Solving Nonconvex SDP Problems of Structural Optimization with Stability Control

MICHAL KOČVARA*

Institute of Information Theory and Automation, Pod vodárenskou věži 4
18208 Praha 4, Czech Republic

MICHAEL STINGL

Institute of Applied Mathematics, University of Erlangen, Martensstr. 3
91058 Erlangen, Germany

(Received date to be inserted)

The goal of this paper is to formulate and solve structural optimization problems with constraints on the global stability of the structure. The stability constraint is based on the linear buckling phenomenon. We formulate the problem as a nonconvex semidefinite programming problem and introduce an algorithm based on the Augmented Lagrangian method combined with the Trust-Region technique. The algorithm is implemented in a code PENNON. The paper is concluded by a series of numerical examples.

KEY WORDS: structural optimization, buckling, stability control, nonconvex semidefinite programming

1 STRUCTURAL OPTIMIZATION WITH STABILITY CONSTRAINTS

Structural optimization deals with engineering design problems with the aim of finding an optimal structure (as specified by a given cost function) that satisfies a number of given constraints. Typically, the designer is faced to the problem of finding optimal design parameters such that the resulting structure is light and stiff. The stiffness is often quantified by a so-called compliance. Further, the structure must be in equilibrium. Hence the designer can choose between the problem of minimizing the weight W subject to upper bound on the compliance C or minimizing the compliance

* Corresponding author. Email: kocvara@utia.cas.cz.

subject to upper bound on the weight. In both formulations, the equilibrium appears as a constraint. Formally, the two problems can be written as follows:

$$\begin{array}{ll} \min\ C(\rho) & \min\ W(\rho) \\ \text{subject to} & \text{subject to} \\ W(\rho) \le \widehat{W} & C(\rho) \le \widehat{C} \\ \text{equilibrium} & \text{equilibrium} \end{array}$$

here ρ stands for the design variable (like shape parameters, thickness,material properties, etc.). It is often the case, in particular when there are no additional constraints, that the above two problems are equivalent. However, when we add more constraints to the problems (like displacement or stress constraints), the equivalence does not hold anymore and we have to carefully distinguish which formulation is more appropriate.

Alternatively, we can look at the problem from the viewpoint of multicriteria optimization and search for the Pareto set of the problem $\min(W(\rho), C(\rho))$ subject to the equilibrium condition (see, e.g., [12]). But, as the Pareto points are computed by solving either of the problems with different values of $\widehat{W}$ or $\widehat{C}$, and as we are primarily interested in efficient numerical solution of the problem, we will concentrate on the single problems.

Arguably, the most serious limitation of the above models is that they do not count with possible instability of the optimal structure. Indeed, elastic instability is often the decisive point when designing a "real-world" structure, like a bridge or an aircraft. Experience showed that such structures may fail in some cases not on account of high stresses but owing to insufficient elastic stability ([25]). In this paper we consider two ways of including stability control in the problem formulation. The first one is based on the so-called *linear buckling model*. This model is based on an assumption that leads to simplification of the nonlinear structural analysis and that is naturally satisfied for many real-world structures. The second one is based on the control of the minimal eigenfrequency of the structure. Both models lead to control of the minimal eigenvalue of a generalized eigenvalue problem

$$K(\rho)u = \lambda Q(\rho)u\,.$$

In the first case (linear buckling), Q is the so-called geometry stiffness matrix that depends in a nonlinear way on the design variable ρ. In the second case (self-vibrations), Q is the mass matrix of the structure and the dependence is linear.

Now the designer has the choice of three different formulations

$\min\ C(\rho)$	$\min\ W(\rho)$	$\max\ \lambda(\rho)$
subject to	subject to	subject to
$W(\rho) \le \widehat{W}$	$C(\rho) \le \widehat{C}$	$W(\rho) \le \widehat{W}$
$\lambda(\rho) \ge \widehat{\lambda}$	$\lambda(\rho) \ge \widehat{\lambda}$	$C(\rho) \le \widehat{C}$
equilibrium	equilibrium	equilibrium

i.e., apart from adding the new constraint to the existing two formulations, he can also maximize stability, subject to upper bounds on weight and compliance.

In the rest of the paper, we will concentrate on the minimum weight formulation. Arguably, the problem of maximizing the stability can lead to unrealistic design, with a wrong choice of upper bounds $\widehat{W}$ and $\widehat{C}$. Also, we usually aim to find the lightest/stiffest structure that is just stable. The minimum compliance problem, on the other hand, can lead to serious difficulties with finding a feasible structure; see Section 2.4 and the discussion in [16].

Our goal is to solve problems with stability constraints based on the linear buckling phenomenon. As mentioned above, this approach leads to a nonconvex matrix inequality constraint, involving the geometry stiffness matrix. We will see later that, due to extremely high computational complexity, we can only solve model problems of relatively low dimension at the moment. Hence, as a viable alternative, we offer the control of self-vibrations of the optimal structure. This results in a formulation with linear matrix inequality constraints (involving the mass matrix) for which the complexity is much lower. In the last section of the paper, we present an example comparing these two approaches.

