Very Large Scale Optimization by Sequential Convex Programming

Authors: Ch. Zillober, K. Schittkowski,
Department of Mathematics
University of Bayreuth
D - 95440 Bayreuth

K. Moritzen
Institute of Applied Mathematics
University of Dortmund
D - 44221 Dortmund

Date: May 2003

Abstract

We introduce a method for constrained nonlinear programming, that is widely used in mechanical engineering and that is known under the name SCP for sequential convex programming. The algorithm consists of solving a sequence of convex and separable subproblems, where an augmented Lagrangian merit function is used for guaranteeing convergence. Originally, SCP methods were developed in structural mechanical optimization, and are particularly applied to solve topology optimization problems. These problems are extremely large and possess dense Hessians of the objective function. The purpose of the paper is to show, that constrained dense nonlinear programs with 10^5 to 10^6 variables can be solved successfully and that SCP methods can be applied also to optimal control problems based on semilinear elliptic partial differential equations after a full discretization.

Keywords: large scale nonlinear programming, sequential convex programming, SCP, topology optimization, optimal control of partial differential equations, semilinear elliptic equations

1 Introduction

We consider the smooth, constrained optimization problem of minimizing a scalar objective function $f(x)$ under nonlinear equality and inequality constraints,

$$x \in I\!\!R^n : \quad \begin{array}{l} \min f(x) \\ g_j(x) = 0 \ , \quad j = 1, \ldots, m_e \ , \\ g_j(x) \leq 0 \ , \quad j = m_e + 1, \ldots, m \ , \\ x_l \leq x \leq x_u \ . \end{array} \tag{1}$$

Moreover, there are lower and upper bounds for the variables, x_l and x_u, respectively. We have m_e equality and $m - m_e$ inequality constraints, which are summarized by the vector $g(x) := (g_1(x), \ldots, g_m(x))^T$. We assume that the functions f and g are continuously differentiable and that the feasible domain of (1) is non-empty.

Despite the success of sequential quadratic programming (SQP) methods, another class of efficient optimization algorithms was proposed mainly by engineers, where the motivation is to optimize mechanical structures. The first method is known under the name CONLIN or convex linearization, see Fleury and Braibant [5] or Fleury [4]. 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.

More general convex approximations are introduced by Svanberg [17] known under the name *method of moving asymptotes* (MMA). The goal is always to construct convex and separable subproblems, for which efficient solvers are available. Thus, we denote this class of methods by SCP, an abbreviation for *sequential convex programming*. The resulting algorithm is very efficient for solving mechanical engineering problems, if a proper starting point is available and if only a crude approximation of the optimal solution needs to be computed because of certain side conditions, for example calculation time or round-off errors in objective function and constraints. 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 [16].

The computer code under investigation is the SCP routine SCPIP, of Zillober [21, 22], a realization of the method of moving asymptotes (MMA). 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. Version 2.3 of SCPIP has been enhanced to use special sparsity information provided by the models.

The general structure of an SCP algorithm and some organizational details are outlined in Section 2, for example the rules by which moving asymptotes are computed. Especially we show how the method can be stabilized to guarantee convergence. Merit function is the augmented Lagrangian function, where violation of constraints is penalized in the L_2-norm. For more information about theoretical convergence results see Zillober [19].

Topology optimization is one of the main domains of applications, where SCP methods are 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, very large nonlinear programs must be solved iteratively, where the number of variables is on the order of 10^5 to 10^6 or even more. In addition, more realistic structures require constraints for each single element leading to a large number of nonlinear inequality constraints on the same order of magnitude. Section 3 contains an outline of the standard power-law approach that allows the construction of scalable test problems for the SCP code under investigation, and some numerical results.

Large nonlinear programming problems are also obtained in a completely different area, the optimal control of partial differential equations. After discretizing state and control variables, equality constrained nonlinear programs are obtained, where finite difference formulae for the state equations lead to large sets of nonlinear equality constraints. In Section 4, we present some numerical results for a series of semilinear elliptic control problems studied by Maurer and Mittelmann [9, 10]. Surprisingly, also these problems can be solved very efficiently by the SCP code SCPIP. To the knowledge of the authors, the solution of problems with a dominating set of equality constraints or of optimal control problems in general by an SCP method has never been tried before.

