8th World Congress on Structural and Multidisciplinary Optimization
June 1 - 5, 2009, Lisbon, Portugal

A Hybrid Particle Swarm - SQP Scheme for the Optimum Design of Truss Structures

Vagelis Plevris, Manolis Papadrakakis

Institute of Structural Analysis & Seismic Research, National Technical University of Athens, Athens, Greece
Email: vplevris@central.ntua.gr, mpapadra@central.ntua.gr

1. Abstract

The Particle Swarm Optimization (PSO) method 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. It is an instance of a successful application of the philosophy of bounded rationality and decentralized decision-making for solving optimization problems. The method has been given considerable attention in recent years among the optimization research community. Compared to other optimization methods it is considered efficient in terms of number of function evaluations as well as robust since it usually leads to better or the same quality of results. However, 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, the gradient descending method can achieve faster convergence speed around global optimum and, at the same time, the convergence accuracy can be higher. This paper presents the implementation of an enhanced PSO algorithm combined with a gradient-based quasi-Newton Sequential Quadratic Programming (SQP) method for handling structural truss 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 the convergence towards the global optimum. A non-linear 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 a benchmark structural optimization problem.

2. Keywords: Swarm intelligence, particle swarm, PSO, SQP, optimization, hybrid methods, truss

3. 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 non-gradient probabilistic-based search algorithms, for addressing global non-convex optimization problems. Many non-gradient methods have been inspired by natural phenomena. The inclination of researchers towards the implementation of methodologies inspired by nature in solving engineering problems has led to the invention of many important algorithms 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. One of these methods is the Particle Swarm Optimization (PSO) method that 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. It was first proposed by Kennedy and Eberhart [1] as a population-based optimization method built on the premise that social sharing of information among the individuals can provide an evolutionary advantage.
A number of advantages over other algorithms make PSO a prospective candidate to be used in structural optimization problems. It can handle non-linear, non-convex design spaces with discontinuities. Compared to other optimization methods it is considered efficient in terms of number of function evaluations as well as robust since 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 easily 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. 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 [2, 3] as well as topology optimization [4]. Perez and Behdinan [5, 6] implemented the PSO algorithm for constrained structural optimization of plane and space truss structures while Li et al [7] 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 evolutionary algorithms with mathematical optimizers have been proposed in the past. Papadrakakis and Lagaros implemented a hybrid GA-MP scheme for structural sizing optimization [8], while Papadrakakis et al. proposed a hybrid ES-SQP scheme for structural shape optimization [9] with very satisfactory results. Hybrid PSO methods have also been proposed recently. Kaveh and Talatahari [10] implemented a hybrid PSO and ant colony optimization for the design of truss structures. Izui et al. [11, 12] combined a PSO scheme with gradients where the members of the swarm were divided into Sequential Linear Programming (SLP) and PSO individuals. Dimopoulos [13] proposed a hybrid GA-PSO scheme for optimizing mathematical functions and optimally designing a welded beam and a pressure vessel for minimum cost, while Jing-Ru Zhang et al. [14] proposed a hybrid particle swarm optimization – back-propagation algorithm for feedforward neural network training.

In this work, a novel hybrid algorithm combining the PSO algorithm with a gradient-based quasi-Newton Sequential Quadratic Programming (SQP) algorithm, referred to as PSO–SQP is proposed to optimize 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 non-linear cubic weight update rule and a simple yet effective constraint handling technique.

4. Formulation of the structural optimization problem

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

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

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 $\boldsymbol{x}^{\mathbf{L}}$, $\boldsymbol{x}^{\mathbf{U}}$ are two vectors of length n containing the lower and upper bounds of the design variables, respectively.

5. 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. 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.

