Large Scale Structural Optimization: Computational Methods and Optimization Algorithms

ARTICLE *in* ARCHIVES OF COMPUTATIONAL METHODS IN ENGINEERING · JANUARY 2001

Impact Factor: 3.68 · DOI: 10.1007/BF02736645

4 AUTHORS:

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

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

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

Vagelis Plevris
School of Pedagogical & Technological Educ…
25 PUBLICATIONS **230** CITATIONS
SEE PROFILE

Vol. **8, 3,** 239-301 (2001)

Archives of Computational Methods in Engineering
State of the art reviews

Large Scale Structural Optimization: Computational Methods and Optimization Algorithms

M. Papadrakakis, Nikolaos D. Lagaros, Y. Tsompanakis and V. Plevris

Institute of Structural Analysis & Seismic Research
National Technical University Athens
Zografou Campus, Athens 157 80, Greece

Summary

The objective of this paper is to investigate the efficiency of various optimization methods based on mathematical programming and evolutionary algorithms for solving structural optimization problems under static and seismic loading conditions. Particular emphasis is given on modified versions of the basic evolutionary algorithms aiming at improving the performance of the optimization procedure. Modified versions of both genetic algorithms and evolution strategies combined with mathematical programming methods to form hybrid methodologies are also tested and compared and proved particularly promising. Furthermore, the structural analysis phase is replaced by a neural network prediction for the computation of the necessary data required by the evolutionary algorithms. Advanced domain decomposition techniques particularly tailored for parallel solution of large-scale sensitivity analysis problems are also implemented. The efficiency of a rigorous approach for treating seismic loading is investigated and compared with a simplified dynamic analysis adopted by seismic codes in the framework of finding the optimum design of structures with minimum weight. In this context a number of accelerograms are produced from the elastic design response spectrum of the region. These accelerograms constitute the multiple loading conditions under which the structures are optimally designed. The numerical tests presented demonstrate the computational advantages of the discussed methods, which become more pronounced in large-scale optimization problems.

1 INTRODUCTION

Since 1970 structural optimization has been the subject of intensive research and several different approaches for optimal design of structures have been advocated [25,35,51,67,76,78,90]. Mathematical programming 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 to construct an approximate model of the initial problem. On the other hand the application of combinatorial optimization methods based on probabilistic searching do not need gradient information and therefore avoid to perform the computationally expensive sensitivity analysis step. Gradient based methods present a satisfactory local rate of convergence, but they cannot assure that the global optimum can be found, while combinatorial optimization techniques, 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.

During the last three decades there has been a growing interest in problem solving systems based on algorithms that rely on analogies to natural processes, called Evolutionary Algorithms (EA). The best-known algorithms in this class include Evolutionary Programming (EP) [24], Genetic Algorithms (GA) [31,41], Evolution Strategies (ES) [69,75]. Evolution- based systems maintain a population of potential solutions. These systems have some selection process based on fitness of individuals and some recombination operators. Both GA and ES imitate biological evolution and combine the concept of artificial survival of the fittest with evolutionary operators to form a robust search mechanism.

Over the last ten years artificial intelligence techniques [13,45] have emerged as a powerful tool that could be used to replace time-consuming procedures in many scientific or

©2001 by CIMNE, Barcelona (Spain). ISSN: 1134–3060 Received: March 2001

engineering applications. The use of NN to predict finite element analysis outputs has been studied previously in the context of optimal design of structural systems [4,5,7,14,34,59,77] and also in some other areas of structural engineering applications, such as structural damage assessment, structural reliability analysis, finite element mesh generation or fracture mechanics [33,44,62,79,84,87].

In this work the efficiency of both mathematical programming and evolutionary algorithms is investigated in sizing, shape and topology optimization problems. In order to benefit from the advantages of both methodologies combinations of EA with mathematical programming are also examined in an attempt to increase further the robustness as well as the computational efficiency of the optimization procedure. Furthermore, combinations of EA and Neural Networks (NN) are also implemented and tested in sizing optimization problems.

Since a great deal of effort is spent for the solution of the finite element equilibrium equations encountered during the optimization process specially tailored solution methods have been applied and tested in this work in a variety of optimization problems. These methods implemented in serial and parallel computing environments can have a paramount effect on the computational performance of the whole optimization procedure.

Structural optimization is also performed under seismic loading. In this case the computational effort for optimization can be orders of magnitude more than the corresponding effort for static loading. A rigorous approach based on a number of artificial accelerograms treated as multiple loading conditions is compared with a simplified response spectrum modal analysis adopted by the seismic codes. The numerical tests presented demonstrate the computational advantages of the discussed methods, which become more pronounced in large- scale and computationally intensive optimization problems.

2 FORMULATION OF THE STRUCTURAL OPTIMIZATION PROBLEM

Structural optimization problems are characterized by various objective and constraint functions that 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 depend 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:

$$\begin{aligned} \min \quad & F(s) \\ \text{subject to} \quad & g_j(s) \leq 0 \quad j = 1, \cdots, m \\ & s_i^1 \leq s_i \leq s_i^u \quad i = 1, \cdots, n \end{aligned} \tag{1}$$

where, s is the vector of design variables, $F(s)$ is the objective function to be minimized, $g_j(s)$ are the behavioural constraints, s_i^1 and s_i^u are the lower and the upper bounds on a typical design variable s_i. Equality constraints are usually rarely imposed. Whenever they are used they are treated for simplicity as a set of two inequality constraints.

There are mainly three classes of structural optimization problems: sizing, shape and topology or layout. Initially structural optimization was focused on sizing optimization, such as optimizing cross sectional areas of truss and frame structures, or the thickness of plates and shells. The next step was to consider finding optimum boundaries of a structure, and therefore to optimize its shape. In the former case the structural domain is fixed, while in the latter case it is not fixed but it has a predefined topology. In both cases a non-optimal starting topology can lead to sub-optimal results. To overcome this deficiency structural topology optimization needs to be employed, which allows the designer to optimize the layout or the topology of a structure by detecting and removing the low-stressed material in the structure which is not used effectively.

2.1 Sizing Optimization

In sizing optimization problems the aim is usually to minimize the weight of the structure under certain behavioural constraints on stresses and displacements. The design variables are most frequently chosen to be dimensions of the cross-sectional areas of the members of the structure. Due to engineering practice demands the members are divided into groups having the same design variables. This linking of elements results in a trade-off between the use of more material and the need of symmetry and uniformity of structures due to practical considerations. Furthermore, it has to be to taken into account that due to fabrication limitations the design variables are not continuous but discrete since cross-sections belong to a certain set.

A discrete structural optimization problem can be formulated in the following form

$$\begin{aligned} \min \quad & F(s) \\ \text{subject to} \quad & g_j(s) \leq 0 \quad j = 1, \cdots, m \\ & s_i \in R^d \quad i = 1, \cdots, n \end{aligned} \tag{2}$$

where R^d is a given set of discrete values representing the available structural member cross-sections and design variables s_i $(i = 1, \cdots, n)$ can take values only from this set.

The sizing optimization methodology proceeds with the following steps: (i) At the outset of the optimization the geometry, the boundaries and the loads of the structure under investigation have to be defined. (ii) The design variables, which may or may not be independent to each other, are also properly selected. Furthermore, the constraints are also defined in this stage in order to formulate the optimization problem as in eq. (2). (iii) A finite element analysis, is then carried out and the displacements and stresses are evaluated. (iv) If a gradient-based optimizer is used then the sensitivities of the constraints and the objective function to small changes of the design variables are computed. (v) The design variables are being optimized. If the convergence criteria for the optimization algorithm are satisfied, then the optimum solution has been found and the process is terminated, else the optimizer updates the design variable values and the whole process is repeated from step (iii).

2.2 Shape Optimization

In structural shape optimization problems the aim is to improve the performance of the structure by modifying its boundaries. This can be numerically achieved by minimizing an objective function subjected to certain constraints [37,68]. All functions are related to the design variables, which are some of the coordinates of the key points in the boundary of the structure. The shape optimization approach adopted in the present study is based on a previous work by Hinton and Sienz [37] for treating two-dimensional problems. More specifically 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 modeled using cubic B-splines that, 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 to 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 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 optimization problem is solved; the design variables are being optimized and the new shape of the structure is defined. 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).

2.3 Topology Optimization

Structural topology optimization assists the designer to define the type of structure, which is best suited to satisfy the operating conditions for the problem in question. It can be seen as a procedure of optimizing the rational arrangement of the available material in the design space and eliminating the material that is not needed. Topology optimization is usually employed in order to achieve an acceptable initial layout of the structure, which is then refined with a shape optimization tool. The topology optimization procedure proceeds step-by-step with a gradual "removal" of small portions of low stressed material, which are being used inefficiently. This approach is treated in this study as a typical case of a structural reanalysis problem with small variations of the stiffness matrix between two subsequent optimization steps.

Many researchers have presented solutions for structural topology optimization problems. Topological or layout optimization can be undertaken by employing one of the following main approaches, which have evolved during the last few years [38]: (i) Ground structure approach [66,77], (ii) homogenization method [12,36,80], (iii) bubble method [19] and (iv) fully stressed design technique [88,92]. The first three approaches have several things in common. They are optimization techniques with an objective function, design variables, constraints and they solve the optimization problem by using an algorithm based on sequential quadratic programming (approach (i)), or on an optimality criterion concept (approaches (ii) and (iii)). However, inherently linked with the solution of the optimization problem is the complexity of these approaches. The fully stressed design technique on the other hand, although not an optimization algorithm in the conventional sense, proceeds by removing inefficient material, and therefore optimizes the use of the remaining material in the structure, in an evolutionary process.

At present only a limited number of studies is devoted to 3-D optimal topology design of structures. For this type of problems the main difficulty when a homogenization method is used is the orientation of the material voids which is more complicated than in the 2-D case. This difficulty is not present in the case of the fully stressed design technique. The work presented in this study is based on the implementation of the evolutionary fully stressed design technique (FSD) proposed by Hinton and Sienz [38] and the improved implementation presented by Papadrakakis *et al.* [65] for 2-D topology optimization problems. This methodology is extended to 3-D topology optimization problems using solid finite elements. Furthermore an investigation is performed on the impact of using effective domain decomposition solution techniques on the overall performance of the FSD topology optimization approach.

The algorithm for topology optimization adopted in this study is based on the simple principle that material which has small stress levels is used inefficiently and therefore it can be removed. Thus, by removing small amounts of material at each optimization step the layout of the structure evolves gradually. In order to achieve convergence of the whole optimization procedure, it is important the amount of material removed at each stage to be small and to maintain a smooth transition from one layout of the structure to the subsequent one.

The domain of the structure, which is called the reference domain, can be divided into the design domain and the non-design domain. The non-design domain covers regions with stress concentrations, such as supports and areas where loads are applied, and therefore it cannot be modified throughout the whole topology optimization process. After the generation of the finite element mesh, the evolutionary fully stressed design cycle is activated, where a linear elastic finite element analysis is carried out. The maximum principal stress σ_{pr} for each element can be computed which for convenience is called stress level and is denoted as σ_{evo}. The maximum stress level $\sigma_{\max}$ of the elements in the structure at the current optimization step is defined, and all elements that fulfill the condition

$$\sigma_{\mathrm{evo}} < \mathrm{ratre} \times \sigma_{\max} \tag{3}$$

are removed, or *switched-off*, where *ratre* is the rejection rate parameter [93]. The elements are removed by assigning them a relatively small elastic modulus which is typically

$$E_{\mathrm{off}} = 10^{-5} \times E_{\mathrm{on}} \tag{4}$$

In this way the elements switched-off virtually do not carry any load and their stress levels are accordingly small in subsequent analyses. This strategy is called “hard kill”, since the low stressed elements are immediately removed, in contrast with the “soft kill” method where the elastic modulus varies linearly and the elements are removed more gradually. The remaining elements are considered *active* and they are sorted in ascending order according to their stress levels before a subsequent analysis is performed.

The iterative process of element removal and addition, if element growth is allowed, is continued until one of several specified convergence criteria are met: (i) All stress levels are larger than a certain percentage value of the maximum stress. This criterion assumes that a fully stressed design has been achieved and the material is used efficiently. (ii) The number of active elements is smaller than a specified percentage of the total number of elements. For uniform meshes, which are commonly used in topology optimization problems, this criterion is equivalent to an area or volume fraction of the initial design, which will be in use in the final layout. (iii) When element growth is allowed the evolutionary process is completed when more elements are switched-on than they are switched-off.

3 MATHEMATICAL PROGRAMMING OPTIMIZATION ALGORITHMS

Mathematical programming algorithms such as the successive quadratic programming method [83], the generalized reduced gradient method [47], the method of moving asymptotes [81], the method of feasible directions [89] have been used for structural optimization problems. Successive Quadratic Programming (SQP) methods are regarded as the standard general purpose mathematical programming algorithms for solving non-linear programming optimization problems [28]. They are also considered to be the most suitable methods for solving structural optimization problems [6,73,83]. 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 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 form

$$\text{minimize} \quad \frac{1}{2} p^T H p + g^T p$$

$$\begin{aligned} \text{subject to} \quad & Ap + h(s) \le 0 \\ & \overline{s}_1 \le p \le \overline{s}_u \end{aligned} \tag{5}$$

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), $\overline{s}_1 = s_1 - s$, $\overline{s}_u = s_u - s$ and H is an approximation of the Hessian matrix of the Lagrangian function

$$L(s, \lambda) = F(s) + \lambda h(s) \tag{6}$$

in which λ are the Lagrange multipliers under the non-negativity restriction ($\lambda \ge 0$) 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 [29], or a dual [23] formulation. The primal algorithm adopted in this study 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. 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 [29]

$$\varphi = F(s) - \sum_i \lambda_i (g_i(s) - \gamma_i) + \frac{1}{2} \sum_i \rho_i (g_i(s) - \gamma_i)^2 \tag{7}$$

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 iterations are increased whenever this is necessary in order to control the violation of the constraints and to ensure that merit function follows a descent path.

The update of the Hessian matrix of the Lagrangian function is performed with a BFGS quasi-Newton update [28] where attention is given to keep 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

$$\tilde{H} = H - \frac{1}{w^T H w} H w w^T H + \frac{1}{y^T w} y y^T \tag{8}$$

where w and y denote the change in the design variable vector s and the gradient vector of the Lagrangian function of eq. (6), 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.

3.1 Sensitivity Analysis

The most time-consuming part of any optimization algorithm based on mathematical programming methods is devoted to the sensitivity analysis phase [65], which is an important ingredient of all mathematical programming optimization methods. 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 [16].

A classification of the discrete methods for sensitivity analysis is the following. (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 semi-analytical and the finite difference approaches are the two most widely used types of sensitivity analysis techniques. From the algorithmic point of view the semi-analytical technique results in a typical linear solution problem with multiple right-hand sides in which the stiffness matrix remains unchanged, while the finite difference technique results in a typical reanalysis problem in which the stiffness matrix is modified due to the perturbations of the design variables. In both shape and sizing optimization problems 60% to 90% of the computations are spent for the solution of equilibrium equations required for the finite element analysis and sensitivity analysis.

3.1.1 The semi-analytical (SA) method

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

$$K\frac{\partial u}{\partial s_k} + \frac{\partial K}{\partial s_k}u = \frac{\partial f}{\partial s_k} \tag{9}$$

which when rearranged results in

$$K\frac{\partial u}{\partial s_k} = f_k^* \tag{10}$$

where

$$f_k^* = \frac{\partial f}{\partial s_k} - \frac{\partial K}{\partial s_k}u \tag{11}$$

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 is implemented in two versions: The conventional SA and the "exact" SA. In the conventional sensitivity analysis (CSA), the values of the derivatives in eq.(9) are calculated by applying the forward difference approximation scheme

$$\frac{\partial K}{\partial s_k} \approx \frac{\Delta K}{\Delta s_k} = \frac{K(s_k + \Delta s_k) - K(s_k)}{\Delta s_k} \tag{12}$$

In the "exact" semi-analytical method (ESA) [54] the derivatives $\partial K/\partial s_k$ are computed on the element level as follows

$$\frac{\partial k}{\partial s_k} = \sum_{j=1}^{n} \frac{\partial k}{\partial \alpha_j}\frac{\partial \alpha_j}{\partial s_k} \tag{13}$$

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 [39]. This approach is used in the present study.

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

$$\frac{\partial \sigma}{\partial s_k} = \frac{\partial D}{\partial s_k}Bu + D\frac{\partial B}{\partial s_k}u + DB\frac{\partial u}{\partial s_k} \tag{14}$$

Since the elasticity matrix D is not a function of the design variables then eq.(14) reduces to

$$\frac{\partial \sigma}{\partial s_k} = D\frac{\partial B}{\partial s_k}u + DB\frac{\partial u}{\partial s_k} \tag{15}$$

In eq. (15), $\partial u/\partial s_k$ and $\partial B/\partial s_k$ may be 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.

3.1.2 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

$$\frac{\partial u}{\partial s_k} \approx \frac{\Delta u}{\Delta s_k} = \frac{u(s_k + \Delta s_k) - u(s_k)}{\Delta s_k} \tag{16}$$

$$\frac{\partial \sigma}{\partial s_k} \approx \frac{\Delta \sigma}{\Delta s_k} = \frac{\sigma(s_k + \Delta s_k) - \sigma(s_k)}{\Delta s_k} \tag{17}$$

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

$$K(s_k + \Delta s_k)u(s_k + \Delta s_k) = f(s_k + \Delta s_k) \tag{18}$$

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

$$\sigma(s_k + \Delta s_k) = DB(s_k + \Delta s_k)u(s_k + \Delta s_k) \tag{19}$$

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}.

3.2 Solving The Sensitivity Analysis And Topology Optimization Problem

Usually, a real world structural optimization problem, whether it is topology, shape, sizing or an integrated structural optimization problem, is a computationally intensive task, where 60% to 90% of the computations are spent for the solution of the finite element equilibrium equations required for the analysis steps within the optimization procedure. Thus, the computational efficiency of the finite element solver has a paramount effect on the efficiency of the optimization algorithm.

The numerical problem encountered in the sensitivity analysis phase can be seen as an algebraic problem with multiple right-hand sides of the type $Ku_i = f_i$ $(i = 1, \cdots, q)$, when the exact semi-analytical (ESA) approach is used, or as a nearby problem of the type $(K + \Delta K_i)u_i = f_i$ $(i = 1, \cdots, q)$, when the global finite difference approach is implemented. In the case of topology optimization with the fully stressed design concept adopted in this study, a nearby problem has to be solved in each optimization step.

The solution methods implemented and tested in this work can be distinguished in single- and multi-domain methods. Single-domain methods perform operation on the global structural level, while in multi-domain methods the operations are performed in subdomains or substructures in a successive or simultaneous fashion. The second case corresponds to the implementation of the solution algorithms in parallel computing environment.

3.2.1 Single-domain methods

Single-domain hybrid solution schemes based on a combination of direct and preconditioned iterative methods are applied in the context of both sensitivity analysis and topology optimization. These schemes combine direct skyline algorithms with preconditioned conjugate gradient and Lanczos methods and are properly modified to address the special features of the particular optimization problem at hand.

The Incomplete Cholesky Preconditioned Conjugate Gradient (ICCG) Method

The PCG method has become very popular for the solution of large-scale finite element problems. An efficient preconditioning matrix makes PCG very attractive even for ill-conditioned problems without destroying the characteristic features of the method. Several global preconditioners have been used in the past for solving finite element linear problems [56]. Preconditioning techniques based on incomplete Cholesky factorization are capable in increasing the convergence rate of the basic iterative method, at the expense of more storage requirements. In the present study the incomplete procedure by magnitude proposed by Bitoulas and Papadrakakis [15], is implemented in a mixed precision arithmetic mode, and a compact storage scheme is used to store both the stiffness and the preconditioning matrices row-by-row.

The reason for performing an incomplete factorization is to obtain a reasonably accurate factorization of the stiffness matrix without generating too many fill-ins, whereas a complete factorization produces the strongest possible preconditioner, which in exact precision arithmetic is actually the inverse of K. In sensitivity analysis the incomplete factorization of the stiffness matrix K can be written: $LDL^T = K_0 + \Delta K - E$, where E is an error matrix which does not have to be formed. For this class of methods, E is defined by the computed positions of "small" elements in L, which do not satisfy a specified magnitude criterion and therefore are discarded. If we have to deal with a nearby problem, such as the case of GFD sensitivity analysis and topology optimization problems, the matrix E is taken as the ΔK matrix, whereas in the case of SA sensitivity analysis both E and ΔK are taken as null matrices. The complete Cholesky factorization of the "initial" stiffness matrix K_0 is used as the preconditioning matrix in its original skyline form and is stored in single precision arithmetic.

