A SUBGRADIENT METHOD FOR CONTACT STRUCTURAL OPTIMIZATION *

MICHAEL PATRIKSSON† AND JOAKIM PETERSSON‡

Abstract. We consider the problem of maximizing the stiffness of a structure in unilateral contact by using a subgradient optimization algorithm. The problem is in general a nonlinear programming problem subject to equilibrium constraints, but can in certain cases be given a convex–concave saddle point formulation. The structure is assumed to have a linear potential energy–design dependence, an assumption which is fulfilled by discretized variable thickness sheets and trusses.

The subgradient optimization scheme is applied to the primal (displacement-only) reformulation of the saddle problem; convergence is guaranteed when design variables are allowed to take zero values, and hence topology optimization is included. The algorithm is augmented with a simple averaging scheme for generating an optimal design as well. We establish a finite termination criterion for the subgradient algorithm, and evaluate the numerical properties of the algorithm and its viability for solving large-scale engineering problems through some numerical examples. The present work is a continuation of investigations of this algorithm initiated in [13].

Key words. structural optimization, unilateral contact, nondifferentiable convex programming, subgradient method

1. Introduction. Structural optimization is the problem of optimizing the performance of an assemblage of material which is subject to external loads. The objective function reflects what exactly is desired from the structure, and the design variables are the parameters subject to changes when extremizing this objective. In addition to constraints on these variables (design constraints), it is also necessary to introduce equilibrium conditions; since equilibrium is governed by a variational inequality, structural optimization problems have, in general, an inherent bilevel programming form.

Several performance objectives are of interest from the engineer's viewpoint; minimizing the weight of the structure under conditions on the stresses, and maximizing the fundamental frequency or buckling load subject to a constraint on the available amount of material, are two such examples.

Compliance topology optimization is probably the structural optimization problem which has been most frequently studied and used during the last decade. Changes in topology usually imply drastic improvements in the structural performance, and choosing compliance as a performance measure also implies that the bilevel formulation of the problem reduces to a convex program.

The topology nature of the optimization is usually incorporated by allowing the thickness type design variables to approach zero. The frequently used (heuristic) optimality criterion methods, however, need strictly positive lower design bounds in order to guarantee the solvability of the equilibrium problems. (When design bounds are zero, this problem may not have a solution, or it may have several solutions.)

In this paper we present an iterative method, based on subgradient optimization, for the solution of structural optimization problems which is convergent also when

*This work was supported by the Swedish Research Council for Engineering Sciences (TFR), grants 92-426 and 282-93-1195.

† Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195-4350, USA. Currently at Department of Mathematics, Division of Optimization, Linköping Institute of Technology, S-581 83 Linköping, Sweden.

‡Department of Mechanical Engineering, Division of Mechanics, Linköping Institute of Technology, S-581 83 Linköping, Sweden.

allowing voids in the structural domain, and when enforcing unilateral displacement constraints. The compliance objective is extended to include contact. Despite the fact that the rate of convergence of the algorithm is slow measured in the number of iterations needed, it works well also in a large-scale setting. The main reason for this is that the problem structure is exploited maximally, thus resulting in a very low computational cost per iteration. Moreover, the convergence of the algorithm is monitored through the evaluation of both lower and upper bounds on the optimal objective value, and both optimal designs and displacements are generated.

This paper supplements [13], which establishes the convergence and complexity of the iterative scheme and provides mechanical evaluations of the solutions to ($S\Pi$). We concentrate here on one hand on providing new theoretical results concerning the generation of an optimal design and the finite termination of the subgradient algorithm, and on the other hand on evaluating the numerical properties of the algorithm.

The study of mathematical programs with equilibrium constraints is a fairly new discipline, which is largely unknown to the structural optimization community. While one aim of the present paper is to illustrate some features of the proposed algorithm when applied to a topology optimization problem, an equally important aim is to facilitate an increase of interdisciplinary flows of information between the structural optimization and mathematical programming communities.

2. Problem Statements.

2.1. The Equilibrium Problem. We consider a linear–elastic, discrete (or, discretized) structure whose state of displacements is represented by a vector $u \in \Re^n$. Forces acting on the body, $F_{\text{tot}} \in \Re^n$, are composed of forces due to contact with a rigid body, $F_{\text{c}} \in \Re^n$, and external prescribed forces $f \in \Re^n$:

$$F_{\text{tot}} = F_{\text{c}} + f, \tag{1}$$

and related to the state of displacement through the linear relation

$$F_{\text{tot}} = Ku, \tag{2}$$

where K is an $n \times n$ symmetric and positive semidefinite stiffness matrix.

The contact displacements $w \in \Re^r$ can be calculated from u through

$$w = Cu, \tag{3}$$

where C is a $r \times n$ kinematic transformation matrix which is quasi-orthogonal, that is, $CC^T = I$ holds, where I is the $r \times r$ unit matrix. When no friction is present between the body and the rigid obstacle, F_{c} is related to the vector $\lambda \in \Re^r$ of contact force magnitudes by

$$F_{\text{c}} = -C^T \lambda. \tag{4}$$

Combining (1), (2) and (4) yields

$$f = Ku + C^T \lambda, \tag{5}$$

which is the equation of *force equilibrium.*

We let $g \in \Re^r$ be a vector of initial distances between contact boundary nodes of the body and the rigid obstacle. We then have the frictionless Signorini contact conditions:

$$w \leq g; \quad \lambda \geq 0; \quad \lambda^T (w - g) = 0, \tag{6}$$

due to requirements of *non-penetration*, *no adhesion* and *no contact action at a distance*, respectively.

It is well-known that (3), (5) and (6) constitute the necessary and sufficient Karush–Kuhn–Tucker (KKT) conditions for u to solve the quadratic program (QP)

$$\min_{u \in \mathcal{U}} \Pi(u), \tag{7}$$

where

$$\Pi(u) = \frac{1}{2} u^T K u - f^T u$$

is the total potential energy, and

$$\mathcal{U} = \{ u \in \Re^n \mid Cu \leq g \} \tag{8}$$

is the set of kinematically admissible displacements. This is the well-known *principle of minimum potential energy* in solid mechanics. We remark that equilibrium is taken care of by either one of the QP (7) or the mixed LCP (3), (5) and (6).

2.2. The Design Optimization Problem. A structural optimization problem can be given the following general formulation:

$$\min_{\substack{(s,h) \in \mathcal{Z} \\ (3),(5),(6)}} F(s,h), \tag{9}$$

in which (3), (5) and (6) constitute the *equilibrium constraints*. Here, F is the objective function reflecting what is desired from the design, s are the state variables, h the design variables, and $\mathcal{Z}$ is defined by the *design* and *behavioural constraints*. This is a mathematical program with equilibrium constraints (MPEC) [10].

We will consider a particular instance of (9). The state variables are displacements and contact forces; hence $s = (u^T, \lambda^T)^T$. We omit behavioural constraints (one such will instead be represented in the objective), and let

$$\mathcal{Z} = \Re^{n+r} \times \mathcal{H}, \tag{10}$$

where

$$\mathcal{H} = \left\{ h \in \Re^m \;\middle|\; \sum_{i=1}^m h_i \equiv 1^T h = V; \quad \underline{h} \leq h \leq \overline{h} \right\} \tag{11}$$

is the set of permissible designs. The design variables h are structural thicknesses or volumes, such as bar volumes for trusses or thicknesses representing the discretized thickness function for a sheet, and V is the available amount of material. These variables affect the structural stiffness matrix (only), and we write $K = K(h)$. In fact, $h \mapsto K(h)$ is linear:

$$K(h) = \sum_{i=1}^m h_i K_i, \tag{12}$$

where K_i is the global specific element stiffness matrix for the ith structural element. The matrices K_i are symmetric and positive semidefinite, and we assume that there exists at least one stable structure, that is, at least one $h \in \mathcal{H}$ such that $K(h)$ is positive definite.

