FP–6 STREP 30717 PLATO–N (Aeronautics and Space)

PLATO–N

A PLAtform for Topology Optimisation incorporating Novel, Large-Scale, Free-Material Optimisation and Mixed Integer Programming Methods

Sequential Convex Programming for Free Material Optimization with Displacement and Stress Constraints

PLATO–N Public Report PU–R–2–2008

March, 2008

Authors:

Sonja Ertel

Klaus Schittkowski

Christian Zillober

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)

Dissemination level: Public

Sequential Convex Programming for Free Material Optimization with Displacement and Stress Constraints

Deliverable D14

Sonja Ertel

Department of Computer Science, University of Bayreuth, D-95440 Bayreuth
sonja.ertel@uni-bayreuth.de

Klaus Schittkowski

Department of Computer Science, University of Bayreuth, D-95440 Bayreuth
klaus.schittkowski@uni-bayreuth.de

Christian Zillober

Department of Mathematics, University of Würzburg, D-97074 Würzburg
christian.zillober@uni-wuerzburg.de

Date:
March, 2008

Task:
3.4

Abstract: We consider free material optimization (FMO) problems, which are defined in form of semidefinite programs. The objective is to compute the stiffest structure subject to given loads. Constraints are bounds for compliances, displacements or stresses. Optimization variables are the entries of element material matrices in two or three dimensions. FMO problems are solved by a sequential convex programming algorithm, which is frequently applied in mechanical structural design optimization. The idea is to construct convex and separable subproblems, which are either solved by an interior point method or by an external solver, which is able to treat positive semidefinite variables. In the case of isotropic materials, additional linear constraints guarantee positive definite material matrices. For anisotropic materials, we propose to optimize over Cholesky factors of the elasticity matrices. Some preliminary numerical results are presented for these two situations. Finally we discuss the integration of two semidefinite subproblem solvers called PENNON and CONERML and the possibility to add stress and displacement constraints.

Keywords: nonlinear programming; sequential convex programming; method of moving asymptotes; free material optimization; semidefinite programming; displacement constraints; stress constraints; SCPIP; PENNON; CONERML

1 Introduction

We consider a constrained optimization method that is widely used in mechanical engineering and that is known under the name sequential convex programming (SCP). The algorithm computes an optimal solution by solving a sequence of convex and separable subproblems, where an augmented Lagrangian merit function is used to guarantee convergence. Originally SCP methods were developed in structural mechanical optimization and are particularly applied to solve topology optimization problems.

The algorithm is based on the observation that in some special cases typical structural constraints become linear in the inverse variables. Although this special situation is rarely observed in practice, a suitable substitution by inverse variables depending on the sign of the corresponding partial derivatives and subsequent linearization is expected to linearize model functions somehow.

The approach goes back to the method of moving asymptotes (MMA) introduced by Svanberg [29]. The algorithm does not apply a line search but is very efficient for solving mechanical engineering problems, if a proper starting point is available. Some comparative numerical tests of SCP, SQP and some other nonlinear programming codes are available for test problems from mechanical structural optimization, see Schittkowski, Zillober, and Zotemantel [27] and Schittkowski, Zillober, Moritzen [26].

The computer code under investigation is the SCP routine SCPIP of Zillober [30, 33, 34, 31]. Strictly convex and fully separable subproblems are solved by an interior point method combined with an active set strategy. General sparsity of the Jacobian matrix of the constraints is taken into account.

Topology optimization is one of the main domains of applications where SCP methods are frequently used. The idea is to distribute mass within a given volume, so that the global compliance of the structure is minimized. Since the number of the finite elements is often very large depending on the desired discretization accuracy, nonlinear programs must be solved iteratively with up to 10^5 to 10^6 or even more variables. In addition, more realistic structures require constraints for each single element leading to a large number of nonlinear inequality constraints of the same order of magnitude.

Free material optimization (FMO) is introduced in a couple of papers by Bendsøe [4], Bendsøe and Sigmund [5] and Zowe, Kočvara, and Bendsøe [35]. FMO tries to find the *best* mechanical structure with respect to one or more given loads in the sense that a design criterion, e.g., minimal weight or maximal stiffness, is obtained. The material properties as well as material distributions in the available space are varied. In this context FMO belongs to topology optimization. As shown by Kočvara and Stingl [17], the FMO problem can be formulated for a given set of loads by a nonlinear semidefinite programming (NSDP) problem.

The standard FMO formulation is to minimize the maximum compliance $f_j^T K^{-1}(E) f_j$ for a load f_j, $j = 1, \ldots, l$, where l is the number of load cases and $K(E)$ the global stiffness matrix. A more detailed description is found in Hörnlein, Kočvara and Werner [11] and Kočvara and Zowe [20]. As a measure of the material stiffness in the coordinate directions, we use the traces of the material matrices E_i, which are the design or optimization variables of our approach. The elasticity matrices $E_i, i = 1, \ldots, m$ have to be physically reasonable, i.e., symmetric and positive semidefinite. Thus there are additional volume constrains, see Ben-Tal, Kočvara, Nemirovski, and Zowe [2] and further constraints to prevent singularities.

The FMO problem leads to a special formulation of the convex and separable subproblem of the SCP method. Besides of certain ways to enforce the positive semi-definiteness (PSD) of the anisotropic material matrices, e.g., by optimizing Cholesky factors, a new alternative is to handle

PSD constraints directly by applying special solvers. In our situation the PSD code PENNON of Kočvara and Stingl [14] and the convex optimization solver CONERML of Beck, Ben-Tal, and Tetruashvili [1] are applied.

The FMO problem is extended by displacement and stress constraints. Displacement constraints are motivated by the technological background and provide a means to keep the deformation of the structure within certain boundaries. They are considered at special nodes unlike the compliance, and act as additional constraints. Moreover, we analyze the so-called von Mises stress constraints. Stresses acting in different directions are mapped into one equivalent stress vector which is then tested for stability.

In the subsequent section, we present the SCP method in more detail since it provides the basis for all numerical investigations. The convex approximations are presented which generate convex and separable subproblems.

An interior point method for solving these problems is presented in Section 3, which is implemented as the standard solver in the corresponding code SCPIP of Zillober [34].

The standard formulation of FMO is outlined in Section 4. Derivatives of compliance constraints are given which are needed to formulate the convex and separable subproblem discussed in Section 3 in general terms. It is shown how the PSD conditions are passed to the subproblem.

Some solution methods for solving the standard FMO problem are introduced in Section 5. In case of isotropic materials, the PSD constraints are easily replaced by a set of linear constraints. For the general anisotropic case, a Cholesky decomposition of the material matrices allows a *black box* application of the SCP code SCPIP. However, there are disadvantages and it might be more reasonable to apply directly either a PSD solver like PENNON or a convex programming solver like CONERML to solve the PSD subproblems. A particular advantage is that second derivatives are available and that the Hessian of the Lagrangian is diagonal.

Possible extensions towards displacement and stress constraints are discussed in Section 6. It is shown how these constraints are included in the available FMO framework and how derivatives can be computed.

The software architecture is briefly outlined in Section 7, more details are presented in an appendix. The section contains also some numerical results obtained for the standard FMO formulation and two simple academic test cases. So far, two-dimensional FMO problems with up to 30,000 variables can be solved.

2 Sequential Convex Programming

We consider nonlinear optimization problems of the kind

$$\text{(P0)} \left\{ \begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ \text{s.t.} & c(x) \le 0 \end{array} \right. \tag{1}$$

where the scalar objective function $f(x)$ and the scalar constraint functions $c_j(x)$, $j = 1, \ldots, m$, are twice continuously differentiable, $c(x) := (c_1(x), \ldots, c_m(x))^T$. Without loss of generality we omit equality constraints to facilitate the notation. We do not need them for free material optimization (FMO) discussed subsequently.

To understand the discussions and notations of the subsequent sections, we introduce the Lagrangian function

$$L(x, y) := f(x) + y^T c(x) \tag{2}$$

where $x \in \mathbb{R}^n$ is the primal variable and $y = (y_1, \ldots, y_m)^T \in \mathbb{R}^m$ the dual variable, also called the vector of the Lagrange multipliers. Under some additional assumptions, necessary and sufficient optimality conditions, the so-called KKT conditions, can be derived. A local stationary point $x^\star \in \mathbb{R}^n$ is characterized by finding a corresponding multiplier vector $y^\star \in \mathbb{R}^m$ satisfying

$$\begin{aligned} y^\star &\geq 0 \\ c(x^\star) &\leq 0 \\ \bigtriangledown_x L(x^\star, y^\star) &= 0 \\ c(x^\star)^T y^\star &= 0 \end{aligned} \tag{3}$$

For modeling more realistic situations we introduce upper and lower bounds for the variables

$$\underline{x} \leq x \leq \overline{x} \ .$$

Moreover, linear constraints defined by a matrix $A \in \mathbb{R}^{m_l \times n}$ and a vector $b \in \mathbb{R}^{m_l}$ are handled separately and lead to the following extended nonlinear program which is used subsequently,

$$(\mathrm{P}) \left\{ \begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ \text{s.t.} & c(x) \leq 0 \\ & Ax \leq b \\ & \underline{x} \leq x \leq \overline{x} \end{array} \right. \tag{4}$$

First, we consider the so-called method of moving asymptotes (MMA) for solving the nonlinear program (P), see Fleury [8] and Svanberg [29]. Although the algorithm is not stabilized, e.g., by a line search, and convergence of the algorithm cannot be proved in a rigorous mathematical sense, the algorithm is quite successful in practice. The basic idea is to generate a sequence of convex and separable subproblems which can be solved by any available algorithm taking the special structure into account. The procedure is illustrated in Figure 1.

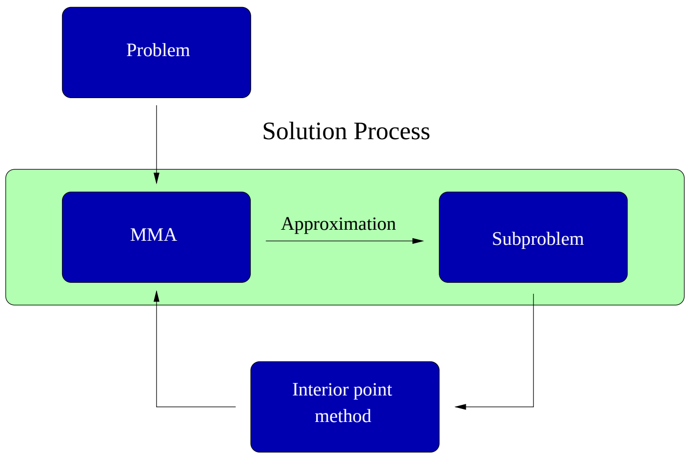

Figure 1: Procedure of MMA-Algorithm

The idea behind MMA is the segmentation of the n-dimensional problem space into n one-dimensional spaces. One of the most important features is the introduction of two flexible asymptotes for each optimization variable x_i which restrains the feasible region and prevents the algorithm performing too long steps.

Objective function and nonlinear inequality constraints are linearized with respect to reciprocal bounded variables $(U_i^{(k)} - x_i)^{-1}$ and $(x_i - L_i^{(k)})^{-1}$. The resulting approximation of the objective function at an iterate k is

$$\begin{aligned} f^{(k)}(x) \;:=\; & f\left(x^{(k)}\right) \\ + & \sum_{I_+^{(k)}} \left[\frac{\partial f\left(x^{(k)}\right)}{\partial x_i} \left(U_i^{(k)} - x_i^{(k)}\right)^2 \left(\frac{1}{U_i^{(k)} - x_i} - \frac{1}{U_i^{(k)} - x_i^{(k)}} \right) \right] \\ - & \sum_{I_-^{(k)}} \left[\frac{\partial f\left(x^{(k)}\right)}{\partial x_i} \left(x_i^{(k)} - L_i^{(k)}\right)^2 \left(\frac{1}{x_i - L_i^{(k)}} - \frac{1}{x_i^{(k)} - L_i^{(k)}} \right) \right] \\ + & \sum_{I_+^{(k)}} \tau_i^{(k)} \frac{\left(x_i - x_i^{(k)}\right)^2}{U_i^{(k)} - x_i} + \sum_{I_-^{(k)}} \tau_i^{(k)} \frac{\left(x_i - x_i^{(k)}\right)^2}{x_i - L_i^{(k)}} \end{aligned} \tag{5}$$

with $I_+^{(k)} := \left\{ i \,\middle|\, \frac{\partial f\left(x^{(k)}\right)}{\partial x_i} \geq 0 \right\}, \quad I_-^{(k)} := \left\{ i \,\middle|\, \frac{\partial f\left(x^{(k)}\right)}{\partial x_i} < 0 \right\}, \quad L_i^{(k)} < x_i < U_i^{(k)}.$

A positive parameter τ is introduced to guarantee strict convexity, see Zillober [32]. The lower and upper bounds $L_i^{(k)}$ and $U_i^{(k)}$ serve as lower and upper asymptotes which cannot be exceeded by the algorithm.