Another important factor affecting the performance of the PCG iterative procedure for solving $Kx = b$ is the determination of the residual vector. The accuracy achieved and the computational labor of the method is largely determined by how this is calculated. A study performed in [56] revealed that the computation of the residual vector of the equilibrium equations $Kx = b$ from its defining formula $r^{(m)} = Kx^{(m)} - b$ with an explicit or a first order differences matrix-vector multiplication $Ku^{(m)}$ offers no improvement in the accuracy of the computed results. In fact, it was found that, contrary to previous recommendations, the calculation of the residuals by the recursive expression $r^{(m+1)} = r^{(m)} + \alpha_m K d^{(m)}$, where d is the direction vector, produces a more stable and well-behaved iterative procedure. Based on this observation a mixed precision arithmetic PCG implementation is proposed in which all computations are performed in single precision arithmetic, except for double precision arithmetic computation of the matrix-vector multiplication involved in the recursive evaluation of the residual vector. This implementation is a robust and reliable solution procedure even for handling large and ill-conditioned problems, while it is also computer storage-effective. It was also demonstrated to be more cost-effective, for the same storage demands, than double precision arithmetic calculations [56].

The Neumann Series-CG Method (NSCG)

The approximation of the inverse of the stiffness matrix using a Neumann series expansion has been used in the framework of stochastic finite element analysis, structural reanalysis and damage analysis problems. In all these cases the method was implemented on an "as is" basis, without any corrections to improve the quality of the solution, thus the results were satisfactory only in the vicinity of the initial design and unacceptable for large modifications of the stiffness matrix. In a recent study by Papadrakakis and Papadopoulos [61] the method was successfully combined with the conjugate gradient algorithm resulting in an improvement on the accuracies achieved with low additional computational cost. It can also handle cases with significant changes of the stiffness matrix.

The solution of a typical reanalysis problem

$$(K_0 + \Delta K)u = f \tag{20}$$

yields

$$u = (I + K_0^{-1} \Delta K)^{-1} K_0^{-1} f \tag{21}$$

The term in parenthesis can be expressed in a Neumann expansion giving

$$u = (I - P + P^2 - P^3 + \cdots) K_0^{-1} f \tag{22}$$

with $P = K_0^{-1} \Delta K$. The response vector can now be represented by the following series

$$u = u_0 - P u_0 + P^2 u_0 - P^2 u_0 + \cdots \tag{23}$$

or

$$u = u_0 - u_1 + u_2 - u_3 + \cdots \tag{24}$$

The series solution can also be expressed by the following recursive equation

$$K_0 u_i = \Delta K u_{i-1} \quad i = 1, 2, \cdots \tag{25}$$

The advantage of this expression is that the stiffness matrix has to be factorized once while the additive terms u_i to the solution of eq. (24) can be computed by successive backward and forward substitutions.

In order to improve the quality of the preconditioning matrix C used in the PCG method, a Neumann series expansion is implemented for the calculation of the preconditioned vector $z^{(m)} = C^{-1} r^{(m)}$ of the PCG algorithm. The preconditioning matrix is now defined as the complete global stiffness matrix $K = K_0 + \Delta K$, but the solution for z is performed approximately using a truncated Neumann series expansion. Thus, the preconditioned vector z of the PCG algorithm is obtained at each iteration by

$$z = z_0 - z_1 + z_2 - z_3 + \cdots \tag{26}$$

z_0 is given by

$$z_0 = K_0^{-1} r_0 \tag{27}$$

and

$$K_0 z_i = \Delta K z_{i-1} \quad i = 1, 2, \cdots \tag{28}$$

The incorporation of the Neumann series expansion in the preconditioned step of the PCG algorithm can be seen from two different perspectives. From the PCG point of view an improvement of the quality of the preconditioning matrix is achieved by computing a better approximation to the solution of $u = (K_0 + \Delta K)^{-1} f$ than the one provided by the preconditioning matrix K_0. From the Neumann series expansion point of view, the inaccuracy entailed by the truncated series is alleviated by the conjugate gradient iterative procedure.

The Preconditioned Lanczos Method

When a sequence of right-hand sides has to be processed direct methods possess a clear advantage over the conventional application of iterative methods. The major effort concerned with the factorization of the stiffness matrix is not repeated and only a back and forward substitution is required for each subsequent right-hand side. In the case of iterative methods the whole work has to be repeated from the beginning for every right-hand side.

Papadrakakis and Smerou [63] presented an implementation of the Lanczos algorithm for solving linear systems of equations with a sequence of right-hand sides. This algorithm handles all approximations to the solution vectors simultaneously without the necessity for keeping in fast or secondary storage the tridiagonal matrix or the orthonormal basis produced by the Lanczos method. Thus, when the first solution vector has converged to a required accuracy, good approximations to the remaining solution vectors have simultaneously been obtained. It then takes fewer iterations to reach the final accuracy by working separately on each of the remaining vectors.

The equilibrium equations for multiple right-hand sides can be stated as follows:

$$K[u_1 \cdots u_k] = [f_1 \cdots f_k] \tag{29}$$

or

$$KU = F \tag{30}$$

and the characteristic equations of the Lanczos algorithm become

$$T_j Y_j = Q_j^T R_0 \tag{31}$$

and

$$U_j = Q_j Y_j \tag{32}$$

where $Y_j = [y_1, \cdots, y_k]_j$, $U_j = [u_1, \cdots, u_k]_j$ consist of the jth approximation to the k auxiliary and solution vectors y and u respectively, and $R_0 = [r_0^1, \cdots, r_0^k]$ with $r_0^i = f^i - K u_0^j$ consists of the residual vectors. By using a Cholesky root-free decomposition of T_j we get

$$L_j D_j Z_j = Q_j^T R_0 \tag{33}$$

$$U_j = B_j Z_j \tag{34}$$

with $Z_j = [z_1, \cdots, z_k]_j$ and $L_j B_j^T = Q_j^T$. The last components of matrix Z_j are now given by

$$\zeta_{ji} = (q_j^T r_0^i - \delta_j d_{j-1} \zeta_{j-1,i})/d_j \tag{35}$$

with $\delta_j = \beta_j / d_{j-1}$, $\beta_j = (r_j^T C^{-1} r_j)^{1/2}$ and C is the preconditioning matrix. The new approximation to the solution vectors by

$$[x_1 \cdots x_k]_j = [x_1 \cdots x_k]_{j-1} + b_j [\zeta_{j1} \cdots \zeta_{jk}] \tag{36}$$

If converge is achieved for the first right hand side

$$\frac{\|r_j^{(1)}\|}{\|r_1^{(1)}\|} < \varepsilon \Rightarrow \frac{|\zeta_{j1}| \, \|r_{j+1}^{(1)}\|}{\|r_1^{(1)}\|} < \varepsilon \tag{37}$$

then continue iterations (separately) for the remaining $f^{(i)}$ $(i = 2, \cdots, k)$ with the PCG algorithm [63].

3.2.2 Multi-domain methods

In computational structural mechanics there are basically three domain decomposition formulations combined with the PCG method for solving linear finite element problems in parallel computing environments. The first approach is the global subdomain implementation (GSI) in which a subdomain-by-subdomain PCG algorithm is implemented on the global stiffness matrix. In the second approach the PCG algorithm is applied on the interface problem after eliminating the internal degrees of freedom of each subdomain. This Schur complement scheme is called the primal subdomain implementation (PSI) on the interface to distinguish from the third approach which is called the dual subdomain implementation (DSI) on the interface. The most efficient DSI is the FETI method [21] which incorporates a projection re-orthogonalization scheme for handling problems with multiple or repeated right-hand sides.

The FETI method operates on totally disconnected subdomains, while the governing equilibrium equations are derived by invoking stationarity of the energy functional subject to displacement constraints which enforce the compatibility conditions on the subdomain interface. The augmented equations are solved for the Lagrange multipliers after eliminating the unknown displacements. The resulting interface problem is in general indefinite, due to the presence of floating subdomains which do not have enough prescribed displacements to eliminate the local rigid body modes. The solution of the indefinite problem is performed by a preconditioned conjugate projected gradient (PCPG) algorithm.

Solving $Ku_i = f_i (i = 1, 2, \cdots, q)$

The modified Lanczos method proposed in [63] can handle simultaneously with the first right-hand side a sequence of right-hand sides. This means that all right-hand sides vectors must be known in advance. When the multiple right-hand sides are not known in advance a reorthogonalization procedure has been proposed by Farhat *et al.* [21], for extending the PCG method to problems with repeated right-hand sides based on the K-conjugate property of the search directions ($d_m = d_m^T K d_i = 0$ for $m < i$).

The implementation of the reorthogonalization technique is impractical when applied to the full problem $Ku^{(i)} = f^{(i)}$ due to excessive storage requirements for keeping the direction vectors d_m. This methodology, however, has been efficiently combined with the DSI-FETI method [22] where the size of the interface problem can be order(s) of magnitude less than the size of the global problem. Thus, the cost of reorthogonalization is negligible compared to the cost of the solution of the local problems associated with the matrix-vector products of the FETI method, while the additional memory requirements are not excessive. The modified search direction of the PCPG algorithm is given by

$$d'_{m+1} = d_{m+1} - \sum_{i=1}^{m} \frac{d_i^T F_I d_{m+1}}{d_i^T F_I d_i} d_i \tag{38}$$

which enforces explicitly the orthogonality condition $d'_{m+1} F_I d_i = 0$, $i = 1, \cdots, m$. $F_I = \sum_{j=1}^{s} B^{(j)} K^{(j)} B^{(j)^T}$, $K^{(j)}, B^{(j)}$ are the subdomain matrices and the signed Boolean matrices which localize the subdomain displacements on the interface, while "s" is the total number of subdomains. The initial estimate $\lambda_0^{(i+1)}$ of the solution vector of the subsequent right-hand side $[f_\lambda^{(i+1)}\ f_\gamma^{(i+1)}]^T$ is given by

$$\lambda_0^{(i+1)} = D_k^T x + x' \tag{39}$$

where $D_k^T F_I D_k x = D_k^T (f_\lambda^{(i+1)} - F_I x')$ and $x' = G_I (G_I^T G_I)^{-1} f_\gamma^{(i+1)}$. $G_I = [B^{(1)} \cdot R^{(1)} \cdots B^{(s_f)} \cdot R^{(s_f)}]$ with $R^{(j)}$ being the rigid body modes and "s_f" is the total number of floating subdomains.

Solving $(K_0 + \Delta K_i) u_i = f (i = 1, 2, \cdots, q)$

The hybrid solution schemes proposed in [65] for treating nearby problems, based on the global formulation and solution of the problem of eq. (20), proved to be very efficient compared with the standard direct skyline solver in sequential computing environment. Their parallel implementation, however, is hindered by the inherent scalability difficulties encountered during the preconditioning step of single-domain methods which incorporates forward and backward substitutions of a fully factorized stiffness matrix. In order to alleviate this deficiency the GSI subdomain-by-subdomain PCG algorithm is implemented in this study on the global stiffness matrix. The dominant matrix-vector operations of the stiffness and the preconditioning matrices are performed in parallel on the basis of a multi-element group partitioning of the entire domain.

In order to exploit the parallelizable features of the GSI-PCG method and to take advantage of the efficiency of a fully factorized preconditioning matrix, the following two-level methodology is proposed based on the combination of the GSI and the DSI approaches. The GSI-PCG method is employed, using a multi-element group partitioning of the entire finite element domain, in which the solution required during the preconditioning step is performed by the FETI method operating on the same mesh partitioning of the GSI-PCG method. In the proposed methodology the preconditioning step of the GSI-PCG method

$$z_{m+1} = C_k^{-1} r_{m+1} \tag{40}$$

is performed by the FETI solution procedure. For the solution of this problem two methodologies, namely the GSI(ICCG)-FETI and the GSI(NSCG)-FETI are proposed. The second approach is based on a Neumann series expansion of the preconditioning step.

The GSI(ICCG)-FETI method

In the GSI(PCG)-FETI method the iterations are performed on the global level with the GSI-PCG method, using a complete Cholesky factorization of a nearby stiffness matrix as preconditioner. Thus, the incomplete factorization of the stiffness matrix $K_0 + \Delta K$ can be written as $LDL^T = K_0 + \Delta K - E$, where E is an error matrix which does not have to be formed. Matrix E is usually defined by the computed positions of "small" elements in L which do not satisfy a specified magnitude criterion and therefore are discarded [15]. For the typical reanalysis problem

$$(K_0 + \Delta K_i)u_i = f \quad (i = 1, \cdots, q) \tag{41}$$

matrix E is taken as ΔK, so that the preconditioning matrix becomes the complete factorized initial stiffness matrix $C_k = K_0$. Therefore, the solution of the preconditioning step of the GSI-ICCG algorithm, which has to be performed at each GSI-ICCG iteration, can be effortlessly executed, once K_0 is factorized, by a forward and backward substitution.

With the parallel implementation of the two-level GSI(ICCG)-FETI method the preconditioning step can be solved in parallel by the interface FETI method for treating the repeated solutions required in eq. (40), using the same decomposition of the domain employed by the external GSI-PCG method. The procedure continues this way for every reanalysis problem, while the FETI direction vectors are being reorthogonalized in order to further decrease the number of FETI iterations within the preconditioning step. The solution of eq.(40) is performed $n_i \cdot n_r$ times via the FETI method, where n_i and n_r correspond to the number of GSI-PCG iterations and the number of reanalysis steps, respectively.

The GSI(NSCG)-FETI method

The quality of the preconditioning step of eq.(40) can be improved by computing the inverse approximation of the preconditioning matrix via a Neumann series expansion. The preconditioning matrix is defined in this case as the complete stiffness matrix $(K_0 + \Delta K)$, but the solution for z_{m+1} of eq. (40), which can be written as

$$z_{m+1} = (I + K_0^{-1}\Delta K)^{-1} K_0^{-1} r_{m+1} \tag{42}$$

is performed approximately using a truncated Neumann series expansion

$$z_{m+1} = z'_0 - z'_1 + z'_2 - z'_3 + \cdots \tag{43}$$

with

$$z'_0 = K_0^{-1} r_{m+1} \tag{44}$$

$$z'_i = K_0^{-1}(\Delta K z'_{i-1}), \quad i = 1, 2 \cdots \tag{45}$$

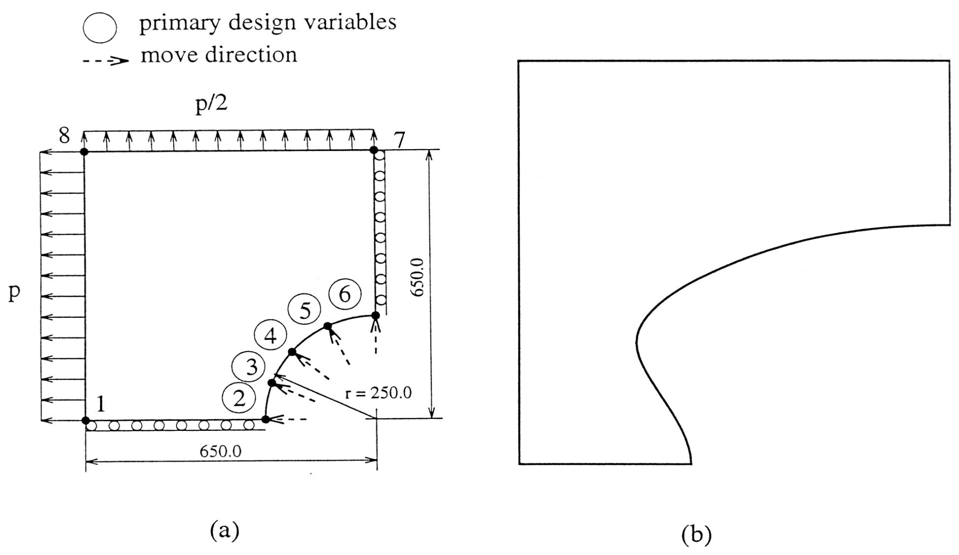


Figure 1. Square plate: (a) initial shape. (b) final shape

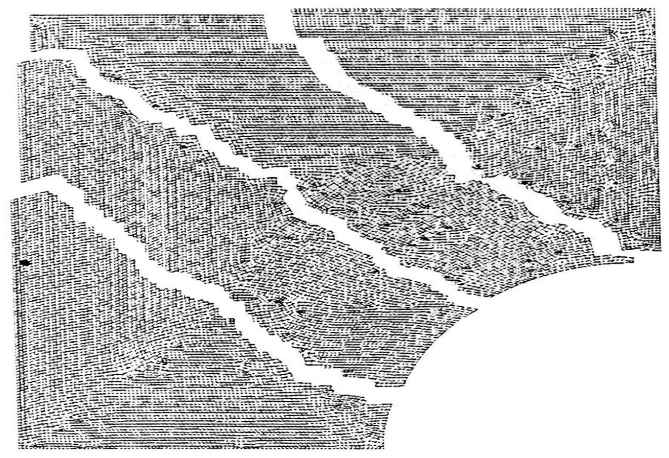
3.3 Numerical Tests

For the test examples considered in this section, isotropic material properties are assumed (elastic modulus $E = 210,000$ N/mm^2 and Poison's ratio $\nu = 0.3$). The performance of the parallel solution methods was examined using an SGI Power Challenge XL computer with 14 R4000 processors. The convergence tolerance for all solution methods was taken as 10^{-3}.

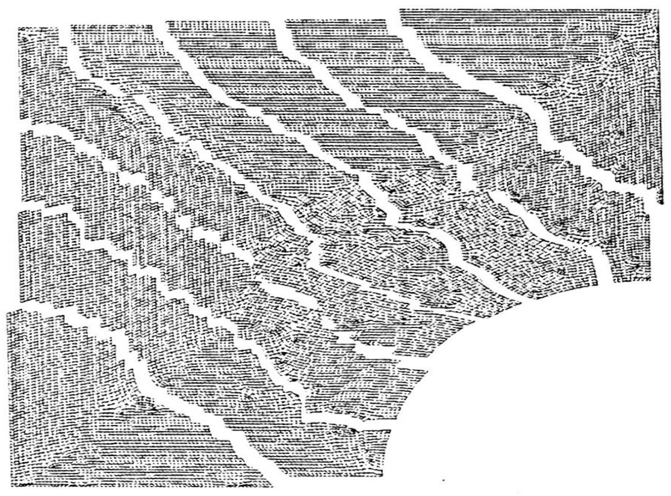
3.3.1 Shape optimization with gradient-based optimizers

The performance of the optimization methods discussed is investigated in one characteristic plane stress test example with isotropic material properties. The SQP method used for the mathematical programming based optimization is taken from the NAG library [53]. The problem definition of this example is given in Figure 1 where, due to symmetry, only a quarter of the plate is modeled. The plate is under biaxial tension with one side loaded with a distributed loading $p = 0.65$ N/mm^2 and the other side loaded only with half of this value, as shown in Figure 1. The objective is to minimize the volume of the structure subject to an equivalent stress limit of $\sigma_{max} = 7.0$ N/mm^2. The design model consists of 8 key points and 5 primary design variables (2, 3, 4, 5, 6) which can move along radial lines. The movement directions 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 the key points 2, 3, 4, 5, 6 and 8. The problem is analyzed with a fine mesh of 38,800 d.o.f. giving a sparse global stiffness matrix with relatively large bandwidth. The characteristic d.o.f. for 4 and 8 subdomains, as depicted in Figure 2a and 2b, are given in Table 1. The ESA and the GFD methods are used to compute the sensitivities with $\Delta s = 10^{-5}$.

The performance of the solution methods presented in section 3.2 is investigated first in serial computing mode with the conventional direct skyline and ICCG, NSCG and Lanczos solvers. Furthermore, the parallel performance of the standard FETI (S-FETI) and a modified version (M-FETI) is investigated in both types of sensitivity analysis problems [55], while the two-level PCG method is applied for the GFD sensitivity analysis test cases. In M-FETI the rigid body modes are computed explicitly and are not obtained as a by-product of the factorization procedure as in the S-FETI, while the local problem is solved via the



(a)



(b)

Figure 2. Square plate: (a) finite element mesh in 4 subdomains. (b) finite element mesh in 8 subdomains

subdomains	4	8
Total d.o.f.	38,800	38,800
Internal d.o.f.*	9,738	5,122
Ineterface d.o.f.	998	2,290

*of the larger subdomain

Table 1. Square plate: Characteristic d.o.f. for 4 and 8 subdomains

PCG algorithm with preconditioner the complete factorized stiffness matrix stored in single precision arithmetic. In all test cases FETI methods are applied with re-orthogonalization unless otherwise stated, while the lumped type preconditioner is used for the PCPG algorithm for the solution of the constrained problem.