Henceforth we take (12) into account and in all previous expressions where K appears, it is replaced by $K(h)$. For example, $\Pi(u,h)$ will mean $\Pi(u)$ with $K(h)$ in place of K.

As cost function we choose an extension of the objective used in the established area of *compliance minimization*, namely

$$F(s,h) = \overline{F}(u,\lambda) = f^T u + g^T \lambda, \tag{13}$$

which clearly is design-independent. In case $g = 0$ we obtain the classical compliance term, but, in general, the choice (13) means a strive for small contact force magnitudes in addition to small displacements.

It is a main result of [12] that (9) with the choices (10), (11), (13), and mixed LCP equilibrium constraints (3), (5) and (6), can be solved by finding a saddle point $(u^*, h^*) \in \mathcal{U} \times \mathcal{H}$ to Π:

$$(S\Pi) \quad \begin{cases} \text{Find } (u^*, h^*) \in \mathcal{U} \times \mathcal{H} \ : \\ \Pi(u^*, h) \leq \Pi(u^*, h^*) \leq \Pi(u, h^*), \qquad \forall (u,h) \in \mathcal{U} \times \mathcal{H}. \end{cases}$$

If (u^*, h^*) solves $(S\Pi)$, and $\lambda^* \in \Re^r$ are Lagrange multipliers associated with the contact conditions describing the set $\mathcal{U}$, then (u^*, λ^*, h^*) solves (9) (with the particular choices above).

Associated with the saddle problem $(S\Pi)$ is the primal–dual pair of convex programs

$$(P) \quad \begin{cases} \text{Find } u^* \in \mathcal{U} \ : \\ \psi(u^*) \leq \psi(u), \qquad \forall u \in \mathcal{U}, \end{cases}$$

where $\psi(u) \stackrel{\text{def}}{=} \sup_{h \in \mathcal{H}} \Pi(u,h)$, $u \in \mathcal{U}$, and

$$(D) \quad \begin{cases} \text{Find } h^* \in \mathcal{H} \ : \\ \varphi(h^*) \geq \varphi(h), \qquad \forall h \in \mathcal{H}, \end{cases}$$

where $\varphi(h) \stackrel{\text{def}}{=} \inf_{u \in \mathcal{U}} \Pi(u,h)$, $h \in \mathcal{H}$. There exist nonempty solution sets $\mathcal{U}^*$ and $\mathcal{H}^*$ to (P) and (D), respectively, and $\mathcal{U}^* \times \mathcal{H}^*$ is the solution set of $(S\Pi)$. We denote by Π^* the saddle value of $(S\Pi)$.

In the next section, we present an iterative algorithm for the solution of $(S\Pi)$ through the solution of the primal problem (P). The reason for utilizing (P) is that the evaluations of objective values and subgradients of ψ are very simple operations; one merely solves a highly structured linear program. Although the algorithm supplies a solution u^* to the primal problem only, an optimal solution h^* to the dual problem (D) may be easily obtained as a byproduct of the algorithm.

3. An iterative algorithm for the saddle problem.

3.1. Motivation. When choosing a numerical tool for the solution of $(S\Pi)$, one must bear in mind the special properties of the problem and the purposes of solving it.

First, the fact that zero design volumes are allowed implies that the saddle function Π is non-strictly convex–concave. This implies, in turn, that the analysis problem may fail to have a solution or have multiple solutions for certain designs, and that an optimal design cannot be generated through the solution of the linear design problem given an optimal state vector. A consequence is that the classical nested approach in structural optimization (e.g., [7]) does not produce a solution to $(S\Pi)$.

Second, in order to draw qualitative conclusions regarding solutions to a continuous problem, one needs to solve the discretized problem $(S\Pi)$ using a very fine mesh. Hence, the algorithm chosen should have a low time complexity in the parameter m. For truss topology design problems without contact, efficient algorithms have been proposed recently in [3, 4], the first being an ε-descent algorithm and the second an interior point scheme. These algorithms have a favourable complexity, due to the sparsity of truss structures, that is, $n \ll m$ holds. [For example, the time complexity per iteration in the algorithm described in [4] is $O(n^3 + mn^2)$.] For finite-element problems such as $(S\Pi)$, however, n is of the same order as m, in which case the resulting complexity is rather high. (A very recent contribution to the literature of interior point methods is, however, [5], the algorithm of which utilizes the sparsity of the finite-element stiffness matrix to reach a time complexity only slightly higher than quadratic in m.) Further, the presence of contact conditions would make the search direction finding problems even more complex in both approaches, possibly resulting in a questionable efficiency.

The algorithm chosen is a *subgradient optimization* algorithm for the solution of the program (P). This algorithm has several advantages pertaining to the above discussion: zero design volumes are allowed, and the presence of the contact conditions has very little effect on the complexity of the algorithm. In fact, it utilizes the problem structure very efficiently, to the extent that both the storage requirements and the time complexity per iteration are *linear* in the parameters m, n, and r [13]. (The total number of iterations needed to reach a given accuracy was, from experiments, found to roughly be linear in m, resulting in a quadratic overall time complexity.) Further, it is easy to implement, and includes few algorithm parameters that need to be calibrated. (We note that the making of efficient implementations of interior point methods often is associated with the careful selection of several algorithm parameters; an automatic means for choosing the parameters a and b in the step length formula (16) below is given in [16, p. 17].)

The algorithm also has potential disadvantages. The number of iterations needed to reach a satisfactory design solution certainly is high, as demonstrated in theory for subgradient algorithms by Poljak [14]; in our experience, however, the very low time complexity per iteration seems to counterbalance this (inherent) property. (For engineering applications, the accuracy required in the design solution is such that the engineer is able to perform qualitative analyses regarding its topological structure; a design solution accurate to a few digits is then sufficient.)

Another potential drawback is that, contrary to the algorithms in [3, 4], the subgradient scheme does not provide an optimal design automatically. This problem was solved in [13], where we presented a simple averaging scheme to be used within the subgradient method for (P) for the simultaneous generation of an optimal design.

3.2. The subgradient algorithm. The subgradient optimization algorithm [14, 15, 18] for solving (P) generates a sequence $\{u^k\}$ of feasible solutions by the formula

$$u^{k+1} = P_{\mathcal{U}}(u^k - \alpha_k \xi_\psi(u^k)), \qquad k = 1, 2, \ldots, \tag{14}$$

where

$$\xi_\psi(u^k) \in \partial\psi(u^k) \tag{15}$$

is an element of the subdifferential (that is, a subgradient) of ψ at u^k, and $P_{\mathcal{U}}$ denotes the Euclidean projection map onto $\mathcal{U}$. We choose step lengths α_k according to the harmonic series formula

$$\alpha_k = \frac{1}{a + bk}, \quad a \geq 0, \quad b > 0, \qquad k = 1, 2, \ldots; \tag{16}$$

the reason for this rather restrictive choice of step lengths is that the averaging scheme for generating an optimal design, according to the convergence analysis in [13], apparently requires this.

How to compute a subgradient of ψ and perform the projection operation will be discussed next.

3.3. Generating Subgradients. Using a well-known result on subgradients of max-functions (e.g., [8]), a subgradient of ψ is given by

$$\xi_\psi(u^k) = K(h^k)u^k - f,$$

where

$$h^k \in \mathcal{H}(u^k) \stackrel{\text{def}}{=} \arg\max_{h \in \mathcal{H}} \Pi(u^k, h). \tag{17}$$

[Hence, h^k is a solution to the linear program defing the value $\psi(u^k)$.]