5.1 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:

$$\boldsymbol{v}^{\boldsymbol{j}}(t+1)=w\boldsymbol{v}^{\boldsymbol{j}}(t)+c_1r_1(\boldsymbol{x}^{\boldsymbol{Pb,j}}-\boldsymbol{x}^{\boldsymbol{j}}(t))+c_2r_2(\boldsymbol{x}^{\boldsymbol{Gb}}-\boldsymbol{x}^{\boldsymbol{j}}(t)) \tag{2}$$

$$\boldsymbol{x}^{\boldsymbol{j}}(t+1)=\boldsymbol{x}^{\boldsymbol{j}}(t)+\boldsymbol{v}^{\boldsymbol{j}}(t+1) \tag{3}$$

where $\boldsymbol{v}^{\boldsymbol{j}}(t)$ denotes the velocity vector of particle j at time t, $\boldsymbol{x}^{\boldsymbol{j}}(t)$ represents the position vector of particle j at time t, vector $\boldsymbol{x}^{\boldsymbol{Pb,j}}$ 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 $\boldsymbol{x}^{\boldsymbol{Gb}}$ 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 r_1 and r_2 are two random numbers with uniform distribution in the interval [0, 1].

In the above formulation, the global best location found by the entire swarm up to the current iteration ($\boldsymbol{x}^{Gb}$) 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.

The term w of Eq. (2) 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 [1], as it was introduced later by Shi and Eberhart [15] 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 6.

Particles' velocities in each dimension i (i=1, …, n) are restricted to a maximum velocity $v^{\max}_i$. The vector $\boldsymbol{v}^{\mathbf{max}}$ 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 Eq. (2) causes the absolute velocity for dimension i to exceed $v^{\max}_i$, then the velocity on that dimension is limited to $\pm v^{\max}_i$. The vector parameter $\boldsymbol{v}^{\mathbf{max}}$ is employed to protect the cohesion of the system, in the process of amplification of the positive feedback.

5.2 Design variables bounds handling

As shown in Eq. (1), constraints also apply in the available space for every design variable x_i, as in vector terms $\boldsymbol{x}^{\mathbf{L}} \leq \boldsymbol{x} \leq \boldsymbol{x}^{\mathbf{U}}$. If, after the velocity update rule of Eq. (2), the position update of Eq. (3) forces a particle to move outside the bounds for a dimension i ($x_i \leq x^L_i$ or $x_i \geq x^U_i$), then the design variable x_i is reset to the closest bound ($x_i = x^L_i$ or $x_i = x^U_i$). In order to avoid considering any points outside the specified design space, the corresponding coefficient v_i of the velocity vector $\boldsymbol{v}$ is reset to zero, to be used for the next iteration.

5.3 Main PSO parameters

The basic PSO has only a few parameters to adjust. These parameters are the number of particles NP (a typical range is 10 – 40), the dimension of the particles n (determined by the problem to be optimized), the inertia weight w (usually set to a value less than 1), the vectors containing the lower and upper bounds of the n design variables, $\boldsymbol{x}^{\mathbf{L}}$, $\boldsymbol{x}^{\mathbf{U}}$, respectively, the vector containing the maximum allowable velocity for each dimension during one iteration $\boldsymbol{v}^{\mathbf{max}}$, and the cognitive and social parameters, c_1 and c_2, respectively. Usually $c_1 = c_2 = 2$. Other values can also be used, provided that $0 < c_1 + c_2 < 4$, as suggested in [6].

5.4 Convergence criteria

Due to the repeated process of the PSO search, convergence criteria have to be applied for the termination of the optimization procedure. In the present study, together with the maximum number of iterations $t_{\max}$, 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:

$$\frac{Gbest_{t-k_f+1} - Gbest_t}{Gbest_{t-k_f+1}} \leq f_m \tag{4}$$

6. 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 it much capable of dealing with more demanding constrained optimization problems such as those customary present in optimum design of engineering structures.

6.1 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. In order to accomplish the above strategy, Shi and

Eberhart [15] proposed a time-dependent value of the inertia weight. A commonly used inertia update rule is the linearly-decreasing, calculated by the formula:

$$w_{t+1} = w_{\max} - \frac{w_{\max} - w_{\min}}{t_{\max}} t \qquad (5)$$

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

In the present work, a new non-linear weight update strategy is adopted: 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, ending point and two intermediate points), as shown in Figure 1.