The following abbreviations are used: Direct is the conventional direct skyline solver; *ICCG* (ψ) and *Lanczos* (ψ) are the PCG and Lanczos solvers respectively, preconditioned with a Cholesky factorized matrix controlled by the rejection parameter ψ. A value of ψ between 0 and 1 corresponds to an incomplete Cholesky preconditioner, while $\psi = 0$ gives the complete factorized matrix. *NSCG-i* is the NSCG solver with i terms of the Neumann series expansion. The variants of the two-level implementation, namely the *GSI(ICCG)-FETI* and *GSI(NSCG)-FETI*, are compared with the other solvers, both in serial and parallel computing modes, in the case of GFD sensitivity analysis problems.

Table 2 demonstrates the performance of the methods operated in a sequential mode for the case of ESA sensitivity analysis. In the standard FETI method and its variant the operations are carried out in 4 subdomains. Table 3 shows the performance of S-FETI and M-FETI, for the case of ESA sensitivity analysis, operated on parallel computing mode in 4 and 8 processors using 4 and 8 subdomains, respectively. The benefit from the use of the reorthogonalization is also evident both in terms of FETI iterations and computing time. Tables 4 and 5 depict the performance of the methods, for the case of GFD sensitivity analysis, operated on sequential and parallel computing modes, respectively. In Tables 3 and 5 the iteration history is also depicted for six right-hand sides, which correspond to the initial finite element solution and the sensitivity analysis for the five design variables of the problem.

method (4 subdomains-1 processor)	time (s)	storage (Mbytes)
Direct skyline	502	95
Lanczos (0)	524	63
Lanczos (1E-9)	417	29
ICCG (0)	514	61
ICCG (1E-9)	425	27
S-FETI	486	43
M-FETI	414	26

Table 2. Square plate: Performance of the methods in sequential mode in ESA sensitivity analysis

right-hand sides	1	2	3	4	5	6		
method (4 processors)	iterations						time (s)	storage (Mbytes)
S-FETI-no reorth	67	60	65	51	53	53	420	41
S-FETI	33	16	13	10	9	8	150	43
M-FETI-no reorth	67	60	65	51	53	53	306	24
M-FETI	33	16	13	10	9	8	120	26
method (8 processors)	iterations							
S-FETI-no reorth	271	266	253	219	199	204	398	20
S-FETI	64	24	18	14	11	11	92	23
M-FETI-no reorth	269	267	253	220	199	205	290	13
M-FETI	64	24	18	11	11	11	70	16

Table 3. Square plate: Performance of the methods in parallel mode in ESA sensitivity analysis

method (4 subdomains)	time (s)	storage (Mbytes)
Direct skyline	2,790	95
Lanczos (0)	732	65
Lanczos (1E-9)	745	61
ICCG (0)	714	27
ICCG (1E-9)	2,108	43
S-FETI	1,782	26
M-FETI	779	29

Table 4. Square plate: Performance of the methods in parallel mode in GFD sensitivity analysis

right-hand sides	1	2		3		4		5		6			
method (4 processors)	iterations											time (s)	storage (Mbytes)
S-FETI	33	33		33		33		33		33		667	43
M-FETI	33	33		33		33		33		33		574	26
GSI(ICCG) M- FETI	33	17	13	12	10	10	8	8	7	7	6	256	27
GSI(NSCG) M-FETI	33	17	14	12	11	11	9	7	7	6	6	249	29
method (8 processors)	iterations												
S-FETI	64	64		64		64		64		64		396	23
M-FETI	64	64		64		64		64		64		332	16
GSI(ICCG) M- FETI	64	24	19	17	14	12	11	9	9	8	7	165	17
GSI(NSCG) M-FETI	64	24	18	16	14	13	11	10	9	8	7	163	18

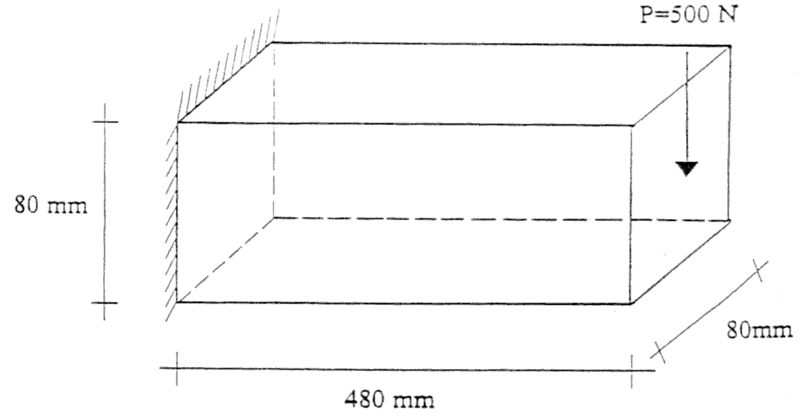
Table 5. Square plate: Performance of the methods in parallel mode in GFD sensitivity analysis

3.3.2 Topology optimization

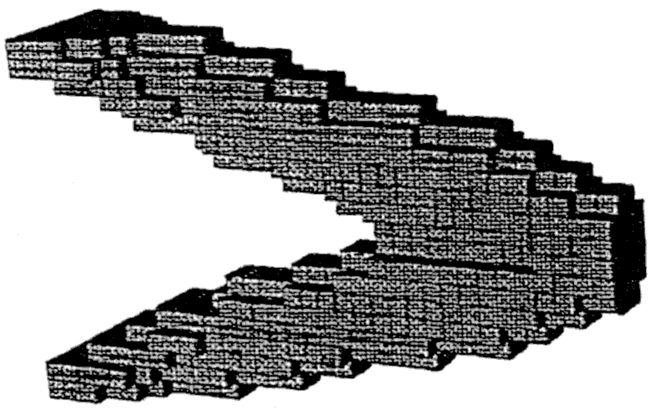
The performance of the proposed multi-domain two-level methodologies is demonstrated and compared with the one-level dual decomposition solvers in sequential and parallel computing modes. They are also compared with the single-domain direct skyline solver and the ICCG and NSCG hybrid solvers in sequential computing mode. For the single-domain iterative solvers the following abbreviations are used: *ICCG-n* stands for the ICCG solver in which the preconditioning matrix is formed with a complete Cholesky factorization and is updated with a refactorization when the number of PCG iterations becomes greater than n. *NSCG-n* stands for the NSCG solver using one term in the Neumann series expansion in which a refactorization of the stiffness matrix is performed when the number of PCG iterations becomes greater than n.

A cantilever beam is taken as the initial design domain for the topology optimization test example. The design domain for the optimum layout of the structures covers a large portion of the reference domain. Around areas of loading and support a non-design domain is used in order to take into account manufacturing constraints and to avoid taking into consideration high stress concentrations. The domain of the example is discretised using a fine mesh of hexahedral elements in order to give a good resolution of the final topology and to compute the stresses accurately. The initial values for the rejection rate, the evolution rate and the cut-off stress are taken as: 1%, 1% and 2,000 N/mm^2, respectively. The basic control parameters are chosen as follows: minimum and maximum number of elements to be switched-off in every optimization step are taken as 0.5% and 2% of the total elements, respectively. Convergence is achieved when all stress levels are within 70% of the maximum stress or when the number of active elements is less than 30% of the total number of elements, while isolated active elements are suppressed and no element growth is allowed. The switching-off of the elements in all examples is accomplished by dividing the elastic modulus for active elements by a factor of 10^5.

The cantilever beam is clamped on one face and loaded at the middle of the right-hand face with a vertical load as shown in Figure 3. The finite element mesh consists of 9,600 solid elements, 11,737 nodes and 34,848 d.o.f. resulting in a dense global stiffness matrix with narrow bandwidth. The characteristic d.o.f. for 4 and 8 subdomains are given in Table 6. The final layout of the structure is almost identical for all solution methods considered, and it is depicted in Figure 3. Table 7 depicts the total optimization time in sequential computing mode and the storage requirements. Table 8 demonstrates the performance of the one-level and two- level multi-domain methods operated on parallel computing mode in 4 and 8 processors using 4 and 8 subdomains, respectively.



(a)



(b)

Figure 3. 3D cantilever: (a) initial topology. (b) final topology

Subdomains	4	8
Total d.o.f.	34,848	34,848
Internal d.o.f.*	9,075	4,719
Ineterface d.o.f.	1,079	2,541

*of the larger subdomain

Table 6. 3D cantilever: Characteristic d.o.f. for 4 and 8 subdomains

method (4 subdomains-1 processor)	time (s)	storage (Mbytes)
Direct skyline	83,164	112
ICCG-4	41,785	69
NSCG-3	42,367	72
S-FETI	72,135	108
M-FETI	61,457	67
GSI(ICCG) M-FETI	43,863	71
GSI(NSCG) M-FETI	44,386	73

Table 7. 3D cantilever: Performance of the methods in sequential mode

method (4 subdomains-4 processors)	time (s)	storage (Mbytes)
S-FETI	24,281	108
M-FETI	20,693	67
GSI(ICCG) M-FETI	14,086	71
GSI(NSCG) M-FETI	14,278	73
Method (8 subdomains-8 processors)		
S-FETI	15,724	103
M-FETI	13,361	65
GSI(ICCG) M-FETI	9,176	69
GSI(NSCG) M-FETI	9,302	71

Table 8. 3D cantilever: Performance of the methods in parallel mode

4 EVOLUTIONARY OPTIMIZATION ALGORITHMS

Computer algorithms based on the process of natural evolution have been found capable to produce very powerful and robust search mechanisms although the similarity between these algorithms and the natural evolution is based on crude imitation of biological reality. The resulting Evolutionary Algorithms (EA) are based on a population of individuals, each of which represents a search point in the space of potential solutions of a given problem. These algorithms adopt a selection process based on the fitness of the individuals and some recombination operators. The best known EA in this class include evolutionary programming (EP) [24], Genetic Algorithms (GA) [31,41] and Evolution Strategies (ES) [69,75]. The first attempt to use evolutionary algorithms took place in the sixties by a team of biologists [10] and was focused in building a computer program that would simulate the process of evolution in nature.

Both GA and ES imitate biological evolution in nature and have three characteristics that differ from other conventional optimization algorithms: (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 a population of design points in the space of design variables. (iii) They can handle, with minor modifications continuous, discrete or mixed optimization problems. The second characteristic allows for natural implementation of GA and ES on a parallel computing environment [2,57,85].

In structural optimization problems, where the objective function and the constraints are highly non-linear functions of the design variables, the computational effort spent in gradient calculations required by the mathematical programming algorithms is usually large. In two recent studies by Papadrakakis *et al.* [59,64] it was found that probabilistic search algorithms are computationally efficient even if greater number of analyses is needed to reach the optimum. These analyses are computationally less expensive than in the case of mathematical programming algorithms since they do not need gradient information. Furthermore, probabilistic methodologies were found, due to their random search, to be more robust in finding the global optimum, whereas mathematical programming algorithms may be trapped in local optima.

4.1 Genetic Algorithms

GA are probably the best-known evolutionary algorithms, receiving substantial attention in recent years. The GA model used in this study and in many other structural design applications refers to a model introduced and studied by Holland and co-workers [41]. In general the term genetic algorithm refers to any population-based model that uses various operators (selection-crossover-mutation) to evolve. In the basic genetic algorithm each member of this population will be a binary or a real valued string, which is sometimes referred to as a *genotype* or, alternatively, as a *chromosome*.

Different versions of GA have appeared in the literature in the last decade dealing with methods for handling the constraints or techniques to reduce the size of the population of design vectors. In this section the basic genetic algorithms together with some of the most frequently used versions of GA are considered.

4.1.1 The Basic Genetic Algorithms

The three main steps of the basic GA

Step 0 *Initialization*

The first step in the implementation of any genetic algorithm is to generate an initial population. In most cases the initial population is generated randomly. In this study in order to perform a comparison between various optimization techniques the initial population is fixed and is chosen in the neighborhood of the initial design used for the mathematical programming methods. After creating an initial population, each member of the population is evaluated by computing its fitness function.

Step 1 *Selection*

Selection operator is applied to the current population to create an intermediate one. In the first generation the initial population is considered as the intermediate one, while in the next generations this population is created by the application of the selection operator.

Step 2 *Generation*

In order to create the next generation crossover and mutation operators are applied to the intermediate population to create the next population. Crossover is a reproduction operator, which forms a new chromosome by combining parts of each of the two parental chromosomes. Mutation is a reproduction operator that forms a new chromosome by making (usually small) alterations to the values of genes in a copy of a single parent chromosome. The process of going from the current population to the next population constitutes one generation in the evolution process of a genetic algorithm. If the termination criteria are satisfied the procedure stops otherwise returns to step 1.

Encoding

The first step before the activation of any operator is the step of encoding the design variables of the optimization problem into a string of binary digits (l's and 0's) called a *chromosome*. If there are n design variables in an optimization problem and each design variable is encoded as a *L-digit* binary sequence, then a chromosome is a string of $n \times L$ binary digits. In the case of discrete design variables each discrete values is assigned to a binary string, while in the case of continuous design variables the design space is divided into a number of intervals (a power of 2). The number of intervals $L+1$ depends on the tolerance given by the designer. If $s \in [s^\ell, s^u]$ is the decoded value of the binary string $< b_L b_{L-1} \cdots b_0 >$ then

$$s = DE(< b_L b_{L-1} \cdots b_0 >) = s^\ell + \frac{s^u - s^\ell}{2^L - 1} \left(\sum_{i=0}^{L} b_i \cdot 10^i \right) \tag{46}$$

where $DE(\cdot)$ is the function that performs the decoding procedure. In order to code a real valued number into the binary form the reverse procedure is followed.

Evaluation of fitness function

Apart from the objective function the so-called fitness function is also used by a genetic algorithm. The evaluation of a string refers to the evaluation of the objective function value of that string and it is independent to the evaluation of any other string. The fitness of that string, however, is always defined with respect to other members of the current population. The fitness is used to determine the selection probability of this chromosome to become the parent chromosome for the generation of the new chromosomes. In the basic genetic algorithm, fitness is defined by: $F_i'/\overline{F}'$ where F_i' is the penalized objective function associated with string i. $\overline{F}'$ is the average penalized objective function value of all the strings in the population. Fitness can also be assigned based on a string's rank in the population [9] or by sampling methods, such as tournament selection [30].

Selection

There are a number of ways to perform the selection. According to the *Tournament Selection* scheme each member of the intermediate population is selected to be the best member from a randomly selected group of members belonging to the current population. According to the *Roulette Wheel* selection scheme, the population is laid out in random order as in a pie graph, where each individual is assigned space on the pie graph in proportion to fitness. Next an outer roulette wheel is placed around the pie with N equally spaced pointers, where N is the size of the population. A single spin of the roulette wheel will now simultaneously pick all N members of the intermediate population.

Crossover

Crossover is a reproduction operator, which forms a new chromosome by combining parts of each of two "parent" chromosomes. The simplest form is called single-point crossover, in which an arbitrary point in the chromosome is picked. According to this operator two "offspring" chromosomes are generated, the first one is generated by copying all the information from the start up of the parent A to the crossover point and all the information from the crossover point to the end of parent B. The second "offspring" chromosome is generated by the reverse procedure. Variations exist which use more than one crossover point, or combine information from parents in other ways.

Mutation

Mutation is a reproduction operator, which forms a new chromosome by making (usually small) alterations to the values of genes in a copy of a single parent chromosome.

4.1.2 Micro Genetic Algorithms (μGA)

The micro genetic algorithm was introduced by Krishnakumar [46] and applied to simple mathematical test functions and to the wind shear optimal guidance problem. The main objective of this scheme is to reduce the size of the population compared to the basic one. This corresponds, in the case of structural optimization problems discretized with finite elements, to less finite element analyses per generation. It is a known fact that GA generally exhibit poor performance with small population size due to insufficient information processed and premature convergence to non-optimal results. A remedy to this problem, suggested by Goldberg [32], could be to restart the evolution process in case of nominal convergence with a new initial population, which will include the best solution already achieved. Based on this suggestion Krishnakumar proposed the μGA which can be described by the following steps:

Step 0 *Initialization*

The first step generates a population of size 5 either randomly or by generating 4 strings randomly and by selecting 1 good string from any previous search, or according to the experience of the designer.

Step 1 *Fitness evaluation*

In this step the fitness of each individual is evaluated and the best string is determined. The best string is labeled as string 5 and it is carried to the next generation (elitist strategy). In this way there is a guarantee that the information about good strings are not lost.

Step 2 *Generation*

According to the previous step the best individual of the current generation is carried out to the next one. The remaining four members of the next generation are chosen according to the tournament selection operator. After the selection operator is terminated the crossover operator is applied.

Step 3 *Convergence check*

If the termination criteria is satisfied the process ends, otherwise check for nominal convergence which is measured by bit wise convergence in case of binary coding or by comparing the design variables in case of real valued strings. If converged go to step 0, else return to step 1.

A modified version of μGA is tested in this study, where only feasible designs are accepted for the evolution process. This version, which resembles the death penalty treatment of the constraints adopted by ES, is abbreviated to mμGA.

4.1.3 Methods for handling the constraints

Although genetic algorithms are initially developed to solve unconstrained optimization problems during the last decade several methods have been proposed for handling constrained optimization problems as well. The methods based on the use of penalty functions are employed in the majority of cases for treating constraint optimization problems with GA. In this study methods belonging to this category have been implemented and will be briefly described in the following paragraphs.

Method of static penalties

In this simple method the objective function is modified as follows

$$F'(s) = \begin{cases} F^{(n)}(s), & \text{if } s \in \mathcal{F} \\ F^{(n)}(s) + p \cdot \text{viol}^{(n)}(s), & \text{otherwise} \end{cases} \tag{47}$$

where p is the static penalty parameter, $\text{viol}^{(n)}(s)$ is the sum of the violated constraints

$$\text{viol}(s) = \sum_{j=1}^{m} f_j(s) \tag{48}$$

and $F^{(n)}(s)$ is the objective function to be minimized, both normalized in [0,1], while $\mathcal{F}$ is the feasible region of the design space.

The sum of the violated constraints is normalized before it is used for the calculation of the modified objective function. The main advantage of this method is its simplicity. However, there is no guidance on how to choose the single penalty parameter p. If it is chosen too small the search will converge to an infeasible solution otherwise if it is chosen too large a feasible solution may be located but it would be far from the global optimum. A large penalty parameter will force the search procedure to work away from the boundary, where is usually located the global optimum, that divides the feasible region from the infeasible one.

Method of dynamic penalties

The method of dynamic penalties was proposed by Joines and Houck [43] and applied to mathematical test functions. As opposed to the previous method, the penalty parameter does not remain constant during the optimization process. Individuals are evaluated (at the generation g) by the following formula

$$F'(s) = F^{(n)}(s) + (c \cdot g)^{\alpha} \text{viol}^{(n)}(s) \tag{49}$$

with

$$\text{viol}(s) = \sum_{j=1}^{m} f_j^{\beta}(s) \tag{50}$$

where c, α and β are constants. A reasonable choice for these parameters was proposed as follows: $c = 0.5 \div 2.0$, $\alpha = \beta = 1$ or 2. For high generation number, however, the $(c \cdot g)^{\alpha}$ component of the penalty term takes extremely large values which makes even the slightly violated designs not to be selected in subsequent generations. Thus, the system has little chances to escape from local optima. In most experiments reported by Michalewicz [50] the best individual was found in early generations.

Augmented Lagrangian method

The Augmented Lagrangian method (AL-GA) was proposed by Adeli and Cheng [1,3]. According to this method the constrained problem is transformed to an unconstrained one, by introducing two sets of penalty coefficients $\gamma[(\gamma_1, \gamma_2, \cdots, \gamma_{M+N})]$ and $\mu[(\mu_1, \mu_2, \cdots, \mu_{M+N})]$. The modified objective function, for the generation g, is defined as follows

$$F'(s,\gamma,\mu) = \frac{1}{L_f}F(s)+\frac{1}{2}\left\{\sum_{j=1}^{N}\gamma_j^{(g)}[(q_j-1+\mu_j^{(g)})^+]^2+\sum_{j=1}^{M}\gamma_{j+N}^{(g)}\left[\left(\frac{|d_j|}{|d_j^a|}-1+\mu_{j+N}^{(g)}\right)^+\right]^2\right\} \tag{51}$$

where L_f is a factor for normalizing the objective function; q_j is a non-dimensional ratio related to the stress constraints of the j^{th} element group (see eqs. (62), (63)); d_j is the displacement in the direction of the j^{th} examined degree of freedom, while d_j^a is the corresponding allowable displacement; N, M correspond to the number of stress and displacement constraint functions, respectively:

$$(q_j-1+\mu_j^{(I)})^+ = \max(q_j-1+\mu_j^{(I)},0) \tag{52}$$

$$\left(\frac{|d_j|}{|d_j^a|}-1+\mu_{j+N}^{(I)}\right)^+ = \max\left(\frac{|d_j|}{|d_j^a|}-1+\mu_{j+N}^{(I)},0\right) \tag{53}$$

There is an outer step I and the penalty coefficients are updated at each step according to the expressions $\gamma_j^{(I+1)} = \beta\cdot\gamma_j^{(I)}$ and $\mu_j^{(I)} = \mu_j^{(I)}/\beta$, where $\mu_j^{(I+1)} = \mu_j^{(I)}+\max[\text{con}_{j,ave}^{(I)}, -\mu_j^{(I)}]$ and $\text{con}_{j,ave}^{(I)}$ is the average value of the j^{th} constraint function for the I^{th}th outer step, while the initial values of γ's and μ's are set equal to 3 and zero, respectively. Coefficient β is taken equal to 10 as recommended by Belegundu and Arora [11].