2 Sequential Convex Programming Methods

Sequential convex programming or SCP methods are developed mainly for mechanical structural optimization. The first approach of Fleury and Braibant [5] and Fleury [4] is known under the name convex linearization (CONLIN), and exploits the observation that in some special cases, typical structural constraints become linear in the inverse variables. We start our investigations from the inequality-constrained nonlinear program

$$x \in I\!\!R^n : \quad \begin{array}{l} \min f(x) \\ g(x) \le 0 \ , \\ x_l \le x \le x_u \ . \end{array} \tag{2}$$

Equality constraints are dropped to simplify the analysis. They are linearized by the method we are interested in, and would not lead to any new insight.

To illustrate the motivation, we consider the most simple example, two bars fixed at a wall and connected at the other end. An external load p is applied at this node, see Figure 1. The two design variables are the cross sectional areas a_i scaled by elasticity modulus E and length l_i, $i = 1, 2$, i.e., $x_i = Ea_i/l_i$. If s_i and c_i denote the sine and cosine function values of the corresponding angles of the trusses, $i = 1, 2$, the horizontal and vertical displacements are given in the form

$$h(x) = |p|(\cos\psi(s_1^2/x_2 + s_2^2/x_1) - \sin\psi(s_1c_1/x_2 + s_2c_2/x_1))/\sin^2(\phi_1 - \phi_2) \ ,$$

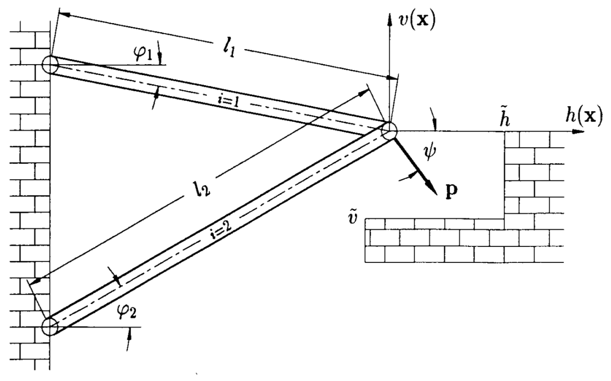

Figure 1: 2-Bar-Truss

$$v(x) = |p|(\sin\psi(c_1^2/x_2 + c_2^2/x_1) - \cos\psi(s_1c_1/x_2 + s_2c_2/x_1))/\sin^2(\phi_1 - \phi_2) \ .$$

If we assume now that our optimization problem consists of minimizing the weight of the structure under some given upper bounds for these displacements, we get nonlinear constraints that are linear in the reciprocal design variables.

Although this special situation is always found in case of statically determinate structures, it is rarely observed in practice. However, a suitable substitution by inverse variables depending on the sign of the corresponding partial derivatives and subsequent linearization is expected to linearize constraints somehow.

For the CONLIN method, Nguyen, Strodiot, and Fleury [12] gave a convergence proof but only for the case that (2) consists of a concave objective function and concave constraints which is of minor practical interest. They also showed that a generalization to non-concave constraints is not possible. More general convex approximations are introduced by Svanberg [17], which are known under the name *method of moving asymptotes* (MMA). The overall goal is to construct nonlinear convex and separable subproblems, for which efficient solvers are available. The choice of the asymptotes influences the curvature of the approximations, and must be adapted carefully to improve the quality of the convex approximation.