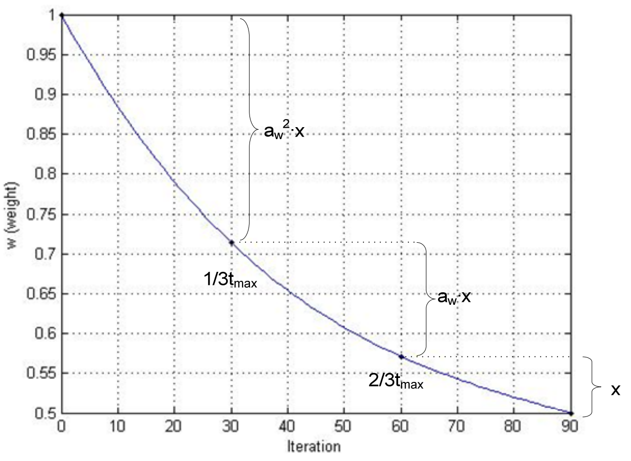

Figure 1: The proposed non-linear weight update rule drawn for $t_{\max} = 90$, $w_{\min} = 0.5$, $w_{\max} = 1$ and $a_w = 2$.

Compared to the linear update rule, the proposed non-linear 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. 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].

6.2 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 non-linear constraints by Evolutionary Algorithms in general. Koziel et al. [16] grouped them into four categories: methods based on preserving feasibility of the solutions; methods based on penalty functions; methods that search for feasibility; other hybrid methods. Very 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 particle using new random numbers r_1 and r_2, until the new position of the particle becomes feasible. 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 [17]. This "death penalty" approach that guarantees the feasibility of the final optimum has the disadvantage that is 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 [18].

Venter and Sobieszczanski-Sobieski [3] 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 re-setting it to zero. Then, the velocity vector $\boldsymbol{v}^j(t+1)$ for next iteration $t+1$ is obtained from Eq. (2) 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 ($\boldsymbol{x}^{Pb,j}$) and the current best point found by the entire swarm ($\boldsymbol{x}^{Gb}$). 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 at 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 [6, 19]. 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.

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

For the typical constraint k in structural optimization of the form $g_k(\boldsymbol{x}) = |q_k(\boldsymbol{x})| - q_{\text{allow},k} \leq 0$, where $q_k(\boldsymbol{x})$ is a response measure (usually stress or displacement) for design $\boldsymbol{x}$ and $q_{\text{allow},k}$ is its maximum allowable absolute value, the penalty function $\Phi_k(\boldsymbol{x})$ for this constraint is defined as:

$$\Phi_k(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} \leq 1 \\ \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} & \text{if } \dfrac{|q_k(\boldsymbol{x})|}{q_{\text{allow},k}} > 1 \end{cases} \tag{6}$$

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:

$$f_p(\boldsymbol{x}) = f(\boldsymbol{x}) \cdot \max\{\Phi_k(\boldsymbol{x})\}, \quad k = 1, \dots, m \tag{7}$$

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 $f_p(\boldsymbol{x})$ 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 a value corresponding to an infeasible design. Indeed, this can happen for an infeasible design if $f(\boldsymbol{x})/Gbest_t < \max\{\Phi_k(\boldsymbol{x})\}$. In order to avoid this undesirable case, $Gbest_t$ is used instead of $f(\boldsymbol{x})$ in Eq. (7), 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.

7. 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 Sequential Quadratic Programming (SQP) method. SQP methods are the standard general purpose mathematical programming algorithms for solving Non-Linear 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 Eq. (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.

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{k=1}^{m} \lambda_k g_k(\boldsymbol{x}) \tag{8}$$

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:

$$\min_{\boldsymbol{p}\in\mathbb{R}^n} \frac{1}{2}\boldsymbol{p}^T \boldsymbol{H}_\ell \boldsymbol{p} + \nabla f(\boldsymbol{x}_\ell)^T \boldsymbol{p}$$
$$\text{Subject to} \tag{9}$$
$$\nabla g_k(\boldsymbol{x}_\ell)^T \boldsymbol{p} + g_k(\boldsymbol{x}_\ell) \leq 0 \quad k = 1,\dots,m$$

Where $\boldsymbol{p}$ is the search direction and $\boldsymbol{H}_\ell$ a positive definite approximation of the Hessian matrix of the Lagrangian function of Eq. (8). In order 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 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 in order to determine the step length parameter a_ℓ. The new design point is then calculated as

$$\boldsymbol{x}_{\ell+1} = \boldsymbol{x}_\ell + a_\ell \cdot \boldsymbol{p}_\ell \tag{10}$$

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.