Segregated GA

The basic idea of the segregated GA (S-GA) [48] is to use two static penalty parameters instead of one, as in the method of static penalties. The two values of the penalty parameters are associated with two populations that have a different level of satisfaction of the constraints. Each of the groups corresponds to the best performing individuals with respect to the associated penalty parameter. The segregated GA can be described as follows:

Step 0 *Initialization*

Random generation of $2N$ designs. The objective functions of the designs $1, 2, \cdots, N$ are evaluated using the ph penalty parameter, while the remaining designs $N+1, \cdots, 2N$ are evaluated using the p_ℓ penalty parameter.

Step 1 *Selection*

An intermediate population of size N is created by selecting the best individuals from the two populations.

Step 2 *Generation*

Generate N offsprings using the basic operators mutation and crossover. The parents are evaluated using the ph penalty parameter while the offsprings using the p_ℓ. The process is then repeated by returning to *step 1.*

This version was used in [48] for the minimal weight design problem of a composite laminated plate.

4.2 Evolution Strategies (ES)

ES were proposed for parameter optimization problems in the seventies by Rechenberg [69] and Schwefel [75]. Some differences between GA and ES stem from the numerical representation of the design variables used by these two algorithms. The basic GA operate on fixed-sized bit strings which are mapped to the values of the design variables, ES work on real-valued vectors. Another difference can be found in the use of the genetic operators. Although, both GA and ES use the mutation and recombination (crossover) operators, the role of these genetic operators is different. In GA mutation only serves to recover lost alleles, while in ES mutation implements some kind of hill-climbing search procedure with self-adapting step sizes σ (or γ). In both algorithms recombination serves to enlarge the diversity of the population, and thus the covered search space. There is also a difference in treating constrained optimization problems where in the case of ES the death penalty method is always used, while in the case of GA only the augmented Lagrangian method can guarantee the convergence to a feasible solution. The ES, however, achieve a high rate of convergence than GA due to their self-adaptation search mechanism and are considered more efficient for solving real world problems [40]. The ES were initially applied for continuous optimization problems, but recently they have also been implemented in discrete and mixed optimization problems [85,86]. The ES algorithms used in the present study are based on the work of Thierauf and Cai who applied the ES methodologies in sizing structural optimization problems having discrete and/or continuous design variables [85,86]. In the following paragraphs different versions of ES algorithms are discussed and compared in some test examples.

4.2.1 ES Algorithms

The ES can be divided into a two-membered evolution strategy (2-ES) or a multi-membered evolution strategy (M-ES).

The two-member ES

The earliest evolution strategies were 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 ES 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 $s_0^{(g)}$, whose genotype is slightly different from that of the parent

$$s_0^{(g)} = s_p^{(g)} + z^{(g)} \tag{54}$$

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

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

$$s_p^{(g+1)} = \begin{cases} s_0^{(g)} & \text{if } g_i(s_0^{(g)}) \leq 0 \quad i = 1, 2, \cdots, 1 \quad \text{and } f(s_0^{(g)}) \leq f(s_p^{(g)}) \\ s_p^{(g)} & \text{otherwise} \end{cases} \tag{55}$$

The question how to choose the random vector $z^{(g)}$ in *Step 1* is very important. This choice has the role of mutation. 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, it is natural choice to use a probability distribution according to which small changes occur frequently, but large ones only rarely. Two requirements arise together by analogy with natural evolution: (i) the expected mean value ξ_i for a component $z_i^{(g)}$ to be zero; (ii) the variance σ_i^2, the average squared deviation from mean value, is small.

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

$$p(z_i^{(g)}) = \frac{1}{\sqrt{(2\pi)}\sigma_i} \exp\left(-\frac{(z_i^{(g)} - \xi_i)^2}{2 - \sigma_i^2} \right) \tag{56}$$

when $\xi_i = 0$ the so-called $(0, \sigma_i)$ 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 further more closely linked to the convergence behaviour.

Multi-membered ES

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 ES:

$(\mu + \lambda)$-ES:	The best μ individuals are selected from a temporary population of $(\mu+\lambda)$ individuals to form the parents of the next generation.
(μ, λ)-ES:	The μ individuals produce 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)-ES 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 $\tilde{s} = [\tilde{s}_1, \tilde{s}_2, \cdots, \tilde{s}_n]^T$ is first built by means of recombination. For continuous problem the following recombination cases can be used

$$\tilde{s}_i = \begin{cases} s_{\alpha,i} \quad \text{or} \quad s_{b,i} \quad \text{randomly} & \text{(a)} \\ 1/2(s_{\alpha,i} + s_{b,i}) & \text{(b)} \\ s_{bj,i} & \text{(c)} \\ s_{\alpha,i} \quad \text{or} \quad s_{bj,i} \quad \text{randomly} & \text{(d)} \\ 1/2(s_{\alpha,i} + s_{bj,i}) & \text{(e)} \end{cases} \tag{57}$$

where $\tilde{s}_i$ is the i-th component of the temporary parent vector $\tilde{s}$, $s_{\alpha,i}$ and $s_{b,i}$ are the i-th components of the vectors s_a and s_b which are two parent vectors randomly chosen from the population. In case (57c), $\tilde{s}_i = s_{bj,i}$ means that the i-th component of $\tilde{s}$ is chosen randomly from the i-th 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 ES (eq. (54)).

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.

4.2.2 ES in structural optimization problems

The ES optimization procedure starts with a set of parent vectors and if any of these parent vectors gives an infeasible design then this parent vector is modified until it becomes feasible. Subsequently, the offsprings are generated and checked if they are in the feasible region. The computational efficiency of the multi-membered ES is affected by the number of parents and offsprings involved. It has been observed that values of μ and λ should be close the number of the design variables produce best results [64].

The ES algorithm for structural optimization applications can be stated as follows:

1. *Selection step*: selection of s_i $(i = 1, 2, \cdots, \mu)$ parent vectors of the design variables
2. *Analysis step*: solve $K(s_i)u_i = f$ $(i = 1, 2, \cdots, \mu)$, where K is the stiffness matrix of the structure and f is the loading vector
3. *Constraints check*: all parent vectors become feasible
4. *Offspring generation*: generate s_j, $(j = 1, 2, \cdots, \lambda)$ offspring vectors of the design variables
5. *Analysis step*: solve $K(s_j)u_j = f$ $(j = 1, 2, \cdots, \lambda)$
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*

4.2.3 Contemporary ES (C-ES) - The (μ, λ, θ) Evolution Strategies

This is a more general ES version, which was proposed by Schwefel and Rudolph [74] for application in continuous problems but has not been applied either to continuous or to discrete optimization problems [71]. Considering the two schemes of the multimembered evolution strategy, namely the $(\mu + \lambda)$ and the (μ, λ) ES, only empirical results have shown that the "plus" version performs better in structural optimization problems [18,75]. The

(μ, λ)-ES version is in danger to diverge because the so far best position is not preserved within the generation cycle (the so-called elitist strategy). The "comma" version implies that each parent can have children only once (duration of life: one generation or one reproduction cycle), whereas in the "plus" version individuals may live eternally if no child achieves a better or at least the same improvement in the objective function.

The C-ES introduce a maximal life span of $\theta \geq 1$ reproduction cycles which gives the "comma" scheme for $\theta = 1$ and the "plus" one for $\theta = \infty$. If $\mu \geq 1$ is the number of parents, $\lambda > \mu$ is the number of offsprings, then ρ with $1 \leq \rho \leq \mu$ is the number of ancestors for each descendant. This ES version differs in two points from the basic one: (i) Free number of parents are involved in reproduction ranging from 1 to μ. (ii) A finite number of reproduction cycles per individual is performed, not one (1) or infinite (∞) as for the "comma" and the "plus" schemes, respectively. The selection operator used in the C-ES can be similar to the one used by the genetic algorithms.

4.2.4 Adaptive ES (A-ES)

The handling of the constraints by the basic ES is based on the death penalty approach [8], where every infeasible design point is discarded. Thus the process is directed to search only in the feasible region of the design space. Due to this approach many designs that are examined by the optimizer during the search process and are close to the acceptable design space are rejected leading to the loss of valuable information. The idea introduced in this work is to use soft constraints during the first stages of the search and as the search approaches the region of the global optimum the constraints to become more severe until they reach their real values.

The implementation of A-ES in structural optimization problems is straightforward and follows the same steps described in the section of the basic ES. The ES optimization procedure starts with a population of parent vectors, while a level of violation of the constraints is determined. If any of these parents corresponds to an infeasible design lying outside the extended design space then this parent is modified until it becomes "feasible". Then the offsprings are generated and checked if they are in the "feasible" region according the current level of violation. In every generation the values of the objective function are compared between 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 termination criterion is satisfied.

In this adaptive scheme a nominal convergence check is adopted for the determination of the level of violation of constraints. Nominal convergence occurs when the mean value of the objective function of the designs of the current population is relatively close to the best design achieved until the current generation, according to the expression

$$\frac{\overline{F}^{(g)} - F_{best}^{(g)}}{\overline{F}^{(g)}} \geq \varepsilon_{ad} \tag{58}$$

where $\overline{F}^{(g)}$ is the mean objective function value, $F_{best}^{(g)}$ is the best objective function value of all parents in the g-th generation, and $\varepsilon_{ad} = 0.05$.

The A-ES steps can be stated as follows:

1. *Initialization step*: selection of s_i $(i = 1, 2, \cdots, \mu)$ parent vectors of the design variables and the percentage of violation of the constraints v0 (usually taken between 20-50%)

2. *Analysis step*: solve $K(s_i)u_i = f$ $(i = 1, 2, \cdots, \mu)$

3. *Constraints check*: all parent vectors become "feasible", within the prescribed level of constraints violation v_0
4. *Offspring generation*: generate s_j, $(j = 1, 2, \cdots, \lambda)$ offspring vectors of the design variables
5. *Analysis step*: solve $K(s_j)u_j = f$ $(j = 1, 2, \cdots, \lambda)$
6. *Nominal convergence check*: if nominal convergence has occurred the level of violation v_g becomes more severe by reducing its value by a small quantity (usually 0.1 or 0.2)
7. *Constraints check*: if satisfied according to the current level of violation v_g continue, else change s_j and return to *step 4*
8. *Selection step*: selection of the next generation parents according to $(\mu + \lambda)$ or (μ, λ)
9. *Convergence check*: if satisfied stop, else go to *step 3*

4.2.5 ES for discrete optimization problems

In engineering practice the design variables are not continuous because usually the structural parts are constructed with certain variation of their dimensions. Thus design variables can only take values from a predefined discrete set. For the solution of discrete optimization problems Thierauf and Cai [85] have proposed a modified ES algorithm. The basic differences between discrete and continuous ES are focused on the mutation and the recombination operators. In the discrete version of ES the random vector $z^{(g)}$ is properly generated in order to force the offspring vector to move to another set of discrete values.

The fact that the difference between any two adjacent values can be relatively large is against the requirement that the variance σ_i^2 should be small. For this reason it is suggested that not all the components of a parent vector, but only a few of them (eg. ℓ) should be randomly changed in every generation. This means that $n - \ell$ components of the randomly changed vector $z^{(g)}$ will have zero value. In other words, the terms of vector $z^{(g)}$ are derived from

$$z_i^{(g)} = \begin{cases} (\kappa + 1)\delta s_i & \text{for } \ell \text{ randomly chosen components} \\ 0 & \text{for } n - \ell \text{ other components} \end{cases} \tag{59}$$

where δs_i is the difference between two adjacent values in the discrete set and κ is a random integer number which follows the Poisson distribution

$$p(\kappa) = \frac{(\gamma)^\kappa}{\kappa!} e^{-\gamma} \tag{60}$$

γ is the standard deviation as well as the mean value of the random number κ. The choice of ℓ depends on the size of the problem and it is usually taken as the 1/5 of the total number of design variables. The ℓ components are selected using uniform random distribution in every generation.

For discrete optimization the procedure terminates when one of the following termination criteria is satisfied: (i) when the best value of the objective function in the last $4 * n * \mu/\lambda$ generations remains unchanged, (ii) when the mean value of the objective values from all parent vectors in the last $2 * n * \mu/\lambda$ generations has not been improved by less than a given value $\varepsilon_b (= 0.0001)$, (iii) when the relative difference between the best objective function value and the mean value of the objective function values from all parent vectors in the current generation is less than a given value $\varepsilon_c(= 0.0001)$, (iv) when the ratio μ_b/μ has reached a given value $\varepsilon_d(= 0.5 to 0.8)$ where μ_b is the number of the parent vectors in the current generation with the best objective function value.

4.3 Hybrid Optimization Algorithms

Several hybrid optimization algorithms which combine evolutionary computation techniques with deterministic procedures for numerical optimization problems have been recently investigated. Papadrakakis *et al.* [64] used evolution strategies with the SQP method, while Waagen *et al.* [91] combined evolutionary programming with the direction set method of Hooke and Jeeves [42]. The hybrid implementation proposed in [64] was found very successful on shape optimization problems, while the method proposed in [91] was applied to unconstrained mathematical test functions. Myung *et al.* [52] considered a similar to Waagen *et al.* approach, but they experimented with constrained mathematical test functions. Myung *et al.* combined a floating-point evolutionary programming technique, with a method-developed by Maa and Shanblatt [49] applied to the best solution found by the evolutionary programming technique. The second method iterates until the system defined by the combination of the objective function, the constraint functions and the design variables reach equilibrium.

A characteristic property of the SQP based optimizers is that they usually capture very fast the right path to the nearest optimum, irrespective of its nature of local or global optimum. However, after locating the area of this optimum it might oscillate until all constraints are satisfied since it is observed that even small constraint violations often slow down the convergence rate of the method. On the other hand EA proceed with slower rate, due to their random search, but the absence of strict mathematical rules, which govern the convergence rate of the mathematical programming methods, make EA less vulnerable to local optima and therefore it is much more likely to converge towards the global optimum in non-convex optimization problems. These two facts gave the motivation to combine EA with MP methodologies. Between the two EA examined in this study the basic genetic algorithms seems to be faster than evolution strategies since they do not always operate on the feasible region of the design space as evolution algorithms. However, they are most often found unable to converge to feasible designs.

In order to benefit from the advantages of both methodologies a hybrid approach is proposed, which combines the two optimization methodologies in an effort to increase the robustness and the computational efficiency of the optimization procedure. Two combinations of SQP and EA methodologies are implemented : (i) In the first approach the SQP method is used first, giving a design very close to the optimum, followed by EA 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

$$\left| \frac{f_{j+1} - f_j}{f_j} \right| \leq \varepsilon \tag{61}$$

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 EA procedure, either GA or ES, is used first in order to locate the region where the global optimum lies, and then the SQP is activated in order to exploit its higher order of accuracy in the neighbourhood of the optimum. In this case the switch is performed when there is a small difference ($\varepsilon = 0.1$) 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 and is perfectly suited to GA since it improves the fast-converged solution to an infeasible design by GA. Furthermore combination of GA and ES are performed in which ES are used to improve the quality of the solution achieved by GA.

4.4 Solving The FE Equilibrium Problems Within Evolutionary Algorithms

In the case of ES the optimization procedure starts with a set of parent vectors. If any of these parent vectors gives an infeasible design then this parent vector is modified until it becomes feasible. Subsequently, the offsprings are generated and checked if they are in the feasible region. According to $(\mu + \lambda)$ selection scheme in every generation the values of the objective function of the parent and the offspring vectors are compared and the worst vectors are rejected, while the remaining ones are considered to be the parent vectors of the new generation. On the other hand, according to (μ, λ) selection scheme only the offspring vectors of each generation are used to produce the new generation. This procedure is repeated until the chosen termination criterion is satisfied.

The algebraic definition of an evolution strategy procedure applied into a structural system with the finite element equation $Ku = f$ may be described as follows:

1. *Initialization*: selection of s_i $(i = 1, 2, \cdots, \mu)$ parent vectors of the design variables
   set $K_0 = K(s_1)$
   solve $K_0 u_1 = f$
   $$(K_0 + \Delta K(s_i))u_i = f, \qquad (i = 2, 3, \cdots, \mu)$$
2. *Constraints check*
3. *Offspring generation*: generate s_j, $(j = 1, 2, \cdots, \lambda)$ offspring vectors of the design variables
4. *Solution step*:
   $$(K_0 + \Delta K(s_j))u_j = f, \qquad (j = 1, 2, \cdots, \lambda)$$
5. *Constraints check*: if satisfied continue, else change sj and go to *step 4*
6. *Selection step*: selection of the next generation parents according to $(\mu + \lambda)$ or (μ, λ) selection schemes
7. *Convergence check*: if satisfied stop, else go to *step 3*

It can be seen that the solution of at least λ systems of finite element equations need to be solved at each generation, where $\Delta K(s_j)$ defines the modification of the stiffness matrix due to the changes on the design variables and is generally small compared to K_0. Similar steps are performed with a GA optimization algorithm in which the constraint checks in steps 2 and 5 are omitted.

4.4.1 Single- and multi-domain methods for solving $(K_0 + \Delta K_i)u_i = f_i$

The finite element equilibrium equations required to be solved a number of times at each optimization step, constitute a typical nearby problem similar to that discussed in the GFD sensitivity analysis and topology optimization problems. The implementation of hybrid solution schemes, based on a combination of direct and preconditioned iterative methods implemented on single structural domains as well as the multi-domain methods implemented on sequential or parallel computing environments, as discussed in sections 3.2.1 and 3.2.2, may drastically reduce the time required for the solution of the finite element equations with an overall beneficial effect on the efficiency of the optimization procedure. The methods are properly modified, as in previous applications, to address the special features of the particular optimization problems at hand, while mixed precision arithmetic operations are proposed resulting in additional savings in computer time and storage without affecting the accuracy of the solution.

4.4.2 Neural Networks

Basic principles of artificial neural networks theory

The aim of the present study is to train a neural network (NN) to provide computationally inexpensive estimates of analysis outputs required during the optimization process. A trained network presents some distinct advantages over the numerical computing paradigm. It provides a rapid mapping of a given input into the desired output quantities, thereby enhancing the efficiency of the redesign process. This major advantage of a trained NN over the conventional procedure, under the provision that the predicted results fall within acceptable tolerances, leads to results that can be produced in a few clock cycles, requiring order(s) of magnitude less computational effort than the conventional computational process. The learning algorithm which was employed for the training is the well known Back Propagation (BP) algorithm [72].

In the present implementation the objective is to investigate the ability of the NN to predict accurate structural analysis outputs that are necessary for the EA optimizer. This is achieved with a proper training of the NN. The NN training comprises the following tasks: (i) select the proper training set, (ii) find a suitable network architecture and (iii) determine the appropriate values of characteristic parameters such as the learning rate and momentum term.

An important factor governing the success of the learning procedure of a NN architecture is the selection of the training set. A sufficient number of input data properly distributed in the design space together with the output data resulting from complete structural analyses are needed for the BP algorithm in order to provide satisfactory results. Overloading the network with unnecessary similar information results to over training without increasing the accuracy of the predictions. A few tens of structural analyses have been found sufficient for the examples considered to produce a satisfactory training of the NN. Ninety percent of those runs are used for training and the rest is used to test the results of the NN.

Most researchers split the design space into subregions and try to combine randomly the values within each subregion in order to obtain a training set which is representative of the whole design space. This procedure leads frequently to a huge number of training patterns in order to ensure that the whole design space is properly represented. In an effort to increase the robustness as well as the computational efficiency of the NN procedure various types of training set selection were investigated in a previous study [59]. In this study two types of training set selection are used: (i) the training set is chosen automatically based on a Gaussian distribution of the design variables around the midpoints of the design space, (ii) the training set is chosen using data from the structural analyses performed in the framework of ES optimization steps until the computed designs reach a plateau near the optimum. The NN training is then activated when the value of objective function remains unchanged for a number of ES generations.

