $\pm y$ directions for each node is $d_{\max} = 0.25$ in, while the maximum allowable stress (absolute value) is $\sigma_{\text{allow}} = 25$ ksi in tension or compression and the objective is to minimize the weight of the structure under the specified constraints.

7.3.1 Comparison of PSO with GA. This test example is considered to compare the proposed PSO methodology with GAs. Table 18 shows the PSO parameters used in this study. At first, the PSO algorithm itself is implemented with the convergence criterion $f_m = 10^{-6}$ connected to the relative improvement of the objective function over the last $k_f = 150$ iterations. The objective is to compare the convergence properties of the proposed PSO methodology with the corresponding GA results of Sarma and Adeli (2000). The characteristics of the PSO scheme used are shown in Table 3.

The convergence history of PSO is shown in Figure 15 (PSO line). The optimum design achieved is 364.01 lb, obtained in 1,512 iterations. The optimum design achieved is shown in Table 19 and compared with the results obtained by Adeli and Park (1998) and Sarma and Adeli (2000). It is shown that the proposed PSO algorithm converged to a slightly better value of the objective function, in less iterations than the two GA algorithms.

Furthermore, it can be seen in Table 20 that the optimum design of the proposed PSO methodology is truly feasible, confirming that the proposed method yields always feasible optimum designs. Violated constraints are highlighted in bold.

7.3.2 The hybrid PSO-SQP method. Next, the hybrid PSO-SQP scheme is applied. If the relative improvement of the objective function over the last $k_f = 95$ iterations of the PSO optimizer is less or equal to $f_m = 10^{-5}$, the PSO phase is terminated and subsequently the SQP starts from the best estimate of the PSO. The

Table 19
Example 3. Comparison of the optimum design with results from the literature

Design variable (in^2)	*(Adeli and Park, 1998)*	*(Sarma and Adeli, 2000) Simple GA*	*(Sarma and Adeli, 2000) Fuzzy GA*	*The proposed PSO*	*The proposed PSO-SQP*
A1	2.7547	2.1407	1.7321	1.8497	1.8875
A2	0.5102	0.5098	0.5215	0.5217	0.5169
A3	0.0100	0.0538	0.0100	0.0100	0.0100
A4	0.0100	0.0100	0.0129	0.0101	0.0100
A5	1.3696	1.4889	1.3451	1.3041	1.2901
A6	0.5070	0.5507	0.5507	0.5225	0.5170
A7	0.0100	0.0568	0.0100	0.0100	0.0100
A8	0.0100	0.0129	0.0129	0.0100	0.0100
A9	0.4807	0.5653	0.4923	0.5215	0.5211
A10	0.5084	0.5273	0.5449	0.5014	0.5181
A11	0.0100	0.0100	0.0655	0.0119	0.0100
A12	0.0643	0.0655	0.0129	0.1257	0.1140
A13	0.2151	0.1737	0.1778	0.1651	0.1665
A14	0.5179	0.4250	0.5244	0.5442	0.5362
A15	0.4190	0.4367	0.3958	0.4465	0.4457
A16	0.5039	0.6413	0.5952	0.5783	0.5759
Weight (lb)	**376.50**	**372.40**	**364.40**	**364.01**	**363.82**
Iterations	-	**2,776**	**1,758**	**1,512**	-

Table 20
Example 3. Comparison of the constraints of the optimum design with results from the literature

Constraints	*Allowable value*	*(Adeli and Park, 1998)*	*(Sarma and Adeli, 2000) Simple GA*	*(Sarma and Adeli, 2000) Fuzzy GA*	*The proposed PSO*	*The proposed PSO-SQP*
Max Abs. x-displ. (in)	0.25	0.2494	0.2500^{+}	**0.2523**	0.2500^{-}	0.2500^{+}
Max Abs. y-displ. (in)	0.25	0.2494	0.2500^{+}	**0.2523**	0.2500^{-}	0.2500^{+}
Min stress (ksi)	−25	−6.8170	−7.2150	−7.2885	−6.9616	−7.1494
Max stress (ksi)	25	20.7658	24.8790	24.0550	24.9759	25.0000^{+}

SQP termination criterion is the same as the one used for the previous test example. The results of the hybrid method are also reported in Tables 19 and 20. It is shown that the optimum result achieved by the PSO-SQP is slightly better than that of PSO, without violating the constraints. The main advantage of the PSO-SQP method is its fast convergence rate, as shown in Figure 15, where the convergence history of the hybrid PSO-SQP scheme is depicted in terms of objective function value vs. objective function evaluations.

