Structural shape optimization using evolution strategies

ARTICLE *in* ENGINEERING OPTIMIZATION · MARCH 1999

Impact Factor: 1.08 · DOI: 10.1080/03052159908941385

3 AUTHORS:

Papadrakakis Manolis
National Technical University of Athens
249 PUBLICATIONS **2,792** CITATIONS
SEE PROFILE

Yiannis Tsompanakis
Technical University of Crete
65 PUBLICATIONS **357** CITATIONS
SEE PROFILE

Nikos Lagaros
National Technical University of Athens
129 PUBLICATIONS **1,298** CITATIONS
SEE PROFILE

Engineering Optimization
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713641621

STRUCTURAL SHAPE OPTIMIZATION USING EVOLUTION STRATEGIES

Manolis Papadrakakis [a]; Yiannis Tsompanakis [a]; Nikos D. Lagaros [a]
[a] Institute of Structural Analysis and Seismic Research, National Technical University Athens. Athens. Greece

To cite this Article: Papadrakakis, Manolis, Tsompanakis, Yiannis and Lagaros, Nikos D. , 'STRUCTURAL SHAPE OPTIMIZATION USING EVOLUTION STRATEGIES', Engineering Optimization, 31:4, 515 - 540
To link to this article: DOI: 10.1080/03052159908941385
URL: http://dx.doi.org/10.1080/03052159908941385

Eng. Opt., 1999, Vol. 31, pp. 515–540
Reprints available directly from the publisher
Photocopying permitted by license only

STRUCTURAL SHAPE OPTIMIZATION USING EVOLUTION STRATEGIES

MANOLIS PAPADRAKAKIS*, YIANNIS TSOMPANAKIS
and NIKOS D. LAGAROS

Institute of Structural Analysis and Seismic Research, National Technical University Athens, Zografou Campus, Athens 15773, Greece

(Received 5 November 1996; In final form 6 May 1998)

The objective of this paper is to investigate the efficiency of combinatorial optimization methods and in particular algorithms based on evolution strategies, when incorporated into shape optimization problems. Evolution strategy algorithms are used either on a stand-alone basis, or combined with a conventional mathematical programming technique. The numerical tests presented demonstrate the computational advantages of the proposed approach which become more pronounced in large-scale optimization problems and/or parallel computing environment.

Keywords: Structural shape optimization; sequential quadratic programming; evolution strategies; parallel processing

1. INTRODUCTION

In structural shape optimization problems the aim is to improve a given topology by minimizing an objective function subjected to certain constraints [1, 2]. All functions are related to the design variables which are some of the coordinates of the key points in the boundary of the structure. In a gradient-based mathematical programming approach the shape optimization algorithm proceeds with the following steps: (i) a finite element mesh is generated, (ii) displacements and stresses are evaluated, (iii) sensitivities are

*Corresponding author.

computed by perturbing each design variable by a small amount, and (iv) the optimization problem is solved and the new shape of the structure is defined. These steps are repeated until convergence has occurred. The most time-consuming part of this process is devoted to the sensitivity analysis phase which is an important ingredient of all mathematical programming optimization methods [3]. On the other hand the application of combinatorial optimization methods based on probabilistic searching, such as evolution strategies (ESs), do not need gradient information and therefore avoid the need to perform the computationally expensive sensitivity analysis step.

During the last three decades there has been a growing interest in problem solving systems based on algorithms which rely on analogies to natural processes. The best known algorithms in this class include evolutionary programming (EP) [4], genetic algorithms (GAs) [5, 6], evolution strategies (ESs) [7, 8], simulated annealing [9], classifier systems and neural networks [10]. Evolution-based systems maintain a population of potential solutions and use selection processes based on fitness of individuals and recombination operators. ESs like GAs imitate biological evolution and combine the concept of artificial survival of the fittest with evolutionary operators to form a robust search mechanism.

Mathematical programming methods, such as Sequential Quadratic Programming (SQP), 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 to construct an approximate model of the initial problem. These methods present a satisfactory local rate of convergence, but they cannot assure that the global optimum can be found. On the other hand, combinatorial optimization techniques, such as ESs, are in general more robust and present a better global behaviour than the mathematical programming methods. They may suffer, however, from a slow rate of convergence towards the global optimum.

In this work the efficiency of ESs in structural shape optimization problems is investigated. Furthermore, in order to benefit from the advantages of both methodologies a combination of SQP and ESs is also examined in an effort to increase the robustness as well as the computational efficiency of the optimization procedure. The numerical

tests presented demonstrate the computational advantages of the proposed approach which become more pronounced in large-scale and computationally intensive optimization problems as well as in a parallel computing environment.

2. SHAPE OPTIMIZATION

The shape optimization method used in the present study is based on a previous work by Hinton and Sienz [1] for treating two-dimensional problems. It consists of the following essential ingredients: (i) shape generation and control, (ii) mesh generation, (iii) adaptive finite element analysis, (iv) sensitivity analysis, when gradient-based optimization methods are applied, and (v) shape optimization.

Structural optimization problems are characterized by various objective and constraint functions which are generally non-linear functions of the design variables. These functions are usually implicit, discontinuous and non-convex. The mathematical formulation of structural optimization problems with respect to the design variables, the objective and constraint functions depends on the type of the application. However, all optimization problems can be expressed in standard mathematical terms as a non-linear programming problem (NLP) which in general form can be stated as follows:

$$\begin{array}{lll} \min & F(s) & \\ \text{subject to} & h_j(s) \leq 0 & j = 1, \ldots, m \\ & s_i^l \leq s_i \leq s_i^u & i = 1, \ldots, n \end{array} \tag{1}$$

where, s is the vector of design variables, $F(s)$ is the objective function to be minimized, $h_j(s)$ are the behavioural constraints, s_i^l and s_i^u are the lower and the upper bounds on a typical design variable s_i. Equality constraints are usually rarely imposed in this type of problems except in some cases for design variable linking. Whenever they are used they are treated for simplicity as a set of two inequality constraints.

The set of design variables gives a unique definition of a particular design. The selection of design variables is very important in the optimization process. The designer has to decide *a priori* where to

allow design changes and to evaluate how these changes should take place by defining the location of the design variables and the moving directions. The use of the coordinates at key points of the curves that define the shape of the structural model as design variables leads to fewer design variables and more freedom in controlling the shape of the structure.