The first type of the training set selection was motivated from the fact that usually the searching for the optimum and its location lies in the region near the midpoints of the design space. A Gaussian distribution was therefore used for the random selection of input data in order to cover the whole design space and enforce the selection of most input patterns around the midpoints of the design space. This approach proved to be more efficient than choosing randomly combinations of input data from the whole range of the design space using a uniform distribution of the design variables [59]. The second type of the training set selection is based on the fact that in most cases the EA optimizer very fast tracks the path to the optimum and then it may oscillate around it until convergence is achieved at a slower rate. Therefore it is more efficient to produce the training sets in the vicinity of the design point where the optimizer has reached a stationary point. This way a smaller number of training sets are required and the NN training is performed much faster and accurately.

Structural optimization based on EA and NN

After the selection of the suitable NN architecture the training procedure is performed using a number (M) of data sets, selected as described previously, in order to obtain the I/O pairs needed for the NN training. Since the NN based structural analysis can only provide approximate results it is recommended that a correction on the output values should be performed in order to alleviate any inaccuracies entailed, especially when the constraint value is near the limit which separates the feasible and the infeasible region. This is achieved with a relaxation of this limit during the NN testing phase before entering the optimization procedure. A "correction" of the allowable constraint values was therefore performed proportional to the maximum testing error of the NN configuration. The maximum testing error is the largest average error of the output values among testing patterns. Whenever the predicted values were found smaller than those derived from a conventional structural analysis the allowable values of the constraints were decreased according to the maximum testing error of the NN configuration and vice versa.

The proposed EA-NN methodology can be described, for the case of ES, with the following algorithms according to the two types of training set selection schemes that were previously described:

Algorithm 1

The combined ES-NN optimization procedure is performed in two phases. The first phase includes the training set selection, the structural analyses required to obtain the necessary I/O data for the NN training, and finally the selection, training and testing of a suitable NN configuration. The second phase is the ES optimization stage where the trained NN is used to predict the response of the structure in terms of objective and constraints function values, due to different sets of design variables, instead of the standard structural analysis computations.

The proposed methodology ES-NN can be described with the following algorithm:

- NN training phase:
  1. *Training set selection step*: select $(i = 1, 2, \cdots, M)$ input patterns
  2. *Structural analysis step*: solve $K(s_i)u_i = f \ (i = 1, \cdots, M)$
  3. *Training step*: selection and training of a suitable NN architecture
  4. *Testing step*: test NN and "correct" allowable constraint values
- ES-NN optimization phase:
  1. *Selection step*: selection of $s_i \ (i = 1, 2, \cdots, \mu)$ parent vectors of the design variables
  2. *Prediction step*: using NN to compute optimization function values for the μ parent vectors
  3. *Constraints check*: all parent vectors become feasible

4. *Offspring generation*: generate s_j, $(j = 1, 2, \cdots, \lambda)$ offspring vectors of the design variables
5. *Prediction step*: using NN to compute optimization function values for the λ offspring vectors
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*

Algorithm 2

According to the second type of training set selection the proposed ES-NN methodology can be described with the following algorithm: The combined ES-NN optimization procedure is performed in three phases. The first phase is the ES optimization stage until a stationary point is obtained. This is the case when the mean value of the objective values from all parent vectors in the last $n * \mu/\lambda$ generations has not been improved by less than a given value $\varepsilon_d (= 0.05)$. The second phase includes the training set selection in the vicinity of the stationary point from the previous structural analyses during previous ES steps. This way the necessary I/O data required for the NN training are obtained, and finally the selection, training and testing of a suitable NN configuration. The third phase is identical to the second phase of algorithm 1.

The second algorithm is described as follows:

- ES optimization phase:
  1. *Selection step*: selection of s_i $(i = 1, 2, \cdots, \mu)$ parent vectors of the design variables
  2. *Analysis step*: solve $K(s_i)u_i = f$ $(i = 1, 2, \cdots, \mu)$
  3. *Constraints check*: all parent vectors become feasible
  4. *Offspring generation*: generate s_j , $(j = 1, 2, \cdots, \lambda)$ offspring vectors of the design variables
  5. *Analysis step*: solve $K(s_j)u_j = f$ $(j = 1, 2, \cdots, \lambda)$
  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. *Stationarity check*: If satisfied continue, else go to *step 3*
- NN training phase:
  1. *Training set selection step*: choose s_i $(i = 1, 2, \cdots, M)$ I/O data
  2. *Training step*: selection and training of a suitable NN architecture
  3. *Testing step*: test NN and "correct" allowable constraint values
- ES-NN optimization phase: as in algorithm 1

Similar consideration could be applied to GA-NN in a straightforward manner.

4.5 Numerical Tests

4.5.1 Sizing optimization with evolutionary algorithms

A twenty-storey space frame [60] and a double-layered space roof [58] were selected as test problems for comparing various optimization algorithms in sizing optimization. All tests were performed on a SG Power Challenge computer with the R4000 processor.

Twenty-storey space frame

The space frame, shown in Figure 4, has 1,020 members and 2,400 degrees of freedom, with modulus of elasticity $E = 200$ GPa and the yield stress $\sigma_y = 250$ MPa. The cross section of each member is assumed to be a I-shape and for each member two design variables are considered as. The values of b and h are selected from an integer design space, while t and w are given as follows: $f = 0.06h + 0.10(b - 10)$, $w = 0.625f$. Those two expressions make sure that the web thickness is less than b, the opposite of which would have been not acceptable. The objective function of the problems is the weight of the structure. The constraints are the member stresses and the inter-storey drifts. For rigid frames in rolled I-shapes, under allowable stress design requirements specified by Eurocode 3 [20], the stress constraints are defined by the non-dimensional ratio q of interaction formulas

$$q = \frac{f_a}{F_a} + \frac{f_b^y}{F_b^y} + \frac{f_b^z}{F_b^z} \leq 1.0 \qquad \text{if } \frac{f_a}{F_a} \leq 0.15 \tag{62}$$

and

$$q = \frac{f_a}{0.60 \cdot \sigma_y} + \frac{f_b^y}{F_b^y} + \frac{f_b^z}{F_b^z} \leq 1.0 \qquad \text{if } \frac{f_a}{F_a} \leq 0.15 \tag{63}$$

where f_a is the computed compressive axial stress, f_b^y, f_b^z are the computed bending stresses for y and z axis, respectively. F_a is the allowable compressive axial stress, F_b^y, F_b^z are the allowable bending stresses for y and z axis, respectively, and σ_y is the yield stress of the steel. The allowable inter-storey drift is limited to 1.5% of the height of each storey. One load case is considered in all examples.

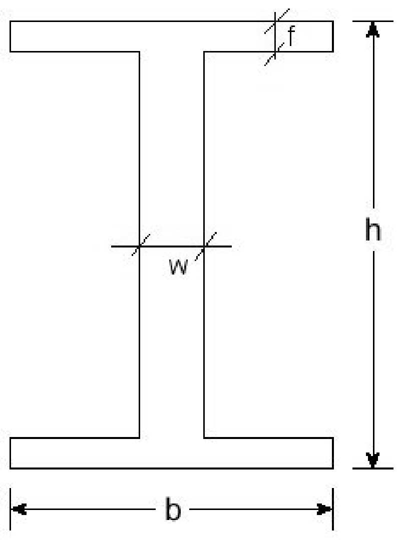


Figure 4. I-shaped cross-section design variables

The loads considered here are uniform vertical forces applied at joints equivalent to uniform load of 4.8 kPa and horizontal forces equivalent to uniform forces of 1.0 kPa on the largest surface. The element members are divided into 11 groups shown in Figure 5 and the total number of design variables is 22. The initial design used in this example was chosen away from the optimum corresponding to the weight of 42,248 kN for every test.

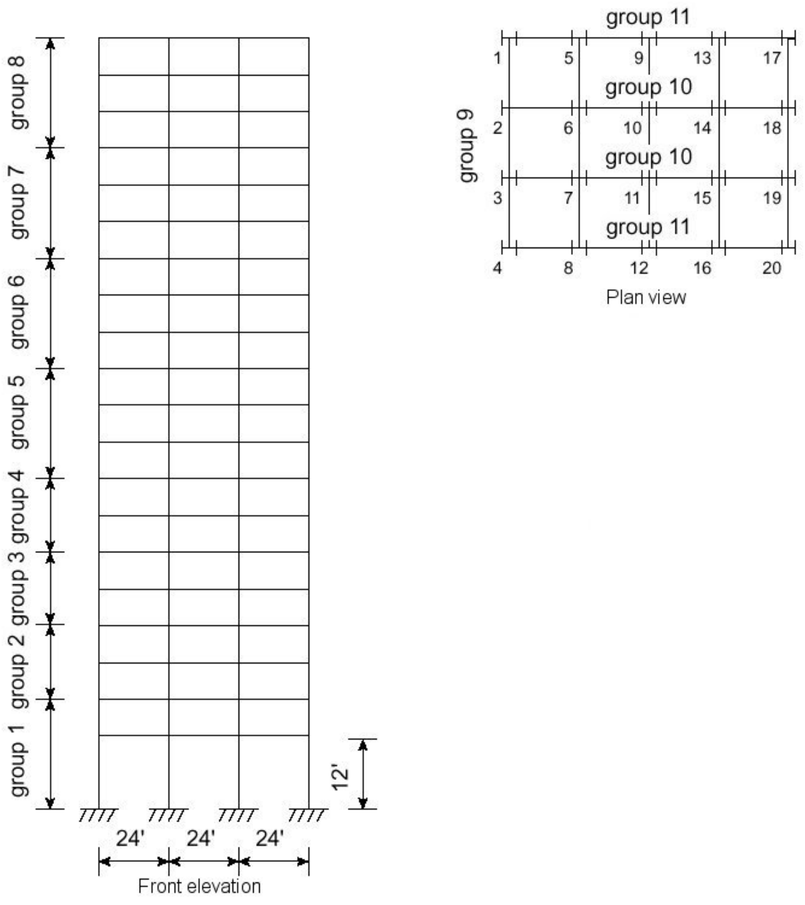
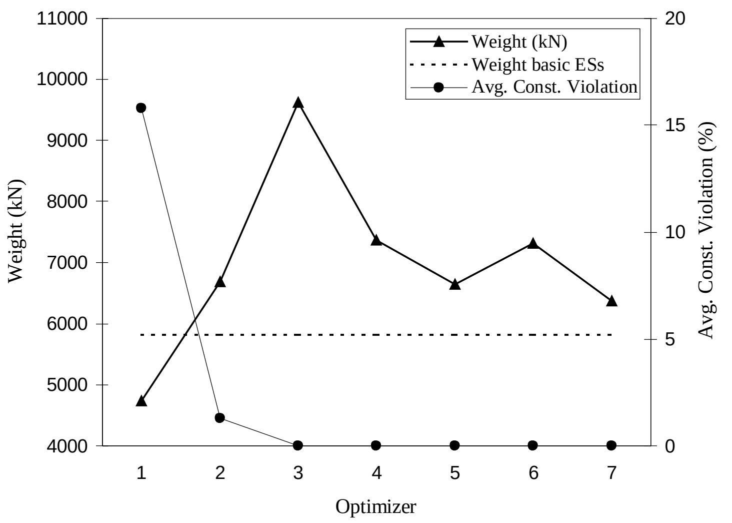


Figure 5. Twenty storey space frame

Table 9 shows the performance of the two types of the multi-membered ES, namely (10+10)-ES and (10,10)-ES used as a basis for comparison. The (10+10) version of ES manages to converge to the best design, which is used as reference design, at approximately half the time required by the (10,10) scheme. In Figures 6 and 7 various techniques for handling the constraints by the GA are presented. It can be seen that all feasible solutions achieved appear to be local optima although for this example the GA with dynamic penalties showed a more robust behaviour.

Optimizer	Weight (kN)	FE analyses	Time (s)
(5+5)-ES	6,028	240	3,708
(5,5)-ES	6,085	452	6,984
(10+10)-ES	5,819	422	6,519
(10,10)-ES	5,877	727	11,230

Table 9. Twenty storey space frame: Performance of the two selection schemes $(\mu + \lambda)$-ES and (μ, λ)-ES



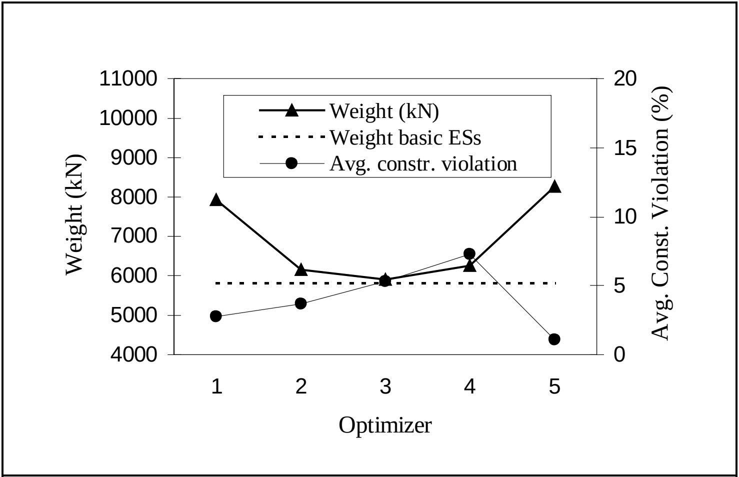
Optimizer	Parameters	FE analyses
1	p=1	360
2	p=50	290
3	p=500	120
4	c=0.5	220
5	c=1	260
6	c=1.5	150
7	c=2	190

Figure 6. Twenty storey space frame: Optimizer (1-3) GA with static penalties; Optimizer (4-7) GA and dynamic penalties ($\alpha = \beta = 2$)

Tables 10 and 11 contain the results of the Augmented Lagrangian GA method (AL-GA). The constraint violation percentage is equal to zero for all tests considered. Table 10 depicts the results of the method with the termination criteria suggested by Adeli and Cheng in [3], while Table 11 includes the results obtained for the termination criteria used for the rest of the methods. For the adopted termination criterion we have examined four different cases according to the allowable number of generations with no improvement of the objective function. It can be seen that both termination criteria converge to the same result which appears to be a local minimal compared to the minimum achieved by the ES as shown in Table 9. It can also be seen that the Augmented Lagrangian GA method is not affected by the value of normalization parameter L_f.

L_f	Weight (kN)	FE analyses	Time (s)
100	7,015	195	3,097
500	7,015	195	3,097
1000	7,015	195	3,097

Table 10. Twenty storey space frame: Performance of the AL-GA (termination criterion of Ref.[3])



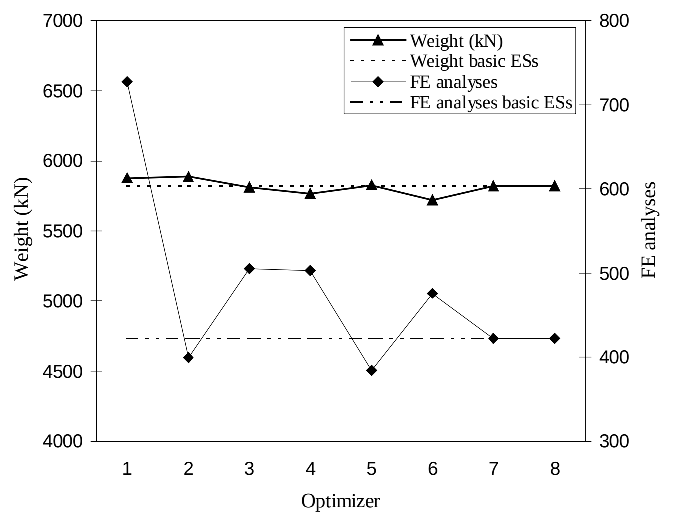
Optimizer	Penalty Parameters (p_h, p_ℓ)	FE analyses
1	(5, 0.5)	65
2	(10,0.5)	141
3	(2,1)	147
4	(25,1)	93
5	(50,1)	77

Figure 7. Twenty storey space frame - Performance of S-GA (p_ℓ, p_h) scheme

L_f	Weight (kN)	FE analyses	Time (s)
	case a: 6μ/λ		
100	8,073	120	1,917
500	8,073	120	1,917
1000	8,073	120	1,917
	case b: 12μ/λ		
100	7,397	140	2,231
500	7,397	140	2,231
1000	7,397	140	2,231
	case c: 18μ/λ		
100	7,015	195	3,097
500	7,015	195	3,097
1000	7,015	195	3,097
	case d: 24μ/λ		
100	7,015	195	3,097
500	7,015	195	3,097
1000	7,015	195	3,097

Table 11. Twenty storey space frame: Performance of the AL-GA with the termination criterion adopted in this work

Figures 8 and 9 depict the performance of the contemporary and adaptive evolution strategies, namely C-ES and A-ES, where all designs appearing in those two figures are feasible. A comparison of the results of Table 9 and those depicted in Figures 6 and 7 indicates that contemporary and adaptive ES outperform in most cases the basic version of evolution strategies in terms of the achieved optimum weight at the expense of more computing time.



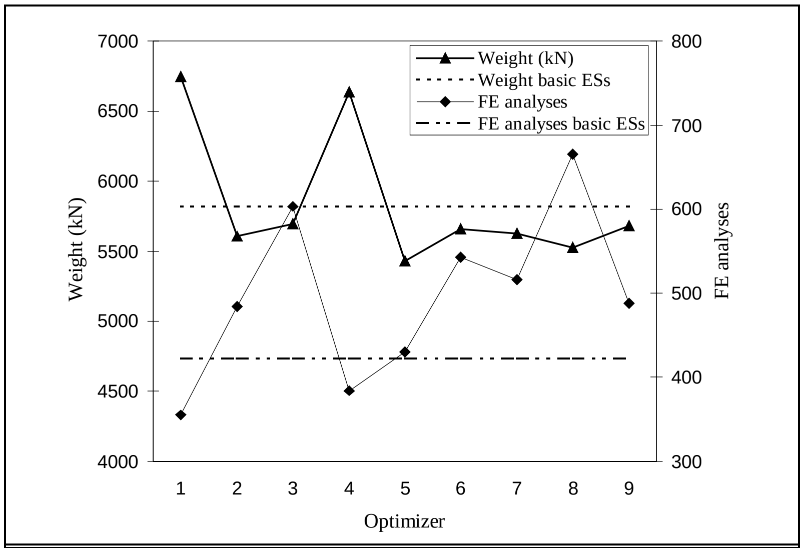
Optimizer	Parameters (μ,λ,θ)
1	(10,10,1)
2	(10,10,2)
3	(10,10,4)
4	(10,10,6)
5	(10,10,8)
6	(10,10,10)
7	(10,10,15)
8	(10,10,20)

Figure 8. Twenty storey space frame - Performance of the C-ES

Tables 12-14 depict the performance of hybrid approaches for this test example. Three initial designs are considered, one close, one medium and one away 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 SQP approach. Furthermore, GA-SQP, ES-SQP and AL-GA-SQP optimizers manage to converge to final designs with 12% less weight than the SQP optimizer at one fifth of computing time. We also examined two EA hybrid methods by combining AL-GA with ES and vice versa. Both perform well in terms of the design achieved and the required computing time. The mathematical optimizer using the ESA sensitivity analysis method is again faster than GFD method while the magnitude of perturbation is equal to 10^{-5}.

Double-layered space truss

The second test example is the three dimensional double-layered space roof truss depicted in Figure 10 with discrete design variables. Space truss structures usually have the topology of single or multi-layered flat or curved grids that can be easily constructed in practice. Most frequently the objective function is the weight or the volume of the structure and the constraints are the member stresses, nodal displacements, or frequencies. The stress constraints can be written as $|\sigma| \leq |\sigma_a|$, where σ is the maximum axial stress in each element



Optimizer	Parameters (ε_{ad}, α_i, α_f)
1	(0.01, 1.2, 1.0)
2	(0.01, 1.4, 1.0)
3	(0.01, 1.6, 1.0)
4	(0.03, 1.2, 1.0)
5	(0.03, 1.4, 1.0)
6	(0.03, 1.6, 1.0)
7	(0.05, 1.2, 1.0)
8	(0.05, 1.4, 1.0)
9	(0.05, 1.6, 1.0)

Figure 9. Twenty storey space frame - Performance of the A-ES ($b = 0.1$)

Optimizer	Initial Design for 2nd optimizer	Final design (kN)	Sensitivity Analysis	Time (s)		Time (s)		
				EA	SQP	EA	SQP	Total
SQP	-	6,427	GFD	-	435	-	21,932	21,932
SQP	-	6,427	ESA	-	438	-	13,655	13,655
S-GA-SQP	7,928v	5,764	GFD	65	116	1,100	5,166	6,266
S-GA-SQP	7,928v	5,764	ESA	65	117	1,100	3,608	4,708
AL-GA-SQP	7,015	6,427	GFD	195	99	3,301	4,415	7,716
AL-GA-SQP	7,015	6,427	ESA	195	99	3,301	3,077	6,378
C-ES-SQP	7,132	5,834	GFD	193	152	3,266	6,770	10,036
C-ES-SQP	7,132	5,834	ESA	193	152	3,266	4,687	7,953
A-ES-SQP	6,531	5,713	GFD	107	111	1,811	4,943	6,754
A-ES-SQP	6,531	5,713	ESA	107	110	1,811	3,392	5,202
AL-GA-ES	7,015	5,819	-	195+65	-	4,401	-	4,401
mμGA	-	5,772	-	395	-	6,117	-	6,117

Table 12. Twenty storey space frame: Hybrid methods (bad initial design)

group for all loading cases, $\sigma_a = 0.60 \times \sigma_y$ is the allowable axial stress and σ_y is the yield stress. Similarly, the displacement constraints can be written as $|d| \leq d_a$, where d_a is the limiting value of the displacement at a certain node, or the maximum nodal displacement.