Let us mention other approaches that, in some sense, control the stability of the optimal structure. The first one is the worst-case multiple load design: given a number of independent load cases, we want to find a design that is optimal with respect to all these loads [4, 2]. It is often the case that the resulting design is also robust with respect to global stability. In many applications, the load cases (typically 2–5) are an inherent part of the problem formulation, while in other applications there is just one "natural" load case. We are aiming at the latter case. Another option is to formulate the optimal design problem as a robust optimization problem [3]. For instance, we can try to find a structure that is robust with respect to small perturbations (and still "almost optimal" with respect to the original objective function). This approach was investigated in [15].

Stability of elastic structures has been extensively studied in many monographs; see, e.g. [24, 25]. The problem of optimal structural design including stability constraints is, however, very difficult, and there are not many references in the literature (see, e.g., [6, 9, 14]). Our approach is close to the one by Ulf Ringertz ([20, 21]) used mainly for optimum design of shells. For a particular case of truss topology design, the problem formulation was thoroughly investigated in [16]; several numerical approaches for its solution were proposed in [1]. We believe that the algorithm introduced in this paper is more effective than the ones proposed in [1]. It is typical for these algorithms (and ours is not an exception) that they only find a KKT point, in case the problem is nonconvex. This should be kept in mind when reviewing the results.

In the rest of the paper, we will concentrate on a particular problem of material optimization, that is of high practical importance (see, e.g., [4]).

2 MATERIAL OPTIMIZATION WITH STABILITY CONSTRAINTS

2.1 Material optimization problem

Material optimization concerns optimal design of elastic structures, where the design variables are material properties. The material can even vanish in certain areas, thus one often speaks of topology optimization.

Let $\Omega \subset \mathbb{R}^{dim}$, $dim = 2, 3$, be a bounded domain (the elastic body) with a Lipschitz boundary Γ. By $u(x) = (u_1(x), \ldots, u_{dim}(x))$ we denote the displacement vector at point x of the body under load f, and by

$$e_{ij}(u(x)) = \frac{1}{2}\left(\frac{\partial u_i(x)}{\partial x_j} + \frac{\partial u_j(x)}{\partial x_i}\right) \quad \text{for } i, j = 1, \ldots, dim$$

the (small-)strain tensor. We assume that our system is governed by the linear Hooke's law, i.e., the stress is a linear function of the strain

$$\sigma_{ij}(x) = E_{ijk\ell}(x) e_{k\ell}(u(x)) \qquad \text{(in tensor notation)},$$

where E is the so-called elasticity tensor.

Consider the following problem

$$\inf_{\substack{\rho \geq 0 \\ \int \rho\, dx \leq 1}} \sup_{u \in U} -\frac{1}{2} \int_\Omega \rho^p \langle E e(u), e(u) \rangle \, dx + \int_{\Gamma_2} f \cdot u \, dx \,, \tag{1}$$

where $\Gamma_2 \subset \Gamma$ and $[H_0^1(\Omega)]^{dim} \subset U \subset [H^1(\Omega)]^{dim}$. For different choices of p, E and dim, we get different classes of problems:

- Free material optimization (FMO) [28, 4]

$$p = 1, \quad E = I, \quad dim = 2, 3$$

  Originally, the design variables in FMO are all elements of the elasticity tensor. It can be shown that (for single-load case) the original problem can be equivalently reformulated as the above problem, where ρ is the trace of the optimal elasticity tensor. The optimal E can be reconstructed from the optimal ρ and u.

- Variable thickness sheet (VTS) [4, 18]

$$p = 1, \quad E \ldots \text{elasticity matrix of an isotropic material}, \quad dim = 2$$

  Here ρ has the meaning of thickness of a two-dimensional isotropic elastic body.

- Solid Isotropic Material with Penalization (SIMP) [4]

$$p \geq 1, \quad E \ldots \text{elasticity matrix of an isotropic material}, \quad dim = 2, 3$$

  In this problem, nowadays widely popular among the engineers, ρ plays the role of artificial density. With increasing p, the optimal density tends to be either on the lower bound (void) or on the upper bound (full material E). Therefore, this approach is used whenever one tries to avoid intermediate densities or anisotropic materials in the optimal structure.

In the following, we will concentrate on the case

$$p = 1, \quad E \ldots \text{elasticity matrix of an isotropic material}, \quad dim = 2, 3$$

that includes VTS and a sub-case of SIMP. Note that we cannot use the FMO formulation for problems with stability constraints. When we add additional constraints to (1), the equivalence of the new problem to the original FMO formulation does not hold anymore, and the optimal E cannot be reconstructed.

2.2 Discretization

We now briefly introduce the discretization of the problem. Let m denote the number of finite elements and n the number of nodes (vertices of the elements). We approximate $\rho(x)$ by a function that is constant on each element, i.e., characterized by a vector $\rho = (\rho_1, \ldots, \rho_m)$ of its element values. Further assume that the displacement vector $u(x)$ is approximated

by a continuous function that is bi- or tri-linear (linear in each coordinate) on every element. Such a function can be written as $u(x) = \sum_{i=1}^{n} u_i \vartheta_i(x)$ where u_i is the value of u at i-th node and ϑ_i is the basis function associated with i-th node (for details, see [10]). Recall that, at each node, the displacement has dim components, so $u \in \mathbb{R}^{dim \cdot n}$.