It is an issue of extreme importance to formulate the optimization problem correctly otherwise unrealistic solutions may be found. Normally, it is necessary to constrain some function of the stresses (*e.g.*, the principal stress) so that it will not exceed a specified value throughout the entire structure. From a practical point of view a finite number of so-called 'stress constraint points' is selected, where the condition is enforceable. These points are either some predefined points within the domain, or some boundary nodes. Also, other type of constraints, like displacement or frequency constraints, can be imposed depending on the type of problem. Usually the constraint functions and their derivatives are normalized in order to improve the performance of the optimizer.

The shape optimization methodology proceeds with the following steps: (i) At the outset of the optimization, the geometry of the structure under investigation has to be defined. The boundaries of the structure are modelled using cubic B-splines which, in turn, are defined by a set of key points. Some of the coordinates of these key points will be the design variables which may or may not be independent of each other. (ii) An automatic mesh generator is used to create a valid and complete finite element model. A finite element analysis, is then carried out and the displacements and stresses are evaluated. In order to increase the accuracy of the analysis an h-type adaptivity analysis may be incorporated in this stage. (iii) If a gradient-based optimizer, like the sequential quadratic programming SQP algorithms, is used then the sensitivities of the constraints and the objective function to small changes of the design variables are computed either with the finite difference, or with the semi-analytical method. (iv) The design variables are optimized. If the convergence criteria for the optimization algorithm are satisfied, then the optimum solution has been found and the process is terminated, else a new geometry is defined and the whole process is repeated from step (ii).

3. GRADIENT-BASED SHAPE OPTIMIZATION

3.1. Sensitivity Analysis

Sensitivity analysis is the most important and time-consuming part of a gradient-based shape optimization procedure. Although sensitivity analysis is mostly mentioned in the context of structural optimization, it has evolved into a research topic of its own. The calculation of the sensitivity coefficients follows the application of a relatively small perturbation to each primary design variable. Several techniques have been developed which can be mainly distinguished by their numerical efficiency and their implementation aspects. The methods for sensitivity analysis can be divided into discrete and variational methods [11]. In the variational approach the starting point is an idealized but continuous structure, such as a beam or a shell. In the discrete approach the derivatives or the sensitivities of the objective and constraint functions, are evaluated using the finite element equations of the discretized structure. The effort required for the computer implementation of the discrete methods is less than implementing the variational methods.

A further classification of the discrete methods is the following [1]: (i) *Global finite difference method*: A full finite element analysis has to be performed for each design variable and the accuracy of the method depends strongly on the value of the perturbation of the design variables. (ii) *Semi-analytical method*: The stiffness matrix of the initial finite element solution is retained during the computation of the sensitivities. This provides an improved efficiency over the finite difference method by a relatively small increase in the algorithmic complexity. The accuracy problem involved with the numerical differentiation can be overcome by using the "exact" semi-analytical method which needs more programming effort than the simple method but it is computationally more efficient. (iii) *Analytical method*: The finite element equations, the objective and constraint functions are differentiated analytically.

The decision on which method to implement depends strongly on the type of problem, the structure of the computer program and the access to the source code. The implementation of analytical and semi-analytical methods is more complex and requires access to the source

code, whereas when a finite difference method is applied the formulation is much simpler and the sensitivity coefficients can be easily evaluated even with general purpose commercial codes. In the present investigation both the global finite difference method and the semi-analytical method have been used.

3.1.1. The Global Finite Difference (GFD) Method

In this method the design sensitivities for the displacements $\partial u/\partial s_k$ and the stresses $\partial\sigma/\partial s_k$, which are needed for the gradients of the constraints, are computed using a forward difference scheme:

$$\partial u/\partial s_k \approx \frac{\Delta u}{\Delta s_k} = \frac{u(s_k + \Delta s_k) - u(s_k)}{\Delta s_k} \tag{2}$$

$$\partial\sigma/\partial s_k \approx \frac{\Delta\sigma}{\Delta s_k} = \frac{\sigma(s_k + \Delta s_k) - \sigma(s_k)}{\Delta s_k} \tag{3}$$

The perturbed displacement vector $u(s_k + \Delta s_k)$ of the finite element equations is evaluated by

$$K(s_k + \Delta s_k)u(s_k + \Delta s_k) = f(s_k + \Delta s_k) \tag{4}$$

and the perturbed stresses $\sigma(s_k + \Delta s_k)$ are computed from

$$\sigma(s_k + \Delta s_k) = DB(s_k + \Delta s_k)u(s_k + \Delta s_k) \tag{5}$$

where D and B are the elasticity and the deformation matrices, respectively.

The GFD scheme is usually sensitive to the accuracy of the computed perturbed displacement vectors which is dependent on the magnitude of the perturbation of the design variables. The magnitude of this perturbation is usually taken between 10^{-3} and 10^{-5}.

3.1.2. The Semi-analytical (SA) Method

The SA method is based on the chain rule differentiation of the finite element equations $Ku = f$:

$$K\frac{\partial u}{\partial s_k} + \frac{\partial K}{\partial s_k}u = \frac{\partial f}{\partial s_k} \tag{6}$$

which when rearranged results in

$$K\frac{\partial u}{\partial s_k} = f_k^* \tag{7}$$

where

$$f_k^* = \frac{\partial f}{\partial s_k} - \frac{\partial K}{\partial s_k}u \tag{8}$$

f_k^* represents a pseudo-load vector. The derivatives of $\partial K/\partial s_k$ and $\partial f/\partial s_k$ are computed for each design variable by recalculating the new values of $K(s_k + \Delta s_k)$ and $f(s_k + \Delta s_k)$ for a small perturbation Δs_k of the design variable s_k. The derivatives of $\partial f/\partial s_k$ are computed using a forward finite difference scheme.

With respect to the differentiation of K the semi-analytical approach can be divided in two methods: The conventional SA and the "exact" SA. In the conventional sensitivity analysis (CSA), the values of the derivatives in (6) are calculated by applying the forward difference approximation scheme

$$\partial K/\partial s_k \approx \frac{\Delta K}{\Delta s_k} = \frac{K(s_k + \Delta s_k) - K(s_k)}{\Delta s_k} \tag{9}$$

In the "exact" semi-analytical (ESA) approach the derivatives $\partial K/\partial s_k$ are computed on the element level as follows [12]

$$\frac{\partial k}{\partial s_k} = \sum_{j=1}^{n} \frac{\partial k}{\partial \alpha_j}\frac{\partial \alpha_j}{\partial s_k} \tag{10}$$

where n is the number of elemental nodal coordinates affected by the perturbation of the design variable s_k and α_j are the nodal coordinates of the element. The ESA method is more accurate and leads the mathematical optimizer to a faster convergence [13]. This approach is used in the present study.