Optimizer	Initial Design for 2nd optimizer	Final design (kN)	Sensitivity Analysis	Time (s)		Time (s)		
				EA	SQP	EA	SQP	Total
SQP	-	6,119	GFD	-	385	-	17,519	17,519
SQP	-	6,119	ESA	-	387	-	12,097	12,097
S-GA-SQP	7,669	5,695	GFD	40	169	681	7,391	8,071
S-GA-SQP	7,669	5,695	ESA	40	169	681	5,291	5,972
AL-GA-SQP	6,816	5,695	GFD	155	98	2,639	4,361	7,000
AL-GA-SQP	6,816	5,695	ESA	155	98	2,639	3,057	5,696
C-ES-SQP	6,835	5,764	GFD	171	111	2,901	4,943	7,844
C-ES-SQP	6,835	5,764	ESA	171	111	2,901	3,469	6,370
A-ES-SQP	6,319	5,764	GFD	91	67	1,552	2,983	4,535
A-ES-SQP	6,319	5,764	ESA	91	67	1,552	2,089	3,641
AL-GA-ES	6,816	5,430	-	155+73	-	3,866	-	3,866
mμGA	-	5,472	-	339	-	5,243	-	5,243

Table 13. Twenty storey space frame: Hybrid methods (medium initial design)

Optimizer	Initial Design for 2nd optimizer	Final design (kN)	Sensitivity Analysis	Time (s)		Time (s)		
				EA	SQP	EA	SQP	Total
SQP	-	6,119	GFD	-	259	-	11,508	11,508
SQP	-	6,119	ESA	-	259	-	8,049	8,049
S-GA-SQP	7,669	5,695	GFD	35	169	590	7,391	7,981
S-GA-SQP	7,669	5,695	ESA	35	169	590	5,291	5,881
AL-GA-SQP	6,816	5,695	GFD	115	98	1,934	4,361	6,295
AL-GA-SQP	6,816	5,695	ESA	115	98	1,934	3,057	4,991
C-ES-SQP	6,835	5,764	GFD	158	111	2,685	4,943	7,628
C-ES-SQP	6,835	5,764	ESA	158	111	2,685	3,469	6,154
A-ES-SQP	6,275	5,764	GFD	69	67	1,198	2,983	4,181
A-ES-SQP	6,275	5,764	ESA	69	67	1,198	2,089	3,287
AL-GA-ES	6,816	5,430	-	115+73	-	3,161	-	3,161
mμGA	-	5,472	-	339	-	5,243	-	5,243

Table 14. Twenty storey space frame: Hybrid methods (good initial design)

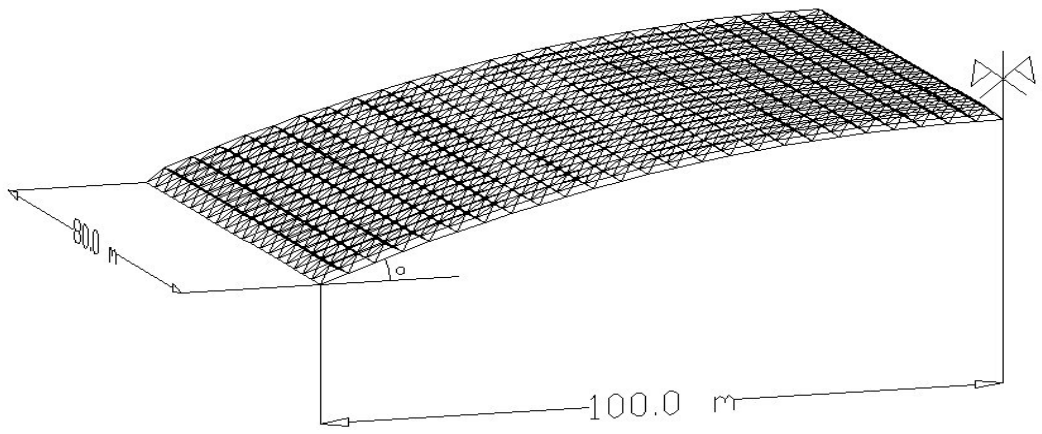


Figure 10. Double-layered space roof truss

Euler buckling occurs in truss structures when the magnitude of a member's compressive stress is greater than a critical stress. For the first buckling mode of a pin-connected member this is equal to

$$\sigma_b = \frac{P_b}{A} = -\frac{1}{A}\left(\frac{\pi^2 EI}{L^2}\right) \tag{64}$$

where P_b is the computed compressive axial force, I is the moment of inertia, L is the member length. For thin-walled tubular members with a diameter-to-thickness ratio $\rho = D/t = 10$ to 20 the cross-sectional area is approximately equal to $A \cong \pi Dt$, and the moment of inertia is approximated by $I \cong \frac{\pi Dt(D^2+t^2)}{8}$. The expression for the buckling stress can, therefore, be written as a function of the cross-sectional areas, which are the design variables of the optimization problem, as follows

$$\sigma_b = -\frac{AE}{L^2} \cdot \frac{\pi(\rho^2+1)}{8\rho} \tag{65}$$

Thus, the compressive stress should be less (in absolute values) than the critical Euler buckling stress $|\sigma| \leq |\sigma_b|$. The values of the constraint functions are normalized in order to improve the performance of the optimization procedure as follows

$$\sigma/\sigma_a \leq 1 \text{ for tension member } \sigma_a = 0.6 \times \sigma_Y$$
$$\sigma/\sigma_b \leq 1 \text{ for compression member } \sigma_b = E(\pi/(1/r))^2$$
$$d/d_a \leq 1$$

The NN tool used in this study is the META-NETS program [18] using the back propagation training algorithm. In Tables 15-18, containing the results of this test example, the following abbreviations are used: *ES* refers to the standard evolution strategies optimization procedure, in which structural analyses are performed in the conventional way. *ES-NN* refers to the ES optimization procedure, where the structural analysis response is predicted by a trained NN. For the two different types of training set selection that have been compared in this study the following abbreviations are used: (i) *GT* stands for the random selection of Training set based on a Gaussian distribution of the design variables in the design space according to Algorithm 1 of section 4.4.2 (ii) *AT* stands for the "Automatic" Training set selection using the results obtained at the initial stages of the ES optimization procedure according to Algorithm 2 of section 4.4.2 The symbol "(c)" is used when the allowable limits of the constraints have been adjusted, as discussed previously, in order to "correct" the NN predictions near the feasible region limits, while symbol "(v)" indicates that the final design is violating the constraints and thus it is infeasible.

In order to investigate the influence of the curvature of the structure in the optimum design four different topologies were tested corresponding to 0°, 5°, 10°, 15° inclination of the curved surface at the supports. The modulus of elasticity is 200 GPa (29,000 ksi) and the yield stress is $\sigma_y = 250$ MPa (36 ksi). Each member is assumed to have a thin-walled tubular cross section. The cross-sectional area is considered to be the design variable of each member. Members are divided to forty eight groups according to their position. For all test cases the finite model consists of 8,000 members, 2,071 nodes and 6,183 degrees of freedom. The loads are taken as uniform vertical forces applied at joints equivalent to uniform load of 0.10 kN and a concentrated vertical load 50 kN at the center of the structure which corresponds to the maximum load of a crane and it is equally distributed

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum volume (mm^3)
			Analysis	Training	NN-ES	Total	
ES	4,150	-	-	-	-	93,125	58.7
NN-ES-GT	480	3,827	10,771	1,546	131	12,448	53.1(v)
NN-ES-GTc	480	4,094	10,771	1,546	140	12,457	59.4
NN-ES-A	472	3174	10,591	345	109	11,045	57.8(v)
NN-ES-Ac	472	3515	10,591	345	120	11,056	59.4

Table 15. Double layered space truss: Test case 1 (0° inclination): Performance of the optimization methods

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum volume (mm^3)
			Analysis	Training	NN-ES	Total	
ES	3,940	-	-	-	-	88,220	44.5
NN-ES-GT	480	4,017	10,771	1,493	138	12,402	46.9
NN-ES-GTc	480	3,862	10,771	1,493	133	12,397	44.7
NN-ES-A	512	3116	11,489	361	106	11,956	41.9(v)
NN-ES-Ac	512	3406	11,489	361	117	11,967	44.8

Table 16. Double layered space truss: Test case 2 (5° inclination): Performance of the optimization methods

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum volume (mm^3)
			Analysis	Training	NN-ES	Total	
ES	4,210	-	-	-	-	95,640	51.3
NN-ES-GT	480	4,132	10,771	1,575	140	12,486	55.5
NN-ES-GTc	480	4,256	10,771	1,575	147	12,493	51.7
NN-ES-A	423	3213	9,492	337	109	9,938	53.5
NN-ES-Ac	423	3147	9,492	337	107	9,936	51.6

Table 17. Double layered space truss: Test case 3 (10° inclination): Performance of the optimization methods

to the central nodes of the roof. The objective function in all test cases is the weight of the structure. The constraints are imposed on the maximum nodal displacement and the maximum axial and buckling stresses in each element group. The values of allowable axial stress is $\sigma_a = 150$ MPa, whereas the maximum allowable displacement is limited to 3 cm.

For all test cases the $(\mu + \lambda)$-ES approach is used with $\mu = \lambda = 20$. The number of NN input units is equal to the number of design variables, whereas the output units are ninety eight corresponding to the two values of axial and buckling stresses, for the forty eight element groups, the value of the maximum nodal displacement and the value of the objective function. The NN configuration has two hidden layers each one consisting of 35 nodes, which results in a 48-35-35-98 NN architecture used for all runs. The performance of the Gaussian and the "automatic" NN training set selection with 480 and 200 training sets,

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum volume (mm^3)
			Analysis	Training	NN-ES	Total	
ES	4,280	-	-	-	-	96,035	54.4
NN-ES-GT	480	4,021	10,771	1,631	138	12,540	51.5(v)
NN-ES-GTc	480	4,267	10,771	1,631	147	12,549	54.6
NN-ES-A	437	3613	9,806	356	124	10,286	55.2
NN-ES-Ac	437	3221	9,806	356	110	10,272	54.6

Table 18. Double layered space truss: Test case 4 (15° inclination): Performance of the optimization methods

respectively, is shown in Tables 15-18 for the four configurations of the roof. It is obvious from the results that the performance of the proposed ES-NN methodology is superior to the performance of the conventional ES optimization procedure, since a dramatic improvement in total computing time required by ES-NN over ES is observed in all test cases examined. A significant improvement is also observed in the performance, both in terms of computing time and optimum values of the objective function, of the proposed ES-NN methodology when the “automatic” type of NN training is used over the Gaussian type of NN training. As it can be observed from the results obtained the curved type of structure is more economical from the flat roof type even though the surface of the structure is longer. For greater slopes, however, the overall weight grows since the surface of the structure increases significantly.

4.5.2 Shape optimization with evolutionary algorithms

So far little effort has been spent in applying probabilistic search methods in shape optimization problems which are usually solved with a mathematical programming algorithms. The use of combinatorial type algorithms appears to be promising even if greater number of analyses is 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 usually limited to few tens or hundreds. Furthermore, the same advantages stated previously for the probabilistic methodologies are also valid in shape optimization problems.

The performance of the optimization methods discussed is investigated in the characteristic plane stress test example considered in section 3.3.1 with isotropic material properties (elastic modulus $E = 210,000$ N/mm^2 and Poisson's ratio $\nu = 0.3$) and five design variables. The problem definition of this example is given in Figure 1 where, due to symmetry, only a quarter of the plate is modeled. The plate is under biaxial tension with one side loaded with a distributed loading $p = 0.65$ N/mm^2 and the other side loaded only with half of this value, as shown in Figure 1. The problem is analyzed with a fine mesh of 38,800 d.o.f. giving a sparse global stiffness matrix with relatively large bandwidth. As previously ESA and GFD methods are used to compute the sensitivities with $\Delta s = 10^{-5}$.

Table 19 depicts the performance of mathematical programming and evolution strategies optimization methods in sequential and parallel computing modes for two test cases of this example corresponding to two initial designs, one close and the other away from the optimum. The parallel computing mode is a single-domain natural parallelization scheme. In Table 19 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 and λ respectively,

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)

Table 19. Square plate example: Performance of the optimization methods

for the "plus" version of ES. 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 given 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 ES $p = \mu$. ES manage to find the optimum solution in both test cases of the square plate example (good and bad initial design), whereas the MP approach failed in the second test case. This could be explained by the fact that the MP optimizer was trapped into the infeasible region due to its inability to overcome severe violations mainly on the global stress constraint.

The computing time spent by the optimizers is affected by the initial design, particularly in the case of the MP approach. It can also be observed that ES achieve better performance to MP optimizer in sequential computing mode and perform much better in parallel computing mode. The use of hybrid approaches, especially the MP-ES, leads to a significant reduction of computing time in both sequential and parallel modes for the first test case. For the second test case, however, since SQP fails both hybrid approaches fail. The mathematical programming optimizer using the ESA sensitivity analysis method is faster than GFD method in sequential mode. In parallel mode, however, GFD becomes competitive to ESA. The natural parallelization scheme implemented has a beneficial effect to all versions of the optimizers used. In particular, this effect is more pronounced in the case of ES where a higher efficiency is achieved.

5 OPTIMUM STRUCTURAL DESIGN UNDER SEISMIC LOADING

Due to the uncertain nature of the earthquake loading, structural designs are often based on design response spectra of the region and on some simplified assumptions of the structural behavior under earthquake. In the case of a direct consideration of the earthquake loading the optimization of structural systems requires the solution of the dynamic equations of motion which can be orders of magnitude more computational intensive than the case of static loading. In this section, both the rigorous approach and the simplified one, with respect to the loading conditions, are implemented and their efficiency is compared in the framework of finding the optimum design of a structure having the minimum weight. In the context of the rigorous approach a number of artificial accelerograms are produced from the design response spectrum of the region for elastic structural response, which constitutes the multiple loading conditions under which the structures is optimally designed. The elastic design response spectrum can be seen as an envelope of response spectra, for a specific damping ratio, of different earthquakes most likely to occur in the region. This approach is compared with the approximate one based on simplifications adopted by the seismic codes. The results obtained for a characteristic test problem indicate the improvement achieved in the final design when the rigorous approach is considered.

The equations of equilibrium for a finite element system in motion can be written in the usual form

$$M(s_i)\ddot{u}_t + C(s_i)\dot{u}_t + K(s_i)u_t = R_t \tag{66}$$

where $M(s_i)$, $C(s_i)$, and $K(s_i)$ are the mass, damping and stiffness matrices for the i-th design vector s_i; R_t is the external load vector, while $u, \dot{u}$ and $\ddot{u}$ are the displacement, velocity, and acceleration vectors of the finite element assemblage, respectively. Design approaches based on direct integration of equations of motion and on the response spectrum modal analysis, which is based on the mode superposition approach, will be considered in the following paragraph.

The Newmark integration scheme is adopted in the present study to perform the direct time integration of the equations of motion where the equilibrium equations (66) are considered at time $t + \Delta t$

$$M(s_i)\ddot{u}_{t+\Delta t} + C(s_i)\dot{u}_{t+\Delta t} + K(s_i)u_{t+\Delta t} = R_{t+\Delta t} \tag{67a}$$

and the variation of velocity and displacement are given by

$$\dot{u}_{t+\Delta t} = \dot{u}_t + [(1-\delta)\ddot{u}_t + \delta\ddot{u}_{t+\Delta t}]\Delta t \tag{67b}$$

$$u_{t+\Delta t} = u_t + \dot{u}_t\Delta t + [(1/2-\alpha)\ddot{u}_t + \alpha\ddot{u}_{t+\Delta t}]\Delta t^2 \tag{67c}$$

where α and δ are parameters that can be determined to obtain integration accuracy and stability. Solving for $\ddot{u}_{t+\Delta t}$ in terms of $u_{t+\Delta t}$ from eq. (67c) and then substituting for $\ddot{u}_{t+\Delta t}$ in eq. (67b) we obtain equations for $\ddot{u}_{t+\Delta t}$ and $\dot{u}_{t+\Delta t}$ each in terms of the unknown displacements $u_{t+\Delta t}$ only. These two relations for $\ddot{u}_{t+\Delta t}$ and $\dot{u}_{t+\Delta t}$ are substituted into eq. (67a) to solve for $u_{t+\Delta t}$. As a result of this substitution the following well-known equilibrium equation is obtained at each Δt

$$K_{\text{eff}}(s_i)u_{t+\Delta t} = R^{\text{eff}}_{t+\Delta t} \tag{68}$$

5.1 Generation of Artificial Accelerograms

The selection of the proper external loading R_t for design purposes is not an easy task due to the uncertainties involved in the seismic loading. For this reason a rigorous treatment of the seismic loading is to assume that the structure is subjected to a set of earthquakes that are more likely to occur in the region where the structure is located.

The seismic excitations that are more likely to occur are produced as a series of artificial accelerograms. In order these artificial accelerograms, that will load the structure, to be representative they have to match some requirements of the seismic codes. The most demanding one is that the accelerograms have to be compatible with the elastic design response spectrum of the region where the structure is located. It is well known that each accelerogram corresponds to a single response spectrum for a given damping ratio that can be defined relatively easy. On the other hand on each response spectrum corresponds an infinite number of accelerograms.

Gasparini and Vanmarke [26,27] originally proposed the creation of artificial accelerograms that correspond to a specific response spectrum. In this study the implementation published by Taylor [82] for the generation of statistically independent artificial acceleration time histories is adopted. This method is based on the fact that any periodic function can be expanded into a series of sinusoidal waves

$$x(t) = \sum_k A_k \sin(\omega_k t + \varphi_k) \tag{69}$$

where A_k is the amplitude, ω_k is the cyclic frequency and φ_k is the phase angle of the k-th contributing sinusoid. By fixing an array of amplitudes and then generating different arrays of phase angles, different motions can be generated which are similar in general appearance but different in the "details". The computer uses a random number generator subroutine to produce strings of phase angles with a uniform distribution in the range between 0 and 2π. The amplitudes A_k are related to the spectral density function in the following way

$$G(\omega_k)\Delta\omega = \frac{A_k^2}{2} \tag{70}$$

where $G(\omega_k)\Delta\omega$ may be interpreted as the contribution to the total power of the motion from the sinusoid with frequency ω_k.

The power of the motion produced by eq. (69) does not vary with time. To simulate the transient character of real earthquakes, the steady-state motion are multiplied by a deterministic envelope function $I(t)$

$$Z(t) = I(t) \sum_k A_k \sin(\omega_k t + \varphi_k) \tag{71}$$

The resulting motion is stationary in frequency content with peak acceleration close to the target peak acceleration. In this study a trapezoidal intensity envelope function is adopted. The generated peak acceleration is artificially modified to match the target peak acceleration, which corresponds to the chosen elastic design response spectrum. An iterative procedure is implemented to smooth the calculated spectrum and improve the matching [82].

The elastic design response spectrum considered in the current study is depicted in Figure 11 for damping ratio $\xi = 2.5\%$. Five artificial uncorrelated accelerograms, produced by the previously discussed procedure and shown in Figure 12, have been used as the input seismic excitation for the numerical tests. The corresponding response spectrum of the first artificial accelerogram is also depicted in Figure 11.