To solve the linear program (17), we observe that it may be written as

$$\max_{h \in \mathcal{H}} \sum_{i=1}^{m} c_i^k h_i, \tag{18}$$

where

$$c_i^k = (u^k)^T K_i u^k \geq 0, \qquad i = 1, 2, \ldots, m, \tag{19}$$

by the positive semidefiniteness of each matrix K_i. From the structure of the feasible set $\mathcal{H}$, it follows that (18) is a continuous knapsack problem, which belongs to the class of problems solvable by greedy algorithms [11].

Greedy algorithm for (18)

Step 0 (Initialization): Let $\hat{h} = \underline{h}$.

Step 1 (Iterative step): Repeat the following until $1^T \hat{h} = V$:

1. Let $\tilde{\imath} \in \arg\max_{i \in \{1,\ldots,m\}} \{ c_i^k \mid \hat{h}_i < \overline{h}_i \}$.
2. Let $\hat{h}_{\tilde{\imath}} := \hat{h}_{\tilde{\imath}} + \min\{V - 1^T \hat{h}, \overline{h}_{\tilde{\imath}} - \hat{h}_{\tilde{\imath}}\}$.

The solution to (18) then is $h^k = \hat{h}$.

To carry out Step 1, the direct approach is to sort the m elements c_i^k in descending order. (The best complexity of a sorting algorithm is known to be $O(m \log m)$ [1, Chap. 8].) A better approach is to identify the active designs through some breakpoint or median search, based on linear programming duality; in both approaches, it is possible to utilize the previous solution to (18) as a starting solution.

In the interesting special case when $\underline{h}_i = 0$ for all i and all $\overline{h}_i$ are equal (to, say, $\breve{h}$), we can, however, do even better. Indeed, to solve the linear program (18) it is only necessary to select the $\lceil V/\breve{h} \rceil$:th largest element of the vector c^k, where $\lceil x \rceil$ denotes the smallest integer larger than or equal to x, and (as a byproduct of this procedure) partition the remaining elements into two groups depending on if they would precede the $\lceil V/\check{h}\rceil$:th largest element or not in a completely sorted list. The solution is then obtained by letting $h_i^k = \check{h}$ for all i corresponding to elements in c^k that precede the breakpoint element, and to let that element receive the remainder of the total volume $V - \check{h}(\lceil V/\check{h}\rceil - 1)$. The complexity of this selection operation [and therefore the complexity of the linear program (18)] is $O(m)$ (see [1, Sec. 8.7]).

We have experimented only with problems with zero lower design limits and a uniform upper design limit, and found the selection procedure to be quite satisfactory. Experimenting also with other solution procedures for (18), we have concluded that the efficient solution of (18) is crucial for the overall efficiency of the subgradient algorithm (compare, for example, the very different computational results in [13] and [16]).

3.4. Performing projections. Given $u^k \in \mathcal{U}$, a new displacement vector u^{k+1} is calculated by projecting $u^k - \alpha_k \xi_\psi(u^k)$ onto $\mathcal{U}$. Due to the quasi-orthogonality of the matrix C, the Euclidean projection of any vector in $\Re^n$ onto $\mathcal{U}$ is performed by (essentially) projecting a vector onto the nonnegative orthant $\Re^r_+$, which is a much more simple operation than a Euclidean projection onto a general polyhedral set; the projection of a point $z \in \Re^r$ onto $\Re^r_+$ is given by $P_{\Re^r_+}(z) = \max\{0, z\}$, where the maximum is taken component-wise.

LEMMA 3.1. [13, Le. 3.1] *Suppose that C is an $r \times n$ quasi-orthogonal matrix and that $\mathcal{U}$ is given by (8). Then, for any $x \in \Re^n$,*

$$P_{\mathcal{U}}(x) = x - C^T P_{\Re^r_+}(Cx - g).$$

3.5. A Design Sequence and Convergence Properties. In general, the sequence $\{h^k\}$ does not accumulate at a design solution; this is due to the linearity of Π in h, and is in applications of Lagrangean relaxation known as the *non-coordinability phenomenon*. We established in [13] that the simple average of the linear programming solutions h^k does converge to the set of optimal designs; this is of computational importance, since previous approaches for introducing coordinability either amounts to solving design problems over the convex hull of the points h^k or to augment Π with a strictly convex function in h (see [13] for more details on these and other potential methods for solving $(S\Pi)$).

We define an auxiliary sequence $\{h(k)\}$ by

$$h(k) = \frac{1}{k}\sum_{s=1}^{k} h^s, \qquad k = 1, 2, \ldots, \tag{20}$$

where h^k is given by (17). Clearly, $h(k) \in \mathcal{H}$ for all k. (In practice, we calculate $h(k)$ according to $h(k) = (1 - \frac{1}{k})h(k-1) + \frac{1}{k}h^k$.)

We next give the main convergence result for the combined algorithm.

THEOREM 3.2. [13, Thm. 4.3] (Convergence properties of the overall scheme) *Let the sequence $\{u^k\}$ be generated by the algorithm (14), (15), (16), and assume that the sequence $\{\xi_\psi(u^k)\}$ is bounded. Let the sequence $\{h(k)\}$ be defined by (20). Then,*

$$\{u^k\} \to u^* \in \mathcal{U}^*,$$

and

$$\{d_{\mathcal{H}^*}(h(k))\} \to 0,$$

where

$$d_{\mathcal{H}^*}(x) = \inf_{h \in \mathcal{H}^*} \|x - h\|.$$

We remark that the boundedness of the sequence $\{\xi_\psi(u^k)\}$ is guaranteed by letting the value of a be sufficiently large, since $\{\alpha_k\}$ is decreasing and the set $\mathcal{U}^*$ of optimal displacement solutions is bounded [18, p. 27].

It is not necessary to introduce the averaging scheme from the very beginning of the subgradient optimization scheme, and there is a good reason for not doing so. An implication of Theorem 3.2 and the Representation Theorem for bounded polyhedral sets (e.g., [2, Thm. 2.6.7]) is that the frequency by which a given extreme point of $\mathcal{H}$ appears as a solution to (18) tends to an optimal portion of the total volume V that should be allocated to that extreme point solution. The initial frequencies are, however, poor estimates of these optimal portions, and, therefore, initializing the averaging scheme (20) too early will result in having to apply it many iterations to filter out the early iterates h^s included in $h(k)$. A theoretical support for postponing the averaging scheme is also provided in the below result.

THEOREM 3.3. (Finite identification of solutions to the linear design problem) *Let the sequence $\{u^k\}$ be generated by the algorithm (14), (15), (16), and assume that the sequence $\{\xi_\psi(u^k)\}$ is bounded. Further, let $u^* \in \mathcal{U}^*$ be the limit point of $\{u^k\}$. Then, there exists a positive integer k_0 such that*

$$\mathcal{H}(u^k) \subset \mathcal{H}(u^*), \qquad k \geq k_0. \tag{21}$$

Proof. According to Theorem 3.2, the sequences $\{c_i^k\}$, $i = 1, \ldots, m$, of cost coefficients in the linear program (19) converges. The result then follows by appealing to [6, Thm. 3.1]. ∎

Consequently, eventually the solutions to (18) are solutions to the linear design problem at optimum. If, therefore, the averaging scheme (20) is postponed until the (unknown) iteration k_0, we can expect a much faster convergence of the sequence $\{h(k)\}$. Experiments regarding this topic are made in Section 4.