Stress gradients can be calculated by differentiating $\sigma = DBu$ as follows

$$\frac{\partial \sigma}{\partial s_k} = \frac{\partial D}{\partial s_k}Bu + D\frac{\partial B}{\partial s_k}u + DB\frac{\partial u}{\partial s_k} \tag{11}$$

Since the elasticity matrix D is not a function of the design variables then Eq. (11) reduces to

$$\frac{\partial \sigma}{\partial s_k} = D\frac{\partial B}{\partial s_k}u + DB\frac{\partial u}{\partial s_k} \tag{12}$$

In Eq. (12), $\partial u/\partial s_k$ may be computed as indicated in Eq. (2), while the term $\partial B/\partial s_k$ is computed using a forward finite difference scheme. Using the values of $\partial \sigma/\partial s_k$ the sensitivities of different types of stresses (*e.g.*, the principal stresses or the equivalent stresses) can be readily calculated by analytically differentiating their expressions with respect to the shape variables.

3.2. Mathematical Optimization Algorithm

Sequential Quadratic Programming (SQP) methods are the standard general purpose mathematical programming algorithms for solving Non-Linear Programming (NLP) optimization problems [14]. They are also considered to be the most suitable methods for solving structural optimization problems [15–17]. 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. Thus a Quadratic Programming (QP) model (or subproblem) is constructed from the initial NLP problem. A local minimizer is found by solving a sequence of these QP subproblems using a quadratic approximation of the objective function. Each subproblem has the following form:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}p^T HP + g^T p \\ \text{subject to} \quad & Ap + h(s) \leq 0 \\ & \bar{s}_l \leq p \leq \bar{s}_u \end{aligned} \tag{13}$$

where p is the search direction subjected to upper and lower bounds, g is the gradient of the objective function, A is the Jacobian of the constraints, usually the *active* ones only (*i.e.*, those that are either violated, or not far from being violated), $\bar{s}_l = s_l - s$, $\bar{s}_u = s_u - s$ and H is an approximation of the Hessian matrix of the Lagrangian function

$$L(s, \lambda) = F(s) + \lambda h(s) \tag{14}$$

in which λ are the Lagrange multipliers under the non-negativity restriction ($\lambda \geq 0$) for the inequality constraints. 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 are computed during the sensitivity analysis phase.

There are two ways to solve this QP subproblem, either with a primal [18], or a dual [19] formulation. In the present study a primal algorithm is employed based on an SQP algorithm from the NAG library [20]. The primal algorithm is divided into three phases: (i) the solution of the QP subproblem to obtain the search direction, (ii) the line search along the search direction p, (iii) the update of the Hessian matrix H.

The solution of the QP subproblem is performed in two steps, first by minimizing the sum of the infeasibilities and then by minimizing the quadratic objective function within the feasible region. The latter step is achieved using a reduced Hessian matrix of the active constraints only in order to obtain a good search direction and minimize the computational cost. The method is more efficient when many constraints or bounds are active at the optimum.

Once the direction vector p is found a line search is performed, involving only the nonlinear constraints, in order to produce a "sufficient decrease" to the merit function φ. This merit function is an augmented Lagrangian function of the form [19]

$$\varphi = F(s) - \sum_i \lambda_i (g_i(s) - \gamma_i) + \frac{1}{2} \sum_i \rho_i (g_i(s) - \gamma_i)^2 \qquad (15)$$

where γ_i are the non-negative slack variables of the inequality constraints derived from the solution of the QP subproblem. These slack variables allow the active inequality constraints to be treated as equalities and avoid possible discontinuities. Finally, ρ_i are the penalty parameters which are initially set to zero and in subsequent interations are increased whenever this is necessary in order to control the violation of the constraints and to ensure that the merit function follows a descent path.

Finally a BFGS quasi-Newton update [14] of the approximate Hessian of the Lagrangian function L is implemented, where attention is given to keeping the Hessian matrix positive definite. In order to

incorporate the new curvature information obtained through the last optimization step, the updated Hessian $\tilde{H}$ is defined as a rank-two modification of H:

$$\tilde{H} = H - \frac{1}{w^T H w} H w w^T H + \frac{1}{y^T w} y y^T \tag{16}$$

where w and y denote the change in the design variable vectors s and the gradient vector of the Lagrangian function in Eq. (14), respectively. If the quadratic function is convex then the Hessian is positive definite, or positive semi-definite and the solution obtained will be a global optimum, else if the quadratic function is non-convex then the Hessian is indefinite and if a solution exists it is only a local optimum.

4. EVOLUTION STRATEGIES (ESs)

There are three classes of algorithms that imitate nature by using biological methodologies in order to find the optimum solution of a problem: evolutionary programming (EP), genetic algorithms (GAs) and evolution strategies (ESs). Their main difference is that GAs deal with bit-strings of fixed sizes, ESs with real vectors and evolutionary programming (EP) with finite state automata. GAs basic assumption is that the optimal solution can be found by assembling building blocks, *i.e*, partial pieces of solutions, while ESs and EP simply ensure the emergence of the best solutions. The most visible consequence of this debate deals with the recombination operator, viewed as essential for GAs, as potentially useful for ESs and as possibly harmful for EP. The modern tendencies are more pragmatic since GA users have turned to real number representations when dealing with real numbers, following experimental results or heuristic demonstrations, whereas ESs researchers have included recombination as a standard operator, and designed specific operators for non real-valued problems [21].

Evolution strategies were proposed for parameter optimization problems in the 1970s [7, 8]. Similar to genetic algorithms, ESs imitate biological evolution in nature and have three characteristics that make them differ from other conventional optimization algorithms [22]: (i)

in place of the usual deterministic operators, they use randomized operators: mutation, selection and recombination; (ii) instead of a single design point, they work simultaneously with population of design points in the space of variables; (iii) they can handle continuous, discrete and mixed optimization problems [23, 24]. The second characteristic allows for natural implementation of GAs and ESs on parallel computing environments. The ESs, however, achieve a higher rate of convergence than GAs due to their self-adaptation search mechanism and are considered more efficient for solving real world problem [22].