5.2 Response Spectrum Modal Analysis

The response spectrum modal analysis is based on a simplification of the mode superposition approach with the aim to avoid time history analyses which are required by both, the direct integration and mode superposition approaches. In the case of the response spectrum modal analysis eq. (66) is modified according to the modal superposition approach in the following form

$$\overline{M}(s_i)\ddot{u}_t + \overline{C}(s_i)\dot{u}_t + \overline{K}(s_i)u_t = \overline{R}_t \tag{72}$$

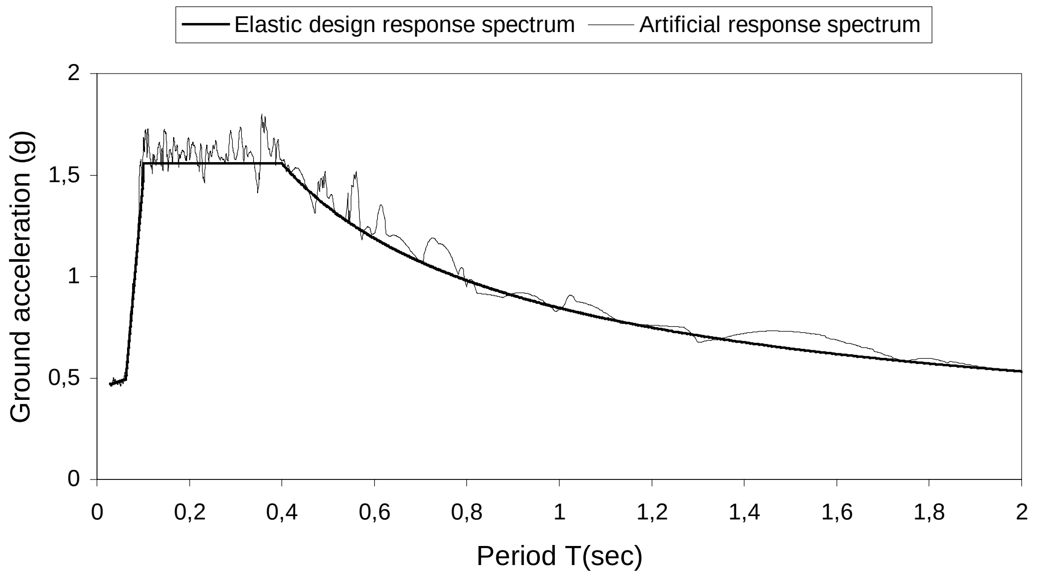


Figure 11. Elastic design response spectrum of the region and response spectrum of the first artificial accelerogram ($\xi = 2.5\%$)

where

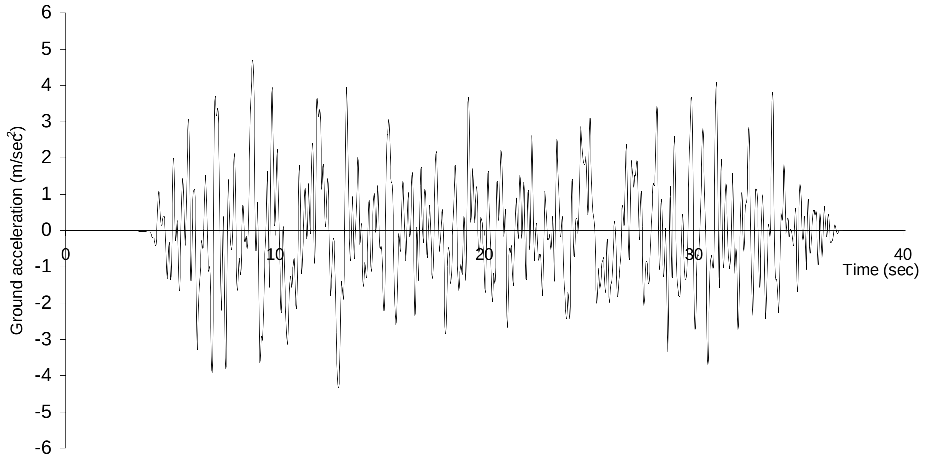
$$\overline{M}_i = \Phi_i^T M_i \Phi_i \tag{73}$$
$$\overline{C}_i = \Phi_i^T C_i \Phi_i \tag{74}$$
$$\overline{K}_i = \Phi_i^T K_i \Phi_i \tag{75}$$
$$\overline{R}_t = \Phi_i^T R_t \tag{76}$$

are the generalized values of the corresponding matrices and the loading vector, while Φ_i is an eigenmode shape matrix to be defined later. For simplicity $M(s_i)$, $C(s_i)$, $K(s_i)$ are denoted by M_i, C_i, K_i, respectively. These matrices correspond to the design, which is defined by the i-th vector of the design parameters, also called design vector. According to the modal superposition approach the system of N simultaneous differential equations, which are coupled with the off-diagonal terms in the mass, damping and stiffness matrices, is transformed to a set of N independent normal-coordinate equations. The dynamic response can therefore be obtained by solving separately for the response of each normal (modal) coordinate and then superimposing these to obtain the response in the original coordinates.

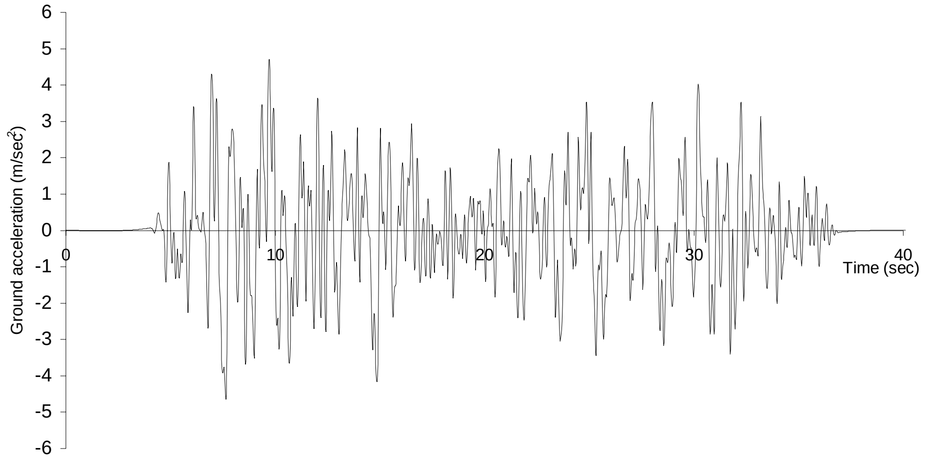
In the response spectrum modal analysis a number of different formulas have been proposed to obtain reasonable estimates of the maximum response based on the spectral values without performing time history analyses for a considerable number of transformed dynamic equations. The simplest and most popular of these is the square root of the sum of the squares (SRSS) of the modal responses. According to this estimate the maximum total displacement is approximated by

$$u_{\max} = \sqrt{u_{1,\max}^2 + u_{2,\max}^2 + \cdots u_{N,\max}^2} \tag{77}$$

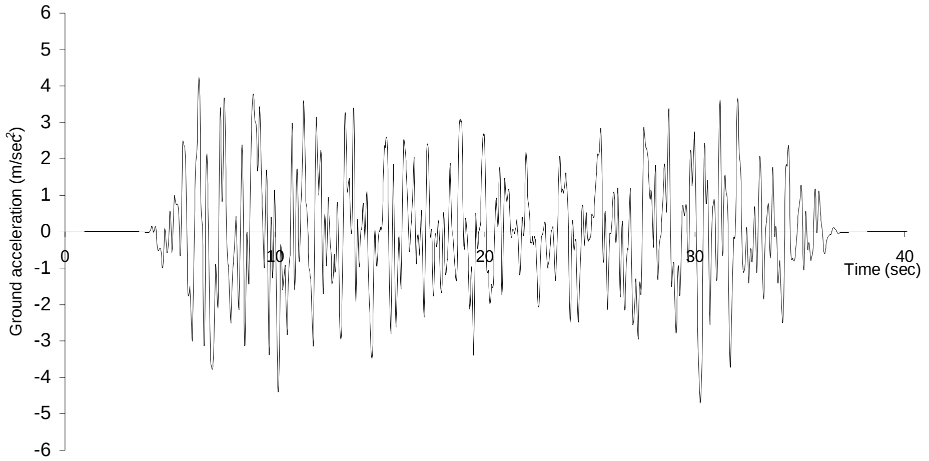
where $u_{j,\max}$ corresponds to the maximum displacement calculated from the j-th transformed dynamic equations over the complete time period. The use of the eq. (77) permits this type of "dynamic" analysis by knowing only the maximum modal coordinates $u_{j,\max}$.



(a)

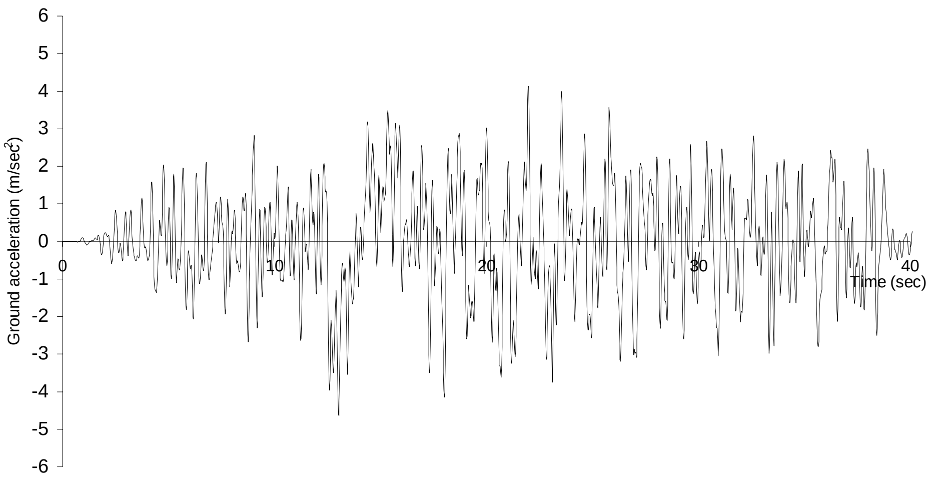


(b)

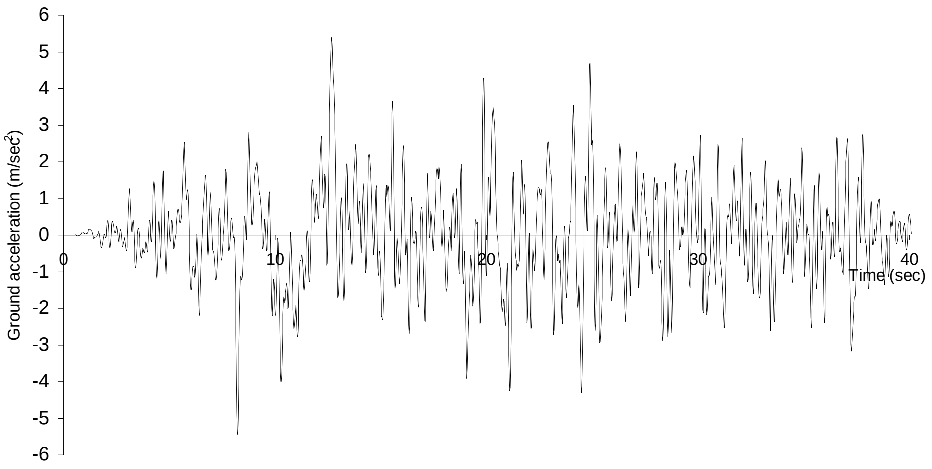


(c)

Figure 12. The five artificial accelerograms



(d)



(e)

Figure 12. continued

The following steps summarize the response spectrum modal analysis adopted in this study and by a number of seismic codes around the world:

1. Calculate a number $m' < N$ of eigenfrequencies and the corresponding eigenmode shape matrices, which are classified in the following order $(\omega_i^1, \omega_i^2, \cdots, \omega_i^{m'})$ and $\Phi_i = [\phi_i^1, \phi_i^2, \cdots, \phi_i^{m'}]$, respectively, where ω_i^j, ϕ_i^j are the j-th eigenfrequency-eigenmode corresponding to the i-th design vector. m' is a user specified number, based on experience or on previous test analyses, which has to satisfy the requirement of step 6.

2. Calculate the generalized masses, according to the following equation

$$\overline{m}_i^j = \phi_i^{j^T} M_i \phi_i^j \tag{78}$$

3. Calculate the coefficients L_i^j, according to the following equation

$$L_i^j = \phi_i^{j^T} M_i r \tag{79}$$

   where r is the influence vector, which represents the displacements of the masses resulting from static application of a unit, ground displacement.

4. Calculate the modal participation factor Γ_i^j, according to the following equation

$$\Gamma_i^j = \frac{L_i^j}{\overline{m}_i^j} \tag{80}$$

5. Calculate the effective modal mass for each design vector and for each eigenmode, by the following equation

$$m_{\text{eff},i}^j = \frac{L_i^{j^2}}{\overline{m}_i^j} \tag{81}$$

6. Calculate a number $m < m'$ of the important eigenmodes. According to Eurocode the minimum number of the eigenmodes that has to be taken into consideration is defined by the following assumption: The sum of the effective eigenmasses must not be less than the 90% of the total vibrating mass m_{tot} of the system, so the first m eigenmodes that satisfy the equation

$$\sum_{j=1}^{m} m_{\text{eff},i}^j \geq 0.90 m_{\text{tot}} \tag{82}$$

   are taken into consideration.

7. Calculate the values of the spectral acceleration $R_d(T_j)$ that correspond to each eigenperiod T_j of the important modes.

8. Calculate the spectral displacements according to equation

$$(SD)_j = \frac{R_d(T_j)}{\omega_j^2} = \frac{R_d(T_j) \cdot T_j^2}{4\pi^2} \tag{83}$$

9. Calculate the modal displacements

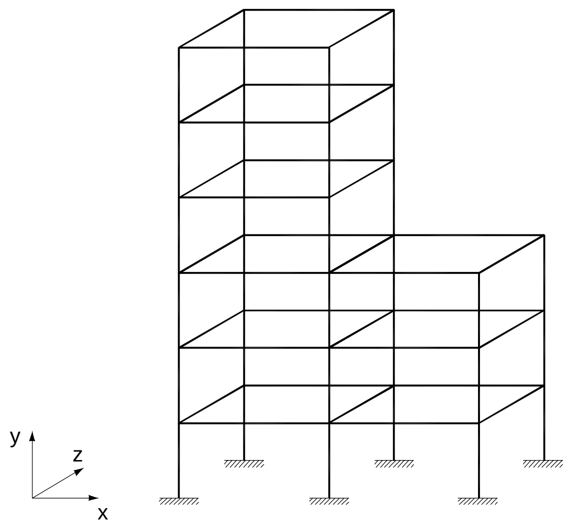
$$u_{j,\max} = \Gamma_i^j \cdot \phi_i^j \cdot (SD)_j \tag{84}$$

   The total maximum displacement is then calculated by superimposing the maximum modal displacements according to eq. (77).

5.3 Numerical Tests of Sizing Optimization Under Seismic Loading

One benchmark test example of space frame with six storeys has been considered to illustrate the efficiency of the proposed methodology in sizing optimization problems with discrete design variables. The modulus of elasticity is 200 GPa and the yield stress is $\sigma_y = 250$ MPa. The cross section of each member is assumed to be a I-shape and for each member two design variables are considered as shown in Figure 4. The objective function of the problem is as previously the weight of the structure. The constraints are imposed on the inter-storey drifts and on the maximum non-dimensional ratio q of eqs. (62) and (63) for each element group which combines axial force and bending moments. The values of allowable axial and bending stresses are $F_a = 150$ MPa and $F_b = 165$ MPa, respectively, whereas the maximum allowable inter-storey drift is limited to 4.5 cm which corresponds to 1.5% of the height of each storey. All tests were performed on a SG Power Challenge computer with the R4000 processor.

The space frame consists of 63 elements with 180 degrees of freedom as shown in Figure 13a. The beams have length $L_1 = 7.32$ m and the columns $L_2 = 3.66$ m. The structure is loaded with a 19.16 kPa gravity load on all floor levels and a static lateral load of 109 kN applied at each node in the front elevation along the z direction. The element members are divided into 5 groups, as shown in Figure 13b, each one having two design variables resulting in ten total design variables. The constraints are imposed on the maximum allowable inter-storey drift and the non-dimensional ratio q for each element group. For this test case both $(\mu + \lambda)$-ES and (μ, λ)-ES schemes are implemented.



(a)

Figure 13. (a) Six storey space frame

The convergence history with respect to the finite element analyses performed by the optimization procedure using the (5+5)-ES scheme is shown in Figure 14 for both the direct time integration and the response spectrum modal analysis methods. It can be seen that the optimum design achieved by the direct time integration approach under the multiple loading conditions of the five artificial accelerograms given in Figure 12 is 20% less than the corresponding design given by the response spectrum modal analysis.

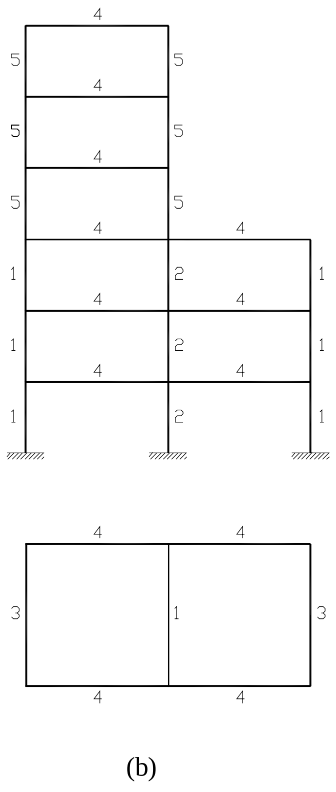


Figure 13. (b) Element groups

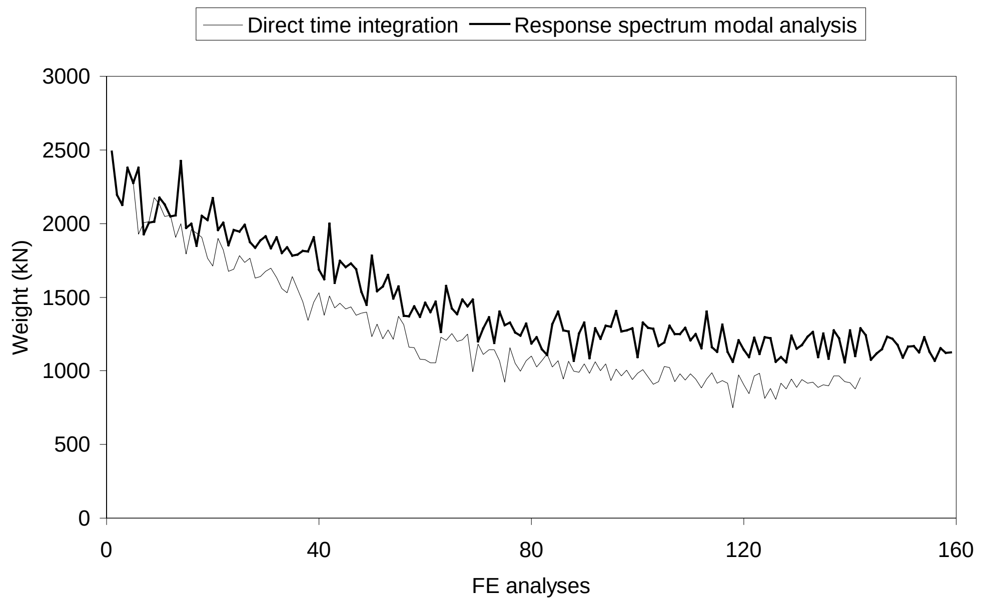


Figure 14. Six storey space frame: Convergence histories of the optimization procedure

The results obtained by the $(\mu+\lambda)$-ES and (μ,λ)-ES schemes with $\mu=\lambda=5$ are shown in Table 20 for the direct integration approach and the response spectrum modal analysis. The upper values of the design parameters are taken as the initial design ($W_{\text{initial}}=2486$ kN), while the termination criterion (i) of section 4.2.5 is adopted for both schemes. The results indicate that the $(\mu+\lambda)$-ES scheme appears to be more robust than the (μ,λ)-ES scheme. Furthermore, the direct time integration approach is about two times more computationally expensive than the response spectrum modal analysis.

ESs scheme	Weight (kN)	Time (sec)	Generations	FE analyses
(μ+λ)	944	13818	40	142
(μ, λ)	842	39657	40	359

ESs scheme	Optimum solution achieved (design variables-cm)									
	h1	b1	h2	b2	h3	b3	h4	b4	h5	b5
(μ+λ)	46	38	58	46	51	35	20	15	46	33
(μ, λ)	51	35	46	43	51	35	30	13	51	20

(a) The (μ+λ)-ESs and (μ,λ)-ESs schemes for the direct time integration approach

ESs scheme	Weight (kN)	Time (sec)	Generations	FE analyses
(μ+λ)	1126	5674	40	157
(μ, λ)	1316	5284	21	140

ESs scheme	Optimum solution achieved (design variables-cm)									
	h1	b1	h2	b2	h3	b3	h4	b4	h5	b5
(μ+λ)	51	41	53	53	51	41	28	20	35	33
(μ, λ)	43	43	66	56	51	46	33	23	41	35

Table 20. Six storey space frame: Comparison of $(\mu+\lambda)$-ESs and (μ,λ)-ESs schemes

The influence of the number of parents and offsprings is shown in Table 21. The results indicate that the schemes close to (5+5)-ES scheme gave better convergence. This confirms the empirical rule that the sum of the parents and offsprings should be roughly equal to the number of the design parameters of the problem. Schemes with larger number of parents and offsprings consume much more time until they reach convergence but they can give good results in some cases, as far as the optimum is concerned.

