Preprint of the paper

"An efficient MP algorithm for structural shape optimization problems"
F. Navarrina, R. Tarrech, I. Colominas, G. Mosqueira, J. Gómez-Calviño, M. Casteleiro (2001)
En "Computer Aided Optimum Design of Structures VII", 247-256, Hernández S., Brebbia C.A. (Editors); WIT Press, Southampton, UK. (ISBN: 1-85312-868-6)

http:/caminos.udc.es/gmni

An efficient MP algorithm for structural shape optimization problems

F. Navarrina[1], R. Tarrech[1], I. Colominas[1], G. Mosqueira[1], J. Gómez-Calviño[1] and M. Casteleiro[2]

[1] *Dept. Applied Math., Universidad de La Coruña*
[2] *Dept. Mech. Engrg., Northwestern University*

Abstract

Integral methods —such as the Finite Element Method (FEM) and the Boundary Element Method (BEM)— are frequently used in structural optimization problems to solve systems of partial differential equations. Therefore, one must take into account the large computational requirements of these sophisticated techniques at the time of choosing a suitable Mathematical Programming (MP) algorithm for this kind of problems. Among the currently available MP algorithms, Sequential Linear Programming (SLP) seems to be one of the most adequate to structural optimization. Basically, SLP consist in constructing succesive linear approximations to the original non linear optimization problem within each step. However, the application of SLP may involve important malfunctions. Thus, the solution to the approximated linear problems can fail to exist, or may lead to a highly unfeasible point of the original non linear problem; also, large oscillations often occur near the optimum, precluding the algorithm to converge.

In this paper, we present an improved SLP algorithm with line-search, specially designed for structural optimization problems. In each iteration, an approximated linear problem with aditional side constraints is solved by Linear Programming. The solution to this linear problem defines a search direction. Then, the objective function and the non linear constraints are quadratically approximated in the search direction, and a line-search is performed. The algorithm includes strategies to avoid stalling in the boundary of the feasible region, and to obtain alternate search directions in the case of incompatible linearized constraints. Techniques developed by the authors for efficient high-order shape sensitivity analysis are referenced.

1 Introduction

The general optimum design problem can be simbolically stated in terms of a non linear constrained minimization problem [1] as:

$$\begin{aligned}
&\text{OBTAIN } \boldsymbol{x} = \{x_i\} && i = 1, \ldots, m && \text{(primal variables)} \\
&\text{THAT MINIMIZES } F(\boldsymbol{x}) && && \text{(objective function)} \\
&\text{SUBJECT TO } G_j(\boldsymbol{x}) \leq 0, && j = 1, \ldots, p_g && \text{(inequality constraints)} \\
&H_l(\boldsymbol{x}) = 0, && l = 1, \ldots, p_h && \text{(equality constraints)} \\
&a_i \leq x_i \leq b_i, && i = 1, \ldots, m. && \text{(side constraints)}
\end{aligned} \tag{1}$$

The Kuhn-Tucker necessary conditions define the optimality requirements for the solution to this problem [2], but nevertheless one can seldom find the optimum by means of analytical techniques, except in the most simple academic cases.

A number of MP algorithms for the numerical solution to problem (1) have been proposed during the last decades [2]. However, the situation has not significantly changed since Sandgren & Ragsdell published their comparative study [3] about twenty years ago. Thus, one still can say that none of the currently available methods is capable of solving a wide range of different problems. In fact, most of basic algorithms normally fail to converge (even for unconstrained cases!). Hence, a successful strategy (to solve any of a certain kind of problems) usually requires the selective combination and subsequent application of different basic MP algorithms, according to the characteristics of the problem being solved.

MP algorithms are iterative actualization procedures type

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \theta^k \boldsymbol{s}^k, \tag{2}$$

in which the iterates $\boldsymbol{x}^k$ are intented to move steadily towards the solution. Thus, different basic algorithms correspond to different simple methods of choosing the direction of search $\boldsymbol{s}^k$ within each iteration. For a given $\boldsymbol{s}^k$, (we assume $|\boldsymbol{s}^k| = 1$) the step θ^k must be subsequently computed, normally by performing a line-search (that is a one-dimensional minimization) in the given direction.

Our goal is to develop an algorithm to solve structural shape optimization problems that must be **reliable**, **robust** and **efficient**. In other words, we expect the algorithm to find the real solution to any properly posed problem, with acceptable requeriments of memory storing and computing time. It is clear that statement (1) is a simplified symbolic formulation, in which a number of variables and dependence relations have been obviated.