The nonlinear inequality constraints $c_j(x)$ for $j = 1, \ldots, m$ are approximated by

$$\begin{aligned} c_j^{(k)}(x) \;:=\; & c_j\left(x^{(k)}\right) \\ + & \sum_{I_+^{(j,k)}} \left[\frac{\partial c_j\left(x^{(k)}\right)}{\partial x_i} \left(U_i^{(k)} - x_i^{(k)}\right)^2 \left(\frac{1}{U_i^{(k)} - x_i} - \frac{1}{U_i^{(k)} - x_i^{(k)}} \right) \right] \\ - & \sum_{I_-^{(j,k)}} \left[\frac{\partial c_j\left(x^{(k)}\right)}{\partial x_i} \left(x_i^{(k)} - L_i^{(k)}\right)^2 \left(\frac{1}{x_i - L_i^{(k)}} - \frac{1}{x_i^{(k)} - L_i^{(k)}} \right) \right] \end{aligned} \tag{6}$$

with $I_+^{(j,k)} := \left\{ i \,\middle|\, \frac{\partial c_j\left(x^{(k)}\right)}{\partial x_i} \geq 0 \right\}, \quad I_-^{(j,k)} := \left\{ i \,\middle|\, \frac{\partial c_j\left(x^{(k)}\right)}{\partial x_i} < 0 \right\}$ and $L_i^{(k)} < x_i < U_i^{(k)}$.

With these definitions we obtain the subproblem

$$\left(\mathrm{P}_{\text{sub}}^{(k)}\right) \begin{cases} \min\limits_{x \in \mathbb{R}^n} & f^{(k)}(x) \\ \text{s.t.} & c^{(k)}(x) \leq 0 \\ & Ax \leq b \\ & \underline{x}^{(k)} \leq x \leq \overline{x}^{(k)} \end{cases} \tag{7}$$

where $\underline{x}_i^{(k)} := \max\left\{\underline{x}_i, x_i^{(k)} - \omega\left(x_i^{(k)} - L_i^{(k)}\right)\right\}$ and $\overline{x}_i^{(k)} := \min\left\{\overline{x}_i, x_i^{(k)} + \omega\left(U_i^{(k)} - x_i^{(k)}\right)\right\}$ for $i = 1, \ldots, n$ and a suitable constant ω with $0 < \omega < 1$.

The approximation of the nonlinear program (P) is strictly convex and of first order, i.e.,

$$\begin{array}{ll}
f^{(k)}\left(x^{(k)}\right) = f\left(x^{(k)}\right) & c^{(k)}\left(x^{(k)}\right) = c\left(x^{(k)}\right) \\
\nabla f^{(k)}\left(x^{(k)}\right) = \nabla f\left(x^{(k)}\right) & \nabla c^{(k)}\left(x^{(k)}\right) = \nabla c\left(x^{(k)}\right) \\
f^{(k)} \text{ strictly convex} & c^{(k)} \text{ convex} \\
f^{(k)} \text{ separable} & c^{(k)} \text{ separable}
\end{array}$$

Due to the strict convexity of the subproblem $\left(\mathrm{P}_{\mathrm{sub}}^{(k)}\right)$ it exhibits a unique solution. The first and second derivatives are available in analytical form and the Hessian of the corresponding Lagrangian function is diagonal. The asymptotes have to be adapted in a special way, see Zillober [34] for more details and Figure 2 for an illustration.

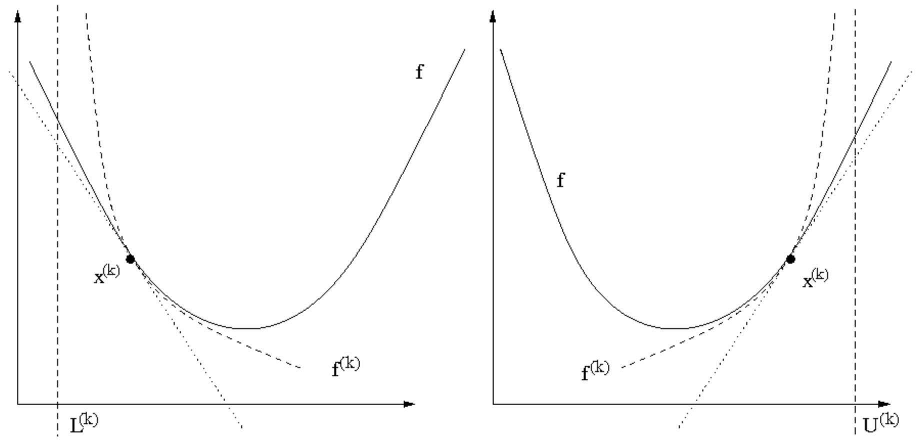

Figure 2: Method of Moving Asymptotes, Ertel [7].

To guarantee convergence we consider an extension of the MMA method, the sequential convex programming (SCP) method. To simplify the subsequent notation, we neglect the linear constraints $Ax \leq b$ in (P), i.e., $m_l = 0$, and proceed from

$$(\mathrm{PS}) \begin{cases} \min\limits_{x \in \mathbb{R}^n} & f(x) \\ \text{s.t.} & c(x) \leq 0 \\ & \underline{x} \leq x \leq \overline{x} \end{cases} \tag{8}$$

The corresponding convex and separable subproblem now becomes

$$\left(\mathrm{PS}_{\mathrm{sub}}^{(k)}\right)\begin{cases} \min\limits_{x\in\mathbb{R}^n} & f^{(k)}(x) \\ \text{s.t.} & c^{(k)}(x) \le 0 \\ & \underline{x}^{(k)} \le x \le \overline{x}^{(k)} \end{cases} \tag{9}$$

An additional merit function is introduced and a line search is performed along the descent direction, obtained from subproblem $\left(\mathrm{PS}_{\mathrm{sub}}^{(k)}\right)$. In this case the augmented Lagrangian merit function is chosen, which is defined by

$$\Phi_\rho(x,y) \;=\; f(x) \;+\; \sum_{j=1}^{m}\begin{cases} y_j c_j(x) \;+\; \dfrac{\rho_j}{2}c_j^2(x), & \text{if } -\dfrac{y_j}{\rho_j} \le c_j(x) \\ -\dfrac{y_j^2}{2\rho_j}, & \text{otherwise.} \end{cases} \tag{10}$$

for a given set of penalty parameters $\rho_j > 0$, $j = 1, \ldots, m$.

The merit function combines the objective function and constraints in a suitable way controlled by penalty parameters ρ_j, which must be carefully adapted during the solution process to guarantee descent search directions and global convergence of the algorithm. The same merit function is used by Schittkowski [25] in the frame of a sequential quadratic programming (SQP) method.

The SCP algorithm is summarized as follows:

Algorithm 1 *Sequential Convex Programming*

```
Step 0: Choose starting value $x^{(0)}$, select suitable parameter values to control the algorithm
        and let $k := 0$.
Step 1: Determine $L_i^{(k)}$, $U_i^{(k)}$ for $i = 1, \ldots, n$, and compute $f^{(k)}(x)$, $c^{(k)}(x)$.
Step 2: Solve $\left(P_{sub}^{(k)}\right)$. Let $z^{(k)}$ be the optimal solution and $v^{(k)}$ be the multiplier.
Step 3: If $z^{(k)} = x^{(k)}$, then stop. $x^{(k)}$ and $y^{(k)}$ is a KKT point.
Step 4: Let $p^{(k)} := \begin{pmatrix} z^{(k)} - x^{(k)} \\ v^{(k)} - y^{(k)} \end{pmatrix}$ be the search direction.
Step 5: Perform a line search with respect to $\Phi\left(x^{(k)}, y^{(k)}\right)$ along the direction $p^{(k)}$ to get a step
        length $\sigma^{(k)}$ and a sufficient reduction of the merit function.
Step 6: Let $\begin{pmatrix} x^{(k+1)} \\ y^{(k+1)} \end{pmatrix} = \begin{pmatrix} x^{(k)} \\ y^{(k)} \end{pmatrix} + \sigma^{(k)} p^{(k)}$ be the new iterate, replace $k$ by $k+1$ and goto Step 1.
```

The asymptotes restrict the feasible region and allow to control the curvature of a merit function, e.g., the augmented Lagrangian function (10), see Zillober [31]. Under some mild assumptions a convergence proof can be given. For further details see Zillober [32].

3 The Convex and Separable Subproblem

In principle, the strictly convex and separable subproblem $\left(\mathrm{PS}_{\mathrm{sub}}^{(k)}\right)$ can be solved by any available algorithm. Besides of the solution the applied code has to provide the optimal multipliers. To simplify the notation, we omit again linear inequality constraints. They are either handled in the same way as general nonlinear constraints or are passed directly to the subproblem solver.

The code SCPIP of Zillober [34] calls a standard routine based on an interior point method (IPM), see Zillober [31]. Since predictor-corrector interior point methods are very efficient in linear programming, see e.g., Lustig, Marsten, and Shanno [22] and Mehrota [23], the same basic idea is modified to take the special structure into account.

A particular advantage is that first and second order partial derivatives are analytically given and that the Hessian matrix of the Lagrangian is diagonal due to separability. For the matter of completeness, these derivatives are listed below for the objective function $f^{(k)}(x)$,

$$\begin{aligned}\frac{\partial f^{(k)}(x)}{\partial x_i} &= \sum_{j\in I_+^{(k)}} \frac{\partial f(x^{(k)})}{\partial x_j}\frac{(U_j^{(k)}-x_j^{(k)})^2}{(U_j^{(k)}-x_j)^2} + \tau_j^{(k)}\underbrace{\frac{\left(x_j-x_j^{(k)}\right)\left(2U_j^{(k)}-x_j-x_j^{(k)}\right)}{\left(U_j^{(k)}-x_j\right)^2}}_{\text{only objective}} \\ &+ \sum_{j\in I_-^{(k)}} \frac{\partial f(x^{(k)})}{\partial x_j}\frac{(x_j^{(k)}-L_j^{(k)})^2}{(x_j-L_j^{(k)})^2} + \tau_j^{(k)}\underbrace{\frac{\left(x_j-x_j^{(k)}\right)\left(x_j-2L_j^{(k)}+x_j^{(k)}\right)}{\left(x_j-L_j^{(k)}\right)^2}}_{\text{only objective}}\end{aligned} \qquad (11)$$

$$\begin{aligned}\frac{\partial^2 f^{(k)}(x)}{\partial x_i^2} &= \sum_{j\in I_+^{(k)}} \frac{\partial f(x^{(k)})}{\partial x_j}\frac{2(U_j^{(k)}-x_j^{(k)})^2}{(U_j^{(k)}-x_j)^3} + 2\tau_j^{(k)}\underbrace{\left(\frac{\left(U_j^k-x_j^{(k)}\right)^2}{\left(U_j^{(k)}-x_j\right)^3}\right)}_{\text{only objective}} \\ &- \sum_{j\in I_-^{(k)}} \frac{\partial f(x^{(k)})}{\partial x_j}\frac{2(x_j^{(k)}-L_j^{(k)})^2}{(x_j-L_j^{(k)})^3} + 2\tau_j^{(k)}\underbrace{\left(\frac{\left(x_j^{(k)}-L_j^{(k)}\right)^2}{\left(x_j-L_j^{(k)}\right)^3})\right)}_{\text{only objective}}\end{aligned} \qquad (12)$$

with $I_+^{(k)} := \left\{ j \,\middle|\, \frac{\partial f\left(x^{(k)}\right)}{\partial x_j} \geq 0 \right\}$, $I_-^{(k)} := \left\{ j \,\middle|\, \frac{\partial f\left(x^{(k)}\right)}{\partial x_j} < 0 \right\}$, $L_j^{(k)} < x_j < U_j^{(k)}$ and a positive parameter τ.