In the hybrid scheme, the PSO needs 579 iterations, or 23,160 objective function evaluations to reach an objective function value of 399.78, while from that point on, SQP needs 715 additional objective function evaluations to converge to an optimum value of 363.82. The total number of objective function evaluations for the hybrid scheme is 23,875 compared to 60,480 for the PSO scheme.

8 CONCLUSIONS

This article introduces optimization algorithms for optimum structural design based on the Particle Swarm algorithm. A nonlinear weight update rule for PSO, an efficient constraint handling technique and a hybrid PSO-SQP scheme for global structural optimization are proposed and evaluated in three benchmark problems. The nonlinear weight update rule for PSO showed better performance than the fixed or the linear rule, especially in cases where a termination criterion connected to the relative improvement of the objective function was used, exhibiting smoother convergence.

The constraint handling technique used in this study, based on a linear segment penalty function, showed excellent performance, because it always led to feasible optimal designs, while taking also advantage of infeasible designs during the optimization procedure.

The proposed hybrid algorithm based on the PSO and SQP is a well-suited optimization tool for solving nonconvex optimization problems in identifying the global optimum from multiple local ones. The numerical results demonstrated the efficiency of the proposed hybrid PSO-SQP algorithm for structural optimization problems. In the standard PSO procedure, the characteristic parameters have to be fine-tuned carefully, based on the experience of the designer, or on trial and error and any other information available for the specific problem at hand. The selection of the PSO parameters plays a significant role in the result of the process in terms of both convergence rate and final optimum design achieved. In general, a bad selection of these parameters can lead to a poor result. By using the proposed hybrid PSO-SQP methodology, the significance of the PSO parameters is substantially alleviated. There is no need of fine-tuning the PSO algorithm for obtaining a high quality final result since the SQP optimization phase can improve drastically the PSO solution and increase significantly the robustness of the optimization scheme. The hybrid optimization algorithm performs better than the standard PSO in terms of efficiency and the convergence rate, while leading to the same, or sometimes even better final optimum result.

ACKNOWLEDGMENTS

This work has been funded by the project PENED 2003. The project is part of the Operational Programme "Competitiveness" (measure 8.3) of the 3rd Community Support Programme and is co-funded, 75% of public expenditure through EC—European Social Fund, 25% of public expenditure through the Greek Ministry of Development—General Secretariat of Research and Technology and through the private sector. The support of the "John Argyris International Centre for Computer Applications in Engineering" is greatly acknowledged.

REFERENCES

Adeli, H. & Cheng, N.-T. (1994), Concurrent genetic algorithms for optimization of large structures, *Journal of Aerospace Engineering*, **7**(3), 276–96.

Adeli, H. & Kamal, O. (1986), Efficient optimization of space trusses, *Computers and Structures*, **24**(3), 501–11.

Adeli, H. & Kamal, O. (1991), Efficient optimization of plane trusses, *Advances in Engineering Software and Workstations*, **13**(3), 116–22.

Adeli, H. & Kumar, S. (1995a), Concurrent structural optimization on a massively parallel supercomputer, *Journal of Structural Engineering (ASCE)*, **121**(11), 1588–97.

Adeli, H. & Kumar, S. (1995b), Distributed genetic algorithms for structural optimization, *Journal of Aerospace Engineering*, **8**(3), 156–63.

Adeli, H. & Park, H. S. (1998), *Neurocomputing for Design Automation*, CRC Press, Inc., Boca Raton, FL, USA.

Angeline, P. J. (1998), *Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences in Evolutionary Programming VII*, Springer Berlin/Heidelberg, 1447, pp. 601–10.

Chen, S., Mei, T., Luo, M. & Yang, X. (2007), Identification of nonlinear system based on a new hybrid gradient-based PSO algorithm, in *Proceedings of the 2007 International Conference on Information Acquisition*, ICIA.

Coelho, L. D. S. & Mariani, V. C. (2007), Particle swarm optimization with quasi-Newton local search for solving economic dispatch problem, in *Conference Proceedings—IEEE International Conference on Systems*, Man and Cybernetics.

Das, S., Koduru, P., Gui, M., Cochran, M., Wareing, A., Welch, S. M. & Babin, B. R. (2006), Adding local search to particle swarm optimization, in *2006 IEEE Congress on Evolutionary Computation*, CEC 2006.