With the basis functions $\vartheta_j, j = 1, \dots, n$, we define (3×2) and (6×3) matrices

$$\widehat{B}_j = \begin{pmatrix} \frac{\partial \vartheta_j}{\partial x_1} & 0 \\ 0 & \frac{\partial \vartheta_j}{\partial x_2} \\ \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_2} & \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_1} \end{pmatrix} \qquad \widehat{B}_j = \begin{pmatrix} \frac{\partial \vartheta_j}{\partial x_1} & 0 & 0 \\ 0 & \frac{\partial \vartheta_j}{\partial x_2} & 0 \\ 0 & 0 & \frac{\partial \vartheta_j}{\partial x_3} \\ \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_2} & \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_1} & 0 \\ 0 & \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_3} & \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_2} \\ \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_3} & 0 & \frac{1}{2}\frac{\partial \vartheta_j}{\partial x_1} \end{pmatrix}$$

for $dim = 2$ and $dim = 3$, respectively.

Now, for the i-th finite element, let $\mathcal{D}_i$ be an index set of nodes belonging to this element. Let nig denotes the number of Gauss integration points in each element. By $B_{i,k}$ we denote the $(1 \times n)$ block matrix composed of $(d \times dim)$ blocks $\widehat{B}_j$ at the j-th position, $j \in \mathcal{D}_i$, (evaluated at the k-th integration point) and zeros otherwise. Hence the full dimension of $B_{i,k}$ is $(d \times dim \cdot n)$.

The (global) stiffness matrix A is a linear combination of element stiffness matrices A_i:

$$A(\rho) = \sum_{i=1}^{m} \rho_i A_i, \qquad A_i = \sum_{k=1}^{nig} B_{i,k}^T E_{i,k} B_{i,k}\,.$$

After the discretization, problem (1) becomes

$$\inf_{\substack{\rho \geq 0 \\ \sum_{i=1}^{m} \rho_i \leq 1}} \sup_{u \in \mathbb{R}^{dim \cdot n}} \Pi(\rho, u) := -\frac{1}{2} \sum_{i=1}^{m} \rho_i \langle A_i u, u \rangle + \langle f, u \rangle\,.$$

It is well-known that the above problem can be formulated as a minimum

compliance problem

$$\begin{aligned} &\min_{u,\rho} f^T u \\ &\text{subject to} \\ &\qquad \rho_i \geq 0, \quad i = 1, \ldots, m \\ &\qquad \sum_{i=1}^{m} \rho_i \leq 1 \\ &\qquad A(\rho)u = f\,. \end{aligned} \tag{2}$$

This is based on the following variational principle (see, e.g., [3, Sect. 3.4.3]): For a given ρ, the equilibrium displacement u (satisfying $A(\rho)u = f$) is a minimizer of $\Pi(\rho, u)$; the compliance $f^T u$ is finite if and only if $\Pi(\rho, \cdot)$ is bounded above, and in this case one has $f^T u = \max_u \Pi(\rho, u)$.

Problem (2) is further equivalent to the following minimum weight problem (up to a scaling factor; compare KKT conditions)

$$\begin{aligned} &\min_{u,\rho} \sum_{i=1}^{m} \rho_i \\ &\text{subject to} \\ &\qquad \rho_i \geq 0, \quad i = 1, \ldots, m \\ &\qquad f^T u \leq 1 \\ &\qquad A(\rho)u = f\,. \end{aligned} \tag{3}$$

2.3 Stability constraint

The stability constraint, in the sense of critical buckling force, requires that all eigenvalues of the generalized eigenvalue problem

$$(A(\rho) + \lambda G(\rho, u))w = 0 \tag{4}$$

are either smaller than zero or bigger than one. Here $G(\rho, u)$ is so-called geometry stiffness matrix; its precise form is given below (see also, e.g., [27]). Condition (4) is equivalent to the following matrix inequality (see, e.g., [16]):

$$A(\rho) + G(\rho, u) \succeq 0\,. \tag{5}$$

The notation $Q \succeq 0$ (or $Q \preceq 0$) means that a symmetric matrix Q is positive (or negative) semidefinite.

We now combine the optimization problem (3) with the constraint (5) to get the minimum weight material optimization problem with stability constraint. Before writing down the full problem formulation, we rewrite, using the Schur complement theorem, the compliance constraint and the equilibrium equation in one matrix inequality constraint

$$Z(\rho) := \begin{pmatrix} 1 & f^T \\ f & A(\rho) \end{pmatrix} \succeq 0.$$

We further eliminate the variable u from the stability constraint by assuming that $A(\rho)$ is nonsingular and setting

$$\widetilde{G}(\rho) := G(\rho, A^{-1}(\rho)f).$$

The minimum weight problem with stability constraints reads

$$\begin{aligned} &\min_{\rho} \sum_{i=1}^{m} \rho_i \\ &\text{subject to} \\ &\qquad Z(\rho) \succeq 0 \\ &\qquad A(\rho) + \widetilde{G}(\rho) \succeq 0 \\ &\qquad \rho_i \geq 0, \quad i = 1, \dots, m\,. \end{aligned} \tag{6}$$