Derivatives of the inequality constraints $c_j(x)$, $j = 1, \ldots, m$ are computed in the same way with $\tau^{(k)} = 0$ and corresponding $I_+^{(j,k)}$ and $I_+^{(j,k)}$, see (6).

The key idea of any IPM is to add nonnegative slack variables to all inequalities constraints,

$$\left(\mathrm{PS}_{\mathrm{sub}}^{(k)} - \mathrm{slack}\right)\left\{\begin{array}{ll} \min\limits_{x,s,t\in\mathbb{R}^n,h,r\in\mathbb{R}^m} & f^{(k)}(x) \\ \text{s.t.} & c^{(k)}(x)+h=0 \\ & -h+r=0 \\ & \underline{x}^{(k)}-x+s=0 \\ & x-\overline{x}^{(k)}+t=0 \\ & r,s,t\geq 0 \end{array}\right. \qquad (13)$$

with slack variables $r = (r_1, \ldots, r_m)^T$, $s = (s_1, \ldots, s_n)^T$, $t = (t_1, \ldots, t_n)^T$, and $h = (h_1, \ldots, h_m)^T$. The variable r is introduced for some technical reasons, see Zillober [32]. The corresponding Lagrange function is

$$\begin{aligned} L_\mu^{(k)}(x, y, h, r, s, t, d_r, d_s, d_t) \quad := \quad & f^{(k)}(x) - \mu \sum_{j=1}^{m} \ln r_j - \mu \sum_{i=1}^{n} \ln s_i - \mu \sum_{i=1}^{n} \ln t_i \\ & + y^T(c^{(k)}(x) + h) \\ & + d_r^T(-h + r) + d_s^T(\underline{x}^{(k)} - x + s) + d_t^T(x - \overline{x}^{(k)} + t) \end{aligned} \tag{14}$$

where $c^{(k)}(x) = (c_1^{(k)}(x), \ldots, c_m^{(k)}(x))^T$ and y, d_r, d_s, d_t are the dual variables of the corresponding constraints. μ is a positive penalty parameter. Subsequently we denote this Lagrangian function by $L_\mu^{(k)}(x)$.

¿From the KKT optimality condition for (14) we obtain

$$\begin{array}{lcll} \nabla_x L_\mu^{(k)}(x) & : & \nabla f^{(k)}(x) + J^{(k)}(x) y - d_s + d_t & = \;\; 0 \\ \nabla_y L_\mu^{(k)}(x) & : & c^{(k)}(x) + h & = \;\; 0 \\ \nabla_h L_\mu^{(k)}(x) & : & y - d_r & = \;\; 0 \\ \nabla_r L_\mu^{(k)}(x) & : & D_r R e - \mu e & = \;\; 0 \\ \nabla_s L_\mu^{(k)}(x) & : & D_s S e - \mu e & = \;\; 0 \\ \nabla_t L_\mu^{(k)}(x) & : & D_t T e - \mu e & = \;\; 0 \\ \nabla_{d_r} L_\mu^{(k)}(x) & : & -h + r & = \;\; 0 \\ \nabla_{d_s} L_\mu^{(k)}(x) & : & \underline{x}^{(k)} - x + s & = \;\; 0 \\ \nabla_{d_t} L_\mu^{(k)}(x) & : & x - \overline{x}^{(k)} + t & = \;\; 0 \end{array} \tag{15}$$

where R, S, T, D_r, D_s, and D_t are diagonal matrices with entries of the vectors r, s, t, d_r, d_s and d_t respectively and where $J^{(k)}(x)$ denotes the Jacobian matrix of the nonlinear constraint functions $c^{(k)}(x)$. Thus we get the following sparse system of linear equations by which a Newton step is defined,

$$\begin{pmatrix} \nabla_{xx}^2 L^{(k)}(x,y) & J^{(k)}(x) & & & & & & -I & I \\ J^{(k)}(x)^T & & I & & & & & & \\ & I & & & & & -I & & \\ & & & D_r & & & R & & \\ & & & & D_s & & & S & \\ & & & & & D_t & & & T \\ & & -I & I & & & & & \\ -I & & & & I & & & & \\ I & & & & & I & & & \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta h \\ \Delta r \\ \Delta s \\ \Delta t \\ \Delta d_r \\ \Delta d_s \\ \Delta d_t \end{pmatrix} = -\nabla L_\mu^{(k)}(x) \tag{16}$$

where $L^{(k)}(x, y) = f^{(k)}(x) + c^{(k)}(x)^T y$ is the Lagrangian function of subproblem $\left(\mathrm{PS}_{\mathrm{sub}}^{(k)}\right)$, see also (2) for a more general definition. The diagonal elements of the submatrices are all positive and $\nabla_{xx}^2 L^{(k)}(x, y)$ is diagonal due to separability. Subsequently the main steps of the IPM are outlined.

```
Algorithm 2 The IPM predictor-corrector method for solving $\left(PS^{(k)}_{sub}\right)$

Step 0: Choose starting values $x^{(0)}$, $h^{(0)}$, $r^{(0)}$, $s^{(0)}$, $t^{(0)}$, $d_r^{(0)}$, $d_s^{(0)}$, $d_t^{(0)}$ and let $k := 0$.
Step 1: Let $\mu^{(0)} := \max_{j=1,\dots,m,i=1,\dots,n} \left\{ r_j^{(0)}(d_r^{(0)})_j, s_i^{(0)}(d_s^{(0)})_i, t_i^{(0)}(d_t^{(0)})_i, \right\}$.
Step 2: Solve Newton System (16) for the predictor step.
Step 3: Add perturbation parameter to the right hand side of the system.
Step 4: Solve Newton System (16) for the corrector step using the results of the predictor step
        and $\mu^{(k+1)}$.
Step 5: Compute independent primal and dual step sizes $\delta_p$ and $\delta_d$.
Step 6: Update iterates by
            $x^{(k+1)} := x^{(k)} + \delta_p \Delta x$,    $y^{(k+1)} := y^{(k)} + \delta_d \Delta y$,
            $h^{(k+1)} := h^{(k)} + \delta_p \Delta h$,    $r^{(k+1)} := r^{(k)} + \delta_p \Delta r$,
            $s^{(k+1)} := s^{(k)} + \delta_p \Delta s$,    $t^{(k+1)} := t^{(k)} + \delta_p \Delta t$,
            $d_r^{(k+1)} := d_r^{(k)} + \delta_d \Delta d_r$,    $d_s^{(k+1)} := d_s^{(k)} + \delta_d \Delta d_s$,
            $d_t^{(k+1)} := d_t^{(k)} + \delta_d \Delta d_t$
        Let $k := k + 1$. If $\|L_\mu^{(k)}(x)\| \le \epsilon$ stop, else goto Step 2.
```

The predictor step is used to estimate a proper value for the parameter μ and the nonlinearity terms for the corrector step. Thus the linear system is solved without taking μ into account, which is part of the right hand side. The algorithm stops as soon as a stopping criterion based on the KKT conditions for $\left(\mathrm{PS}^{(k)}_{\mathrm{sub}}\right)$ is satisfied subject to a predetermined accuracy, see Zillober [34, 31, 32, 33] for details.

The main computational work of the SCP algorithm consists of solving the linear equation systems (16). Due to its special structure it is possible to reduce the size either to an $(n + m) \times (n + m)$, $n \times n$ or $m \times m$ system, where m denotes the number of constraints and n the number of variables. However, the resulting reduced system has special properties depending on the performed reduction, see the subsequent Table 1.

	dual	(S2)	(S3)	(S4)
dimension	$m \times m$	$(n+m) \times (n+m)$	$m \times m$	$n \times n$
definite	positive	indefinite	positive	positive
sparsity	dense	sparse	-	-

Table 1: Dimension of Subproblems, Zillober [34].

4 Free Material Optimization

The SCP algorithm is to be applied for solving free material optimization (FMO) problems, see Bendsøe [4] and Zowe, Kočvara, and Bendsøe [35] for a deeper treatment. FMO tries to find the *best* mechanical structure with respect to one or more given loads in the sense that a design criterion, e.g., minimal weight or maximal stiffness, is obtained. The material properties as well as material distribution in the available space is varied. In this context FMO belongs to topology optimization.

As shown by Kočvara and Stingl [17], the FMO problem can be formulated for a given set of loads by a convex nonlinear semidefinite programming (NSDP) problem. Other problem formulations are given by Kočvara, Beck, Ben-Tal and Stingl in [13].

We proceed from m elements of an FE discretization of the underlying mechanical design structure. The optimization variable E consists of a set of elasticity matrices E_i that represent material properties in each finite element i, $i = 1, \ldots, m$. These matrices must be symmetric and positive semidefinite for physical reasons.

We denote an $dm \times dm$ matrix E by

$$E := diag(E_1, \ldots, E_m) = \begin{pmatrix} E_1 & 0 & \ldots & 0 \\ 0 & E_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & E_m \end{pmatrix} \tag{17}$$

where $d = 3$ in the two-dimensional and $d = 6$ in the three-dimensional case. For a two-dimensional structure we get the material matrix

$$E_i = \begin{pmatrix} e_{i1} & e_{i2} & e_{i3} \\ e_{i2} & e_{i4} & e_{i5} \\ e_{i3} & e_{i5} & e_{i6} \end{pmatrix} \tag{18}$$

and for the three-dimensional we get

$$E_i = \begin{pmatrix} e_{i1} & e_{i2} & e_{i3} & e_{i4} & e_{i5} & e_{i6} \\ e_{i2} & e_{i7} & e_{i8} & e_{i9} & e_{i10} & e_{i11} \\ e_{i3} & e_{i8} & e_{i12} & e_{i13} & e_{i14} & e_{i15} \\ e_{i4} & e_{i9} & e_{i13} & e_{i16} & e_{i17} & e_{i18} \\ e_{i5} & e_{i10} & e_{i14} & e_{i17} & e_{i19} & e_{i20} \\ e_{i6} & e_{i11} & e_{i15} & e_{i18} & e_{i20} & e_{i21} \end{pmatrix} \tag{19}$$

for $i = 1, \ldots, m$.

The so-called compliance function is a measure for the stiffness of a mechanical structure. The smaller the compliance, the more robust is the structure subject to external loads f_j, for a load case j, $j = 1, \ldots, l$. The stiffness of the structure depends on the material properties of each element E_i, $i = 1, \ldots, m$ and is given by $f_j^T K^{-1}(E) f_j$ for $j = 1, \ldots, l$, where

$$K(E) := \sum_{i=1}^{m} \sum_{k=1}^{nig} B_{i,k}^T E_i B_{i,k} \tag{20}$$

is the global stiffness matrix, see Kočvara and Stingl [16]. The matrices $B_{i,k}$ are needed to determine the element stiffness matrices over nig Gaussian integration points. A more detailed description is found in Kočvara and Zowe [20], and in Zowe, Kočvara, Bendsøe [35].

The design goal is to minimize the maximum compliance over a set of l different load cases by introducing an additional design variable α, which is to be minimized subject to

$$f_j^T K^{-1}(E) f_j \leq \alpha \tag{21}$$

for $j = 1, \ldots, l$.

To ensure that the global stiffness matrix $K(E)$ is invertible and to prevent numerical instabilities, we require

$$E_i - \underline{\rho} I \succeq 0 \tag{22}$$

with $i = 1, \ldots, m$, the identity matrix I and $\underline{\rho}$ a small positive value. For dual methods, (22) is often replaced by

$$\underline{\rho}' \leq \mathit{trace}(E_i). \tag{23}$$

Note that (22) is more restrictive than (23).