Finally, the behaviour of the (5+5)-ES scheme for different initial designs is depicted in Table 22. The initial designs correspond to one feasible and one infeasible design. The results show that the final optimum design could be affected by the initial parameters in the range of 10% at the most.

6 CONCLUDING REMARKS

The proposed hybrid optimization algorithms proved to be robust and efficient methods for structural optimization. Both combinations of genetic algorithms with successive quadratic

ESs scheme	Weight (kN)	Time (sec)	Generations	FE analyses
(3+3)	863	9839	65	135
(3+5)	917	11308	35	113
(5+3)	963	12816	56	123
(5+5)	944	13818	40	142
(5+10)	835	20574	38	248
(10+5)	824	29363	78	306
(10+10)	844	32130	48	381

(a) Direct time integration approach

ESs scheme	Weight (kN)	Time (sec)	Generations	FE Dynamic analyses
(3+3)	1207	3110	37	82
(3+5)	1103	3527	29	92
(5+3)	1082	9853	129	299
(5+5)	1126	5674	40	157
(5+10)	1165	4897	18	130
(10+5)	1253	4154	23	109
(10+10)	1108	8646	29	235

(b) Response spectrum modal analysis

Table 21. Six storey space frame: Influence of the number of parent and offspring for the $(\mu+\lambda)$-ESs schemes

Initial design	Weight (kN)	Time (sec)	Generations	FE analyses
feasible	944	9473	35	142
infeasible	1037	13818	40	178

(a) Direct time integration approach

Initial design	Weight (kN)	Time (sec)	Generations	FE analyses
feasible	1126	5674	40	157
infeasible	1104	9510	46	246

(b) Response spectrum modal analysis

Table 22. Six storey space frame: Influence of the number of the starting point of the (5+5)-ESs scheme

programming and of evolution strategies with successive quadratic programming manage to converge to better designs than those achieved by evolution strategies or successive quadratic programming alone at a reduced computational effort compared to the successive quadratic programming procedure. The combination of genetic algorithms with successive quadratic programming is particularly promising for bad initial designs due to the fast convergence of genetic algorithms towards the neighborhood of the optimum and the property of successive quadratic programming to compute quickly the nearest optimum once in the neighborhood of the solution. However, the proposed adaptive evolution strategies when coupled with successive quadratic programming and the combination of the augmented Lagrangian genetic algorithms as the first stage optimizer followed by evolution strategies proved to be the more efficient optimization algorithm for the test cases considered.

The techniques for handling the constraints in genetic algorithms are based on the use of penalty functions, which transform the constraint optimization problem into an unconstraint one. Techniques for handling the constraints based on static and dynamic penalties as well as the micro genetic algorithms and segregated genetic algorithms proved to be rather sensitive to the values of the characteristic parameters for handling the constraints. The computational effort that is required by genetic algorithms is less than the corresponding effort by evolution strategies but they are hindered by premature convergence either to non-optimal or to infeasible designs. The contemporary evolution strategies and the adaptive evolution strategies, did manage to improve the final design achieved by the basic evolution strategies for a number of values of the characteristic parameters at almost no increase of the computational effort.

The proposed multi-domain methods for solving the finite element equilibrium equations outperform the direct and hybrid single-domain methods in sequential computing mode. In parallel computing mode both one-level and two-level multi-domain methods exhibit satisfactory speed-ups in relation to the computational effort for solving the overall optimization problem.

The computational effort involved in the optimization procedure using evolutionary algorithms becomes excessive in large-scale problems and the use of neural networks to "predict" the necessary optimization data for evolutionary algorithms can practically eliminate any limitation on the size of the problem, while the predicted structural response corresponding to different optimization simulations falls within acceptable tolerances. The methodology presented is an efficient, robust and generally applicable optimization procedure capable of finding the global optimum design of complicated structural optimization problems. Additionally, it was found that the proposed hybrid optimization methodology of combining evolution strategies with successive quadratic programming can reach the optimum for large and computationally intensive problems at a fraction of the computing time required by the standard evolution strategies optimization algorithm and the conventional method based on mathematical programming technique.

The proposed optimization algorithms can be considered as robust and efficient tools for design optimization of structures under seismic loading. The presented results indicate the improvement that can be achieved in the final design of structures under seismic loading when the proposed optimization procedure is implemented. Both design methodologies based on a number of artificially generated earthquakes and the response spectrum modal analysis adopted by the seismic codes have been implemented and compared. The more rigorous dynamic approach based on time history analyses gives more economic designs than the approximate response spectrum modal analysis, at the expense of requiring more computational effort.

REFERENCES

1 Adeli H. and Cheng N.T. (1994), "Augmented Lagrangian Genetic Algorithm for structural optimization", *Journal of Aerospace Engineering*, ASCE, **7**(**1**), 104-118.

2 Adeli H. and Cheng N.T. (1994), "Concurrent Genetic Algorithms for optimization of large structures", *Journal of Aerospace Engineering*, ASCE, **7**(**3**), 276-296.

3 Adeli H. and Cheng N.T. (1993), "Integrated Genetic Algorithm for optimization of space structures", *Journal of Aerospace Engineering*, ASCE, **6**(**4**), 315-328.

4 Adeli H. and Hyo Seon Park (1995), "Neural dynamics model for structural optimisation. - Theory", *Computers & Structures*,**57**(**3**), 383-399.

5 Adeli H. and Hyo Seon Park (1995), "Optimization of space structures by Neural dynamics", *Neural Networks*, **8**(**5**), pp. 769-781.

6 Arora J.S. (1990), "Computational Design Optimization: A review and future directions", *Structural Safety*, **7**, 131-148.

7 Arslan M.A. and Hajela P. (1997), "Counterpropagation Neural Networks in decomposition based optimal design", *Computers & Structures*, **65**(**5**), 641-650.

8 Back T., Hoffmeister F. and Schwefel H.-P. (1991), "A survey of evolution strategies", in R.K. Belew and L.B. Booker (Eds.), *Proceedings of the 4th International Conference on Genetic Algorithms*, Morgan Kaufmann, 2-9.

9 Baker J. (1985), "Adaptive selection methods for genetic algorithms", in J.J. Grefenstette (Ed.), *Proc. International Conf. On Genetic Algorithms and their applications*, Lawrence Erlbaum.

10 Barricelli N.A. (1962), "Numerical testing of evolution theories", *ACT A Biotheoretica*, **16**, 69-126.

11 Belegundu A.D. and Arora J.S. (1984), "A computational study of transformation methods for optimal design', *AIAA Journal*, **22**(**4**), 535-542.

12 Bendsoe M.P. and Kikuchi N. (1988), "Generating optimal topologies in structural design using a homogenization method", *Computer Methods in Applied Mechanics and Engineering*, **71**, 197-224.

13 Berke L. and Hajela P. (1990), "Applications of Artificial Neural Nets in Structural Mechanics", NASA 331-348 TM-102420.

14 Berke L., Patnaik S.N. and Murthy P.L.N. (1993), "Optimum Design of Aerospace Structural Components using Neural Networks", *Computers & Structures*, **48**, 1001-1010.

15 Bitoulas N. and Papadrakakis, M. (1994), "An optimised computer implementation of the incomplete Cholesky factorization", *Comp. Systems in Engineering*, **5**(**3**), 265-274.

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

17 Cai J. and Thierauf G. (1993), "Discrete structural optimization using evolution strategies", in B.H.V. Topping and A.I. Khan (Eds.), *Neural networks and combinatorial in civil and structural engineering*, Edinburgh, Civil-Comp Limited, 95-10.

18 Embrechts M.J. (1994), *Metaneural-Version 4.1*, Rensselear Polytechnique Institute.

19 Eschenauer H.A., Schumacher A. and Vietor T. (1993), "Decision makings for initial designs made of advanced materials", in Bendsoe M.P. and Soares C.A.M. (Eds.), NATO ARW "Topology design of structures", Sesimbra, Portugal, Kluwer Academic Publishers, Dordrecht, Netherlands, 469-480.

20 Eurocode 3, Design of steel structures, Part1.1: General rules for buildings, CEN, ENV 1993-1-1/1992.

21 Farhat C. and Roux F.-X. (1994), "Implicit Parallel Processing in Structural Mechanics", *Comp. Mec. Adv.*, **2**, 1-124.

22 Farhat C., Crivelli L. and Roux F.-X. (1994), "Extending substructure based iterative solvers to multiple load and repeated analyses", *Comp. Meth. Appl. Mech. Eng.*, 195-209.

23 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, 509-530.

24 Fogel L.J., Owens A.J. and Walsh M.J. (1966), *Artificial intelligence through simulated evolution*, Wiley, New York.

25 Gallagher R.H. and Zienkiewicz O.C. (1973), *Optimum Structural Design: Theory and Applications*, John Wiley & Sons, New York.

26 Gasparini D.A. and Vanmarke E.H. (1976), "Simulated earthquake motions compatible with prescribed response spectra", Massachusetts Institute of Technology (MIT), Department of Civil Engineering, Publication No. R76-4, January.

27 Gasparini D.A. (1976), *SIMQKE - A program for artificial motion generation*, User's manual and documentation, Massachusetts Institute of Technology (MIT), Department of Civil Engineering, November.

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

29 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.

30 Goldberg D. (1990), "A note on boltzmann tournament selection for Genetic Algorithms and population-oriented Simulated Annealing", TCGA 90003, *Engineering Mechanics*, Alabama University.

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

32 Goldberg D.E. (1988), "Sizing populations for serial and parallel genetic algorithms", TCGA Report No 88004, University of Alabama.

33 Gunaratnam D.J. and Gero J.S. (1994), "Effect of representation on the performance of Neural Networks in structural engineering applications", *Microcomputers in Civil Engineering*, **9**, 97-108.

34 Hajela, P. and Berke, L. (1991), "Neurobiological computational models in structural analysis and design", *Computers & Structures*, **41**, 657-667.

35 Haug E.J. and Arora J.S. (1974), "Optimal mechanical design techniques based on optimal control methods", ASME paper No 64-DTT-10, *Proceedings of the 1st ASME Design Technology Transfer Conference*, 65-74, New York, October.

36 Hinton E. and Hassani B. (1995), "Some experiences in structural topology optimization", in B.H.V. Topping (Ed.), *Developments in Computational Techniques for Structural Engineering*, CIVIL-COMP Press, Edinburgh, 323-331.

37 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, 1-26.

38 Hinton E. and Sienz J. (1993), "Fully stressed topological design of structures using an evolutionary procedure", *Journal of Engineering Computations*, **12**, 229-244.

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

40 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*, Springel-Verlag, Berlin, Germany, 455-469.

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

42 Hooke R. and Jeeves T.A. (1961), "Direct Search Solution of Numerical and Statistical Problems", *J. ACM*, Vol. **8**.

43 Joines J. and Houck C. (1994), "On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA", in Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel and H. Kitano (Eds.), *Proceedings of the First IEEE International Conference on Evolutionary Computation*, 579-584. IEEE Press.

44 Khan, A.I., Topping, B.H.V. and Bahreininejad, A. (1993), "Parallel training of neural networks for finite element mesh generation", *Neural Networks and Combinatorial Optimisation in Civil and Structural Engineering*, B.H.V. Topping and A. I. Khan (Eds.), Civil-Comp Press, 81-94.

45 Kirkpatrick S., Gelatt C.D. and Vecchi M.P. (1983), "Optimization by simulated annealing", *Science*, **220**, 671-680.

46 Krishnakumar K. (1989), "Micro genetic algorithms for stationary and non stationary function optimization", in *SPIE Proceedings Intelligent control and adaptive systems*, 1196.

47 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. Soft.*, **4** (**1**), 34-50.

48 Le Riche R. G., Knopf-Lenoir C. and Haftka R.T. (1995), "A segregated genetic algorithm for constrained structural optimization", in L. J. Eshelman (Ed.), *Proceedings of the 6th International Conference on Genetic Algorithms*, 558-565.

49 Maa C. and Shanblatt M. (1992), "A two-phase optimization neural network", *IEEE Transactions Neural Networks*, **3**(**6**), 1003-1009.

50 Michalewicz Z. (1995), "Genetic algorithms, numerical optimization and constraints", in L.J. Eshelman (Ed.), *Proceedings of the 6th International Conference on Genetic Algorithms*, 151-158, Morgan Kaufmann.

51 Moses F. (1974), "Mathematical programming methods for structural optimization", *ASME Structural Optimisation Symposium AMD*, **7**, 35-48.

52 Myung H., Kim J.-H. and Fogel D. (1995), "Preliminary investigation into a two-stage method of evolutionary optimization on constrained problems", in J.R. McDonnell, R.G. Reynolds and D.B. Fogel (Eds.), *Proceedings of the 4th Annual Conference on Evolutionary Programming*, 449-463, MIT Press.

53 NAG (1988), *Software manual*, NAG Ltd, Oxford, UK.

54 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.

55 Papadrakakis M. (1996), "Domain decomposition techniques for computational structural mechanics", in M. Papadrakakis (Ed.), *Solving Large-Scale Problems in Mechanics*, Volume II, John Wiley.

56 Papadrakakis M. and Bitoulas, N. (1993), "Accuracy and effectiveness of preconditioned conjugate gradient method for large and ill-conditioned problems", *Comp. Meth. Appl. Mech. Eng.*, **109**, 219-232.

57 Papadrakakis M., Lagaros N. D., Thierauf G. and Cai J. (1998), "Advanced Solution Methods in Structural Optimization Based on Evolution Strategies", *J. Engng. Comput.*, **15**(**1**), 12-34.

58 Papadrakakis M., Lagaros N.D. and Tsompanakis Y. (1999), "Optimization of large-scale 3D trusses using evolution strategies and neural networks", Special Issue of the *International Journal of Space structures*, **14**(**3**), 211-223.

59 Papadrakakis M., Lagaros N.D. and Tsompanakis Y. (1998), "Structural optimization using evolution strategies and neural networks", *Comp. Meth. Appl. Mechanics & Engrg*, **156**, 309-333.

60 Papadrakakis M. and Papadopoulo V. (1995), "A computationally efficient method for the limit elasto plastic analysis of space frames", *Computational Mechanics*, **16**(**2**), 132-141.

61 Papadrakakis, M. and Papadopoulos, V. (1996), "Efficient solution procedures for the stochastic finite element analysis of space frames using the Monte Carlo simulation", to appear in *Comp. Meth. Appl. Mech. Eng.*

62 Papadrakakis M., Papadopoulos V. and Lagaros N.D. (1996), "Structural reliability analysis of elastic-plastic structures using neural networks and Monte Carlo simulation", *Comp. Meth. Appl. Mechanics & Engrg*, **136**, 145-163.

63 Papadrakakis, M. and Smerou, S. (1990), "A new implementation of the Lanczos method in linear problems", *Int. J. Num. Meth. Eng.*, **29**, 141-159.

64 Papadrakakis M., Tsompanakis Y. and Lagaros N.D. (1999), "Structural shape optimization using Evolution Strategies", *Engineering Optimization Journal*, **31**, 515-540.

65 Papadrakakis M., Tsompanakis Y., Hinton E. and Sienz J. (1996), "Advanced solution methods in topology optimization and shape sensitivity analysis", *Journal of Engineering Computations*, **3**, No. **5**, 57-90.

66 Pedersen P. (1993), "Topology optimization of three dimensional trusses", in Bendsoe M.P. and Soares C.A.M. (Eds.), NATO ARW *Topology design of structures*, Sesimbra, Portugal, Kluwer Academic Publishers, Dordrecht, Netherlands, 19-31.

67 Pope G.G. and Schmit L.A. (Eds.) (1971), *Structural Design Applications of Mathematical Programming Techniques*, AGARDOgraph 149, Technical Editing and Reproduction Ltd., London, February.

68 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, 27-52.

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

70 Rozvany G.I.N. and Zhou M. (1993), "Layout and generalised shape optimization by iterative COC methods", in Rozvany G.I.N. (Ed.), NATO/DFG ASI *Optimization of large structural systems*, Berchtesgaden, Germany, Kluwer Academic Publishers, Dordrecht, Netherlands, 103-120.

71 Rudolph G. (1999), *Personal communication.*

72 Rummelhart D.E. and McClelland J.L. (1986), *Parallel Distributed Processing*, *Volume 1: Foundations*, The MIT Press, Cambridge.

73 Schittkowski K., Zillober C. and Zotemantel R. (1994), "Numerical comparison of non-linear algorithms for structural optimization", *Strustural Optimization*, **7**, 1-19.

74 Schwefel H.-P. and Rudolph G. (1995), "Contemporary evolution strategies", in F. Morgan, A. Moreno, J.J. Merelo and P. Chacion (Eds.), *Advances in Artificial Life*, Proc. Third European Conf. on Artificial Life Granada, Spain, June 4-6, Springer, Berlin, 893-907.

75 Schwefel H.P. (1981), *Numerical optimization for computer models*, Wiley & Sons, Chichester, UK.

76 Sheu C.Y. and Prager W. (1968), "Recent development in optimal structural design", *Applied Mechanical Reviews*, **21**(**10**), 985-992.

77 Shieh R.C. (1994), "Massively parallel structural design using stochastic optimization and mixed neural net/finite element analysis methods", *Computing Systems in Engineering*, **5**, **4-6**, 455-467.

78 Spunt L. (1971), *Optimum Structural Design*, Prentice-Hall, Englewood Cliffs, New Jersey, 41-42.

79 Stephens, J.E. and VanLuchene, D. (1994), "Integrated assessment of seismic damage in structures", *Microcomputers in Civil Engineering*, **9**(**2**), 119-128.

80 Suzuki K. and Kikuchi N. (1993), "Layout optimization using the homogenization method", in Rozvany G.I.N. (Ed.), NATO/DFG ASI *Optimization of large structural systems*, Berchtesgaden, Germany, Kluwer Academic Publishers, Dordrecht, Netherlands, 157-175.

81 Svanberg K. (1987), "The method of moving asymptotes, a new method for structural optimization", *Int. J. Num. Meth. Eng.*, **23**, 359-373.

82 Taylor C.A. (1989), *EQSIM, A program for generating spectrum compatible earthquake ground acceleration time histories*, Reference Manual, Bristol Earthquake Engineering Data Acquisition and Processing System, December.

83 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. Journal of Num. Meth. Engng*, **23**, 2187-2203.

84 Theocharis P.S. and Panagiotopoulos P.D. (1993), "Neural networks for computing in fracture mechanics. Methods and prospects of applications", *Comp. Meth. Appl. Mechanics & Engrg.*, pp. 106, pp. 213-228.

85 Thierauf G,. and Cai J. (1995), "A two level parallel evolution strategy for solving mixed-discrete structural optimization problems", *The 21th ASME Design Automation Conference*, Boston MA, September 17-221.

86 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, 266-280.

87 Topping, B.H.V. and Bahreininejad, A. (1997), *Neural computing for structural mechanics*, Saxe Coburg, UK.

88 Van Keulen F. and Hinton E. (1996), "Topology design of plate and shell structures using the hard kill method", in B.H.V. Topping (Ed.), *Advances in optimization for Structural Engineering*, CIVIL-COMP Press, Edinburgh, 177-188.

89 Vanderplaats G.N. (1984), *Numerical optimisation techniques for engineering design: with applications*, McGraw-Hill, New York.

90 Venkayya V.B., Khot N.S. and Berke L. (1973), "Application of optimality criteria approaches to automated design of large practical structurs", *2nd Symposium on Structural Optimisation AGARD-CP-123*, Milan, Italy, April.

91 Waagen D., Diercks P. and McDonnell J. (1992), "The stochastic direction set algorithm: A hybrid technique for finding function extrema", in D.B. Fogel and W. Atmar (Eds.), *Proceedings of the 1st Annual Conference on Evolutionary Programming*, 35-42, Evolutionary Programming Society.

92 Xie Y.M. and Steven G.P. (1993), "A simple evolutionary procedure for structural optimization", *Computers & Structures*, **49**, 885-896.

93 Xie Y.M. and Steven G.P. (1994), "Optimal design of multiple load case structures using an evolutionary procedure", *Journal of Engineering Computations*, **11**, 295-302.

Please address your comments or questions on this paper to:
International Center for Numerical Methods in Engineering
Edificio C-1, Campus Norte UPC
Grand Capitán s/n
08034 Barcelona, Spain
Phone: 34-93-4106035; Fax: 34-93-4016517
E-mail: onate@cimne.upc.es