2.4 Operators A and $\widetilde{G}$

For ease of notation, let us only consider the case $dim = 3$. The geometry stiffness matrix is defined as follows:

$$G(w, \rho) = P^T \left(\sum_{i=1}^{m} G_i \right) P, \qquad G_i = \sum_{k=1}^{nig} Q_{i,k}^T \, S_{i,k} \, Q_{i,k}\,,$$

where P is the permutation matrix transforming vectors (e.g. displacements) from the ordering

$$(u_{x1}\, u_{x2} \dots u_{xn}\, u_{y1}\, u_{y2} \dots u_{yn}\, u_{z1}\, u_{z2} \dots u_{zn}\,)^T$$

into the default ordering

$$(u_{x1}\, u_{y1}\, u_{z1}\, u_{x2}\, u_{y2}\, u_{z2} \dots u_{xn}\, u_{yn}\, u_{zn})^T\,.$$

Further,

$$Q_{i,k} = \begin{pmatrix} Q_{i,k}^{(r)} & 0 & 0 \\ 0 & Q_{i,k}^{(r)} & 0 \\ 0 & 0 & Q_{i,k}^{(r)} \end{pmatrix} \quad \text{and} \quad S_{i,k} = \begin{pmatrix} S_{i,k}^{(r)} & 0 & 0 \\ 0 & S_{i,k}^{(r)} & 0 \\ 0 & 0 & S_{i,k}^{(r)} \end{pmatrix}.$$

Recall that $\mathcal{D}_i$ denotes the index set of nodes belonging to the eth finite element. The $(3 \times n)$ submatrix $Q_{i,k}^{(r)}$ is composed of columns

$$\begin{pmatrix} \frac{\partial \vartheta_j}{\partial x_1} \\ \frac{\partial \vartheta_j}{\partial x_2} \\ \frac{\partial \vartheta_j}{\partial x_3} \end{pmatrix}, \quad j \in \mathcal{D}_i$$

at j-th position, evaluated at the k-th integration point, and zeros otherwise. $S_{i,k}^{(r)}$ is the element stress tensor

$$S_{i,k}^{(r)} = \begin{pmatrix} \sigma_1 & \sigma_4 & \sigma_6 \\ \sigma_4 & \sigma_2 & \sigma_5 \\ \sigma_6 & \sigma_5 & \sigma_3 \end{pmatrix}_{i,k}$$

composed of items of the element "stress vector"

$$\sigma_{i,k} = (\sigma_1\ \sigma_2\ \sigma_3\ \sigma_4\ \sigma_5\ \sigma_6\ )_{i,k}^T = \rho_i\, E_{i,k}\, B_{i,k}\, u$$

which can be further written as

$$\sigma_{i,k} = \rho_i\, E_{i,k}\, B_{i,k}\, A(\rho)^{-1} f\,. \tag{7}$$

In the last formula we clearly see the nonlinear dependence of $\widetilde{G}$ on ρ.

Given formulas for A and $\widetilde{G}$, we can derive a simple algorithm for finding an initial feasible point for problem (6).

ALGORITHM 2.1 *Let $\rho \geq \varepsilon > 0$ be given.*

- (i) *If $A(\rho) + \widetilde{G}(\rho) \succeq 0$ and $Z(\rho) \succeq 0,$ stop and return current ρ; otherwise go to Step (ii).*
- (ii) $\rho \leftarrow 2\rho$
- (iii) *Go to Step (i).*

The termination of this algorithm is guaranteed by the following facts. We know that $A(\rho) \succeq 0$ for any $\rho > 0$ (as the elasticity matrix E is positive semidefinite). Multiplying ρ in $\widetilde{G}(\rho)$ by 2 does not change the value of $\widetilde{G}(\rho)$ (see (7)) thus, eventually, for big enough ρ, the eigenvalues of $A(\rho)$ will dominate those of $\widetilde{G}(\rho)$ and the nonconvex constraint will become feasible. Similarly, with increasing ρ, the eigenvalues of $A(\rho)$, and thus of $Z(\rho)$, will increase till, eventually, $Z(\rho) \succeq 0$.

3 THE ALGORITHM

Our method is based on the algorithm for convex SDP problems described in [17]. Here we briefly recall it and introduce its generalization to nonconvex problems. Our goal is to solve optimization problems with a linear objective function subject to nonlinear matrix inequalities as constraints:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f^T x \\ &\text{subject to} \\ &\qquad \mathcal{A}(x) \preccurlyeq 0\,. \end{aligned} \tag{8}$$

Here $f \in \mathbb{R}^n$ and $\mathcal{A} : \mathbb{R}^n \to \mathbb{S}^m$ is a (generally nonconvex) matrix operator. Clearly, our structural design problem with stability constraints (6) can be written in this form.

The algorithm is based on a choice of a smooth penalty/barrier function $\Phi_p : \mathbb{S}^m \to \mathbb{S}^m$ that satisfies a number of assumptions (see [17]) guaranteeing, in particular, that

$$\mathcal{A}(x) \preccurlyeq 0 \Longleftrightarrow \Phi_p(\mathcal{A}(x)) \preccurlyeq 0\,.$$

Thus for any $p > 0$, problem (8) has the same solution as the following "augmented" problem

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f^T x \\ &\text{subject to} \\ &\qquad \Phi_p(\mathcal{A}(x)) \preccurlyeq 0\,. \end{aligned} \tag{9}$$

The Lagrangian of (9) can be viewed as a (generalized) augmented Lagrangian of (8):

$$F(x, U, p) = f^T x + \langle U, \Phi_p\left(\mathcal{A}(x)\right)\rangle_{\mathbb{S}_m}\,; \tag{10}$$

here $U \in \mathbb{S}^m$ is a Lagrangian multiplier associated with the inequality constraint.