3.6. A finite termination criterion. In this section we establish a natural finite termination criterion for the solution of $(S\Pi)$; the subgradient optimization algorithm is terminated when the difference between an upper and a lower bound on the saddle value Π^* is small enough.

The function ψ provides upper bounds on Π^*. (For example, an upper bound $\psi(u^k)$ is generated automatically in the subgradient optimization algorithm.) Analogously, on its effective domain φ provides lower bounds on Π^*. Its calculation however amounts to solving a quadratic program over $\mathcal{U}$, which further does not always have an optimal solution. Instead, we propose to use the (under-estimating) function $\underline{\varphi} : \mathcal{H} \times \Re^r_+ \to \Re \cup \{-\infty\}$, where

$$\underline{\varphi}(h, \lambda) = \inf_{u \in \Re^n} \left\{ \Pi(u, h) + \lambda^T (Cu - g) \right\}. \tag{22}$$

The advantage of using $\underline{\varphi}$ in place of φ is that the quadratic program (22) is equivalent to solving a system of linear equations. Note, however, that there may be many solutions to this system (if any), and that the coefficient matrix may have many zeros on the main diagonal, so that a direct equation solver is not suitable. (We have chosen the Fletcher–Reeves algorithm (e.g., [2, Sec. 8.8]) in our implementation.) Obviously, lower bounds are not generated initially, and only with regular intervals.

3.6.1. Assumptions. The results of this section are valid for any convergent method for $(S\Pi)$. We will assume throughout the development that the following assumptions hold:

$$\mathcal{U} \supset \{u^k\} \to u^* \in \mathcal{U}^*, \quad \{h(k)\} \subset \mathcal{H}, \quad \{d_{\mathcal{H}^*}(h(k))\} \to 0, \quad \{\lambda^k\} \subset \Re^r_+, \tag{23}$$

where $\lambda \in \Re^r_+$ denotes a vector of Lagrange multipliers for the contact conditions.

We will also at times introduce additional assumptions. Concerning the sequence $\{u^k\}$, this will never be necessary, but the design sequence will sometimes be assumed to be convergent, that is,

$$\{h(k)\} \to h^* \in \mathcal{H}^*. \tag{24}$$

According to Theorem 3.2, this is the case if $\mathcal{H}^*$ is a singleton, which seems to be common in practice. At some stage, we will also need

$$\{\lambda^k\} \to \lambda^*, \tag{25}$$

for some optimal vector of Lagrange multipliers. One such sequence is given by

$$\lambda^k = P_{\Re^r_+}(C[f - K(h(k))u^k]), \qquad k = 1, \ldots, \tag{26}$$

which, in fact, satisfies (25) whenever (24) holds (cf. [13]).

We next present some assumptions on the optimal structure and the load cases.

Assume that $(S\Pi)$ is the result of a FE-discretization of an infinite-dimensional problem. Then, in this problem, it is stipulated that the contact boundary and traction boundary are disjoint, which leads us to the assumption

$$f^T \nu_i = 0, \qquad i = 1, \ldots, r, \tag{27}$$

where ν_i^T is the ith row of C. This means that the external loads are not applied at potential contact nodes, or if they are, they are parallel to the contact surfaces. (A detailed description of the infinite-dimensional problem is found in [13].)

Let, for $i = 1, \ldots, r$, $J(i) \subset \{1, \ldots, m\}$ be the set of all element numbers for which the corresponding element is connected to contact node i. Define also for any $\lambda \in \Re^r_+$ the index set $\tilde{J}(\lambda) = \{\, j \in \{1, \ldots, r\} \mid \lambda_j > 0\}$, and let $\mathcal{N}(K)$ denote the null space of the matrix K. The final assumption is that of a "non-singularity" of an optimal design h^*:

$$u \in \mathcal{N}(K(h^*)) \text{ with } \nu_i^T u \neq 0 \text{ for some } i \in \{1, \ldots, r\} \Longrightarrow h_j^* = 0, \; j \in J(i). \tag{28}$$

An interpretation of (28) is that "zero energy" displacements at a certain part of the optimal structure are possible only if this part of the structure has no material (that is, a zero volume). Hence, we assume (28) to hold if we do not expect the optimal design to contain mechanisms or to be displaceable in rigid body motions. Experience shows that optimal trusses might include mechanisms, but not such that the corresponding displacement is at the contact nodes and with a non-zero component perpendicular to the obstacle (cf. Fig. 5 in [9]), so (28) still holds. Concerning rigid body motions, however, it is not very difficult to imagine optimal structures containing them; if the available volume in the case shown in Fig. 9 below is sufficiently small, then the optimal structure will rest on the unilateral supports only and will not be attached to the "ceiling", hence being displaceable in, for example, rigid body translations.

3.6.2. Propositions. In order to establish the continuity of $\underline{\varphi}$, we first establish its finiteness. The following is an application of a well-known result in quadratic programming theory.

LEMMA 3.4. *Let $h \geq 0$. The function $u \mapsto \Pi(u,h)$ attains its minimum over $\mathcal{U}$ if and only if*

$$u \in \mathcal{N}(K(h)),\ Cu \leq 0 \quad \Longrightarrow \quad f^T u \leq 0. \tag{29}$$

(The necessity of (29) is quite obvious, and the sufficiency follows from [17, Thm. 27.3] and the fact that the recession cones of $\mathcal{U}$ and $u \mapsto \Pi(u,h)$ are $\{ u \in \Re^n \mid Cu \leq 0 \}$ and $\{ u \in \Re^n \mid Ku = 0,\ f^T u \geq 0 \}$, respectively.)

LEMMA 3.5. *Let (24) hold. Then, there exists a nonempty polyhedral set $S_1 \subset \mathcal{H}$ such that $h(k) \in S_1$ for all but finitely many k, and*

$$h \in S_1 \quad \Longrightarrow \quad \mathcal{N}(K(h)) \subset \mathcal{N}(K(h^*)). \tag{30}$$

Proof. Let $I^* = \{ i \in \{1,\dots,m\} \mid h_i^* > 0 \}$ and $\varepsilon = \min_{i \in I^*}\{h_i^*\}$. Define the set $S_1 = \{ h \in \Re^m \mid |h_i - h_i^*| \leq \varepsilon/2,\ i \in I^* \} \cap \mathcal{H}$. By the convergence of the sequence $\{h(k)\}$, $h(k) \in S_1$ holds for all but finitely many indices. To complete the proof, it remains to establish (30).

Let $h \in S_1$ and take any $u \in \mathcal{N}(K(h))$. Due to the symmetry and positive semidefiniteness of $K(h^*)$ it suffices to show that $u^T K(h^*) u = 0$. From the definition of S_1, $h_i^* \leq 2\overline{h} h_i/\varepsilon$ holds for all $i \in I^*$, where $\overline{h} = \max_i\{\overline{h}_i\}$. Hence,

$$0 \leq u^T K(h^*) u = \sum_{i \in I^*} h_i^* u^T K_i u \leq \frac{2\overline{h}}{\varepsilon} \sum_{i \in I^*} h_i u^T K_i u \leq \frac{2\overline{h}}{\varepsilon} u^T K(h) u = 0,$$

that is, $u \in \mathcal{N}(K(h^*))$. ☐

THEOREM 3.6. (Bound inequalities and convergence) *For all k,*

$$\underline{\varphi}(h(k), \lambda^k) \leq \varphi(h(k)) \leq \Pi^* \leq \psi(u^k) \tag{31}$$

holds, and

$$\{\psi(u^k)\} \to \Pi^*. \tag{32}$$

Further,

$$\{\varphi(h(k))\} \to \Pi^*. \tag{33}$$

Proof. The last two inequalities of (31) follow from the definition of $(S\Pi)$, which, together with weak duality, also yields the first inequality. The function ψ is continuous, so (32) is immediate.