8. 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 non-convex optimization problems are frequently proposed, but have often exhibited inacceptable 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.

9. Numerical example

The performance of the proposed optimization algorithm is examined in a benchmark test example, a space truss with 25 members and 8 design variables. The PSO scheme used in the study is the fully informed PSO, equipped with the non-linear inertia update rule described in section 6.1 and the constraint handling technique of section 6.2, unless otherwise stated. The structure is depicted in Figure 2. Variations of this test example can be found in the literature [6, 20]. The nodal coordinates are shown in Table 1.

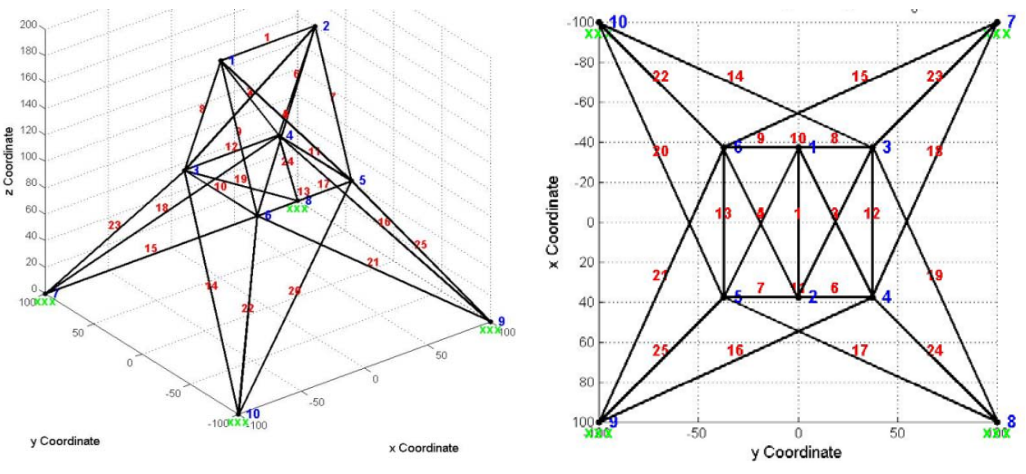

Figure 2: 3D view and top view of the 25 bar space truss

Table 1: Nodal coordinates (in inches)

Node	x	y	z	Node	x	y	z
1	-37.5	0	200	6	-37.5	-37.5	100
2	37.5	0	200	7	-100	100	0
3	-37.5	37.5	100	8	100	100	0
4	37.5	37.5	100	9	100	-100	0
5	37.5	-37.5	100	10	-100	-100	0

The structural characteristics are the following: Modulus of Elasticity $E = 100{,}000$ ksi, material weight $\rho = 0.1$ lb/in^3. The structural members are divided into 8 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 2.

Table 2: Design variable groups and allowable stresses (ksi)

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

The maximum allowable displacement in the $\pm x$, $\pm y$ and $\pm z$ directions for each node is $d_{max} = 0.35$ in. Two load cases have been considered. The nodal loads in kip units in the format (#Node_ID, F_x, F_y, F_z) are for load case 1: (#1, 1, 10, -5), (#2, 0, 10, -5), (#3, 0.5, 0, 0), (#6, 0.5, 0, 0) and for load case 2: (#1, 0, 20, -5), (#2, 0, -20, -5). The objective is to minimize the weight of the structure under the constraints described above for both load cases simultaneously.

9.1 Inertia weight update rule investigation

The influence of the inertia update rule on the optimization process of the PSO schemes will be investigated first. Three PSO schemes are considered. The basic PSO used has the following parameters: $NP = 15$, $n = 8$, $w_{\max} = 0.95$, $w_{\min} = 0.5$, $x^L_i = 0.01$, $x^U_i = 5$ for all dimensions, $\mathrm{v}_{\max,i} = 2.5$ for all dimensions, $c_1 = c_2 = 2$, $t_{\max} = 200$. Since 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 3 which shows the objective function values obtained for the best run, the worst run and the average of the 10 runs for each case.