Therefore, we must deepen into the specific characteristics of our problem, before choosing an adequate MP algorithm to solve it.

2 The Optimum Design Problem

The complete statement of a general optimum design problem [4,5] can be expressed in the following terms:

$$
\begin{array}{rlll}
\text{GIVEN} & \boldsymbol{c} = \{c_i\} & i = 1, \ldots, m_c & \text{(design constants)} \\
\text{OBTAIN} & \boldsymbol{x} = \{x_i\} & i = 1, \ldots, m & \text{(design variables)} \\
\text{THAT MINIMIZES} & f(\boldsymbol{\gamma}) & & \text{(objective function)} \\
\text{SUBJECT TO} & g_j(\boldsymbol{\gamma}) \leq 0, & j = 1, \ldots, p_g & \text{(inequality constraints)} \\
& h_l(\boldsymbol{\gamma}) = 0, & l = 1, \ldots, p_h & \text{(equality constraints)} \\
& a_i \leq x_i \leq b_i, & i = 1, \ldots, m & \text{(side constraints)} \\
\text{WHERE} & \boldsymbol{\gamma} = \boldsymbol{\gamma}(\boldsymbol{\varphi}, \boldsymbol{\beta}, \boldsymbol{\omega}), & & \text{(control variables)} \\
& \boldsymbol{\varphi} = \boldsymbol{\varphi}(\boldsymbol{c}, \boldsymbol{x}), & & \text{(fundamental props.)} \\
& \boldsymbol{\beta} = \boldsymbol{\beta}(\boldsymbol{c}, \boldsymbol{x}), & & \text{(environmental props.)} \\
& \boldsymbol{\omega} = \{\omega_i\}, & i = 1, \ldots, m_\omega & \text{(state variables)} \\
& \boldsymbol{\Psi}(\boldsymbol{\alpha}, \boldsymbol{\omega}) = \boldsymbol{0}, & & \text{(state equation)} \\
& \boldsymbol{\alpha} = \boldsymbol{\alpha}(\boldsymbol{\varphi}, \boldsymbol{\beta}), & & \text{(input data)}
\end{array}
\tag{3}
$$

where the critical point is solving the state equation (that describes the underlying physical phenomena) to obtain the state variables $\boldsymbol{\omega}$ for known values of the input data $\boldsymbol{\alpha}$.

In this kind of problems, the number of primal (design) variables is normally small, while the objective function is simple (quasi-linear) and easy to handle. Thus, neither computing the objective function nor performing its corresponding sensitivity analysis imply an important computing effort. On the contrary, the number of constraints is frequently very large, and many of them could be highly non linear and extremely difficult to handle. On the other hand, the state equation in engineering practice is frequently a linear or non linear discretized form of a certain boundary-value problem, that must be solved by means of a wide purpose FEM or BEM code. Thus, computing the constraints and their first order corresponding derivatives will normally involve a high computing effort. Moreover, the viability of performing a full second order (or higher) sensitivity analysis must be discarded, since the computational cost of the process increases dramatically with the order of differentiation.

Furthermore, problem (3) can be very difficult to handle. Namely: it can exhibit an indefinite number of local minima; establishing the local convexity or non convexity of the design space is normally unviable; trying to predict *a priori* how many and which constraints will determine the optimum is extremely difficult; and the initial design could be highly undersized as much as oversized, or excessively safe as much as highly non feasible, since it could lie quite far from the optimum in the design space.

3 Outline of a Potential Algorithm

Basically, two factors determine the applicability of a given MP algorithm to solve a certain problem. First, we must consider the cost of repeatedly sampling the objective function and the constraints. And second, we must consider the cost of the sensitivity analysis (that is, repeteadly sampling the derivatives of those functions), as much as the order of differentiation that must be achieved. As a general fact, a higher order algorithm will normally require less iterations to converge, in return for the need to compute higher order derivatives. However, this may preclude the implementation of the method or reduce the global efficiency, due to the associated computational cost. Moreover, it is widely accepted that higher order algorithms are normally less robust, while no substantial improvements are obtained over a certain order of convergence.