The algorithm below can be seen as a generalization of the Augmented Lagrangian method.

ALGORITHM 3.1 *Let x^1 and U^1 be given. Let $p^1 > 0$. For $k = 1, 2, \ldots$ repeat until a stopping criterion is reached:*

(i) *Find x^{k+1} satisfying $\|\frac{\partial}{\partial x} F(x, U^k, p^k)\| \leq \alpha^k$ for given $\alpha^k > 0$*

(ii) $U^{k+1} = D_{\mathcal{A}} \Phi_p(\mathcal{A}(x); U^k)$

(iii) $p^{k+1} < p^k$.

Imposing standard assumptions on problem (8), it can be proved that any cluster point of the sequence $\{(x_k, U_k)\}_{k>0}$ generated by Algorithm 3.1 is a KKT point of problem (8). The proof is based on extensions of results by Polyak [19] and Breitfeld and Shanno [8]; for the full version we refer to [22].

In the next section, we will take a closer look at the approximate unconstrained minimization in Step (i), multiplier update in Step (ii) and stopping criteria of the algorithm.

3.1 Choice of Φ_p

The penalty function Φ_p of our choice is defined as follows:

$$\Phi_p(\mathcal{A}(x)) = -p^2(\mathcal{A}(x) - pI)^{-1} - pI\,. \tag{11}$$

The advantage of this choice is that it gives closed formulas for the first and second derivatives of Φ_p. Defining

$$\mathcal{Z}(x) = -(\mathcal{A}(x) - pI)^{-1} \tag{12}$$

we have (see [17]):

$$\frac{\partial}{\partial x_i}\Phi_p(\mathcal{A}(x)) = p^2\mathcal{Z}(x)\frac{\partial \mathcal{A}(x)}{\partial x_i}\mathcal{Z}(x) \tag{13}$$

$$\frac{\partial^2}{\partial x_i \partial x_j}\Phi_p(\mathcal{A}(x)) = p^2\mathcal{Z}(x)\left(\frac{\partial \mathcal{A}(x)}{\partial x_i}\mathcal{Z}(x)\frac{\partial \mathcal{A}(x)}{\partial x_j} + \frac{\partial^2 \mathcal{A}(x)}{\partial x_i \partial x_j} + \frac{\partial \mathcal{A}(x)}{\partial x_j}\mathcal{Z}(x)\frac{\partial \mathcal{A}(x)}{\partial x_i}\right)\mathcal{Z}(x)\,. \tag{14}$$

3.2 Minimizing the augmented Lagrangian

We have implemented two algorithms for the (approximate) unconstrained minimization of the (possibly nonconvex) augmented Lagrangian in Step (i) of Algorithm 3.1. The first one is the standard Trust-Region method, in particular, Algorithm 7.3.4 from [11]. The second one is a variant of the Levenberg-Marquardt method. Here we use the following algorithm to calculate the search direction d.

One step of the minimization method in Step (i) is defined as follows:

1. Given a current iterate (x, U, p), compute the gradient g and Hessian H of F at x.
2. Try to factorize H by Cholesky decomposition. If H is factorizable, set $\widehat{H} = H$ and go to Step 4.
3. Compute $\beta \in [-\lambda_{\min}, -2\lambda_{\min}]$, where $\lambda_{\min}$ is the minimal eigenvalue of H and set
$$\widehat{H} = H + \beta I.$$
4. Compute the search direction
$$d = -\widehat{H}^{-1} g.$$
5. Perform line-search in direction d. Denote the step-length by s.
6. Set
$$x_{\rm new} = x + sd.$$

The step-length s in direction d is calculated by a gradient free line-search that tries to satisfy the Armijo condition.

For a convex F, this is just a Newton step with line-search. If, in the nonconvex case, the Cholesky factorization in Step 2 fails, the value of β in Step 3 is calculated in the following way: We start with $\beta_0 > 0$, set $\beta = \beta_0$ and try to factorize $H + \beta I$ by the Cholesky method. If the factorization fails due to a negative pivot, the matrix $H + \beta I$ is considered indefinite and we multiply β by 2. We repeat this process until the Cholesky factorization exists. If the Cholesky factorization is successful already for $\beta = \beta_0$, we divide β by 2 until Cholesky factorization fails. We denote the corresponding β by $\beta_{\min}$ and set $\beta = 2\beta_{\min}$. In both cases we end up with $\beta \in [-\lambda_{\min}, -2\lambda_{\min}]$.

Although this technique is just heuristics, it proved to work well on a large set of nonconvex NLP and nonconvex SDP problems. It is not possible, though, to prove convergence of Algorithm 3.1 combined with this simple approach without making additional (restrictive) assumptions. Therefore we also implemented the trust-region algorithm for which, in order to

prove convergence of Algorithm 3.1, we needed just standard assumptions. When compared to the above heuristic approach on a set of NLP problems, the trust-region variant turned out to be more robust but often slower.