As a measure of the material stiffness in the coordinate directions, we use the traces of the matrices $E_i, i = 1, \ldots, m$. Thus, the sum of all these traces must not be larger than the available volume V, i.e.,

$$\sum_{i=1}^{m} \mathit{trace}(E_i) \leq V \tag{24}$$

see Ben-Tal, Kočvara, Nemirovski, and Zowe [2]. To prevent singularities we bound the trace of each element material matrix by a positive constant $\overline{\rho}$,

$$\mathit{trace}(E_i) \leq \overline{\rho} \tag{25}$$

for $i = 1, \ldots, m$.

We obtain the basic formulation of the FMO problem

$$\text{(FMO)} \left\{ \begin{array}{lll} \min\limits_{E,\alpha} & \alpha & \\ s.t. & \sum\limits_{i=1}^{m} \mathit{trace}(E_i) \leq V & \\ & \mathit{trace}(E_i) \leq \overline{\rho} & i = 1, \ldots, m \\ & f_j^T K(E)^{-1} f_j \leq \alpha & j = 1, \ldots, l \\ & E_i - \underline{\rho} I \succeq 0 & i = 1, \ldots, m \end{array} \right. \tag{26}$$

Here the optimization variables are the artificial variable α and the entries of the elementary material matrices $E_i \in \mathbb{R}^{d \times d}$, $i = 1, \ldots, m$, where $d = 3$ or $d = 6$, see (18) and (19). E is the diagonal matrix of $E_1, \ldots, E_m$, confer also (17).

For matter of completeness, we also regard an alternative formulation of the FMO problem resulting from replacing (22) by (23),

$$\text{(FMO}_2\text{)} \left\{ \begin{array}{lll} \min\limits_{E,\alpha} & \alpha & \\ s.t. & \sum\limits_{i=1}^{m} \mathit{trace}(E_i) \leq V & \\ & \underline{\rho}' \leq \mathit{trace}(E_i) \leq \overline{\rho} & i = 1, \ldots, m \\ & f_j^T K(E)^{-1} f_j \leq \alpha & j = 1, \ldots, l \\ & E_i \succeq 0 & i = 1, \ldots, m \end{array} \right. \tag{27}$$

with $E_i \in \mathbb{R}^{d\times d}$, $i = 1, \ldots, m$, $d = 3$ or $d = 6$, see (18) and (19), and E analogue to (FMO). Since this approach causes numerical instabilities, we omit this formulation and consider problem (FMO) in the sequel.

For the numerical realization, we first compute the compliance vector $u_j(E)$ for each load f_j, $j = 1, \ldots, l$, from

$$K(E)u_j(E) = f_j \tag{28}$$

leading to compliance constraints

$$c_j(E, \alpha) := f_j^T u_j(E) - \alpha \leq 0, \tag{29}$$

$j = 1, \ldots, l$.

Note that (FMO) is formulated over semidefinite variables. To get first order approximations, the computation of derivatives for the only nonlinear constraints, the compliances, is required. The other constraints are linear and can be handled directly. We get

$$\begin{aligned} \frac{\partial}{\partial\left(E_i\right)_{pq}} f_j^T K(E)^{-1} f_j &= -u_j(E)^T \left(\frac{\partial K(E)}{\partial\left(E_i\right)_{pq}}\right) u_j(E) \\ &= -u_j(E)^T \left(\sum_{k=1}^{nig} B_{i,k}^T \left(\frac{\partial E_i}{\partial\left(E_i\right)_{pq}}\right) B_{i,k}\right) u_j(E) \end{aligned} \tag{30}$$

for $i = 1, \ldots, m$, $p, q = 1, \ldots, d$, $j = 1, \ldots, l$ and $u_j(E) = K(E)^{-1} f_j$.

The crucial part of the SCP algorithm is the solution of the convex and separable subproblems $\left(\mathrm{PS}_{\mathrm{sub}}^{(k)}\right)$. We proceed now from an actual iterate $E^{(k)}$, see (17), and $\alpha^{(k)}$, retain linear constraints and get the subproblem

$$\left(\mathrm{FMO}_{\mathrm{sub}}^{(k)}\right) \left\{ \begin{array}{lll} \min\limits_{E,\alpha} & f^{(k)}(\alpha) & \\ s.t. & \sum\limits_{i=1}^{m} \mathit{trace}(E_i) \leq V & \\ & \mathit{trace}(E_i) \leq \overline{\rho} & i = 1, \ldots, m \\ & c_j^{(k)}(E, \alpha) \leq 0 & j = 1, \ldots, l \\ & E_i - \underline{\rho} I \succeq 0 & i = 1, \ldots, m \\ & \left(\underline{E_i}\right)_{pq}^{(k)} \leq \left(E_i\right)_{pq}^{(k)} \leq \left(\overline{E_i}\right)_{pq}^{(k)} & i = 1, \ldots, m;\ p, q = 1, \ldots, d \end{array} \right. \tag{31}$$

with

$$\left(\underline{E_i}\right)_{pq}^{(k)} := \left(E_i\right)_{pq}^{(k)} - w\left(\left(E_i\right)_{pq}^{(k)} - L_{ipq}^{(k)}\right) \tag{32}$$

$$\left(\overline{E_i}\right)_{pq}^{(k)} := \left(E_i\right)_{pq}^{(k)} + w\left(U_{ipq}^{(k)} - \left(E_i\right)_{pq}^{(k)}\right) \tag{33}$$

where $0 < w < 1$ fixed, $i = 1, \ldots, m$, $p, q = 1, \ldots, d$, see also $\left(\mathrm{P}_{\mathrm{sub}}^{(k)}\right)$,

$$f^{(k)}(\alpha) = \alpha^{(k)} + \frac{\left(U_{\overline{m}+1}^{(k)} - \alpha^{(k)}\right)^2}{U_{\overline{m}+1}^{(k)} - \alpha} - \left(U_{\overline{m}+1}^{(k)} - \alpha^{(k)}\right) + \tau_{\overline{m}+1}^{(k)} \frac{\left(\alpha - \alpha^{(k)}\right)^2}{U_{\overline{m}+1}^{(k)} - \alpha} \tag{34}$$

and

$$
\begin{aligned}
& c_j^{(k)}\left(E,\alpha\right) \\
= \quad & c_j\left(E^{(k)},\alpha^{(k)}\right) \\
+ \quad & \sum_{I_+^{(j,k)}} \left(-u_j(E^{(k)})^T \frac{\partial K(E^{(k)})}{\partial\left(E_i\right)_{pq}} u_j(E^{(k)})\right) \left(\frac{\left(U_{ipq}^{(k)} - \left(E_i^{(k)}\right)_{pq}\right)^2}{U_{ipq}^{(k)} - \left(E_i\right)_{pq}} - \left(U_{ipq}^{(k)} - \left(E_i^{(k)}\right)_{pq}\right)\right) \\
- \quad & \sum_{I_-^{(j,k)}} \left(-u_j(E^{(k)})^T \frac{\partial K(E^{(k)})}{\partial\left(E_i\right)_{pq}} u_j(E^{(k)})\right) \left(\frac{\left(\left(E_i^{(k)}\right)_{pq} - L_{ipq}^{(k)}\right)^2}{\left(E_i\right)_{pq} - L_{ipq}^{(k)}} - \left(\left(E_i^{(k)}\right)_{pq} - L_{ipq}^{(k)}\right)\right) \\
- \quad & \frac{\left(\alpha^{(k)} - L_{\overline{m}+1}^{(k)}\right)^2}{\alpha - L_{\overline{m}+1}^{(k)}} - \left(\alpha^{(k)} - L_{\overline{m}+1}^{(k)}\right)
\end{aligned}
\tag{35}
$$

for $j = 1, \ldots, l$, $p, q = 1, \ldots, d$ and $u_j(E) = K(E)^{-1} f_j$. Here U_{ipq} and L_{ipq} are the corresponding upper and lower asymptotes of the variable $\left(E_i\right)_{pq}$, $\overline{m} = 6m$ in case of two-dimensional and $\overline{m} = 21m$ in case of three-dimensional structures, $\tau_{\overline{m}+1}$ a positive parameter, and the index sets are defined by

$$
I_+^{(j,k)} := \left\{ ipq \;\middle|\; -u_j(E^{(k)})^T \frac{\partial K(E^{(k)})}{\partial\left(E_i\right)_{pq}} u_j(E^{(k)}) \geq 0 \right\} \tag{36}
$$

$$
I_-^{(j,k)} := \left\{ ipq \;\middle|\; -u_j(E^{(k)})^T \frac{\partial K(E^{(k)})}{\partial\left(E_i\right)_{pq}} u_j(E^{(k)}) < 0 \right\} \tag{37}
$$

However, the PSD conditions $E_i - \underline{\rho} I \succeq 0, i = 1, \ldots, m$ violate the standard SCP methodology. In the sequel, we will propose either variants to replace this condition by another one, which is directly handled by the code SCPIP, or we pass $\left(\mathrm{FMO}_{\mathrm{sub}}^{(k)}\right)$ as it stands to a special convex SDP solver.

1. Diagonal dominance: In case of isotropic materials the material matrices are diagonally dominant. This condition can be enforced by imposing suitable linear constraints.

2. Cholesky decomposition: In the more general case, i.e., anisotropic materials, the elementary material matrices E_i are replaced by their Cholesky decomposition, $E_i = L_i L_i^T$, where L_i is a lower triangular matrix, $i = 1, \ldots, m$. Each occurrence of E_i is replaced by its decomposition and the optimization is performed over variables L_i.

3. PENNON: The code PENNON of Kočvara and Stingl [14] is able to handle general nonlinear PSD matrix constraints of the form $A(E) \succeq 0$. In our situation, we have $A(E) := E - \underline{\rho} I$. The advantage is that more complex stability constraints can be handled directly, if the model is extended in future.

4. CONERML: The code CONERML, see Ben-Tal and Nemirovski [3] and Beck, Ben-Tal and Tetruashvili [1], is able to solve very large convex optimization problems over positive definite matrices and can be applied directly to solve the convex subproblem.

5 Solution Methods

5.1 Isotropic Materials

We first consider two-dimensional isotropic materials, which possess the same properties in all directions. The numerical results serve as benchmark tests for the subsequently discussed anisotropic materials.

The advantage of isotropic materials is that the elasticity matrices have a special structure and that they are diagonally dominant, i.e.,

$$E_i = \begin{pmatrix} e_{i1} & e_{i2} & 0 \\ e_{i2} & e_{i1} & 0 \\ 0 & 0 & e_{i1} - e_{i2} \end{pmatrix} \tag{38}$$

and $e_{i1} \geq 0$, $e_{i1} \geq |e_{i2}|$, $i = 1, \ldots, m$, see Pedersen [24] and Horn and Johnson [9]. Note that corresponding to (22) the matrices $E_i - \underline{\rho} I$, $i = 1, \ldots, m$ have to be positive semidefinite. These requirements can be reformulated in terms of additional linear constraints added to our FMO problem (FMO),

$$\text{(DDA)} \left\{ \begin{array}{lll} \min\limits_{E,\alpha} & \alpha & \\ s.t. & \sum\limits_{i=1}^{m} trace(E_i) \leq V & \\ & trace(E_i) \leq \overline{\rho} & i = 1, \ldots, m \\ & f_j^T K(E)^{-1} f_j \leq \alpha & j = 1, \ldots, l \\ & e_{i1} \geq \underline{\rho} & i = 1, \ldots, m \\ & e_{i1} - \underline{\rho} \geq -e_{i2} & i = 1, \ldots, m \\ & e_{i1} - \underline{\rho} \geq e_{i2} & i = 1, \ldots, m \end{array} \right. \tag{39}$$

with $trace(E_i) = 3e_{i1} - e_{i2}$ for $i = 1, \ldots, m$. The resulting problem possesses a large number of sparse linear constraints and as it is nonlinear it is solved by SCPIP using the standard interior point method to compute the optimal solution of the subproblem.