Let $\{h(k_i)\}$ be any subsequence of $\{h(k)\}$ such that $\{h(k_i)\}$ converges to some $h^* \in \mathcal{H}^*$. The second inequality of $(S\Pi)$ yields, according to Lemma 3.4,

$$u \in \mathcal{N}(K(h^*)),\ Cu \leq 0 \quad \Longrightarrow \quad f^T u \leq 0. \tag{34}$$

Lemma 3.5 yields a set S_1 including all but finitely many $h(k_i)$. For any $h \in S_1$ and any $u \in \mathcal{N}(K(h))$ such that $Cu \leq 0$, $f^T u \leq 0$ holds by (30) and (34). Hence, from

Lemma 3.4, φ is finite on S_1. Now, $-\varphi$ is a convex function on a locally simplicial set $S_1 \subset \mathtt{dom}(-\varphi)$, and hence φ is lower semi-continuous relative to S_1 (cf. [17, Thm. 10.2]). This, together with the second inequality of (31), implies that $\{\varphi(h(k_i))\} \to \Pi^*$, and since $\{h(k_i)\}$ was an arbitrary convergent subsequence, (33) follows. □

The following is the main result of this section.

THEOREM 3.7. (Convergence of the lower bound) *Suppose that there exists a nonempty polyhedral set $S = S_1 \times S_2 \subset \mathcal{H} \times \Re^r_+$ such that $(h(k), \lambda^k) \in S_1 \times S_2$ for all but finitely many indices. Further, suppose that for arbitrary $(h, \lambda) \in S_1 \times S_2$,*

$$(f - C^T\lambda)^T u = 0, \qquad \forall u \in \mathcal{N}(K(h)). \tag{35}$$

If (25) holds, then

$$\{\underline{\varphi}(h(k), \lambda^k)\} \to \Pi^*. \tag{36}$$

Proof. From Lemma 3.4 we may conclude that $\underline{\varphi}(h, \lambda)$ is finite if and only if

$$u \in \mathcal{N}(K(h)) \quad \Longrightarrow \quad (f - C^T\lambda)^T u = 0,$$

that is, (35). Hence, $\mathtt{dom}(\underline{\varphi}) \supset S$, and S is locally simplicial. The function $-\underline{\varphi}$ is convex on $\Re^{m+r}$, and from [17, Thm. 10.2], $\underline{\varphi}$ is lower semi-continuous with respect to S. The rest of the proof is along the same lines as in the end of the proof of Theorem 3.6. □

The sufficient conditions of Theorem 3.7 are rather abstract. We next present some concrete examples of assumptions that lead to the conclusion of the theorem.

COROLLARY 3.8. *Suppose that at least one of the three assumptions below holds:*

(i) $\underline{h}_i > 0, \quad i = 1, \ldots, m;$

(ii) $r = 0$ *(that is, there are no contact constraints);*

(iii) *there is a* $\delta > 0$ *such that* $\min_{j \in \tilde{J}(\lambda^k)}\{\lambda_j^k\} \geq \delta$ *for all but finitely many* k, *and* (24), (27) *and* (28) *hold.*

If (25) *holds, then (36) holds.*

Proof.

(i) Let $S = \mathcal{H} \times \Re^r_+$, observe that $\mathcal{N}(K(h)) = \{0\}$ for all h in $\mathcal{H}$, and apply Theorem 3.7.

(ii) This case reduces to (33) of Theorem 3.6.

(iii) Lemma 3.5 (and (24)) implies the existence of a set $S_1 \subset \mathcal{H}$ such that (30) holds. Let $S_2 = \{\, \lambda \in \Re^r \mid |\lambda_i - \lambda_i^*| \leq \delta/2,\, i \in \tilde{J}(\lambda^*);\ \lambda_i = 0,\, i \notin \tilde{J}(\lambda^*)\,\}$. The set $S = S_1 \times S_2$ satisfies all conditions of Theorem 3.7 except possibly (35). Since (25) holds, (36) follows as soon as we have established (35).

Let $(h, \lambda) \in S_1 \times S_2$ be arbitrary. Then, for any $u \in \mathcal{N}(K(h))$,

$$(f - C^T\lambda)^T u = \sum_{i=1}^{r} (\lambda_i^* - \lambda_i)\nu_i^T u, \tag{37}$$

since, by Lemma 3.4 (applied to $u \mapsto \frac{1}{2}u^T K(h^*)u - (f - C^T\lambda^*)^T u$ over $\Re^n$),

$$f^T u = (\lambda^*)^T C u, \qquad \forall u \in \mathcal{N}(K(h)) \subset \mathcal{N}(K(h^*)).$$

Assume that $\nu_i^T u \neq 0$. We will show that then $\lambda_i^* = 0$ must hold. From the second inequality of $(S\Pi)$,

$$K(h^*)u^* = f - C^T\lambda^*,$$

which, premultiplied by ν_i^T, yields

$$(u^*)^T K(h^*)\nu_i = f^T\nu_i - (\lambda^*)^T C\nu_i. \tag{38}$$

By (27), (38) reduces to

$$(u^*)^T K(h^*)\nu_i = -\lambda_i^*.$$

Continuing,

$$\lambda_i^* = -(u^*)^T\left(\sum_{j=1}^m h_j^* K_j\nu_i\right) = -(u^*)^T\left(\sum_{j\in J(i)} h_j^* K_j\nu_i\right) = 0, \tag{39}$$

by (28). By the definition of S_2 we also get $\lambda_i = 0$, and therefore

$$\sum_{i=1}^r (\lambda_i^* - \lambda_i)\nu_i^T u = 0$$

which by (37) establishes (35). □

REMARK 3.9. In (iii), the existence of a $\delta > 0$ can be ensured, for example, via the following multiplier rule (with a sufficiently small $\delta > 0$):

$$\lambda_i^k = \begin{cases} v_i^k, & \text{if } v_i^k \geq \delta, \\ 0, & \text{otherwise}, \end{cases} \qquad i = 1,\dots,r, \qquad k = 1,\dots,$$

with $v_i^k = -(u^k)^T K(h(k))\nu_i$. □

The convergence of the lower bound to the saddle value may fail if the function $\underline{\varphi}$ has a discontinuity in its second argument; this is possible only if both complete material removal and contact constraints are included. Suppose that material has been removed in a neighborhood of contact node i, so that $u := \alpha\nu_i \in \mathcal{N}(K(h))$, and the "true" value of λ_i is 0. Suppose that the value of λ_i is slightly incorrect, that is, $\lambda_i = \varepsilon > 0$. Then, $\Pi(u,h) + \lambda^T Cu = \alpha\varepsilon$ (if $f^T\nu_i = 0$), and hence, by letting $\alpha \to -\infty$, we realize that $\underline{\varphi}(h,\lambda) = -\infty$. In our numerical tests, we have used (26), and problems have occurred only in one case when (28) did not hold (for reasons mentioned in the text following (28)).

4. Numerical Investigations. We apply the method to three numerical examples. The first one is a very small truss example for which we have a closed-form solution, and the second one is a small finite-element discretized sheet example; these are used to illustrate some features of the algorithm. The third one is a large finite-element discretized sheet example, which constitutes an interesting engineering example and demonstrates the usefulness of the algorithm in a large-scale setting. Experiments were conducted on a Fortran-77 coding of the algorithm and executed on an HP 9000 Model 712/60 (Envizex X station).