On the other hand, the currently available MP algorithms have been derived with the aim of solving certain problems. Thus, we can find suitable optimization methods for a kind of important problems in economics (i.e. Linear and Quadratic Programming). We can also find quite effective —although not 100% robust— methods for inverse problems in engineering (i.e. typically unconstrained problems with highly non linear objective functions). But these techniques do not conform to the specific characteristics of problem (3). It is also clear that computing higher order derivatives of a quasi-linear objective function do not contribute valuable information on how to proceed towards a new iterate. Therefore, higher order refinements such as using SQP instead of SLP [6], or performing a BFGS [2,6] hessian approximation, will not normally improve the performance of the algorithms, nor the final results. Furthermore, we must discard the so-called zero order (no sensitivity analysis) algorithms, as much as those that require an exact line-search, due to the computational cost of solving the state equation to repeatedly sample the objective function and the constraints.

The authors have developed a highly efficient and accurate formulation to perform high order sensitivity analysis computations [5,7,8] for arbitrary directions in the design space. Conceptually, a high order sensitivity analysis does not involve a much higher level of complexity than the first order one. However, the amount of memory storing and the computational work raise exponentially with the order of differentiation, due to the increasing number of derivatives that must be computed. This precludes the use of MP algorithms that require full second (or higher) order information, such as the Lagrange-Newton method [2]. Anyway, the above mentioned techniques allow to perform a full first order sensitivity analysis, as well as a second order sensitivity analysis for a given direction in the design space, with relatively small computational requirements. It is the authors' belief that this refinement represents an extremely useful tool to improve MP algorithms that initially require only first order sensitivity analysis. The MP algorithm proposed below is based on the previously outlined concepts.

4 Proposed MP Algorithm

In the SLP [6] algorithm (which is also referred to as Kelley's Cutting Plane method) the original problem (1) is repeteadly linearized within each iteration via a first order Taylor expansion. Move limits are added to ensure that the truncation error of the approximation is kept under control. The resulting sequence of linear problems:

$$\begin{aligned}
&\text{GIVEN } \boldsymbol{x}^k = \{x_i^k\} && i = 1, \ldots, m \\
&\text{OBTAIN } \boldsymbol{s}^k = \{s_i^k\} && i = 1, \ldots, m \\
&\text{THAT MINIMIZES } \mathcal{F}(\boldsymbol{s}) = F(\boldsymbol{x}^k) + \left[\left. \frac{dF}{d\boldsymbol{x}} \right|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \boldsymbol{s} && \\
&\text{SUBJECT TO } \mathcal{G}_j(\boldsymbol{s}) = G_j(\boldsymbol{x}^k) + \left[\left. \frac{dG_j}{d\boldsymbol{x}} \right|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \boldsymbol{s} \leq 0, && j = 1, \ldots, p \\
&\qquad a_i \leq x_i^k + s_i \leq b_i, && i = 1, \ldots, m \\
&\qquad \delta_i^- \leq s_i \leq \delta_i^+, && i = 1, \ldots, m
\end{aligned} \tag{4}$$

can be easily solved using linear programming methods. The corresponding step is obviously $\theta^k = 1$, since no line-search is performed. Thus, the successive iterates $\boldsymbol{x}^{k+1}$ can be easily computed by means of the actualization formula (2). For the sake of notation simplicity, we assume that each equality constraint has been equivalently replaced by two opposite inequality constraints.

SLP has been applied to many practical problems during the last decades, habitually in association with a finite difference computation of the required derivatives. In spite of this, the method has performed quite well. Obviously, the overall performance of the algorithm can be improved by using more sophisticated sensitivity analysis techniques [5,7,8], provided that the associated computational cost is reduced and the accuracy of the information being handled is raised.

However, the algorithm exhibits some major drawbacks. When the number of active constraints at the optimum is large, SLP produces normally a sequence of improving but slightly unfeasible iterates that converge rapidly to the solution. On the contrary, when the number of active constraints at the optimum is lower than the number of primal variables, large oscillations occur near the optimum and the convergence is precluded. Furthermore, the linear approximated problems can be unbounded as much as inconsistent (the linearized constraints can prevent the existence of feasible points), and the algorithm can get stalled.

One of the proposed convergence aids for the SLP algorithm is the imposition of move limits. This ensures that the optimum will eventually be reached within a certain tolerance, and may help to deal with unbounded linear approximated problems. Unfortunately, move limits do not automatically suppress oscillations near the optimum, unless the limits are reduced

during the optimization process as the algorithm proceedes. But this is not obvious whatsoever in practical cases. On the other hand, move limits make more difficult to deal with those cases in which no feasible points exist in the vicinity of the actual iterate $\boldsymbol{x}^k$.