5.2 Cholesky Decomposition

The idea of the second approach is to create an optimization problem without semidefinite constraints which is directly solved by the code SCPIP. The conditions $E_i \succeq 0$, for $i = 1, \ldots, m$ are satisfied if we use Cholesky factors as optimization variables, i.e., if $E_i = L_i L_i^T$ with lower triangular matrices L_i. It is important to note that linear constraints in E_i become nonlinear in the new variables L_i.

In the two-dimensional case we have

$$L_i = \begin{bmatrix} l_{i1} & 0 & 0 \\ l_{i2} & l_{i3} & 0 \\ l_{i4} & l_{i5} & l_{i6} \end{bmatrix} \tag{40}$$

and

$$E_i = \begin{bmatrix} l_{i1}^2 & l_{i1}l_{i2} & l_{i1}l_{i4} \\ l_{i1}l_{i2} & l_{i2}^2 + l_{i3}^2 & l_{i2}l_{i4} + l_{i3}l_{i5} \\ l_{i1}l_{i4} & l_{i2}l_{i4} + l_{i3}l_{i5} & l_{i4}^2 + l_{i5}^2 + l_{i6}^2 \end{bmatrix} \tag{41}$$

In the three-dimensional case $L_i,\ i = 1, \ldots, m$ are lower triangular matrices of the size 6×6 and the product is obtained similarly. Note that the trace of E_i is the sum of all squared elements of a Cholesky factor. As we require $E_i - \underline{\rho} I \succeq 0$, $i = 1, \ldots, m$, i.e., $E_i \succ 0$, see (22), all diagonal entries of L_i have to be positive.

Similar to (17) we denote an $dm \times dm$ matrix L by

$$L := diag(L_1, \ldots, L_m) = \begin{pmatrix} L_1 & 0 & \ldots & 0 \\ 0 & L_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & L_m \end{pmatrix} \tag{42}$$

where $d = 3$ in the two-dimensional and $d = 6$ in the three-dimensional case. The resulting FMO problem (CA) is then obtained as follows,

$$\text{(CA)} \begin{cases} \min\limits_{L,\alpha} & \alpha \\ s.t. & \sum\limits_{i=1}^{m} \|L_i\|_F^2 \leq V \\ & \|L_i\|_F^2 \leq \overline{\rho} \qquad i = 1, \ldots, m \\ & f_j^T K(LL^T)^{-1} f_j \leq \alpha \quad j = 1, \ldots, l \\ & diag(L_i) \geq \sqrt{\underline{\rho}} I \qquad i = 1, \ldots, m \end{cases} \tag{43}$$

where $\|.\|_F$ denotes the Frobenius norm. Moreover, we get from

$$K(LL^T) = \sum_{i=1}^{m} K_i(LL^T) = \sum_{i=1}^{m} \sum_{k=1}^{nig} B_{i,k}^T L_i L_i^T B_{i,k} \tag{44}$$

see also (20), the derivatives of the compliance

$$\frac{\partial}{\partial\,(L_i)_{pq}} \left(f_j^T K(LL^T)^{-1} f_j - \alpha \right) = -u_j(LL^T)^T \frac{\partial K(LL^T)}{\partial E_i} \frac{\partial E_i}{\partial (L_i)_{pq}} u_j(LL^T) \tag{45}$$

for $i = 1, \ldots, m$, $u_j(LL^T) = K(LL^T)^{-1} f_j$ and $p, q = 1, \ldots, d$. The problem is now in standard form and can be solved by SCPIP using the interior point method to solve the subproblem.

5.3 The SDP Solver PENNON

The next approach is to apply the semidefinite solver PENNON which is described by Kočvara and Stingl [14]. The algorithm solves optimization problems of the form

$$\text{(PENNON)} \begin{cases} \min\limits_{X \in \mathbb{R}^N} & F(X) \\ s.t. & A(X) \succeq 0 \\ & G_j(X) \leq 0 \quad j = 1, \ldots, M \end{cases} \tag{46}$$

with differentiable functions $F(X)$ and $G_j(X)$, $j = 1, \ldots, M$. $A(X)$ is a nonlinear mapping into the set of all symmetric $M_A \times M_A$ matrices.

The algorithm penalizes the inequality constraints by a penalty-barrier function, see Kočvara and Stingl [15] and Stingl [28]. An eigenvalue decomposition of $A(X)$ of the form $A(X) = S(X)^T\Lambda(X)S(X)$ is computed with $\Lambda(X) = diag(\lambda_1(X), \ldots, \lambda_{M_A}(X))^T$ and negative eigenvalues are penalized. This is achieved by formulating an augmented Lagrangian function to be minimized by PENNON.

PENNON is to be applied to the separable, convex subproblem $\left(\text{FMO}^{(k)}_{\text{sub}}\right)$ computed by SCP. The standard formulation proceeds from the identity $A(E) = E$, but PENNON allows future extensions towards stability and related constraints.

5.4 The Convex Programming Solver CONERML

The fourth approach is to apply a convex programming solver called CONERML, which is described in Ben-Tal and Nemirovski [3] and Beck, Ben-Tal and Tetruashvili [1]. The algorithm solves problems of the form

$$(\text{CONERML}) \begin{cases} \min\limits_{X\in\mathcal{X}} & F(X) \\ s.t. & H_j(X) \le 0 \quad j = 1, \ldots, M \end{cases} \tag{47}$$

where $\mathcal{X} := \{X \,|X \succeq \epsilon I\}$, $\epsilon > 0$, the identity matrix I and $F(X), H_j(X), j = 1, \ldots, M$ twice continuously differentiable.

This solver is to be integrated into SCPIP to solve the convex, separable and semidefinite subproblem $\left(\text{FMO}^{(k)}_{\text{sub}}\right)$ after some reformulations. The advantage of CONERML is that the program is able to solve large convex semidefinite optimization problems, while the rate of convergence is nearly independent on the dimension.

6 Displacement and Stress Constraints

As additional constraints, the maximum displacement in special nodes is to be restricted, see Kočvara [12]. Moreover there are unilateral contact conditions, see Kočvara, Zibulevsky, and Zowe [19], describing an obstacle near the design space which cannot be passed by the material. Displacement constraints prevent the material to exceed an imaginary border. This limit is due to technical and mechanical reasons described for special nodes and special directions. The displacement constraints can be formulated by

$$CK(E)^{-1}f_j \le \delta \tag{48}$$

for $j = 1, \ldots, l$, where C is a matrix with entries at the bounded nodes in the critical direction and zero otherwise. The gap between the contact surface and the imaginary boarder is described by δ

and we obtain the subsequent FMO problem, see (FMO), extended by displacement constraints,

$$
\text{(FMO-D)}\left\{\begin{array}{lll}
\min\limits_{E,\alpha} & \alpha & \\
s.t. & \sum\limits_{i=1}^{m} trace(E_i) \leq V & \\
 & trace(E_i) \leq \overline{\rho} & i = 1,\ldots,m \\
 & f_j^T K(E)^{-1} f_j \leq \alpha & j = 1,\ldots,l \\
 & CK(E)^{-1} f_j \leq \delta & j = 1,\ldots,l \\
 & E_i - \underline{\rho} I \succeq 0 & i = 1,\ldots,m
\end{array}\right. \tag{49}
$$

Displacement constraints are handled in the same way as the compliance constraints. We consider now the k-th displacement constraint $g_k(E)$ formulated for a load case j_k and the k-th row c_k of C,

$$
g_k(E) := c_k^T K(E)^{-1} f_{j_k} - \delta \tag{50}
$$

We obtain the partial derivatives subject to an entry of an elasticity matrix E_i by

$$
\frac{\partial g_k(E)}{\partial\,(E_i)_{pq}} = -c_k^T K(E)^{-1} \frac{\partial K(E)}{\partial\,(E_i)_{pq}} K(E)^{-1} f_{j_k} \tag{51}
$$

for $i = 1, \ldots, m$ and $p, q = 1, \ldots, d$. Matrix $K(E)$ depends linearly on each E_i and $\dfrac{\partial K(E)}{\partial\,(E_i)_{pq}}$ is computed only once. As usual, $c_k^T K(E)^{-1}$ is computed by solving $K(E)x = c_k$ proceeding from an available decomposition of $K(E)$.

Subsequently, we consider von Mises stresses, see Li, Steven, and Xie [21]. Stresses act in different directions at any point within the design space where the magnitude and the direction vary. Even if none of these stresses exceeds the yield stress of the material, it is possible that the combination of the stresses does. The von Mises criterion combines these different directions into an equivalent stress such that it is comparable to the yield stress of the material, see Li, Steven and Xie [21].

In the two-dimensional space, the von Mises stress $\sigma_{vm,j}(E_i)$ in an element i, $i = 1, \ldots, m$ can be formulated dependent on the load case j, $j = 1, \ldots, l$ by

$$
\sigma_{vm,j}(E_i) = \sqrt{\sigma_{xx,j}(E_i)^2 + \sigma_{yy,j}(E_i)^2 - \sigma_{xx,j}(E_i)\sigma_{yy,j}(E_i) + 3\sigma_{xy,j}(E_i)^2} \tag{52}
$$

or in vector notation

$$
\sigma_{vm,j}(E_i)^2 = \sigma_j(E_i)^T T \sigma_j(E_i) \tag{53}
$$

where $\sigma_j(E_i) = (\sigma_{xx,j}(E_i), \sigma_{yy,j}(E_i), \sigma_{xy,j}(E_i))^T$ is the element stress vector and

$$
T = \begin{bmatrix} 1 & -0.5 & 0 \\ -0.5 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{54}
$$

the coefficient matrix, see Li, Steven and Xie [21]. In a similar way the stress vector $\sigma_j(E_i)$ and the coefficient matrix is defined in the three-dimensional case.

The element stress vector can also be computed by a compliance vector $u_j(E)$ dependent on the load case j, $j = 1, \dots, l$ by

$$\sigma_j(E_i) = \sum_{k=1}^{nig} E_i B_{i,k} u_j(E) \tag{55}$$

with $u_j(E) = K(E)^{-1} f_j$, see also (28). If we replace (55) into (53) we get

$$\sigma_{vm,j}(E_i)^2 = \sum_{k=1}^{nig} u_j(E)^T B_{i,k}^T E_i^T T E_i B_{i,k} u_j(E) \tag{56}$$

for $i = 1, \dots, m$ and $j = 1, \dots, l$.

To ensure stability the von Mises stress may not exceed a special value s_σ in each element i, $i = 1, \dots, m$, and for each load f_j, $j = 1, \dots, l$. These nonlinear constraints can be added to the standard formulation (FMO) or the displacement formulation (FMO-D) to get

$$\text{(FMO-DS)} \left\{ \begin{array}{lll} \min\limits_{E,\alpha} & \alpha & \\ s.t. & \sum\limits_{i=1}^{m} trace(E_i) \leq V & \\ & trace(E_i) \leq \overline{\rho} & i = 1, \dots, m \\ & f_j^T K(E)^{-1} f_j \leq \alpha & j = 1, \dots, l \\ & C K(E)^{-1} f_j \leq \delta & j = 1, \dots, l \\ & \sum\limits_{k=1}^{nig} u_j(E)^T B_{i,k}^T E_i^T T E_i B_{i,k} u_j(E) \leq s_\sigma & j = 1, \dots, l, i = 1, \dots, m \\ & E_i - \underline{\rho} I \succeq 0 & i = 1, \dots, m \end{array} \right. \tag{57}$$

where $u_j(E)$ is implicitly defined by solving the system of linear equation $K(E)u_j(E) = f_j$ for a load j, $j = 1, \dots, l$, see also (28). Derivatives of the stress constraints are computed by

$$\begin{aligned} \frac{\partial}{\partial \left(E_i\right)_{pq}} & \sum_{k=1}^{nig} u_j(E)^T B_{i,k}^T E_i^T T E_i B_{i,k} u_j(E) = \\ & - \quad 2 \sum_{k=1}^{nig} u_j(E)^T \frac{\partial K(E)}{\partial \left(E_i\right)_{pq}} K^{-1}(E) B_{i,k}^T E_i^T T E_i B_{i,k} u_j(E) \\ & + \quad 2 \sum_{k=1}^{nig} u_j(E)^T B_{i,k}^T \frac{\partial E_i}{\partial \left(E_i\right)_{pq}} T E_i B_{i,k} u_j(E) \end{aligned} \tag{58}$$

for $i = 1, \dots, m$, $p, q = 1, \dots, d$, and $j = 1, \dots, l$.