3.3 Multiplier and penalty update, stopping criteria

For the penalty function Φ_p from (11), the formula for update of the matrix multiplier U in Step (ii) of Algorithm 3.1 reduces to

$$U^{k+1} = (p^k)^2 \mathcal{Z}(x) U^k \mathcal{Z}(x) \tag{15}$$

with $\mathcal{Z}$ defined as in (12).

Numerical tests indicate that big changes in the multipliers should be avoided for the following reasons. Big change of U means big change of the augmented Lagrangian that may lead to a large number of Newton steps in the subsequent iteration. It may also happen that already after few initial steps, the multipliers become ill-conditioned and the algorithm suffers from numerical difficulties. To overcome these, we do the following:

1. Calculate U^{k+1} using the update formula in Algorithm 3.1.
2. Choose a positive $\mu_A \leq 1$, typically 0.5.
3. Compute $\lambda_A = \min\left(\mu_A, \mu_A \frac{\|U^k\|_F}{\|U^{k+1}-U^k\|_F}\right)$.
4. Update the current multiplier by

$$U^{new} = U^k + \lambda_A (U^{k+1} - U^k).$$

Given an initial iterate x^1, the initial penalty parameter p^1 is chosen large enough to satisfy the inequality

$$p^1 I - \mathcal{A}(x^1) \succ 0.$$

Let $\lambda_{\max}(\mathcal{A}(x^k)) \in \left(0, p^k\right)$ denote the maximal eigenvalue of $\mathcal{A}(x^k)$, $\pi < 1$ be a constant factor, depending on the initial penalty parameter p^1 (typically chosen between 0.3 and 0.6) and x_{feas} be a feasible point calculated by Algorithm 2.1. Using this notation, our strategy for the penalty parameter update can be described as follows:

1. If $p < p_{eps}$, set $\gamma = 1$ and go to 6.
2. Calculate $\lambda_{\max}(\mathcal{A}(x^k))$.
3. If $\pi p^k > \lambda_{\max}(\mathcal{A}(x^k))$, set $\gamma = \pi$ and go to 6.
4. If $l < 3$, set $\gamma = \left(\lambda_{\max}(\mathcal{A}(x^k)) + p_k\right)/2$, set $l := l + 1$ and go to 6.

5. Let $\gamma = \pi$, find $\lambda \in (0,1)$ such, that

$$\lambda_{\max}\left(\mathcal{A}(\lambda x^{k+1} + (1-\lambda)x_{\rm feas})\right) < \pi p_k$$

and set $x^{k+1} = \lambda x^{k+1} + (1-\lambda)x_{\rm feas}$.

6. Update current penalty parameter by $p^{k+1} = \gamma p^k$.

The redefinition of x^{k+1} guarantees that the values of the augmented Lagrangian in the next iteration remain finite. The parameter p_{eps} is typically chosen as 10^{-6}. In case we detect problems with convergence of the overall algorithm, p_{eps} is decreased and the penalty parameter is updated again, until the new lower bound is reached.

The unconstrained minimization is stopped when $\|\frac{\partial}{\partial x}F(x,U,p)\| \leq \alpha$, where $\alpha = 0.01$ is a good choice in most cases. Also here, α is decreased if we encounter problems with accuracy.

Algorithm 3.1 is stopped if both of the following inequalities hold:

$$\frac{|f(x^k) - F(x^k,U^k,p)|}{1+|f(x^k)|} < \epsilon\,, \qquad \frac{|f(x^k) - f(x^{k-1})|}{1+|f(x^k)|} < \epsilon\,,$$

where ϵ is typically 10^{-7}.

4 SOLVING STABILITY PROBLEMS BY PENNON

Algorithm 3.1 with the details discussed above was implemented in the computer program PENNON[1]. In this section we will point out the main difficulties when solving structural design problems with stability constraints by PENNON. We will also present results of model numerical examples.

4.1 Complexity

This turns out to be the critical issue of our approach. Recall that the main result of the structural optimization approach introduced in Section 1 is a *picture* that shows the distribution of material and/or its properties. Hence the finite elements serve here as pixels in a digital picture. Obviously, to get a reasonable resolution in the resulting picture, we have to work with a sufficiently fine finite element mesh. Praxis shows that in two-dimensional space, one should use minimum of 5000 elements, though about 15000 is "standard". In three-dimensional models, the number of elements may

[1] `http://www2.am.uni-erlangen.de/~kocvara/pennon/`

easily reach 100000. From now on, let us concentrate on the 2D case—we will see that even this leads to too large problems.

Let us estimate the memory and number of arithmetic operations needed to assemble one Hessian of the Lagrangian; here we refer to Section 3.1. It is easy to see that memory needed to store one Hessian is *quadratically proportional* to the number of finite elements m. Further, to solve a problem with 500 elements, we need about 64 MBytes.

The complexity of Hessian assembling is more critical. It can be shown that the complexity is of order

$$O(nig^2 \cdot dim^2 \cdot m^3)$$

where, again, nig is the number of points of numerical integration in one finite element, dim is the space dimension and m the number of finite elements. It is the cubic dependence on m that is disastrous for practical problems. The high complexity is, in particular, due to many matrix-matrix multiplication involved in formula (14). And, although we carefully implemented all critical parts of the program using fast linear algebra libraries, the resulting code cannot be used for problems of practical size. For instance, to solve a 2D problem with just 400 elements we need, in average, to assemble about 100 Hessians; for that the code spends 8 h 45 min on a PC with 2.4GHz Pentium 4. Using the above complexity estimate, we can guess that to solve a problem with 1000 elements (still far below the "practical minimum") we would need about 130 hours of CPU time.