Figure 1 shows how the SLP algorithm with move limits performs when it is applied to the underconstrained test problem

$$\begin{aligned} &\text{OBTAIN } (x, y) \\ &\text{THAT MINIMIZES } f(x, y) = y \\ &\text{SUBJECT TO } g(x, y) = x^2 - y \leq 0. \end{aligned} \tag{5}$$

Taking a glance at this figure throws some light on these facts. In particular, it is clear that large scale oscillations occur because of the repeated linearization of the original problem.

This suggests that the convergence of the method could be achieved by somehow introducing higher order information. With the aim of keeping the computational requirements under acceptable levels, we propose an algorithm that only requires higher order information in the search direction.

To get the search direction $\boldsymbol{s}^k$, we must previously perform a full first order sensitivity analysis of the objective function and the constraints. This requires to compute

$$\left.\frac{dF}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \quad \text{and} \quad \left.\frac{dG_j}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^k}, \qquad j = 1, \ldots, p$$

what is straightforward in terms of the above mentioned sensitivity analysis techniques [5,7,8].

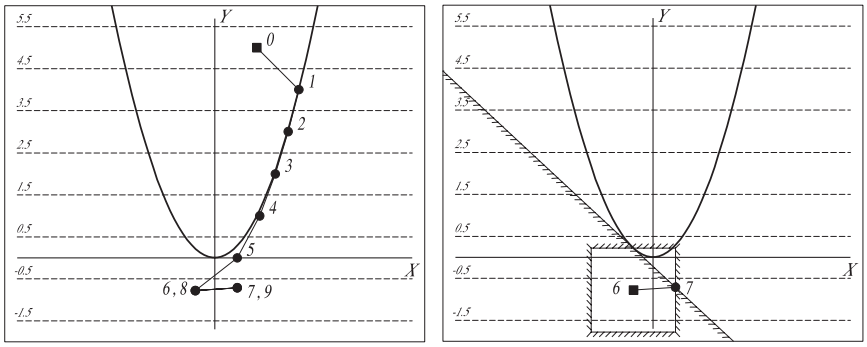
Furthermore, we must classify the actual situation at the iterate $\boldsymbol{x}^k$ depending on the degree of satisfaction of the constraints. Thus, we will say that the constraint number j is

$$\begin{cases} \text{DEACTIVATED} & \text{if} \quad G_j(\boldsymbol{x}^k) < -G^{ACT}, \\ \text{ACTIVATED} & \text{if} \quad G_j(\boldsymbol{x}^k) \geq -G^{ACT}, \\ \text{VIOLATED} & \text{if} \quad G_j(\boldsymbol{x}^k) > \ \ G^{TOL}, \\ \text{STRONGLY VIOLATED} & \text{if} \quad G_j(\boldsymbol{x}^k) > \ \ G^{ACT}, \end{cases}$$

being $0 < G^{TOL} < G^{ACT}$ the parameters that control the performance of the algorithm.

In the most frequent case we will adopt the tentative search direction $\boldsymbol{s}^k$ given by the SLP algorithm with move limits (4), that can be found by means of any linear programming code.

However, it can happen that this procedure fails. Actually, this is quite probable when there are strongly violated constraints at the actual iterate $\boldsymbol{x}^k$. Therefore, we must provide complementary techniques to choose a tentative search direction in these cases. Since the prime objective should



a) **b)**

Fig. 1.– Application of the SLP algorithm with move limits to the test problem. Performance of the algorithm (a). Linearization and large scale oscillations near the optimum (b).

be moving towards the interior of the feasible region, we suggest the linear combination of the strongly violated constraints

$$\boldsymbol{s}^k = \{s_i^k\}, \qquad s_i^k = -\sum_{G_j > G^{ACT}} \beta_j \left.\frac{\partial G_j}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{x}^k}, \qquad i = 1, \ldots, m$$

being

$$\beta_j = G_j(\boldsymbol{x}^k) \Big/ \sum_{i=1}^{m} \left(\left.\frac{\partial G_j}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{x}^k} \right)^2, \qquad j = 1, \ldots, p.$$

On the other hand, if all the constraints are deactivated, any unconstrained minimization method could be used to obtain a tentative search direction. Since this case will be accidental in practical problems, even the steepest descent method can be considered valid for practical purposes, giving

$$\boldsymbol{s}^k = \{s_i^k\}, \qquad s_i^k = -\left.\frac{\partial F}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{x}^k}, \qquad i = 1, \ldots, m.$$

In the last two cases, the tentative search direction has been obtained without taking the side constraints into account. Anyhow, this is quite simple to sort out, since we can just annihilate the inconvenient components *a posteriori*. Thus,

$$s_i^k = 0 \text{ if } [(x_i^k < a_i) \text{ and } (s_i^k < 0)] \text{ or } [(x_i^k > b_i) \text{ and } (s_i^k > 0)].$$

At this point we must check if the found tentative search direction is null. If this happens, we must discern if the last iterate is the optimum, or

the algorithm is stalled (what can happen for inappropriate adjustments of the algorithm parameters, or just because the problem is not well posed). Otherwise, the search direction must be normalized before proceeding to obtain the step θ^k.