Stress constraints do not satisfy the linear independency constraint qualification, see Kočvara and Stingl [17]. Thus the numerical solution can become unstable.

7 Numerical Realization and Test Results

7.1 Implementation and Test Environment

FMO problems are to be solved by the SCP solver SCPIP of Zillober [34]. Design variables are the entries of the elasticity matrices E_i, $i = 1 \ldots, m$ in a given order. The dimension is $6m + 1$ in the two- and $21m + 1$ in the three-dimensional case. Only for isotropic materials we are able to reduce their number significantly. One additional variable is needed to store the artificial variable α.

The main program organizes the problem formulation as well as the function and gradient evaluations and consists of six parts, see Figure 3. The first part retrieves FE- and FMO-data from the platform MOPED, see Kočvara, Stingl and Werner [18] and Hörnlein, Kočvara and Stingl [10]. The second part utilizes these information to compute the derivatives of the global stiffness matrix and its value at the current iterate. The linear system (28) is solved by the NAG routine MA47, see Duff and Reid [6], followed by the computation of function and gradient values. The last step consists of calling SCPIP, which on the other hand calls an appropriate solver for the separable and convex subproblems generated in each iteration step.

The Fortran subroutines are documented in an appendix. Compliance constraints and the objective function are scaled. Moreover, an active set strategy is performed to reduce the number of approximated constraints in the subproblems $\left(\mathrm{P}^{(k)}_{\mathrm{sub}}\right)$. The corresponding tolerance for detecting an active constraint must be carefully selected to reduce on the one hand the number of constraints as much as possible and on the other hand to avoid cycling if the parameter is too small.

Numerical results are obtained under Windows XP Professional x64 and an Intel Core 2 Duo 6600 processor using the Intel Fortran Compiler Version 9.1 including preprocessor.

For checking optimality, we use the KKT conditions shown in (3). The code SCPIP, see Zillober [34], supports additional stopping criteria listed in Table 2.

Stopping condition	Default	Value
KKT condition (3) satisfied, $\|\nabla_x L(x,y)\|_\infty \le \epsilon$	$\epsilon = 1 \cdot 10^{-5}$	0
Maximal number of iterations achieved	500	1
Gradient of objective function at a feasible point close to zero, $\|\nabla f(x)\|_\infty \le \epsilon$	$\epsilon = 1 \cdot 10^{-5}$	2
No alterations of optimization variables and objective function $\lvert f^{(k)} - f^{(k-1)}\rvert \le \epsilon_1$, $\frac{\lvert f^{(k)}-f^{(k-1)}\rvert}{f^{(k-1)}} \le \epsilon_2$, $\max\limits_{i=1,\ldots,n} \frac{\lvert x_i^{(k)}-x_i^{(k-1)}\rvert}{x_i^{(k-1)}} \le \epsilon_2$	$\epsilon_1 = 1 \cdot 10^{-5}$ $\epsilon_2 = 1 \cdot 10^{-5}$	3

Table 2: Stopping Criteria.

Two simple scalable structures are considered for debugging and for testing the standard FMO formulation (FMO) with one and two load cases, see Figure 4.

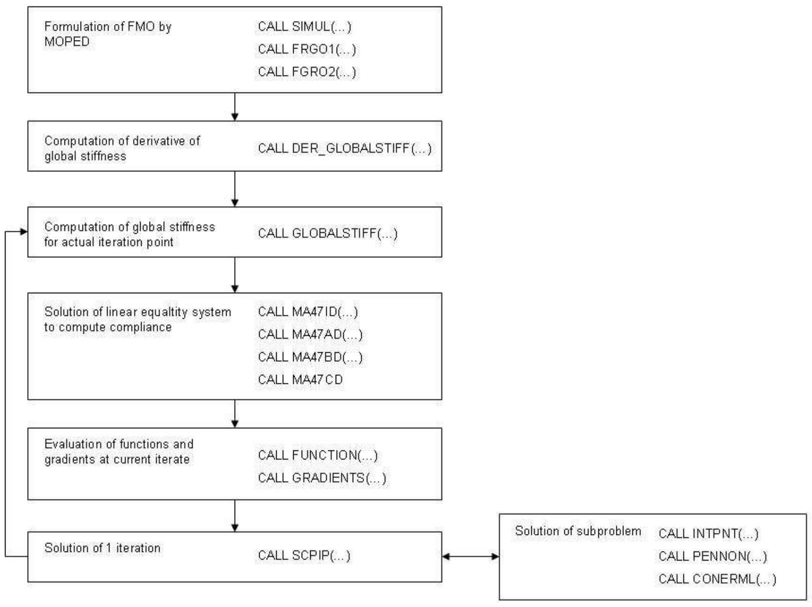

Figure 3: Program Architecture.

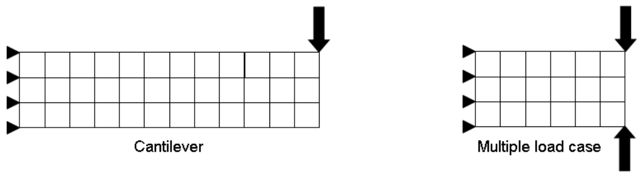

Figure 4: Academic Standard Testcases.

A graphical representation of the results is given by plotting the trace of the elactisity matrices in each element E_i, $i = 1, \ldots, m$. In Figure 5 the colormap of the plot is shown. The left hand side stands for a stiff material, i.e., the trace is large, while the right hand side represents a low value of the trace.

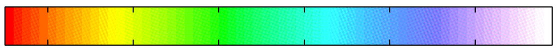

Figure 5: Colormap.

7.2 Isotropic Material

Isotropic materials allow to replace PSD constraints by additional linear constraints, see Section 5.1. Table 3 shows the number of variables and constraints passed to SCPIP. The compliance functions and the objective function are scaled by two parameters called S_1 and S_0. S_0 is dependent on the size of the grid and is given by $S_0 = m \cdot c$, with $c = 10^4$ and m the number of elements. The scaling factor S_1 is shown in Table 3. Numerical results are obtained for $\overline{\rho} = 1$, $V = \frac{m}{2}$ and $\underline{\rho} = 10^{-3}$. The termination accuracy of SCPIP is 10^{-5} and no active set strategy is used. Calculation times, separately for the internal work to solve the convex and separable subproblem, and number of iterations are presented in Table 4 for the MMA version of SCPIP, i.e., without line search. Moreover, the values of the objective function as well as a measure for optimality, i.e., $\|\nabla_x L(x, y)\|_\infty$, and the stopping reason of Table 2 are shown in Table 4. The linear constraints are handled as nonlinear ones, see Section 2. Figures 6 to 11 show the resulting material distributions first for one, then two load cases.

Name	# of elements	# of variables	# of constraints	S_1	Loads
Canti11_i	2,376	4,753	7,130	$0.3 \cdot 10^{-3}$	1
Canti12_i	4,726	9,453	14,180	$0.3 \cdot 10^{-2}$	1
Canti13_i	9,751	19,503	29,255	$0.3 \cdot 10^{-2}$	1
Canti21_i	1,176	2,353	3,531	$0.3 \cdot 10^{-4}$	2
Canti22_i	4,851	9,703	14,556	$0.3 \cdot 10^{-1}$	2
Canti23_i	9,591	19,183	28,776	$0.3 \cdot 10^{-4}$	2

Table 3: Dimensions for Isotropic Material.

Name	Iterations	SCPIP (h:min)	Model (h:min)	Objective Function	KKT Condition	Stopping Reason
Canti11_i	51	0:04	0:07	40.12	$0.99 \cdot 10^{-5}$	0
Canti12_i	384	1:47	3:19	40.17	$0.61 \cdot 10^{-4}$	3
Canti13_i	188	3:38	6:50	40.38	$0.36 \cdot 10^{-5}$	0
Canti21_i	82	0:02	0:03	14.63	$0.97 \cdot 10^{-5}$	0
Canti22_i	137	0:51	1:13	14.97	$0.52 \cdot 10^{-4}$	3
Canti23_i	13	0:18	0:44	20.17	$0.86 \cdot 10^{-5}$	0

Table 4: Results for Isotropic Material.

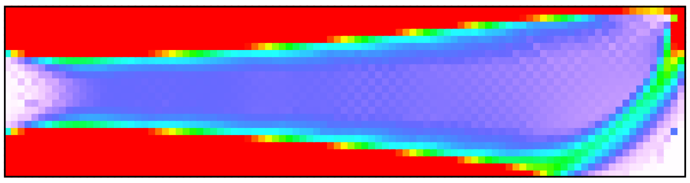
Figure 6: Canti11_i

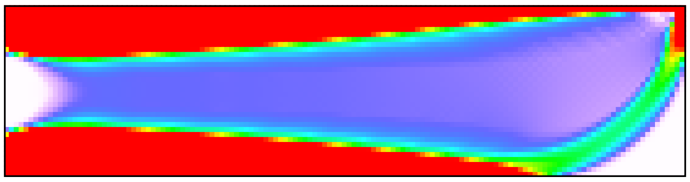

Figure 7: Canti12_i

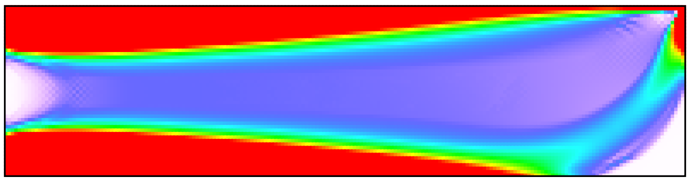

Figure 8: Canti13_i

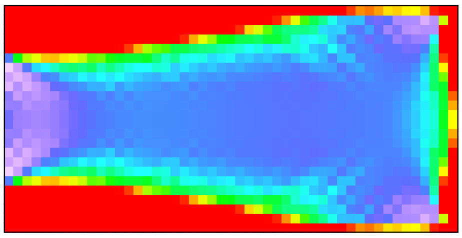

Figure 9: Canti21_i

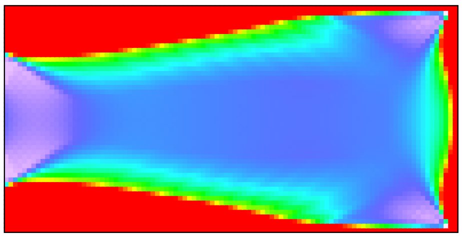

Figure 10: Canti22_i

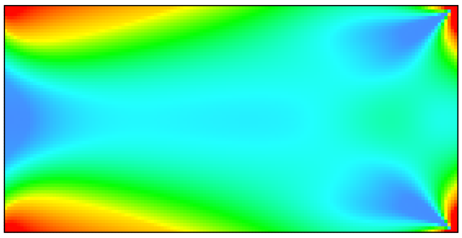

Figure 11: Canti23_i

7.3 Anisotropic Material

The same benchmark problems are used to test the Cholesky approach for anisotropic materials as presented in Section 5.2. Table 5 shows the number of variables and constraints passed to SCPIP. Two scaling parameters S_0 and S_1 are introduced one for the objective function and one for the compliance constraints. Problem parameters are chosen such that $S_0 = m \cdot c$, $c = 10^4$, $\overline{\rho} = 1$, $\underline{\rho} = 0.1$, and $V = \frac{m}{3}$. Termination accuracy of SCPIP is 10^{-5} and the active set method with tolerance $-4 \cdot 10^{-1}$ is selected, i.e., all constraints are taken into account with a function value smaller than $4 \cdot 10^{-1}$ as well as those whose Lagrange multipliers are positive and were active in the previous iteration. The parameter values for S_1 are shown in Table 5. The resulting problem is solved by SCPIP without line search.