4.1. ES Algorithms

ESs were initially applied for continuous optimization problems, but recently they have also been implemented in discrete and mixed optimization problems [23, 24]. ESs can be divided into two-membered evolution strategies (2-ESs) and multi-membered evolution strategies (M-ESs).

4.1.1. The Two-member ESs

The initial formulation of evolution strategies was based on a population consisting of one individual only. The two-membered scheme is the minimal concept for an imitation of organic evolution. The two principles of mutation and selection, which Darwin in 1859 recognized to be most important, are taken as rules for variation of the parameters and for recursion of the iteration sequence respectively.

The two-membered ESs for the solution of the optimization problem works in two steps:

Step 1 (mutation) The parent $s_p^{(g)}$ of the generation g produces an offspring $s_o^{(g)}$, whose genotype is slightly different from that of the parent:

$$s_o^{(g)} = s_p^{(g)} + z^{(g)} \tag{17}$$

where $z^{(g)} = [z_1^{(g)}, z_2^{(g)}, \ldots, z_n^{(g)}]^T$ is a random vector.

Step 2 (selection) The selection chooses the best individual between the parent and the offspring to survive:

$$s_p^{(g+1)} = \begin{cases} s_o^{(g)} \text{ if } g_i(s_o^{(g)}) \leq 0 & i = 1, 2, \ldots, l \text{ and } f(s_o^{(g)}) \leq f(s_p^{(g)}) \\ s_p^{(g)} & \text{otherwise} \end{cases} \tag{18}$$

The question is how to choose the random vector $z^{(g)}$ in *Step 1* which is introduced by the mutation operator. Mutation is understood to be random, purposeless events, which occur very rarely. If one interprets them, as is done here, as a sum of many individual events a rational choice is to use a probability distribution according to which small changes occur frequently and large ones only rarely. Two requirements arise together by analogy with natural evolution: (i) the expected mean value ξ_i for a component $z_i^{(g)}$ should be zero; (ii) the variance σ_i^2, the average squared deviation from mean value, should be small.

The probability density function for normally distributed random events is given by

$$p(z_i^{(g)}) = \frac{1}{\sqrt{(2\pi)}\sigma_i} \exp\left(-\frac{(z_i^{(g)} - \xi_i)^2}{2\sigma_i^2} \right) \tag{19}$$

When $\xi_i = 0$ the so-called standard normal distribution is obtained. By analogy with other deterministic search strategies, σ_i can be called step length, in the sense that it represents average values of the length of the random steps.

If the step length is too small the search takes an unnecessarily large number of iterations. On the other hand, if the step length is too large the optimum can only be crudely approached and the search can even get stuck far away from the global optimum. Thus, as in all optimization strategies, the step length control is the most important part of the algorithm after the recursion formula, and it is closely linked to the convergence behaviour.

The standard deviation σ_i which is considered as the step length can be adjusted during the search as follows (Rechenberg's 1/5 success rule [7]): "*The ratio of successful mutations to all mutations should be* 1/5. *If it is greater, increase; if it is less, decrease the standard deviations* σ_i".

According to Schwefel [8], the check should take place every n mutations over the preceding $10n$ mutations, while the increase and decrease factors of the step length should be (1/0.85) and 0.85, respectively. During the search, not only the design variables s_i, but also the parameters, such as the deviations σ_i, will be modified by the random operator mutation which replaces the 1/5 success rule.

4.1.2. Multi-membered ESs

The multi-membered evolution strategies differ from the previous two-membered strategies in the size of the population. In this case a population of μ parents will produce λ offsprings. Thus the two steps are defined as follows:

Step 1 (recombination and mutation) The population of μ parents at g-th generation produces λ offsprings. The genotype of any descendant differs only slightly from that of its parents.

Step 2 (selection) There are two different types of the multi-membered ESs:

$(\mu + \lambda)$-ESs: The best μ individuals are selected from a temporary population of $(\mu + \lambda)$ individuals to form the parents of the next generation.

(μ,λ)-ESs: The μ individuals produces λ offsprings ($\mu < \lambda$) and the selection process defines a new population of μ individuals from the set of λ offsprings only.

In the second type the existence of each individual is limited to one generation. This allows the (μ, λ)-ESs selection to perform better on problems with an optimum moving over time, or on problems where the objective function is noisy.

In *Step 1* for every offspring vector a temporary parent vector $\tilde{s} = [\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_n]^T$ is first built by means of recombination. For continuous problem the following recombination cases can be used:

$$\tilde{s}_i = \begin{cases} s_{\alpha,i} \text{ or } s_{b,i} \text{ randomly} & \text{(A)} \\ 1/2(s_{\alpha,i} + s_{b,i}) & \text{(B)} \\ s_{bj,i} & \text{(C)} \\ s_{\alpha,i} \text{ or } s_{bj,i} \text{ randomly} & \text{(D)} \\ 1/2(s_{\alpha,i} + s_{bj,i}) & \text{(E)} \end{cases} \tag{20}$$

where $\tilde{s}_i$ is the ith component of the temporary parent vector $\tilde{s}$, $s_{a,i}$ and $s_{b,i}$ are the ith components of the vectors s_a and s_b which are two parent vectors randomly chosen from the population. In case C of Eq. (20) $\tilde{s}_i = s_{bj,i}$ means that the ith component of $\tilde{s}$ is chosen randomly from the ith components of all μ parent vectors. From the temporary parent $\tilde{s}$ an offspring can be created in the same way as in two-membered ESs using Eq. (18).

Multi-membered ES termination criteria are the following: (i) when the absolute or relative difference between the best and the worst objective function values is less than a given value ε_1, or when (ii) the mean value of the objective values from all parent vectors in the last $2 * n$ generations has not been improved by less than a given value ε_2.

4.2. ESs in Shape Optimization Problems

So far little effort has been spent in applying probabilistic search methods in shape optimization problems. Usually this type of problems is solved with a mathematical programming algorithm such as the sequential quadratic programming method SQP [15, 19], the generalized reduced gradient method (GRG) [25], the method of moving asymptotes (MMA) [26], which all need gradient information. Since the objective function and the constraints are highly non-linear functions of the design variables the computational effort required for the solution of the optimization problem, most of which is spent in gradient calculations, is usually large.