4.2 Examples

Material optimization with vibration control Let us start with the simpler problem of material optimization with constraint on minimum free vibrations. In this case, the nonlinear stability constraint in (6) is replaced by a linear one

$$A(\rho) + \widehat{\lambda} M(\rho) \succeq 0\,,$$

where $M(\rho) = \sum_{i=1}^{m} \rho_i M_i$ is a mass matrix, dependent linearly on ρ, and $\widehat{\lambda}$ is the minimum allowed eigenfrequency. The complexity of the linear SDP is much lower than of the nonlinear one. Figure 1 presents result of a model example. Consider a plate (depicted on the left-hand figure) fixed on the left-hand side and subjected to a "horizontal" load concentrated on a small part of the right-hand side. The middle figure shows a result of the minimum weight problem (3) (with no stability/vibration constraint) for a zero-Poisson-ration material. The optimal structure only consists of horizontal fibers and is, as such, extremely unstable to other than the given

load. The right-hand figure presents the results of problem (6) for the same material; the structure is obviously much more stable.

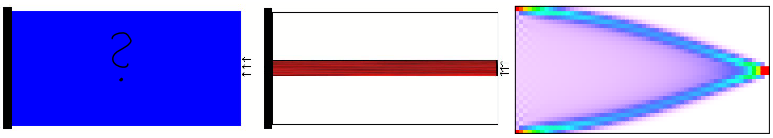

FIGURE 1: SHAPE problem with free vibration control

Because this problem leads to linear SDP, we can solve it by available SDP codes, in order to compare the efficiency of PENNON with existing software. Namely, we used the codes CSDP [7], DSDP [5], SeDuMi [23], SDPA [13], and SDPT3 [26]. We generated a series of problems based on the above example with increasing level of discretization; the problems were encoded using the SDPA input format[2]. Table 1 summarizes the dimensions of these problems, while Table 2 presents the results—CPU time in seconds. The reader can see that PENNON is competitive to the other solvers.

The CPU times in Table 2 were obtained on a 3.2 GHz Pentium 4 with 4GB RDRAM under Linux[3]. The core routines of all codes are written in C/C++ or Fortran. PENNON, SDPA, CSDP and DSDP are linked with ATLAS-BLAS numerical linear algebra library. The stopping criteria of all codes are set to reach about 7 digits of accuracy in the objective function.

TABLE 1: Selected SHAPE problems.

problem	no. of var.	size of matrix
shmup-3	420	1801+840
shmup-4	800	3361+1600
shmup-5	1800	7441+3660

Material optimization with stability control Let us now try to solve the nonlinear SDP problem (6). Due to the limitations given by the complexity

[2] The problems in SDPA format are available on
`www2.am.uni-erlangen.de/~kocvara/pennon`

[3] The table is overtaken from `http://plato.la.asu.edu/bench.html` with a kind permission of the author.

TABLE 2: Results for SHAPE problems. “m” means memory exceeded.

problem	PENNON	SDPT3	SDPA	CSDP	DSDP	SeDuMi
shmup-3	301	307	419	991	7880	10280
shmup-4	1655	1712	1988	2815	2137	109670
shmup-5	10994	15917	m	46185	fail	m

of the algorithm, we can only solve problems with hundreds of finite elements. Figure 2 shows a result for the problem introduced in the previous example, this time with the (nonvonvex) stability constraint; the problem was discretized by 420 elements.

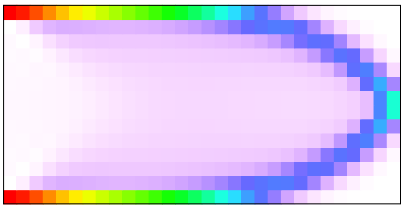

FIGURE 2: SHAPE problem with stability control

Now, while the linear problem of the same size needed 301 seconds (see Table 2), the nonlinear problem required 22 700 seconds, i.e., more than 6 hours on the same computer. Both problems needed about the same number of iterations, though, so the additional time was solely spent in the Hessian assembling routine. To put things into perspective, let us note that the unconstrained problem (3) can be reformulated as a convex nonlinear program ([28]); to solve this NLP (for the same discretization), we needed less than one second on the same computer!

As mentioned in Section 3, for generally nonconvex problems we can only guarantee convergence of Algorithm 3.1 to a KKT point, and this may be the case in this problem. However, the resulting figure looks “reasonable” and we obtained this result for several initial point.

5 CONCLUSIONS

The good news is that Algorithm 3.1, as implemented in PENNON, can reliably solve nonconvex SDP problems. The behavior of the code does not differ too much from the linear version and allows us to reach accurate solutions within 60–100 internal iterations. This good behavior was

confirmed by solving truss design problems with stability constraints and problems with bilinear matrix inequalities coming from nonlinear systems control.

On the other hand, notwithstanding the nice convergence behavior, the complexity and memory requirements of a second-order method are just too high to solve nonconvex SDPs of the kind and size needed in our application. We believe that the way out can be the use of a first order method for the unconstrained minimization in Step (i) of Algorithm 3.1. This is the subject of our future research.