Calculation time and number of iterations are presented in Table 6, separately for the internal work to solve the convex and separable subproblem, and the function and gradient evaluations including the solution of the linear system. Objective function values, a measure for optimality, i.e., $\|\nabla_x L(x, y)\|_\infty$, and the stopping criteria, see Table 2, are shown in Table 6. Figures 12 to 17 show the resulting material distributions first for one, then for two load cases.

Name	# of elements	# of variables	# of constraints	S_1	Loads
Canti11_a	351	2,107	353	1.5	1
Canti12_a	2,376	14,257	2,378	100	1
Canti13_a	4,726	28,357	4,728	100	1
Canti21_a	406	2,437	409	1.5	2
Canti22_a	1,176	7,057	1,179	1.5	2
Canti23_a	4,851	29,107	4,854	100	2

Table 5: Dimensions for Anisotropic Material.

Name	Iterations	SCPIP (h:min)	Model (h:min)	Objective Function	KKT Condition	Stopping Reason
Canti11_a	116	0:01	0:01	23.25	$1.10 \cdot 10^{-1}$	3
Canti12_a	88	1:12	0:13	149.94	$0.29 \cdot 10^{-3}$	3
Canti13_a	308	6:48	0:41	154.57	$0.26 \cdot 10^{-3}$	3
Canti21_a	161	0:01	0:01	21.72	$0.26 \cdot 10^{-1}$	3
Canti22_a	126	0:11	0:04	21.67	$0.76 \cdot 10^{-2}$	3
Canti23_a	74	5:41	0:48	35.01	$0.41 \cdot 10^{-1}$	3

Table 6: Results for Anisotropic Material.

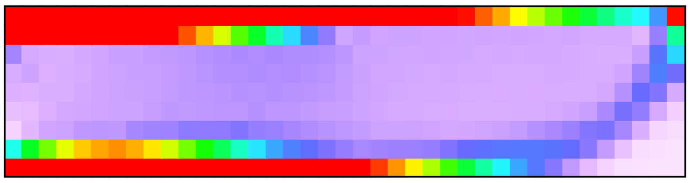
Figure 12: Canti11_a

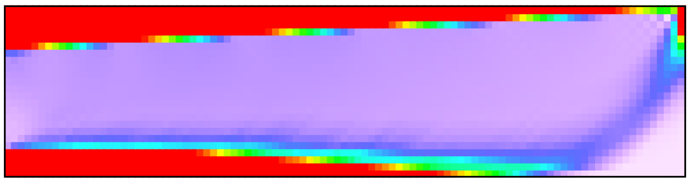
Figure 13: Canti12_a

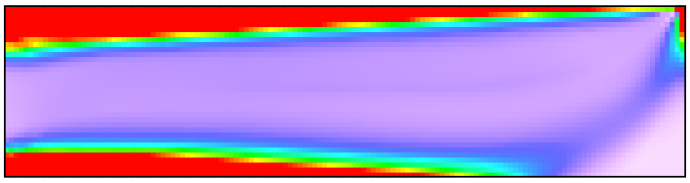
Figure 14: Canti13_a

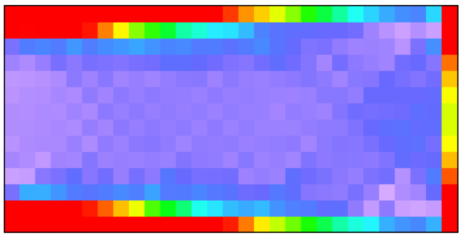

Figure 15: Canti21_a

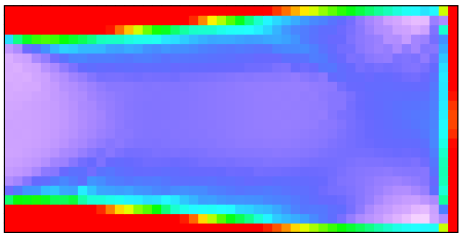

Figure 16: Canti22_a

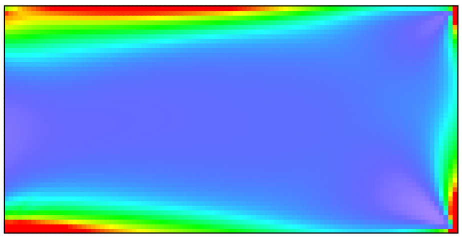

Figure 17: Canti23_a

To show that the optimization problem (FMO_2) introduced in Section 4 is more restrictive and causes instabilities, the corresponding calculations of the Cholesky approach are also considered. The difference to the approach shown in Section 5.2 is an additional lower bound on the trace,

$$\underline{\rho}' \leq \|L_i\|_F^2, \quad i = 1, \dots, m \tag{59}$$

and the lower bounds on the diagonal entries of $L_i, i = 1, \dots, m$ are zero instead of $\underline{\rho}$. The problem is solved using SCPIP without line search. Parameter values are $S_0 = m \cdot c$ with $c = 10^4$, $S_1 = 0.3 \cdot 10^{-3}$, $\overline{\rho} = 1$, $\underline{\rho}' = 0.001$, and $V = \frac{m}{3}$. Termination accuracy of SCPIP is 10^{-5} and the active set method with tolerance $-2 \cdot 10^{-1}$ is selected. Note that using this approach the number of constraints increase by the number of elements compared to problem formulation (FMO). The results are shown in Table 7.

Name	Iterations	SCPIP (h:min)	Model (h:min)	Objective Function	KKT Condition	Stopping Reason
Canti11_a	99	0:03	0:01	18	$0.93 \cdot 10^{-2}$	3
Canti12_a	180	4:11	0:28	101.54	$0.31 \cdot 10^{-2}$	3
Canti13_a	223	12:26	2:14	102.32	$0.67 \cdot 10^{-1}$	3
Canti21_a	193	0:05	0:02	18	$0.34 \cdot 10^{-1}$	3
Canti22_a	110	0:10	0:04	17.38	$0.58 \cdot 10^{-3}$	3
Canti23_a	500	8:56	5:17	20.31	$0.29 \cdot 10^{-1}$	3

Table 7: Results for Anisotropic Material, Formulation (FMO_2).

The large differences in the objective function values result from the large value of $\underline{\rho}$ used in formulation (FMO). This parameter has to be adapted carefully such that on the one hand no instabilities arise and on the other hand the formulation is not too restrictive, i.e., results in worse objective function values.

8 Conclusions

In this paper, we describe different ways to apply a sequential convex programming method to solve free material optimization problems. Interfaces to the external solvers PENNON and CONERML are outlined. Moreover, the formulas for attaching displacement and stress constraints are derived, also for the derivatives. For the standard formulation without displacement and stress constraints, numerical results are presented obtained for two simple academic test problems. The maximum number of optimization variables is about 30.000.

The numerical tests show the feasibility of the approach based on optimizing over the Cholesky factors of the elasticity matrices. However, this approach needs a careful setting of some tolerances. Its disadvantage is that linear constraints, e.g., bounds for traces, become nonlinear. More analytical and numerical investigations are necessary to understand the instabilities.

To solve larger and more practically relevant FMO problems, the linear equation solver MA47 used for computing the compliance vector, needs to be exchanged by a more powerful one.

The bottleneck is the solution of convex and separable subproblems generated by SCPIP and the necessity to handle positive semidefinite matrix variables directly. We hope that alternative approaches based on applying the mentioned external solvers PENNON and CONERML will allow us to solve larger and more realistic FMO problems in a robust way.

References

[1] Beck A., Ben-Tal A., Tetruashvili L. Large scale methods for convex FMO-type problems. Report, MINERVA Optimization Center, Faculty of Industrial Engineering, Technion, Israel, 2007.

[2] Ben-Tal A., Kočvara M., Nemirovski A., Zowe J. Free material design via semidefinite programming. The multi-load case with contact conditions. *SIAM Journal Optimization*, 9(4), 1999.

[3] Ben-Tal A., Nemirovski A. Non-eucliedean restricted memory level method for large-scale convex optimization. *Mathematical Programming*, 102, 2005.

[4] Bendsøe M. *Optimization of Structural Topology, Shape and Material.* Springer-Verlag Berlin, 1995.

[5] Bendsøe M., Sigmund O. *Topology Optimization: Theory, Methods and Applications.* Springer-Verlag Berlin, 2003.

[6] Reid J.K. Duff I.S. MA47, a Fortran code for direct solution of indefinite sparse symmetric linear systems. Technical Report RAL-95-001, Didcot, Oxon, UK, 1995.

[7] Ertel S. Anwendungen von Filtermethoden auf das Optimierungsverfahren SCP. Diplomarbeit, Mathematisches Institut, University Bayreuth, 2006.

[8] Fleury C. CONLIN: an efficient dual optimizer based on convex approximation concepts. *Structural Optimization*, 1, 1989.

[9] Horn R., Johnson C. *Matrix Analysis.* Cambridge University Press, 1985.

[10] Hörnlein H., Kočvara M., Stingl M. *MOPED-3D User's guide with NASTRAN input interface Version 0.9*, 2002.

[11] Hörnlein H., Kočvara M., Werner R. Material optimization: bridging the gap between conceptual and preliminary design. *Aerospace Science and Technology*, 5, 2001.

[12] Kočvara M. Topology optimization with displacement constraints: a bilevel programming approach. *Structural Optimization*, 14, 1997.

[13] Kočvara M., Beck A., Ben-Tal A., Stingl M. PLATO-N Work Package 4: FMO models, Task 2.2: Software specification, selection of FMO problem formulations. Report, 2007.

[14] Kočvara M., Stingl M. PENNON - A generalized augmented lagrangian method for semidefinite programming. Report, University Erlangen, 2001.

[15] Kočvara M., Stingl M. PENNON - A code for convex nonlinear and semidefinite programming. *Optimization Methods and Software*, 18(3), 2003.

[16] Kočvara M., Stingl M. Solving nonconvex SDP problems of structural optimization with stability control. *Optimization Methods and Software*, 19(5), 2004.

[17] Kočvara M., Stingl M. Free material optimization: towards the stress constraints. *Structural and Multidisciplinary Optimization*, 33(4-5), 2007.

[18] Kočvara M., Stingl M., Werner R. *MOPED User's guide Version 2.2*, 2002.

[19] Kočvara M., Zibulevsky M., Zowe J. Mechanical design problems with unilateral contact. *Mathematical modelling and numerical analysis*, 32, 1998.

[20] Kočvara M., Zowe J. Free material optimization: an overview. *Trends in Industrial and Applied Mathematics, Kluwer Academic Publishers*, 2002.

[21] Li Q., Steven G.P., Xie Y.M. On equivalence between stress criterion and stiffness criterion in evolutionary structural optimization. *Structural Optimization*, 18, 1999.

[22] Lustig I.J., Marsten R.E., Shanno D.F. On implementing Mehrota's predictor-corrector interior point method for linear programming. *SIAM J. Optimization*, 2, 1992.

[23] Mehrota S. On the implementation of a primal-dual interior point method. *SIAM J. Optimization*, 2, 1992.

[24] Pedersen P. *Elasticity - Anisotropy - Laminates.* Technical University of Denmark, www.mek.fam.dtu.dk/html/pp.html, 1997.

[25] Schittkowski K. On the convergence of a sequential quadratic programming method with an augmented lagrangian line search function. *Optimization*, 14, 1983.

[26] Schittkowski K., Zillober C., Moritzen K. Very large scale optimization by sequential convex programming. *Optimization Methods and Software*, 18(1), 2004.

[27] Schittkowski K., Zillober C., Zotemantel R. Numerical comparison of nonlinear programming algorithms for structural optimization. *Structural Optimization*, 7(1), 1994.

[28] Stingl M. *On the solution of nonlinear semidefinite programs by augmented lagrangian methods.* PhD thesis, Friedrich-Alexander University of Erlangen-Nuremberg, 2005.

[29] Svanberg K. The method of moving asymptotes - a new method for structural optimization. *International Journal for Numerical Methods in Engineering*, 24(2), 1987.