Table 3: Statistical results for 10 PSO runs after 200 iterations

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

Figure 3 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 4 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 non-linear 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 4: Statistical results for 10 PSO runs

Result	Fixed rule ($w = w_{\max}$)	Linear rule	Non-linear rule ($a_w = 1.3$)
Average no. of iterations	107	144	175
Best	581.72	576.80	546.12
Worst	697.88	692.44	694.15
Average	659.60	633.97	576.16

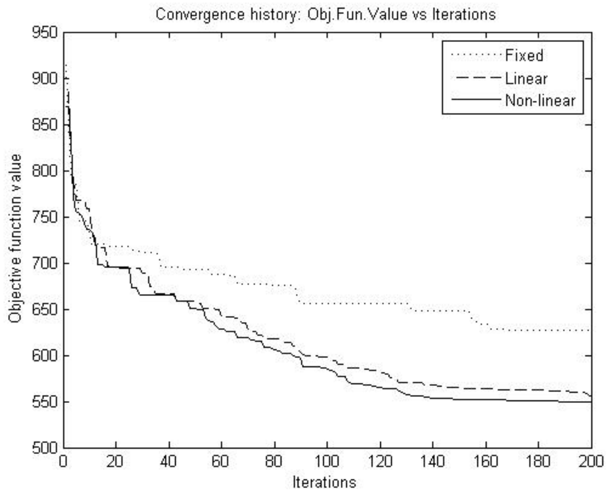

Figure 3: Convergence history for the three PSO schemes

9.2 The hybrid PSO-SQP method

First the SQP is applied 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 for every dimension of the problem. It can be seen that 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. 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.

Table 5: Convergence behavior of SQP

Starting point	Iterations	Obj. fun. evaluations	Obj. function value
“5”	34	396	825.991
“3.5”	41	514	825.991
“2”	18	308	653.357
“0.5”	100	1204	No convergence

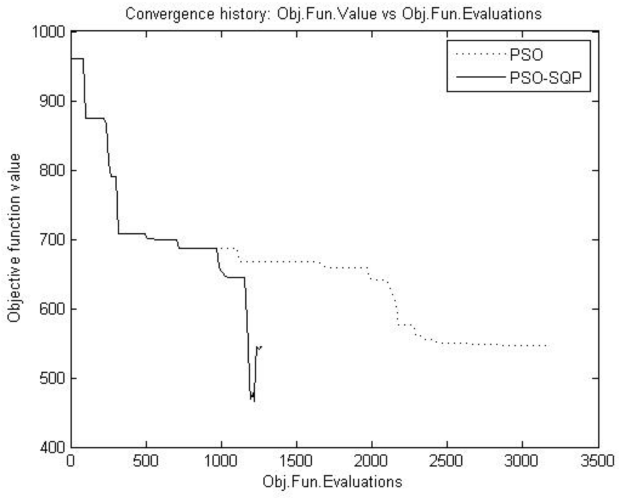

Figure 4: Convergence history for the hybrid PSO-SQP

The convergence history of the hybrid PSO-SQP scheme, compared to the simple PSO scheme is depicted in Figure 4 in terms of objective function value and the 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 1266.
The best design obtained by the hybrid PSO-SQP method, together with the best result of the PSO employing the non-linear rule and the results of Zhou & Rozvany [20] are presented in Table 6.

Table 6: PSO results for the second test example

Design Variable	Zhou & Rozvany (1993)	PSO (This study)	Hybrid PSO-SQP (This study)
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	**545.163**	**545.45**	**545.037**

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 & Rozvany is slightly better than the one obtained with the proposed PSO, there is a slight violation of the maximum nodal displacement constraint. For the optimum designs achieved by PSO and PSO-SQP, all constraints have been met while the maximum nodal displacement constraint is active.

Table 7: Feasibility of the optimum design

		Zhou & Rozvany (1993)		PSO (This study)		Hybrid PSO-SQP (This study)	
Constraints	Allowable value	Value	Active	Value	Active	Value	Active
Max stress (tensile)	40	6.9846	○	7.3735	○	7.1580	○
Max Nodal Displacement	0.35	**0.3500012 (v)**	●	0.3499	●	0.3500^{-}	●

10. Conclusions

This work introduces an optimization algorithm for the optimum structural design of space trusses based on the Particle Swarm algorithm. A non-linear 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 a benchmark problem.
The non-linear 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, since 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 non-convex 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 both convergence rate and final design achieved.