Thus the use of combinatorial type algorithms appears to be promising even if greater numbers of analyses are needed to reach the optimum. This is due to the fact that since the number of design variables in shape optimization problems is relatively small the number of analyses is limited to a few tens or hundreds. Moreover, these analyses are less computationally expensive than in the case of mathematical programming algorithms as they do not need gradient information. Furthermore, probabilistic methodologies, due to their random search, are considered as global optimization methods because they are capable of finding the global optimum, whereas mathematical programming algorithms may be trapped in local optima. Finally, the natural parallelism inherent in combinatorial

algorithms makes them very attractive for application in parallel computer architectures.

The implementation of ESs in shape optimization is straightforward and follows the same steps as described in Section 2 without performing the sensitivity analysis step. The ES optimization procedure starts with a set of parent vectors. If any of these designs gives an infeasible design then this parent is modified until it becomes feasible. Then the offsprings are generated and are also checked if they are in the feasible region. In every generation the values of the objective function are compared among the parent and the offspring vectors and the worst vectors are rejected, while the remaining ones are considered to be the parent vectors of the new generation. This procedure is repeated until the chosen termination criterion is satisfied. The ES steps can be stated as follows:

1. *Selection step:* selection of s_i $(i = 1, 2, \ldots, \mu)$ parent vectors of the design variables
2. *Analysis step:* solve $K(s_i)u_i = f (i = 1, 2, \ldots, \mu)$
3. *Constraints check:* all parent vectors become feasible
4. *Offspring generation:* generate s_j, $(j = 1, 2, \ldots, \lambda)$ offspring vectors of the design variables
5. *Analysis step:* solve $K(s_j)u_j = f (j = 1, 2, \ldots, \lambda)$
6. *Constraints check:* if satisfied continue, else change s_j and go to *Step 4*
7. *Selection step:* selection of the next generation parents according to $(\mu + \lambda)$ or (μ, λ) selection schemes
8. *Convergence check:* if satisfied stop, else go to *Step 3*.

5. THE HYBRID APPROACH

The main advantage of the SQP optimizer is that it captures very fast the right path of the nearest optimum, irrespective of whether it is a local or a global optimum. After locating the area of this optimum it might oscillate until all constraints are satisfied. Even small constraint violations often slow down the convergence rate of the method. On the other hand ESs are not so sensitive as SQP to small constraint violations but proceed at a slower rate, due to their random search,

and usually need a greater number of analyses. However, these analyses are very fast since they do not require expensive gradient calculations. Furthermore, the absence of strict mathematical rules, which govern the convergence rate of the Evolution strategy method, make ESs less vulnerable to local optima and therefore much more reliable to obtain the global optimum in non-convex optimization methods.

In order to benefit from the advantages of both methodologies a hybrid approach is proposed, which combines the two methods in an effort to increase the robustness and the computational efficiency of the optimization procedure. Two combinations of SQP and ES methodologies are implemented: (i) In the first approach the SQP method is used first, giving a design very close to the optimum, followed by ES in order to accelerate convergence and avoid the oscillations of SQP due to small constraint violations around optimum. The transition from one algorithm to the other is performed when

$$\left|\frac{f_{j+1}-f_j}{f_j}\right| \leq \epsilon \tag{21}$$

where ϵ is taken 0.01. This approach appears to be more suitable when the design space is convex, *i.e.*, there is a unique optimum irrespective of the starting design. (ii) In the second approach the sequence of the methods is reversed. An ES procedure is used first in order to locate the region where the global optimum lies, then the SQP is activated in order to exploit its higher order of accuracy in the neighborhood of the optimum. In this case the switch is performed when there is a small difference ($\epsilon = 0.1$) between the best designs of two consecutive generations. This approach appears to be more rational in the general case when more complex and non-convex design problems are to be solved with many local optima.

6. PARALLEL IMPLEMENTATION

The use of Evolution Strategies in structural optimization requires a large number of finite element analyses of the structure for the

evaluation of the objective and constraint functions. An important characteristic of ESs that differs from other conventional optimization algorithms is that in place of a single design point the ESs work simultaneously with a population of design points in the space of variables. This allows for a natural implementation of the evolution procedure in a parallel computer environment. Since a number of finite element analyses of the structure can be performed independently and concurrently, a complete finite element analysis can be assigned to a processor without the need for inter-processor communication during the solution phase. Therefore the parallelization of the ES is based on the fundamental premise that each individual in the population of the offsprings represents an independent group of all design variables and therefore its function evaluation can be done independently and concurrently.

In a distributed memory computing environment this implementation can be realized provided that there is enough memory at each processor to accommodate the storage required by the computer code. There is also need for a host processor to accumulate all information from the other $p-1$ processors in order to select the μ parents of the next generation. In a shared-memory environment, however, there is no storage limitations for each processor but on the total memory of the computer and there is no need for a host processor. In the present study the computations were performed on a shared memory computer where parents and offsprings are of equal number ($\mu = \lambda$) and the number of processors is taken as $p = \mu$.

The same scheme of natural parallelism can be readily applied for sensitivity analysis in the context of the gradient-based optimization approach, since the computations required for the calculation of the gradients for each design variable are entirely independent and thus they can be performed concurrently. In this case the number of processors is taken equal to the number of design variables ($p = n$). The parallelization of mathematical programming optimization methods, apart from the line search procedure, is more involved and when the number of design variables is not large its impact on the overall optimization time is limited. For this reason and since the number of design variables in shape optimization problems is usually small the parallelization of the SQP method is not considered.

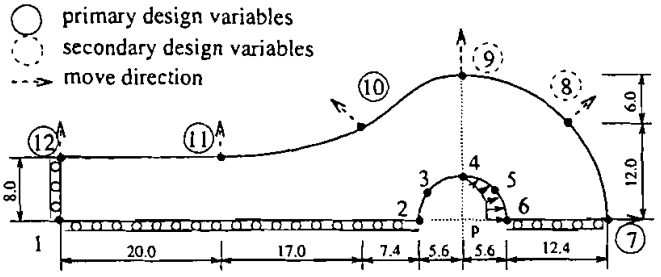
7. EXAMPLES

The performance of the optimization methods discussed is investigated and compared in three characteristic test examples. The SQP method used for the mathematical programming based optimization is taken from the NAG library [20]. For all test cases considered, plane stress conditions and isotropic material properties are assumed (elastic modulus $E = 210{,}000\,\text{N/mm}^2$ and Poisson's ratio $\nu = 0.3$). All examples were run on a Silicon Graphics Power Challenge computer with 16 R4000 processors.