Finally, the original problem is substituted by an approximated univariate optimization problem, via a second order Taylor expansion in the search direction $\boldsymbol{s}^k$. Thus, we must perform a second order sensitivity analysis of the objective function and the constraints in the search direction [5,7,8]. In general, this involves an acceptable cost, since we just need to additionaly compute the directional derivatives

$$D^2_{(s^k)^2} F(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k} \qquad \text{and} \qquad D^2_{(s^k)^2} G_j(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k}, \qquad j = 1, \ldots, p.$$

Then, the solution to the resulting univariate quadratic problem

$$\begin{aligned}
&\text{GIVEN } \boldsymbol{x}^k = \{x_i^k\} \text{ AND } \boldsymbol{s}^k = \{s_i^k\} && i = 1, \ldots, m \\
&\text{OBTAIN } \theta^k \\
&\text{THAT MINIMIZES } \Phi(\theta) = F(\boldsymbol{x}^k) + \\
&\qquad \left[D_{s^k} F(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \theta + \\
&\qquad \left[D^2_{(s^k)^2} F(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \frac{1}{2}\theta^2, \\
&\text{SUBJECT TO } \Gamma_j(\theta) = G_j(\boldsymbol{x}^k) + \\
&\qquad \left[D_{s^k} G_j(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \theta + \\
&\qquad \left[D^2_{(s^k)^2} G_j(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}^k} \right] \frac{1}{2}\theta^2 \leq 0, && j = 1, \ldots, p \\
&\qquad a_i \leq x_i^k + \theta\ s_i^k \leq b_i, && i = 1, \ldots, m \\
&\qquad \delta_i^- \leq \theta\ s_i^k \leq \delta_i^+, && i = 1, \ldots, m
\end{aligned} \tag{6}$$

gives the step θ^k, and the new iterate $\boldsymbol{x}^{k+1}$ can be computed by means of the actualization formula (2). Different basic strategies are used to solve this univariate problem. In particular, we must prevent the algorithm to get stalled when the actual iterate is very close to the boundary of the feasible region, or the above stated approximated problem is inconsistent and has no solution.

Figure 2 shows the performance of the proposed algorithm when it is applied to the test problem (5). A complete description of the algorithm, and a comparative study of its performance, can be found in [9].

It must be clear that the directional second order sensitivity analysis is essential to find the step θ^k, since solving problem (6) in terms of the exact expressions of the objective function and the constraints should involve unacceptable computing requirements.