4.1. Small Truss Example. We choose Example 4.1 in [12], which is a two-bar truss with Young's modulus E and with L being the second bar's length. The problem size parameters are $n = 2$, $m = 2$ and $r = 1$. Moreover, $\underline{h} = 0$, $\overline{h} = V1$, $C = (1,0)$, $g = (g_1)$, $f = (F, -2F)^T$ and

$$K_1 = \frac{E}{4L^2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \text{ and } K_2 = \begin{pmatrix} 0 & 0 \\ 0 & \frac{E}{L^2} \end{pmatrix}.$$

We choose the instance where $E = 200 \cdot 10^3$, $L = 50$, $V = 400 \cdot 10^3$, $F = 40 \cdot 10^3$ and $g_1 = 2.732783FL^2/(EV)$. According to [12], the optimal solutions are:

$$u^* = g_1 \begin{pmatrix} 1 \\ -1 \end{pmatrix}; \quad h^* = \begin{pmatrix} 2V - \frac{4FL^2}{Eg_1} \\ \frac{4FL^2}{Eg_1} - V \end{pmatrix},$$

that is,

$$u^* = \begin{pmatrix} 3.41597875 \cdot 10^{-3} \\ -3.41597875 \cdot 10^{-3} \end{pmatrix}; \quad h^* = \begin{pmatrix} 214.51626 \cdot 10^3 \\ 185.48374 \cdot 10^3 \end{pmatrix}.$$

Fig. 1 shows how the values of the displacement variables converge to the solution. The starting point was chosen as the equilibrium point corresponding to a uniform truss volume, and the step length parameters were $a = 6.0 \cdot 10^6$ and $b = 5.0 \cdot 10^8$.

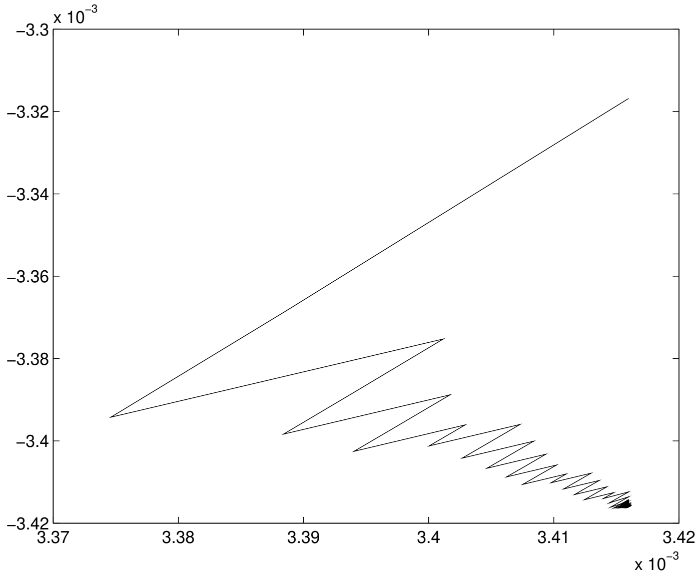

FIG. 1. *Displacement convergence*

Fig. 2 shows how the distance to the displacement solution converges to zero. The two pictures show the distance versus iteration until the Euclidean distance reaches below $0.5 \cdot 10^{-9}$ (which occurs after approximately $58,000$ iterations and less than a second of CPU time) and the same for the first 5% of these iterations. (Note that the Euclidean distance axis is logarithmic.)

To quickly reach a good design solution, it is clear from the discussions made in Section 3.5 that the choice of initial iteration for the averaging scheme (20) is crucial. Ideally, one should postpone its invokation until the iteration k_0 given in Theorem 3.3; the value of k_0 is, however, unknown. We next present some experiments conducted with the aim of providing guidelines to a proper choice of initial iteration for (20).

In addition to the arguments of Section 3.5, we note that in order for the iterates u^k to identify the proper frequencies of the extreme point solutions to the linear program (17), they must visit the corresponding subsets of $\mathcal{U}$ with the proper intervals, and this implies that some amount of zig-zagging or cycling around the solution u^* is necessary.

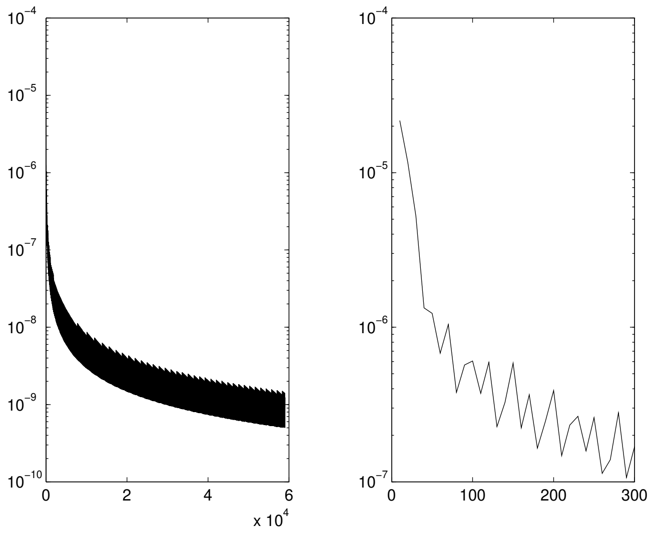

FIG. 2. *Displacement convergence*

Hence, selecting the very best values of the step length parameters a and b in order to maximize the convergence rate of $\{u^k\}$ may not be the best choices for the overall convergence of $\{u^k\}$ and $\{h(k)\}$. Experiments have shown that the convergence rate of $\{h(k)\}$ is slightly more sensitive to the choices of step length parameters than is $\{u^k\}$; a good rule-of-thumb is therefore to use a good design convergence as the target when selecting the step length parameters.

Fig. 3 depicts the design sequences $\{h(k)\}$ corresponding to initializations of the averaging scheme at iteration 0 and 250, respectively. (The optimal designs correspond to the solid lines.)

Fig. 4 is a complementary one showing (the logarithm of) the Euclidean distance from the designs $h(k)$ to the optimal design, for the two starting values 0 and 250 of the averaging schemes, respectively.

Initializing the averaging process at different iterations will not only affect the design convergence, but also the convergence of the lower bound. Fig. 5 shows the upper and lower bounds on the saddle value, obtained from (17) and (22), respectively, with lower bounds corresponding to an initialization at iteration 0 (dotted line) and 250 (dash-dotted line).

4.2. Small Sheet Example. To further illustrate the importance of properly choosing the initialization of the averaging process, we constructed a small sheet example, where $n = 18$, $m = 16$ and $r = 3$. Moreover, the lower design bound $\underline{h}$ is zero and the upper bound $\overline{h}$ a uniform constant; the total design volume V is 48% of $m \cdot \overline{h}$.

The step length parameters were chosen as $a = 6.0 \cdot 10^3$ and $b = 9.0 \cdot 10^5$.

For this example, we recorded upper bounds on the saddle value, and lower bounds for three different choices of initial iteration for the averaging scheme. Fig. 6 shows upper and lower bounds on the saddle value, for three different starting values of the averaging scheme, with the dotted line corresponding to starting iteration 0, dashed line at iteration 500 and dashdotted line at iteration 5000.