In the tables containing the results of the test examples considered the following abbreviations are used: MP corresponds to the Mathematical Programming-SQP method. ESA and GFD refer to exact semi-analytical and global finite difference methods of sensitivity analysis, respectively. ES-$(\mu + \lambda)$ refers to the number of parents and offspring vectors μ, λ respectively for the evolution strategies approach. MP-ES, ES-MP are the two hybrid approaches defining the sequence of the two optimizers, while the number of optimization steps for each optimizer is depicted in parenthesis. Finally, for the parallel implementation of the optimizers the number of processors used for the case of MP optimizer is equal to the number of design variables $p = n$, whereas for the case of ESs is equal to the number of parent vectors $p = \mu$.

7.1. Connecting Rod Example [13]

The problem definition is given in Figure 1a whereas the optimized shape is depicted in Figure 1b. The linearly varying line load between key points 4 and 6 has a maximum value of $p = 500\,\text{N/mm}$. The objective is to minimize the volume of the structure subject to a limit on the equivalent stress $\sigma_{\max} = 1{,}200\,\text{N/mm}^2$. The design model, which makes use of symmetry, consists of 12 key points, 4 primary design variables (7, 10, 11, 12) and 6 secondary design variables (7, 8, 9, 10, 11, 12). The stress constraints are imposed as a global constraint for all Gauss points and as key point constraints for key points 2, 3, 4, 5, 6 and 12. The movement directions of the design variables are indicated by the dashed arrows. Key points 8 and 9 are linked to point 7 so that the shape of the arc is preserved throughout the optimization.



(a)

(b)

FIGURE 1 Connecting rod example – 4 design variables: (a) initial shape; (b) final shape.

Table I depicts the performance of the methods in sequential and parallel computing modes for this test example. Two initial designs are considered, one close and the other a way from the optimum. It can be seen that the computing time spent by the optimizers is affected by the initial design, especially in the case of the MP approach. It can also be observed that ESs are competitive to the MP optimizer in sequential computing mode and perform much better in parallel computing mode. Furthermore, the use of hybrid approaches, especially the MP-ES, leads to a significant reduction of computing time in both sequential and parallel computing modes. Since finding the optimum from a bad initial design is more difficult and time-consuming hybrid approaches are used only for this case. The mathematical optimizer using the ESA sensitivity analysis method is faster than GFD method in sequential mode. In parallel mode, however, GFD improves its efficiency considerably and becomes competitive to ESA. The natural parallelization scheme implemented has a beneficial effect to all versions of the optimizers. In particular this effect is more pronounced in the case of ESs where a higher efficiency is achieved.

TABLE I Connecting rod example–4 design variables: Performance of the optimization methods

Optimization method	*Number of opt. steps*	*Optimum volume* (mm^3)	*Sequential time* (*s*)	*Parallel time in p processors* (*s*)
Good initial design ($V_0 = 559.6\,mm^3$)–2792 d.o.f.				
MP-ESA	26	332	573.4	454.2 ($p = 4$)
MP-GFD	26	332	975.7	562.8 ($p = 4$)
ES (2 + 2)	53	376	842.8	517.3 ($p = 2$)
ES (4 + 4)	48	338	754.6	316.9 ($p = 4$)
ES (6 + 6)	82	334	1450.2	342.3 ($p = 6$)
ES (8 + 8)	92	331	1682.5	402.2 ($p = 8$)
Bad initial design ($V_0 = 726.1\,mm^3$)–3206 d.o.f.				
MP-ESA	47	333	1061.3	796.7 ($p = 4$)
MP-GFD	48	333	1806.1	983.2 ($p = 4$)
ES (2 + 2)	78	348	1420.6	908.2 ($p = 2$)
ES (4 + 4)	53	337	1153.3	421.3 ($p = 4$)
ES (6 + 6)	101	336	2015.2	437.8 ($p = 6$)
ES (8 + 8)	109	332	2079.1	463.4 ($p = 8$)
MP-ES	26 (7 + 19)	332	465.7	259.2 ($p = 4$)
ES-MP	31 (25 + 6)	329	644.7	444.2 ($p = 4$)

7.2. Square Plate Example [13]

The problem definition of this example is given in Figure 2a, where due to symmetry only a quarter of the plate is modelled, whereas the optimized shape is depicted in Figure 2b. The two exterior sides of the plate are loaded with a distributed loading $p = 0.65\,N/mm^2$, as shown in Figure 2a. The objective is to minimize the volume of the structure subject to a limit on the equivalent stress $\sigma_{max} = 7.0\,N/mm^2$. The design model, consists of 8 key points, 5 primary design variables (2, 3, 4, 5, 6) which can move along radial lines. The movement directions of the design variables are indicated by the dashed arrows. The stress constraints are imposed as a global constraint for all the Gauss points and as key point constraints for key points 2, 3, 4, 5, 6 and 8. For this test example the ESA and GFD sensitivity analysis methods are used to compute the sensitivities with $\Delta s = 10^{-5}$.

Table II depicts the performance of the methods for this example in sequential and parallel computing modes. Two initial designs are considered, one close and the other away from the optimum. The computing time spent by the optimizers is affected, as in the case of the connecting rod example, by the initial design, particularly in the case

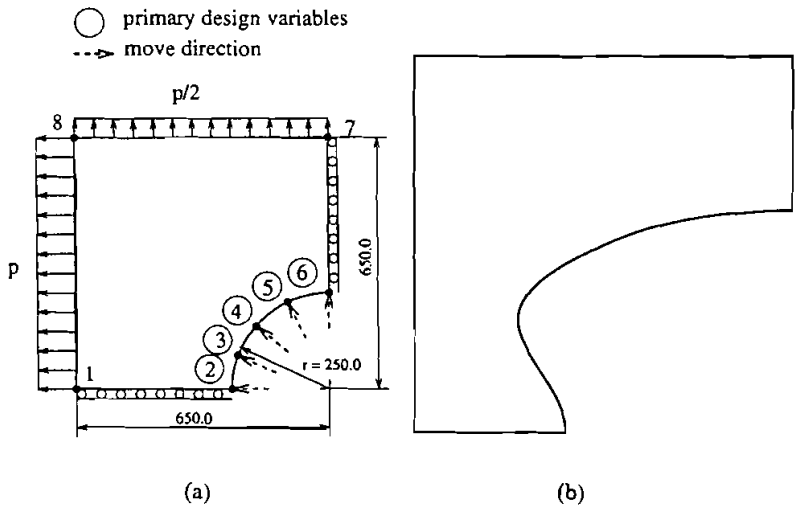


FIGURE 2 Connecting rod example – 9 design variables: (a) initial shape; (b) final shape.