11. 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 private sector.

12. References

[1] J. Kennedy and R. Eberhart, *Particle swarm optimization*, in *IEEE International Conference on Neural Networks*. 1995: Piscataway, NJ, USA. p. 1942–1948.

[2] P.C. Fourie and A. Groenwold, The particle swarm optimization algorithm in size and shape optimization. *Structural and Multidisciplinary Optimization*, 23(4), 259–267, 2002.

[3] G. Venter and J. Sobieszczanski-Sobieski, Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization. *Structural and Multidisciplinary Optimization*, 26, 121-131, 2004.

[4] P. Fourie and A. Groenwold, *The particle swarm optimization in topology optimization*, in *Fourth world congress of structural and multidisciplinary optimization*. 2001: Dalian, China.

[5] R.E. Perez and K. Behdinan, *Particle Swarm Optimization in Structural Design*, in *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, F.T.S. Chan and M.K. Tiwari (Eds.), Itech Education and Publishing: Vienna, Austria. p. 373-394, 2007.

[6] R.E. Perez and K. Behdinan, Particle swarm approach for structural design optimization. *Computers and Structures*, 85, 1579-1588, 2007.

[7] L.J. Li, Z.B. Huang, F. Liu, and Q.H. Wu, A heuristic particle swarm optimizer for optimization of pin connected structures. *Computers and Structures*, 85, 340-349, 2007.

[8] M. Papadrakakis and N.D. Lagaros, *Advances in structural optimization*, in *Recent Advances in Mechanics, Honorary Volume for Professor A.N. Kounadis*, J.T. Katsikadelis, D.E. Beskos, and E.E. Gdoutos (Eds.), NTUA Publics, 2000.

[9] M. Papadrakakis, Y. Tsompanakis, and N.D. Lagaros, Structural Shape Optimization using Evolution Strategies. *Engineering Optimization*, 31(4), 515-540, 1999.

[10] A. Kaveh and S. Talatahari, A hybrid particle swarm and ant colony optimization for design of truss structures. *Asian Journal of Civil Engineering (building and housing)*, 9(4), 329-348, 2008.

[11] K. Izui, S. Nishiwaki, and M. Yoshimura, *Swarm Optimization Algorithms Incorporating Design Sensitivities*, in *6th World Congress on Structural and Multidisciplinary Optimization*. 2005: Rio de Janeiro, Brazil.

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

[13] G.G. Dimopoulos, Mixed-variable engineering optimization based on evolutionary and social metaphors. *Computer Methods in Applied Mechanics and Engineering*, 196, 803-817, 2007.

[14] J.Z. Jing-Ru Zhang, Tat-Ming Lok, Michael R. Lyu, A hybrid PSO–back-propagation algorithm for feedforward NN training. *Applied Mathematics and Computation*, 185, 1026-1037, 2006.

[15] Y. Shi and R. Eberhart, *A modified particle swarm optimizer*, in *IEEE World Congress on Computational Intelligence*. 1998: Anchorage, AK, USA. p. 69-73.

[16] S. Koziel and Z. Michalewicz, Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *IEEE Transaction on Evolutionary Computation*, 7, 19-44, 1999.

[17] X. Hu, R. Eberhart, and Y. Shi, *Engineering Optimization with Particle Swarm*, in *IEEE Swarm Intelligence Symposium (SIS 2003)*. 2003: Indianapolis, Indiana, USA. p. 53-57.

[18] Z. Michalewicz, *A survey of constraint handling techniques in evolutionary computation*, in *4th Annual Conference on Evolutionary Programming*. 1995, MIT Press, Cambridge, MA, USA: San Diego, California. p. 135-155.

[19] K.E. Parsopoulos and M.N. Vrahatis, *Particle swarm optimization method for constrained optimization problems*, in *Intelligent Technologies - Theory and Applications: New Trends in Intelligent Technologies*, P. Sincak, et al. (Eds.), IOS Press. p. 214-220, 2002.

[20] M. Zhou and G.I.N. Rozvany, DCOC: An optimality criteria method for large systems. Part II: Algorithm. *Structural Optimization*, 6, 250-262, 1993.