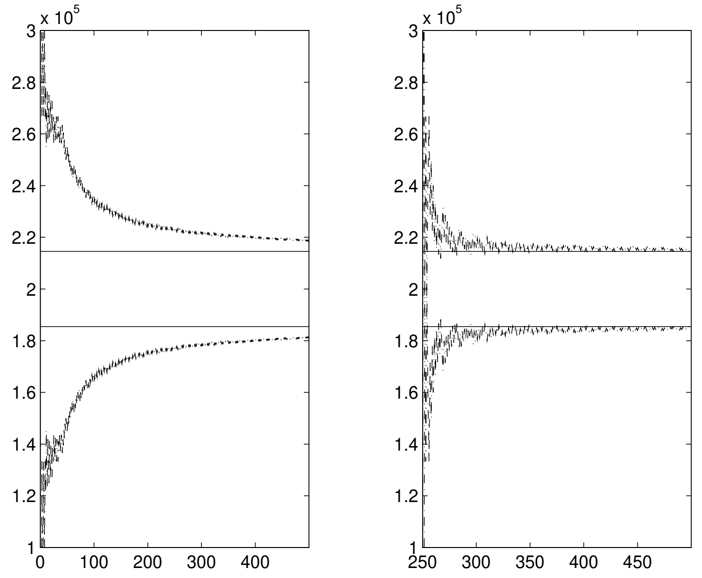

FIG. 3. *Design convergence*

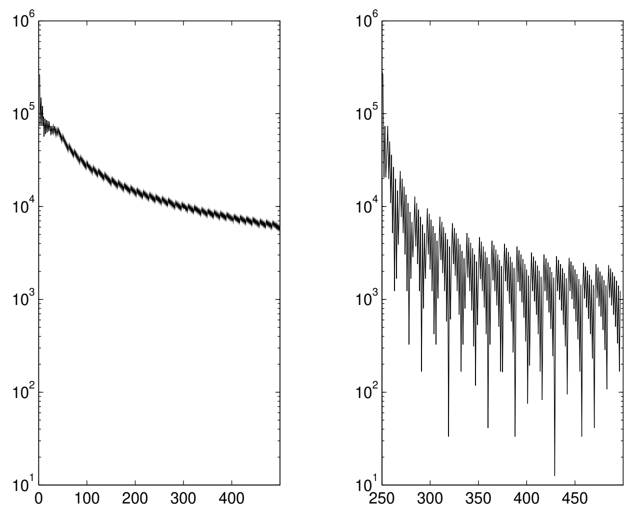

FIG. 4. *Design convergence*

The next experiment was made to try to establish an optimal initial iteration for (20) given a desired final accuracy. We recorded the total number of iterations of the subgradient procedure that was necessary to reach a relative accuracy in the saddle value as a function of the initial iteration of the averaging scheme. The result is shown in Fig. 7, where the dashed and dash-dotted lines correspond to the relative accuracies $1.0 \cdot 10^{-2}$ and $5.0 \cdot 10^{-3}$, respectively.

In Table 1 we provide the initial iteration of the averaging scheme that results in

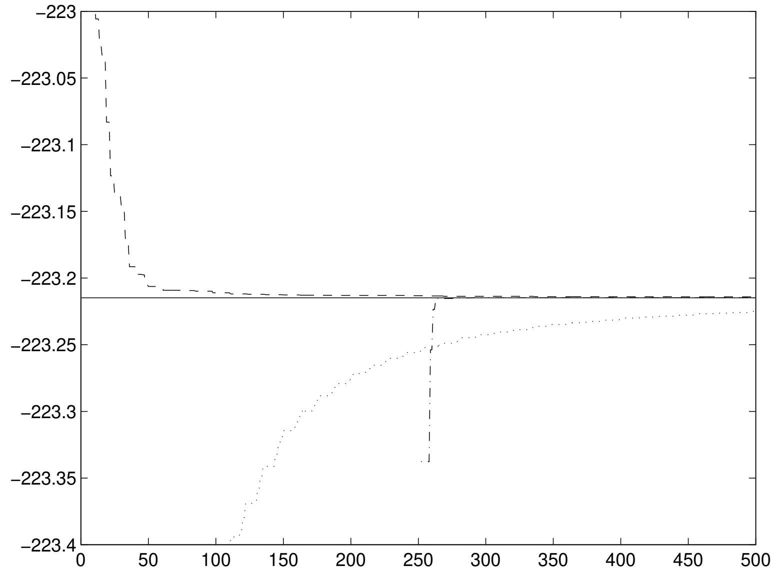

FIG. 5. *Bounds on the saddle value*

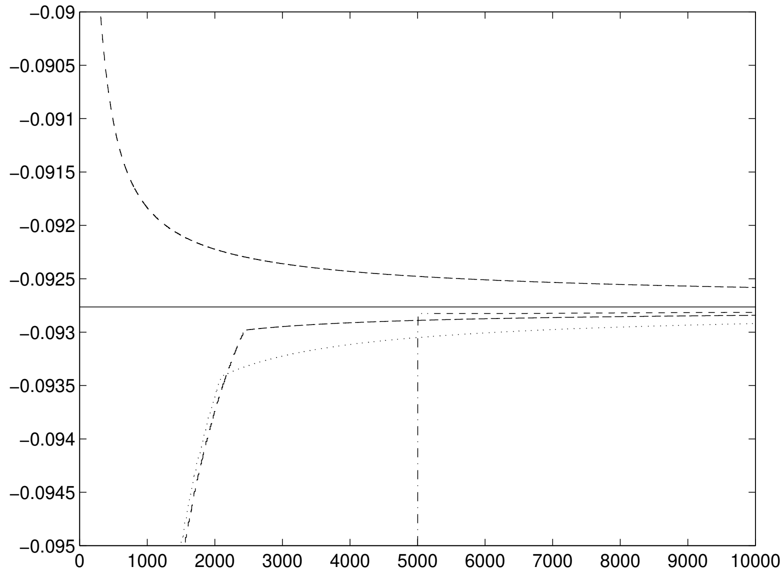

FIG. 6. *Bounds on the saddle value*

the minimal total number of iterations, as well as the latter value, for a number of relative accuracies desired. The table also gives the number of extreme point solutions h^k that was needed asymptotically for obtaining the desired accuracy (that is, the number of design solutions needed when the averaging scheme is initialized at an accurate displacement solution).

Apparently, the optimal ratio between the total number of iterations performed and the initial iteration for the averaging scheme is only slightly larger than one.

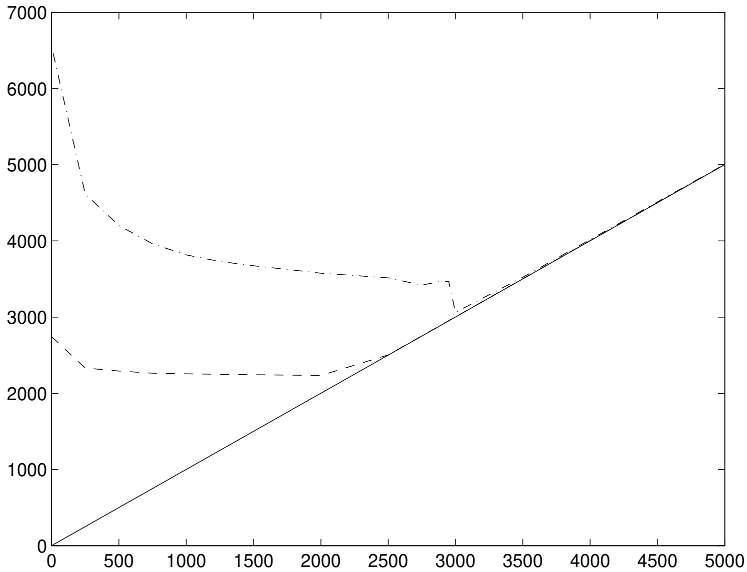

FIG. 7. *Total number of iterations needed for convergence*

TABLE 1
Optimal choice of averaging process initialization

Relative accuracy required	Optimal starting iteration/ Total number of iterations	Number of designs needed asymptotically
$5.0 \cdot 10^{-1}$	$0/2$	1
$7.5 \cdot 10^{-1}$	$0/2$	1
$1.0 \cdot 10^{-2}$	$2,000/2,234$	2
$7.5 \cdot 10^{-3}$	$2,000/2,375$	3
$5.0 \cdot 10^{-3}$	$2,750/3,420$	4
$2.5 \cdot 10^{-3}$	$9,000/9,010$	5
$1.0 \cdot 10^{-3}$	$32,500/36,935$	6
$7.5 \cdot 10^{-4}$	$60,000/60,534$	7
$5.0 \cdot 10^{-4}$	$100,000/110,004$	14

A further experiment was made with this objective. We recorded the total number of iterations needed for different relative accuracies; in Fig. 8, these figures are shown against the corresponding relative accuracy in the saddle value for the starting values 0 (dotted line), 10 (solid line), 100 (dashed line), 1000 (dashdotted line), and $10,000$ (solid line). The figure shows that for high accuracy demands, it is always profitable to start the averaging scheme later in the process; it also shows that, for these accuracy levels, the total number of iterations grows (roughly) like the relative accuracy required to the power of -1.5.