Dimopoulos, G. G. (2007), Mixed-variable engineering optimization based on evolutionary and social metaphors, *Computer Methods in Applied Mechanics and Engineering*, **196**, 803–17.

Dobbs, M. W. & Nelson, R. B. (1976), Application of optimality criteria to automated structural design, *AIAA Journal*, **14**(10), 1436–43.

El-Sayed, M. E. M. & Jang, T. S. (1994), Structural optimization using unconstrained nonlinear goal programming algorithm, *Computers and Structures*, **52**(4), 723–27.

Fourie, P. & Groenwold, A. (2001), The particle swarm optimization in topology optimization, Fourth World Congress of Structural and Multidisciplinary Optimization, Dalian, China.

Fourie, P. C. & Groenwold, A. (2002), The particle swarm optimization algorithm in size and shape optimization, *Structural and Multidisciplinary Optimization*, **23**(4), 259–67.

Galante, M. (1992), Structures optimization by a simple genetic algorithm, *Numerical Methods in Engineering and Applied Sciences*, CIMNE, Barcelona, Spain, 862–70.

Gellatly, R. A. & Berke, L. (1971), Optimal structural design, AFFDL-TR-70-165, Air Force Flight Dynamics Lab, Wright-Patterson Air Force Base, Ohio.

Ghasemi, M. R., Hinton, E. & Wood, R. D. (1997), Optimization of trusses using genetic algorithms for discrete and continuous variables, *Engineering Computations*, **16**(3), 272–303.

Haftka, R. T. & Gürdal, Z. (1992), *Elements of Structural Optimization*, Kluwer Academic Publishers, Dordrecht, the Netherlands.

Hassan, R., Cohanim, B., de Weck, O. & Venter, G. (2005), A comparison of particle swarm optimization and the genetic algorithm, in *Proceedings of the 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Austin, Texas, USA.

Haug, E. J. & Arora, J. S. (1979), *Applied Optimal Design: Mechanical and Structural Systems*, John Wiley & Sons Inc., New York.

Hu, X. & Eberhart, R. (2002), Solving constrained nonlinear optimization problems with particle swarm optimization, in *Sixth World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Florida, USA.

Hu, X., Eberhart, R. & Shi, Y. (2003), Engineering Optimization with particle swarm, in *IEEE Swarm Intelligence Symposium (SIS 2003)*, Indianapolis, Indiana, USA, 53–57.

Izui, K., Nishiwaki, S. & Yoshimura, M. (2005), Swarm optimization algorithms incorporating design sensitivities, in *6th World Congress on Structural and Multidisciplinary Optimization*, Rio de Janeiro, Brazil.

Izui, K., Nishiwaki, S. & Yoshimura, M. (2007), Swarm algorithms for single- and multi-objective optimization problems incorporating sensitivity analysis, *Engineering Optimization*, **39**(8), 981–98

Kamal, C. S. & Adeli, H. (2000), Fuzzy discrete multicriteria cost optimization of steel structures, *Journal of Structural Engineering (ASCE)*, **126**(11), 1339–47.

Kaveh, A. & Shojaee, S. (2007), Optimal design of scissor-link foldable structures using ant colony optimization algorithm, *Computer-Aided Civil and Infrastructure Engineering*, **22**(1), 56–64.

Kaveh, A. & Talatahari, S. (2008), A hybrid particle swarm and ant colony optimization for design of truss structures, *Asian Journal of Civil Engineering (Building and Housing)*, **9**(4), 329–48.

Kennedy, J. & Eberhart, R. (1995), Particle swarm optimization, in *IEEE International Conference on Neural Networks*, Piscataway, NJ, USA, IV, 1942–48.

Koziel, S. & Michalewicz, Z. (1999), Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization, *Evolutionary Computation*, **7**(1), 19–44.

Li, L. J., Huang, Z. B., Liu, F. & Wu, Q. H. (2007), A heuristic particle swarm optimizer for optimization of pin connected structures, *Computers and Structures*, **85**, 340–49.

Memari, A. M. & Fuladgar, A. (1994), Minimum weight design of trusses by BEHSAZ Program, in *Proceedings of 2nd International Conference on Computational Structures Technology*, Athens, Greece.

Michalewicz, Z. (1995), A survey of constraint handling techniques in evolutionary computation methods, in *Proceedings of the 4th Annual Conference on Evolutionary Programming* San Diego, California, MIT Press, Cambridge, MA, USA, 135–55.