Acknowledgements

The authors would like to thank Hans Mittelmann for his help when testing the code. This research was supported by BMBF project 03ZOM3ER. The first author was partly supported by the Grant Agency of the Academy of Sciences of the Czech Republic under grant No. A 107 5005.

References

[1] A. Ben-Tal, F. Jarre, M. Kočvara, A. Nemirovski, and J. Zowe. Optimal design of trusses under a nonconvex global buckling constraint. *Optimization and Engineering*, 1:189–213, 2000.

[2] A. Ben-Tal, M. Kočvara, A. Nemirovski, and J. Zowe. Free material design via semidefinite programming. The multi-load case with contact conditions. *SIAM J. Optimization*, 9:813–832, 1997.

[3] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization*. MPS-SIAM Series on Optimization. SIAM Philadelphia, 2001.

[4] M. Bendsøe and O. Sigmund. *Topology Optimization. Theory, Methods and Applications*. Springer-Verlag, Heidelberg, 2002.

[5] S. J. Benson and Y. Ye. DSDP4 users manual. Report ANL/MCS-TM-248, Argonne National Laboratory, Argonne, 2002. Available at `http://www-unix.mcs.anl.gov/~benson/`.

[6] T. Birker. *New Developments in Structural Optimization Using Optimality Criteria*. Series 18, No. 199, VDI-Verlag, Düsseldorf, Germany, 1994.

[7] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11:613–623, 1999. Available at `http://www.nmt.edu/~borchers/`.

[8] M. Breitfeld and D. Shanno. A globally convergent penalty-barrier algorithm for nonlinear programming and its computational performance. Technical report, Rutcor Research Report, Rutgers University, New Jersey., 1994.

[9] T. Buhl, C. B. W. Pedersen, and O. Sigmund. Stiffness design of geometrically nonlinear structures using topology optimization. *Struct. Multidisc. Optimization*, 19:93–104, 2000.

[10] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. North-Holland, Amsterdam, New York, Oxford, 1978.

[11] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, 2000.

[12] M. Ehrgott. *Multicriteria Optimization*. Springer, Berlin, 2000.

[13] K. Fujisawa, M. Kojima, K. Nakata, and M. Yamashita. SDPA (SemiDefinite Programming Algorithm) User's Manual — Version 6.00. Available at `http://www.is.titech.ac.jp/~yamashi9/sdpa/index.html`, Department of Mathematical and Computing Science, Tokyo University of Technology, 2002.

[14] D. Gerdes. *Strukturoptimierung unter Anwendung der Optimalitätskriterien auf diskretisierte Tragwerke bei besonderer Berücksichtigung der Stabilität (in German)*. Series 18, No. 171, VDI-Verlag, Düsseldorf, Germany, 1994.

[15] M. Kočvara, J. Zowe, and A. Nemirovski. Cascading—an approach to robust material optimization. *Computers & Structures*, 76:431–442, 2000.

[16] M. Kočvara. On the modelling and solving of the truss design problem with global stability constraints. *Struct. Multidisc. Optimization*, 23(3):189–203, 2002.

[17] M. Kočvara and M. Stingl. PENNON—a code for convex nonlinear and semidefinite programming. *Optimization Methods and Software*, 18(3):317–333, 2003.

[18] J. Petersson. On stiffness maximization of variable thickness sheet with unilateral contact. *Quart. Appl. Math.*, 54:541–550, 1996.

[19] R. Polyak. Modified barrier functions: Theory and methods. *Mathematical Programming*, 54:177–222, 1992.

[20] U. T. Ringertz. An algorithm for optimization of non-linear shell structures. *Internat. J. Numer. Methods Engrg.*, 38:299–314, 1995.

[21] U. T. Ringertz. Eigenvalues in optimum structural design. In A. Conn and F. Santosa, editors, *Large Scale Optimization*. Institute for Mathematics and its Applications, Springer, 1995.

[22] M. Stingl. *On the Solution of Nonlinear Semidefinite Programs by Augmented Lagrangian Methods*. PhD thesis, Institute of Applied Mathematics II, Friedrich-Alexander University of Erlangen-Nuremberg, in preparation.

[23] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11 & 12:625–653, 1999. Available at `http://fewcal.kub.nl/sturm/`.

[24] J. M. T. Thompson and G. W. Hunt. *A General Theory of Elastic Stability*. J. Wiley & Sons, London, 1973.

[25] S. P. Timoshenko and J. M. Gere. *Theory of Elastic Stability*. McGraw-Hill, New York, 1961.

[26] R.H. Tütüncü, K.C. Toh, and M.J. Todd. SDPT3 — A MATLAB software package for semidefinite-quadratic-linear programming, Version 3.0. Available at `http://www.orie.cornell.edu/~miketodd/todd.html`, School of Operations Research and Industrial Engineering, Cornell University, 2001.

[27] O. C. Zienkiewicz and R. L. Taylor. *Finite Element Method*. Butterworth-Heinemann, Oxford, 5th edition, 2000.

[28] J. Zowe, M. Kočvara, and M. Bendsøe. Free material optimization via mathematical programming. *Mathematical Programming, Series B*, 79:445–466, 1997.