4.3. Large Sheet Example. We present a symmetric variable thickness sheet example where the thickness function is defined on a square domain. The upper design bound is constant and the lower is identically zero. The available amount of material is half the volume for the upper design bound. The optimal thickness distribution is shown in Fig. 9 where black color represent thickness values on the upper bound, and

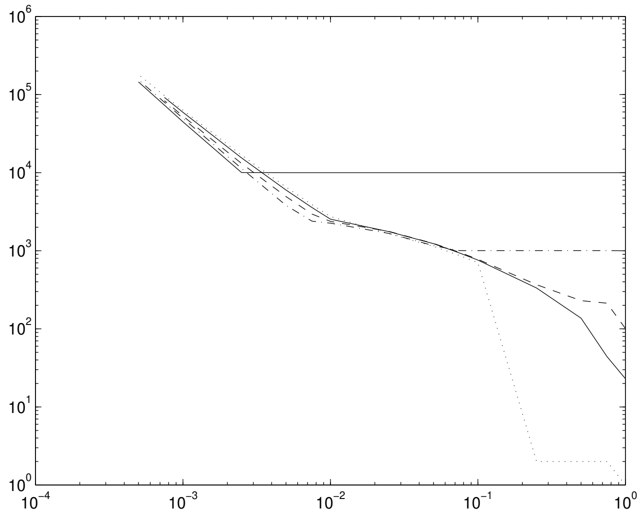

FIG. 8. *Total number of iterations needed for convergence*

white signifies a zero volume (voids in the structure).

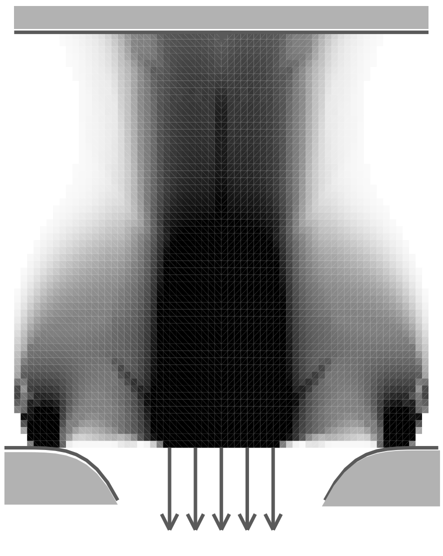

FIG. 9. *The optimal thickness distribution for a square sheet*

The obstacles have gap functions that are fourth order polynomials. The sheet was modelled with $m = 3840$, $n = 3900$, $r = 17$, and we used $a = 5.0 \cdot 10^6$ and $b = 800.0$. Based on the experiments made earlier, we chose the initial iteration for the averaging scheme to be about 90% of the total number of iterations allowed $(500 \cdot 10^3)$. At termination (which occurred after about twelve hours, at the maximal number of iterations), the upper bound on the saddle value was -0.116142 and the lower bound -0.116297.

4.4. Computational experience. We first note that the value of the step length parameter a should be chosen as small as possible, while still ensuring that the sequence of subgradients is bounded. This implies, in effect, that it needs to be larger for larger problems. The value of b should be small; in particular, for larger examples, the value of b should be smaller in order for the step lengths α_k not to converge to zero too quickly.

The algorithm is sensitive to the parameters a and b only in the sense of order of magnitude. The automatic step length selection presented in [16] is based on the mechanically reasonable rule that the first step length should be of a smaller order of magnitude than the displacements present at equilibrium for a non-optimized (uniform) structure with volume V, and the final step length should be less than a few percent of the initial one. This simple, but general, heuristic rule was implemented in a FE program, and it is reported to work for all examples tested.

As remarked earlier, it appears that the averaging scheme should be initialized when around 90% of the total number of iterations have been completed. From our experience, the number of averaged design solutions can be kept very low by using this rule-of-thumb.

REFERENCES

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1985.

[2] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, New York, NY, 2nd edition, 1993.

[3] A. Ben-Tal and M. P. Bendsøe, *A new method for optimal truss topology design*, SIAM Journal on Optimization, 3 (1993), pp. 322–358.

[4] A. Ben-Tal and A. Nemirovskii, *Potential reduction polynomial time method for truss topology design*, SIAM Journal on Optimization, 4 (1994), pp. 596–612.

[5] A. Ben-Tal and G. Roth, *A truncated log barrier algorithm for large-scale convex programming and minimax problems: Implementation and computational results.* Optimization Methods and Software, 6 (1996), pp. 283–312.

[6] J. V. Burke and J. J. Moré, *Exposing constraints.* SIAM Journal on Optimization, 4 (1994), pp. 573–595.

[7] J. Céa and K. Malanowski, *An Example of a Max-Min Problem in Partial Differential Equations*, SIAM Journal on Control, 8 (1970), pp. 305–316.

[8] J. M. Danskin, *The Theory of Max-Min*, Springer-Verlag, Berlin, 1967.

[9] A. Klarbring, J. Petersson, and M. Rönnqvist, *Truss topology optimization involving unilateral contact*, Journal of Optimization Theory and Applications, 87 (1995), pp. 1–31.

[10] Z.-Q. Luo, J.-S. Pang, and D. Ralph, *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, 1996 (forthcoming).

[11] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[12] J. Petersson and A. Klarbring, *Saddle point approach to stiffness optimization of discrete structures including unilateral contact*, Control and Cybernetics, 3 (1994), pp. 461–480.

[13] J. Petersson and M. Patriksson, *Topology optimization of sheets in contact by a subgradient method*, Report, Department of Mechanical Engineering, Division of Mechanics, Linköping Institute of Technology, S-581 83 Linköping, Sweden. To appear in International Journal of Numerical Methods in Engineering.

[14] B. T. Poljak, *A general method of solving extremum problems*, Doklady Akademii Nauk SSSR, 174 (1967), no. 1, pp. 33–36. Translated in Soviet Mathematics Doklady, 8 (1967), no. 3, pp. 593–597.

[15] B. T. Polyak, *Minimization of unsmooth functionals*, USSR Computational Mathematics and Mathematical Physics, 9 (1969), pp. 14–29.

[16] V. Rezek, *Implementation of a subgradient algorithm for topology optimization in the FE-program TRINITAS*, Report LiTH-IKP-Ex-1320, Department of Mechanical Engineering, Linköping Institute of Technology, S-581 83 Linköping, Sweden, 1996.
[17] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, New Jersey, 1970.
[18] N. Z. Shor, *Minimization Methods for Non-Differentiable Functions*, Springer-Verlag, Berlin, 1985. Translated from the Russian by K. C. Kiwiel and A. Ruszczyński.