[30] Zillober C. A globally convergent version of the method of moving asymptotes. *Schwerpunktprogramm der deutschen Forschungsgemeinschaft: Anwendungsbezogene Optimierung und Steuerung*, (409), 1992.

[31] Zillober C. A combined convex approximation - interior point approach for large scale nonlinear programming. *Optimization and Engineering*, 2, 2001.

[32] Zillober C. Global convergence of a nonlinear programming method using convex approximations. *Numerical Algorithms*, 27(3), 2001.

[33] Zillober C. SCPIP - an efficient software tool for the solution of structural optimization problems. *Structural and Multidisciplinary Optimization*, 24, 2002.

[34] Zillober C. *Software manual for SCPIP 3.0.* University Bayreuth, 2004.

[35] Zowe J., Kočvara M., Bendsøe M. Free material optimization via mathematical programming. *Mathematical Programming*, 79, 1997.

APPENDIX: Program Documentation

Subroutine calls:

CALL SIMUL	(DOF, ELEMENTS, NLC, NIG)
CALL FGRO1	(U, DUMMY, FORCE, NONGRO, INTGRO, I)
CALL FGRO2	(I, BGES, NONZERO, ROW, COLUMN, VOLELE)

Usage:
These subroutines are part of MOPED, see Kočvara, Stingl and Werner [18] and Hörnlein, Kočvara and Stingl [10], and provide all relevant FMO data.

Definition of parameters:

DOF	degrees of freedom
ELEMENTS	number of elements
NLC	number of load cases
NIG	number of Gauss integration points
U(DOF*NLC)	compliance vector (here: dummy variable of same size)
DUMMY	dummy variable
FORCE(DOF)	force vector for one special load case
NONGRO(DOF)	position of nonzero entries of the force vector
INTGRO	number of nonzero entries in force vector
I	index
BGES(3*4*8)	values of $B_{i,k}$ to compute global stiffness matrix
NONZERO	number of nonzero entries of BGES
ROW(3*4*8)	row number of entry of BGES
COLUMN(3*4*8)	column number of entry of BGES

Subroutine calls:

CALL DERV_GS	(ELEMENTS, AROW, ACOLUMN, ARRAYB, ABLEITA, ABLEITI, SPALTEGES, NN, A,E, LAENGE, NNT)
CALL GS	(ABLEITA, ABLEITI, E, NN, NNT, A, ACOLUMN, AROW, LENGTH, ELEMENTS)

Usage:
These two subroutines compute the values of the global stiffness matrix. As it depends linearly on the design variable E, it is computed once in the subroutine DERV_GS. The derivative is stored in two arrays for the values and the positions. The second subroutine multiplies the stiffness matrix with the actual value of the design variable. To save storage the global stiffness matrix is defined by three vectors, i.e., one for the row numbers, one for the column numbers, and one for the values.

Definition of parameters:

ELEMENTS	number of elements
E(ELEMENTS*6+1)	design variable E and α
ARRAYB(36,3*ELEMENTS)	array of all matrices $B_{i,k}$
SPALTEGES(36*ELEMENTS)	vector of positions of ArrayB

ABLEITA(36,6*ELEMENTS)	array of derivative values of $K(E)$
ABLEITI(36,2*ELEMENTS)	array of positions of derivative of $K(E)$
A(LENGTH)	global stiffness matrix $K(E)$
AROW(LENGTH)	row number of entry A
ACOLUMN(LENGTH)	column number of entry A
LENGTH	length of vector A, AROW, ACOLUMN

Subroutine calls:

```
CALL MA47ID    (SCNTL,SICNTL)
CALL MA47AD    (DOF,LENGTH,AROW,ACOLUMN,IW,LIW,SKEEP,
               SICNTL,SRINFO,SINFO)
CALL MA47BD    (DOF,LENGTH,ACOLUMN,A,LA,IW,LIW,SKEEP,SCNTL,
               SICNTL,IW1,SRINFO,SINFO)
CALL MA47CD    (DOF,A,LA,IW,LIW,W,RHS,IW1,SICNTL)
```

Usage:
These functions solve a system of linear equations given by $K(E)u_j(E) = f_j$, $j = 1,\ldots,l$, see Section 4. For further details see Harwell Subroutine Library (HSL).

Definition of parameters:

A(LENGTH)	values of left hand side $K(E)$
AROW(LENGTH)	row number of value of A
ACOLUMN(LENGTH)	column number of value of A
LENGTH	number of entries of A, AROW, ACOLUMN
RHS(DOF)	right hand side
DOF	system size

Subroutine calls:

```
CALL FUNCTION     (F_ORG, H_ORG, ELEMENTS, N, MIE, E,
                  VOL, ARRAYFORCE, U, DOF, NLC,
                  BETA,SKAL, NN, SKAL2)
CALL GRADIENTS    (ELEMENTS, ABLEITA, IEDERV, DOF, IERN, IECN,
                  NN, U, NLC,ACTIVE, MIE, DF,IELENG, SKAL,
                  NB, ABLEITI, E, VOL, SKAL2)
```

Usage:
These are subroutines for computing function and gradient values of the FMO problem.

Definition of parameters:

F_ORG	objective function value
H_ORG(MIE)	constraint values
ELEMENTS	number of elements
N	number of variables
MIE	number of inequality constraints
E(6*ELEMENTS +1)	optimization variables

VOL	upper bound for volume constraints
ARRAYFORCE(NLC, DOF)	array containing the force vectors
U(NLC*DOF)	displacement vector
DOF	degrees of freedom
NLC	number of load cases
BETA	upper bound for trace constraints
SKAL	scaling factor for compliance constraints
SKAL2	scaling factor for objective function
DF(ELEMENTS*6+1)	derivative of objective function
IELENG	number of nonzero derivatives of constraints
IEDERV(IELENG)	values of derivatives
IERN(IELENG)	row number of derivative items (variable)
IECN(IELENG)	column number of derivative items (constraint)
ACTIVE(MIE)	vector of active constraints
ABLEITA(36,6*ELEMENTS)	array of derivative values of $K(E)$
ABLEITI(36,2*ELEMENTS)	array of derivative positions of $K(E)$

Subroutine call:

```
CALL SCPIP30   (N, MIE, MEQ, IEMAX, EQMAX, X0, X_L, X_U, F_ORG,
               H_ORG,G_ORG, DF, Y_IE, Y_EQ, Y_L, Y_U, ICNTL, RCNTL,
               RINFO, NOUT,R_SCP, RDIM, R_SUB,RSUBDIM, I_SCP, IDIM,
               I_SUB,INFO, ISUBDIM, ACTIVE,MODE, IERR,IERN, IECN,
               IEDERV, IELPAR, IELENG, EQRN, EQCN,EQCOEF, EQLPAR,
               EQLENG, MACTIV, SPIB, SPIBDIM, SPDW, SPDWDIM,
               SPSTRAT, LINSYS)
```

Usage:

SCPIP is an implementation of the sequential convex programming algorithm of Zillober [34]. The result is the optimal solution or, in case of early termination, an error flag.

Definition of parameters:

N	number of variables
MIE	number of inequality constraints
MEQ	number of equality constraints
IEMAX	dimension of inequality dependent arrays
EQMAX	dimension of equality dependent arrays
X0(N)	initial guess of solution, last iterate on return
X_L(N)	lower bounds of the variables
X_U(N)	upper bounds of the variables
F_ORG	objective function value of last iterate
H_ORG(IEMAX)	values of the inequality constraints at last iterate
G_ORG(EQMAX)	values of equality constraints at last iterate
DF(N)	gradient of the objective function at X0
Y_IE(IEMAX)	Lagrange multipliers for inequality constraints at last iterate
Y_EQ(EQMAX)	Lagrange multipliers for equality constraints at last iterate
Y_L(N)	Lagrange multipliers for lower bounds on variables at last iterate

Y_U(N)	Lagrange multipliers for upper bounds on variables at last iterate
ICNTL(14)	integer parameter array
	ICNTL(1): 1=MMA, 2=SCP
	ICNTL(2): strategy of asymptotes
	ICNTL(3): maximal number of iterations
	ICNTL(4): output level
	ICNTL(5): maximal number of line search evaluations
	ICNTL(6): type of convergence check
	ICNTL(7): linear equation solver
	ICNTL(11): warmstart
	ICNTL(14): subproblem solver
RCNTL(6)	double precision parameter array
	RCNTL(1): accuracy
	RCNTL(2): overflow
	RCNTL(3): parameter for active set strategy
INFO(23)	output information array
RINFO(5)	output real information array
NOUT	output unit number
R_SCP(RDIM)	double precision working array of dimension RDIM
RDIM	dimension of double precision working array
R_SUB(RSUBDIM)	double precision working array of dimension RSUBDIM
RSUBDIM	dimension of working array R_SUB
I_SCP(IDIM)	integer working array of dimension IDIM
IDIM	dimension of I_SCP
I_SUB(ISUBDIM)	integer working array of dimension ISUBDIM
ISUBDIM	dimension of I_SUB
ACTIVE(IEMAX)	indices of active inequality constraints
MODE	direct and reverse communication
IERR	error flag
IERN(IELPAR)	row numbers of non-zero entries of Jacobian of inequality constrains
IECN(IELPAR)	column numbers of non-zero entries of Jacobian of inequality constraints
IEDERV(IELPAR)	non-zero entries of Jacobian of inequality constraints
IELPAR	maximum number of non-zero entries of Jacobian of inequality constraints
IELENG	actual number of non-zero entries of Jacobian of inequality constraints
EQRN(EQLPAR)	row numbers of non-zero entries of Jacobian of equality constrains
EQCN(EQLPAR)	column numbers of non-zero entries of Jacobian of equality constraints
EQCOEF(EQLPAR)	non-zero entries of Jacobian of equality constraints
EQLPAR	maximal number of non-zero entries of Jacobian of equality constraints
EQLENG	actual number of non-zero entries of Jacobian of equality constraints
MACTIV	number of active constraints in subproblem model
SPIB(SPIBDIM)	integer working array for solving the subproblem
SPIBDIM	dimension of SPIB

SPDW(SPDWDIM)	double precision working array for solving the subproblem
SPDWDIM	dimension of SPDW
SPSTRAT	subproblem solution strategy
LINSYS	linear system solver

To pass function or gradient values SCPIP30.f works with reverse communication indicated by negative value of IERR. The requested values are then computed in the main program.

-1	compute only function values
-2	compute only gradient values

Subroutine call:

CALL PENNLPF	(N, 0, MIE, NBLOCKS, BLOCKS, N, N,X_L, X_U, LCONSTRAINTS, UCONSTRAINTS, LMBOUNDS, UMBOUNDS,X, Y1,USRF, USRDF, USRHF, USRG, USRDG, USRHG, IOPTIONS, DOPTIONS, IRESULTS,DRESULTS, IERR)

Usage:
PENNLPF is a nonlinear optimization solver implemented in C and is capable to handle semidefinite constraints. In this context PENNLPF is used to solve the convex and separable subproblems formulated by SCPIP. PENNLPF calls six subroutines, i.e., to compute the objective function value, the derivatives of the objective, the Hessian of the objective, the constraint function values, the derivatives of the constraints, and the Hessian of the constraints. These subroutines are implemented in FORTRAN.

Definition of parameters:

N	number of elements
MIE	number of inequality constraints
NBLOCKS	number of blocks in the positive semidefinite matrix (number of elements)
BLOCKS(NBLOCKS)	size of positive semidefinite block matrix (2D: 3, 3D: 6)
LCONSTRAINTS(MIE)	lower bound of constraints (here: -infinity)
UCONSTRAINTS(MIE)	upper bound of constraints (here: 0)
LMBOUNDS(N)	lower bound for variable (here: 3.333e-3)
UMBOUNDS(N)	upper bound for variable (here: infinity)
X(N)	optimization variable
Y1(MIE)	Lagrangian multipliers
USRF	objective function value
USRDF	first order derivative of objective
USRHF	Hessian of objective
USRG	constraint values
USRDG	first order derivatives of constraints
USRHG	Hessian of constraints
IOPTIONS(19)	integer array of settings

DOPTIONS(13)	double array of settings
IERR	default value