TABLE II Square plate example – 5 design variables: Performance of the optimization methods

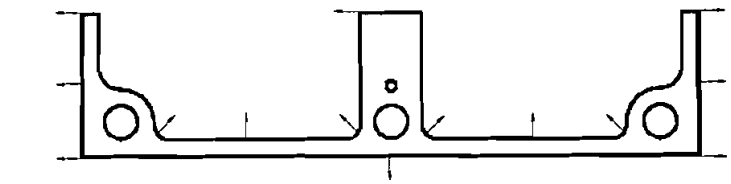
Optimization method	*Number of opt. steps*	*Optimum volume* (mm^3)	*Sequential time* (*s*)	*Parallel time in p processors* (*s*)
Good initial design (V_0 = 307.3 mm^3)–1528 d.o.f.				
MP-ESA	33	280	965.8	702.9 (p = 5)
MP-GFD	33	280	1785.6	873.5 (p = 5)
ES (3 + 3)	134	279	2336.8	1217.2 (p = 3)
ES (5 + 5)	85	279	1271.5	354.2 (p = 5)
ES (10 + 10)	150	279	2780.7	337.4 (p = 10)
Bad initial design (V_0 = 373.4 mm^3)–1546 d.o.f.				
MP-ESA	72	280	1894.7	1198.6 (p = 5)
MP-GFD	75	279	4189.1	1673.4 (p = 5)
ES (3 + 3)	134	279	2406.8	1219.7 (p = 3)
ES (5 + 5)	127	279	2141.3	586.8 (p = 5)
ES (10 + 10)	191	279	2998.5	497.2 (p = 10)
MP-ES	24 (4 + 20)	281	386.1	196.1 (p = 5)
ES-MP	28 (10 + 8)	280	480.1	275.9 (p = 5)

of the MP approach. It can also be observed that ESs present a competitive performance to the MP optimizer in sequential computing mode and perform much better in parallel computing mode. As was observed in the previous example the use of hybrid approaches,

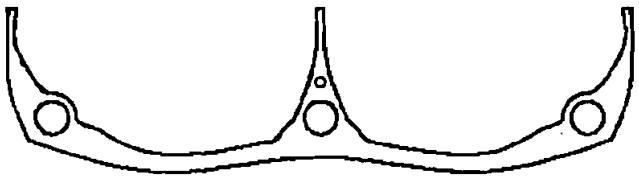
especially the MP-ES, leads to a significant reduction of computing time in both sequential and parallel modes. The mathematical programming optimizer using the ESA sensitivity analysis method is faster than the GFD method in sequential mode. In parallel mode, however, GFD becomes competitive to ESA.

7.3. Engine Block Example [27]

The problem definition of this example is given in Figure 3a, whereas the optimized shape is depicted in Figure 3b. The interior sides of the block are loaded with a distributed loading $p = 1.60\,\text{N/mm}^2$. The objective is to minimize the volume of the structure subject to a limit on the equivalent stress $\sigma_{max} = 5.0\,\text{N/mm}^2$. The design model is in this test more complicated since it consists of 71 key points, 14 primary design variables and 27 secondary design variables. The stress constraints are imposed as a global constraint for all the Gauss points and as key point constraints for a few selected key points. For this test



(a)



(b)

FIGURE 3 Engine block example – 14 design variables: (a) initial shape; (b) final shape.

example the ESA and GFD sensitivity analysis methods are used to compute the sensitivities with $\Delta s = 10^{-5}$.

Table III depicts the performance of the methods for this example in sequential and parallel computing modes. Two initial designs are considered, one close and the other away from the optimum, while the computing time spent by the optimizers is again affected by the initial design, particularly in the case of the MP approach. For this test case it can be observed that the MP optimizer appears to be more efficient than ESs in sequential as well as in parallel computing modes. The performance of hybrid approaches, however, and especially the ES-MP still leads to significant reduction of computing time in both computational modes.

8. CONCLUSIONS

The implementation of ESs in shape optimization problems was found to be very effective. The comparison of ESs, and particularly their hybrid approaches with SQP, over the MP-SQP method, which is considered one of the best mathematical programming methods, is very promising, in both sequential and parallel computing environments. The computational effort required by the optimizers is affected

TABLE III Engine block example – 14 design variables: Performance of the optimization methods

Optimization method	*Number of opt. steps*	*Optimum volume* (mm^3)	*Sequential time* (*s*)	*Parallel time in p processors* (*s*)
Good initial design ($V_0 = 358.8\,mm^3$)–2919 d.o.f.				
MP-ESA	123	294.6	6214.5	2409.6 ($p = 14$)
MP-GFD	124	295.1	13307.6	3563.8 ($p = 14$)
ES (10 + 10)	196	297.2	7801.3	3472.1 ($p = 10$)
ES (15 + 15)	211	294.9	8678.4	2768.5 ($p = 15$)
ES (20 + 20)	247	296.3	10055.3	4265.6 ($p = 10$)
Bad initial design ($V_0 = 483.2\,mm^3$)–3426 d.o.f.				
MP-ESA	241	294.7	12150.6	5441.2 ($p = 14$)
MP-GFD	241	294.8	26119.3	7083.8 ($p = 14$)
ES (10 + 10)	295	296.9	11696.1	8307.5 ($p = 10$)
ES (15 + 15)	309	295.3	12877.0	5675.3 ($p = 15$)
ES (20 + 20)	326	295.2	13009.5	9897.4 ($p = 10$)
MP-ES	176 (71 + 105)	294.9	7431.9	3789.7 ($p = 14$)
ES-MP	154 (125 + 29)	295.0	6257.2	2786.1 ($p = 14$)

by the starting values of the design parameters. This is more pronounced in the case of the MP-SQP method, where the number of optimization steps in most of the test cases examined is increased by a factor of 2 or more, whereas a less significant difference is observed for the ESs optimizer.

