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 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

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

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

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

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

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

$$\text{subject to} \quad Ap + h(s) \leq 0 \qquad (5)$$
$$\overline{s}_1 \leq p \leq \overline{s}_u$$

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

$$L(s, \lambda) = F(s) + \lambda h(s) \qquad (6)$$

in which λ are the Lagrange multipliers under the non-negativity restriction ($\lambda \geq 0$) for the inequality constraints. In order to construct the Jacobian and the Hessian matrices of the QP subproblem the derivatives of the objective and constraint functions are required. These derivatives are computed during the sensitivity analysis phase.

There are two ways to solve this QP subproblem, either with a primal [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]

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

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

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

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 $r^{(m+1)} = r^{(m)} + \alpha_m Kd^{(m)}$, 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

$$u = u_0 - Pu_0 + P^2 u_0 - P^2 u_0 + \cdots \tag{23}$$

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

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

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

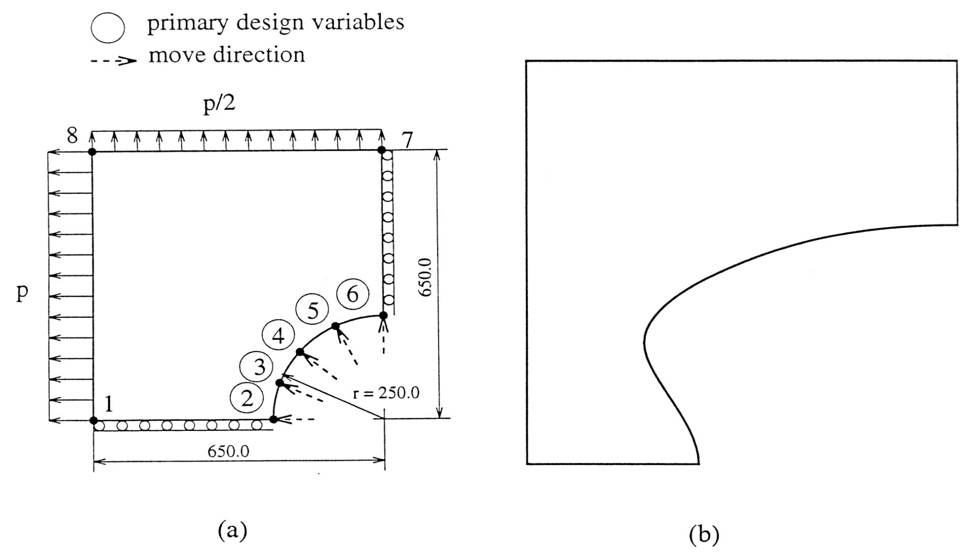


Figure 1. Square plate: (a) initial shape. (b) final shape

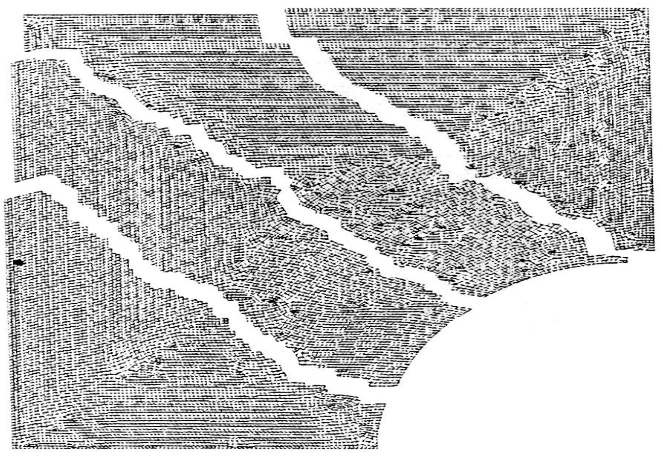
3.3 Numerical Tests

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}.

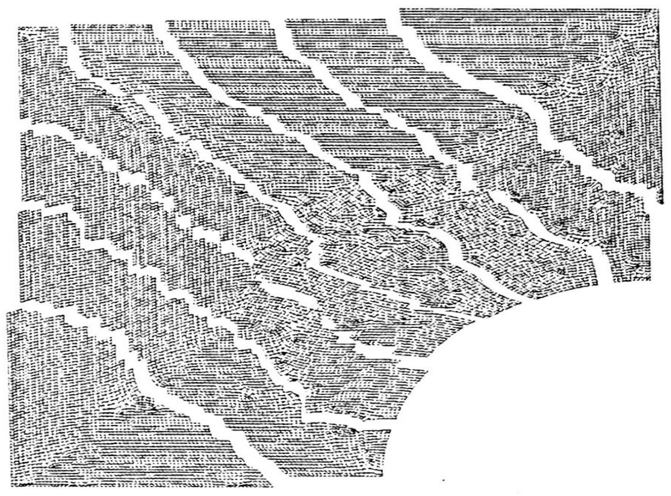
3.3.1 Shape optimization with gradient-based optimizers

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.