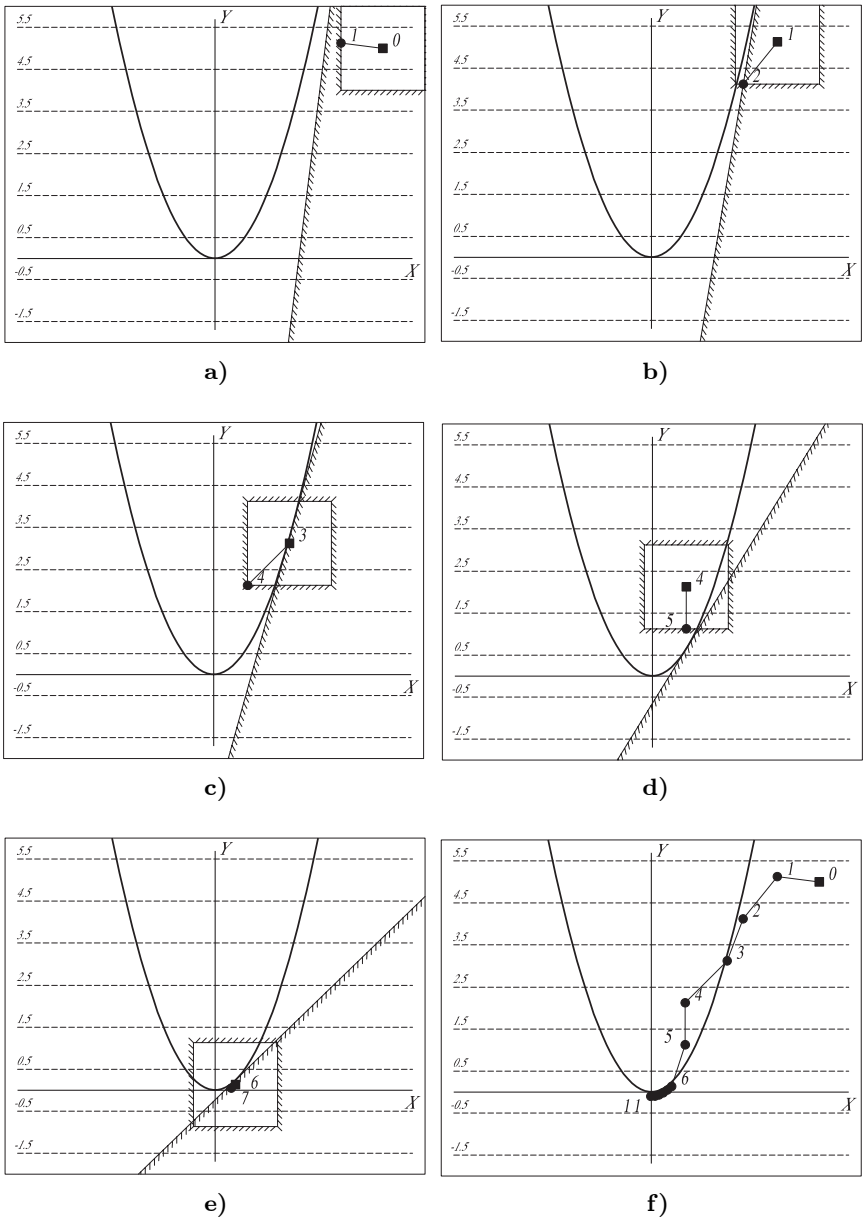


Fig. 2.– Application of the proposed algorithm to the test problem. Performance of the method when the inequality constraint is strongly violated (a), activated (b and c), and deactivated (d). Quadratic information precludes oscillations near the optimum (e). Convergence in 11 iterations (f).

5 Conclusions

The use of high order MP algorithms to solve structural shape optimization problems is precluded in practice, due to the exponential growth of the sensitivity analysis computational cost. However, a high order sensitivity analysis can be performed for a given direction in the design space with relatively small computational requirements.

This provides an extremely useful tool for improving and correcting malfunctions of existing first order MP algorithms, and gives new clues to design new techniques.

Acknowledgements

This work has been partially supported by Grant Number TIC-98-0290 of the "Comisión Interministerial de Ciencia y Tecnología" (CICYT) of the Spanish Government, and Grant Number PGIDT-99MAR11801 of the "Consellería de Educación e Ordenación Universitaria" of the "Xunta de Galicia".

References

[1] Schmit L.A., *Structural Design by Systematic Synthesis*, Proc. of the Second ASCE Conference on Electronic Computation, 105–122, Pittsburgh, Pennsylvania, 1960

[2] Fletcher R.J., *Practical Methods of Optimization*, John Wiley and Sons, 1991

[3] Sandgren E. and Ragsdell K.M., *The Utility of Non Linear Programming Algorithms: A Comparative Study – Part 1 and 2*, Trans. ASME, J. of Mech. Design, **102**, 540–551, 1980

[4] Navarrina F. and Casteleiro M., *A General Methodologycal Analysis for Optimum Design*, Int. J. Num. Meth. Engrg., **31**, 85–111, 1991

[5] Navarrina F., Colominas I., Juanes R., Bendito E. and Casteleiro M., *A Unified Approach for High Order Sensitivity Analysis*, OPTI 2001, Bolonia, 2001

[6] Vanderplaats G.N., *Numerical Optimization Techniques for Engineering Design*, McGraw-Hill, New York, 1984

[7] Navarrina F., Bendito E. and Casteleiro M., *High Order Sensitivity Analysis in Shape Optimization Problems*, Comp. Meth. in App. Mech. and Eng., **75**, 267–281, 1989

[8] Navarrina F., López S., Colominas I., Bendito E. and Casteleiro M., *High Order Shape Design Sensitivity: A Unified Approach*, Comp. Meth. in App. Mech. and Engrg., **188**, 681–696, 2000

[9] Tarrech R., *Algoritmos de Programación Matemática en Optimización Estructural*, Tesina de Especialidad, ETSICCPB, Universidad Politécnica de Cataluña, Barcelona, 1990