The computational efficiency of the multi-membered ESs discussed in this work, namely the $(\mu + \lambda)$-ESs, is affected by the number of parents and offsprings involved. Large number of parents and offsprings produce a computational overhead without a significant improvement on the values of the objective function. The selection of small number of parents and offsprings on the other hand does not produce satisfactory results. In most test cases examined it was observed that values of μ and λ equal to the number of the design variables produced best results.

The comparison of the two hybrid approaches shows that when the design space is convex best results are produced when MP-SQP is used first followed by ESs. This approach appears to be more suitable in this case since the MP-SQP optimizer captures very quickly the proper path to the nearest optimum and ES accelerates the convergence in the close vicinity of the optimum. The reverse approach ES-MP is more effective in the general case when more complex and non-convex design problems are to be solved with many local optima as in the case of the last test example.

The natural parallelization implemented in this study is less effective in the MP approach than in ESs, while the speedup factors achieved are better in the cases with fewer design variables. For large number of design variables the efficiency of the natural parallelization scheme is considerably reduced and a more vigorous parallel handling of the methods is required. The mathematical optimizer using the ESA sensitivity analysis method is faster than GFD method in sequential mode. In parallel mode, however, GFD method improves it's efficiency considerably and becomes competitive to ESA sensitivity analysis method.

Acknowledgements

This work has been supported by HC&M/9203390 project of the EU. The authors wish to thank E. Hinton and J. Sienz of University

College, Swansea and G. Thierauf and J. Cai of the University of Essen for their cooperation. The authors are grateful to Dr. J. Sienz for providing the last example and for his assistance for the completion of this study.

References

[1] Hinton, E. and Sienz, J. (1994). Aspects of adaptive finite element analysis and structural optimization. In: Topping, B. H. V. and Papadrakakis, M. (Eds.), *Advances in Structural Optimization*, CIVIL-COMP Press, Edinburgh, pp. 1–26.

[2] Ramm, E., Bletzinger, K.-U., Reitinger, R. and Maute, K. (1994). The challenge of structural optimization. In: Topping, B. H. V. and Papadrakakis, M. (Eds.), *Advances in Structural Optimization*, CIVIL-COMP Press, Edinburgh, pp. 27–52.

[3] Papadrakakis, M., Tsompanakis, Y., Hinton, E. and Sienz, J. (1996). Advanced solution methods in topology optimization and shape sensitivity analysis. *J. Engineering Computations*, **13**(5), 57–90.

[4] Forgel, L. J., Owens, A. J. and Walsh, M. J. (1966). *Artificial intelligence through simulated evolution*. Wiley, New York.

[5] Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Publishing Co., Inc., Reading, Massachusetts.

[6] Holland, J. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor.

[7] Rechenberg, I. (1973). *Evolution strategy: optimization of technical systems according to the principles of biological evolution*. Frommann-Holzboog, Stuttgart.

[8] Schwefel, H. P. (1981). *Numerical optimization for computer models*. Wiley and Sons, Chichester, UK.

[9] Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, **220**, 671–680.

[10] Berke, L. and Hajela, P. (1990). *Applications of Artificial Neural Nets in Structural Mechanics*. NASA 331–348 TM-102420.

[11] Bletzinger, K. U., Kimmich, S. and Ramm, E. (1991). Efficient modelling in shape optimal design. *Computing Systems in Engineering*, **2**(5/6), 483–495.

[12] Olhoff, N., Rasmussen, J. and Lund, E. (1992). Method of exact numerical differentiation for error estimation in finite element based semi-analytical shape sensitivity analyses. *Special Report No. 10*, Institute of Mechanical Engineering, Aalborg University, Aalborg, DK.

[13] Hinton, E. and Sienz, J. (1995). Studies with a robust and reliable structural shape optimization tool. In: Topping, B. H. V. (Ed.), *Developments in Computational Techniques for Structural Engineering*, CIVIL-COMP Press, Edinburgh, pp. 343–358.

[14] Gill, P. E., Murray, W. and Wright, M. H. (1981). *Practical Optimization*. Academic Press.

[15] Schittkowski, K., Zillober, C. and Zotemantel, R. (1994). Numerical comparison of non-linear algorithms for structural optimization. *Structural Optimization*, **7**, 1–19.

[16] Arora, J. S. (1990). Computational design optimization: A review and future directions. *Structural Safety*, **7**, 131–148.

[17] Thanedar, P. B., Arora, J. S., Tseng, C. H., Lim, O. K. and Park, G. J. (1986). Performance of some SQP methods on structural optimization problems. *Int. J. Num. Meth. Eng.*, **23**, 2187–2203.

[18] Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. (1986). User's guide for NPSOL (Version 4.0): A Fortran Package for Nonlinear Programming. *Technical Report SOL 86-2*, Dept. of Operations Research, Stanford University.
[19] Fleury, C. (1993). Dual methods for convex separable problems. In: Rozvany, G. I. N. (Ed.), *NATO/DFG ASI Optimization of large structural systems*, Berchtesgaden, Germany. Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 509-530.
[20] NAG (1988). *Software manual*, NAG Ltd, Oxford, UK.
[21] Schoenauer, M. (1995). Shape representation for evolutionary optimization and identification in structural mechanics. In: Winter, G., Periaux, J., Galan, M. and Cuesta, P. (Eds.), *Genetic Algorithms in engineering and computer science*, J. Wiley, pp. 443-464.
[22] Hoffmeister, F. and Back, T. (1991). Genetic algorithms and evolution strategies-similarities and differences. In: Schwefel, H. P. and Manner, R. (Eds.), *Parallel Problems Solving from Nature*, Springer-Verlag, Berlin, Germany, pp. 455-469.
[23] Thierauf, G. and Cai, J. (1995). A two level parallel evolution strategy for solving mixed-discrete structural optimization problems. *21st ASME Design Automation Conference*, Boston MA, September 17-21.
[24] Thierauf, G. and Cai, J. (1996). Structural optimization based on Evolution Strategy. In: Papadrakakis, M. and Bugeda, G. (Eds.), *Advanced computational methods in structural mechanics*, CIMNE, Barcelona, pp. 266-280.
[25] Lasdon, L. S., Warren, A. D., Jain, A. and Ratner, R. (1978). Design and testing of a generalized reduced gradient code for nonlinear programming. *ACM Trans. Math. Softw.*, **4**(1), 34-50.
[26] Svanberg, K. (1987). The method of moving asymptotes, a new method for structural optimization. *Int. J. Num. Meth. Eng.*, **23**, 359-373.
[27] Sienz, J. (1997). Private communication.