Papadrakakis, M. & Lagaros, N. D. (2000), Advances in structural optimization, in J. T. Katsikadelis, D. E. Beskos and E. E. Gdoutos (eds.), *Recent Advances in Mechanics, Honorary Volume for Professor A.N. Kounadis*, NTUA Publics, Athens, pp. 229–40.

Papadrakakis, M., Lagaros, N. D., Tsompanakis, Y. & Plevris, V. (2001), Large scale structural optimization: computational methods and optimization algorithms, *Archives of Computational Methods in Engineering*, **8**(3), 239–301.

Papadrakakis, M., Tsompanakis, Y. & Lagaros, N. D. (1999), Structural shape optimization using evolution strategies, *Engineering Optimization*, **31**(4), 515–40.

Parsopoulos, K. E. & Vrahatis, M. N. (2002), Particle swarm optimization method for constrained optimization problems, in P. Sincak, J. Vascak, V. Kvasnicka, and J. Pospichal (eds.), *Intelligent Technologies—Theory and Applications: New Trends in Intelligent Technologies*, vol. 76, IOS Press, Amsterdam, pp. 214–20.

Perez, R. E. & Behdinan, K. (2007a), Particle swarm approach for structural design optimization, *Computers and Structures*, **85**, 1579–88.

Perez, R. E. & Behdinan, K. (2007b), Particle swarm optimization in structural design, in F. T. S. Chan and M. K. Tiwari (eds.), *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, Itech Education and Publishing, Vienna, Austria, pp. 373–94.

Rizzi, P. (1976), Optimization of multiconstrained structures based on optimality criteria, in *17th Structures, Structural Dynamics and Materials Conference*, King of Prussia, PA, USA.

Rodriguez, A. & Reggia, J. A. (2009), A distributed learning algorithm for particle systems, *Integrated Computer-Aided Engineering*, **16**(1), 1–20.

Sarma, K. C. & Adeli, H. (2000), Fuzzy genetic algorithm for optimization of steel structures, *Journal of Structural Engineering (ASCE)*, **126**(5), 596–604.

Sarma, K. C. & Adeli, H. (2001), Bi-level parallel genetic algorithms for optimization of large steel structures, *Computer-Aided Civil and Infrastructure Engineering*, **16**(5), 295–304.

Schmit, L. A. & Miura, H. (1976), A new structural analysis/synthesis capability-ACCESS 1, *AIAA Journal*, **14**(5), 661–71.

Schmit, L. A. J. & Farshi, B. (1974), Some approximation concepts for structural synthesis, *AIAA Journal*, **12**(5), 692–99.

Shi, Y. & Eberhart, R. (1998), A modified particle swarm optimizer, in *IEEE World Congress on Computational Intelligence*, Anchorage, AK, USA, 69–73.

Soegiarso, R. & Adeli, H. (1997), Optimization of large space frame steel structures, *Engineering Journal (AISC)*, **34**(2), 54–60.

Soegiarso, R. & Adeli, H. (1998), Parallel-vector algorithm for optimization of large steel structures on a shared-memory machine, *Computer-Aided Civil and Infrastructure Engineering*, **13**(3), 207–17.

Venter, G. & Sobieszczanski-Sobieski, J. (2004), Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization, *Structural and Multidisciplinary Optimization*, **26**, 121–31.

Victoire, T. A. A. & Jeyakumar, A. E. (2004), Hybrid PSO-SQP for economic dispatch with valve-point effect, *Electric Power Systems Research*, **71**(1), 51–59.

Vitins, B. J. & Axhausen, K. W. (2009), Optimization of large transport networks using the ant colony heuristic, *Computer-Aided Civil and Infrastructure Engineering*, **24**(1), 1–14.

Yang, Z., Yu, B. & Cheng, C. (2007), A parallel ant colony algorithm for bus network optimization, *Computer-Aided Civil and Infrastructure Engineering*, **22**(1), 44–55.

Yeniay, Ö. (2005), Penalty function methods for constraint optimization with genetic algorithms, *Mathematical and Computational Applications*, **10**(1), 45–56.

Zhang, J.-R., Zhang, J., Lok, T.-M. & Lyu, M. R. (2007), A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, *Applied Mathematics and Computation*, **185**, 1026–37.

Zhou, M. & Rozvany, G. I. N. (1993), DCOC: an optimality criteria method for large systems. Part II: algorithm, *Structural Optimization*, **6**, 250–62.