Since we assume that (2) is nonconvex and nonlinear in general, the basic idea is to replace (2) by a sequence of *simpler* problems. Starting from an initial vector $x_0 \in I\!R^n$ and an initial multiplier estimate $u_0 \in I\!R^m$, iterates $x_k \in I\!R^n$ and $u_k \in I\!R^m$ are computed successively by solving subproblems of the form

$$y \in I\!R^n : \begin{array}{l} \min f^k(y) \\ g^k(y) \leq 0 \\ y_l^k \leq y \leq y_u^k \ . \end{array} \quad , \tag{3}$$

where f^k and g^k are approximations of the f and g, respectively, and y_l^k, y_u^k are suitable lower and upper bounds of the variables also depending on the current iterate. Let y_k be the optimal solution and v_k the corresponding Lagrangian multiplier of (3). A new iterate is computed by

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k(y_k - x_k) \ , \\ u_{k+1} &= u_k + \alpha_k(v_k - u_k) \ , \end{aligned} \tag{4}$$

where α_k is a steplength parameter discussed subsequently.

Simpler means in this case that the subproblem is solvable by an available *black box* technique, more or less independently of the underlying model structure. In particular, it is assumed that the numerical algorithm for solving (3) does not require any additional function or gradient evaluations of the original ones $f(x)$ and $g_j(x)$, $j = 1, \ldots, m$. The approach indicates also that we are looking for a simultaneous approximation of an optimal solution $x^\star$ and of the corresponding multiplier $u^\star$. Thus, we require that

1. (3) is strictly convex and smooth, i.e., the functions $f^k(x)$ and $g_j^k(x)$ are twice continuously differentiable, $j = 1, \ldots, m$,
2. (3) is a first order approximation of (2) at x_k, i.e., $f(x_k) = f^k(x_k)$, $\nabla f(x_k) = \nabla f^k(x_k)$, $g(x_k) = g^k(x_k)$, and $\nabla g(x_k) = \nabla g^k(x_k)$,
3. the search direction $(y_k - x_k, v_k - u_k)$ is a descent direction for an augmented Lagrangian merit function introduced below.

Strict convexity of (3) means that the objective function $f^k(x)$ is strictly convex and that the constraints $g_j^k(x)$ are convex functions for all iterates k and $j = 1, \ldots, m$. If the feasible domain is non-empty, (3) has a unique solution $y_k \in I\!R^n$ with Lagrangian multiplier $v_k \in I\!R^m$. A further important consequence is that if $y_k = x_k$, then x_k and v_k are at least a stationary point of the general nonlinear programming problem (2).

A line search is introduced to stabilize the solution process, particularly helpful when starting from a bad initial guess. We are looking for an α_k, see (4), $0 < \alpha_k \le 1$, so that a step along a merit function $\Psi_k(\alpha)$ from the current iterate to the new one becomes *acceptable*. The idea is to penalize the Lagrangian function in the L_2 norm, as soon as constraints are violated, by defining

$$\Phi_r(x,u) = f(x) + \sum_{j\in J}\left(u_j g_j(x) + \frac{1}{2} r_j g_j(x)^2\right) - \frac{1}{2}\sum_{j\in K} u_j^2/r_j \ , \tag{5}$$

and we set

$$\Psi_k(\alpha) = \Phi_{r_k}\left(\begin{pmatrix} x_k \\ u_k \end{pmatrix} + \alpha \begin{pmatrix} y_k - x_k \\ v_k - u_k \end{pmatrix}\right) \ , \tag{6}$$

where $J = \{j : g_j(x) \ge -u_j/r_j\}$ and $K = \{1, \ldots, m\} \setminus J$ define the constraints considered as active or inactive, respectively.

The steplength parameter α_k is required in (4) to enforce global convergence of the optimization method, i.e., the approximation of a point satisfying the necessary Karush-Kuhn-Tucker optimality conditions when starting from arbitrary initial values. The merit function (5) is also called augmented Lagrangian function, see for example Rockafellar [14]. The corresponding penalty parameter r_k at the k-th iterate that controls the degree of constraint violation, must be chosen carefully to guarantee a descent direction of the merit function, so that the line search is well-defined,

$$\Psi_k'(0) < -\mu \|y_k - x_k\|^2 \tag{7}$$

with a suitable constant μ. The line search consists of a successive reduction of α starting at 1, usually combined with a quadratic interpolation, until a sufficient decrease condition is obtained. For a more detailed discussion of line search and global convergence aspects, see Ortega and Rheinboldt [13].

The basic idea of the method of moving asymptotes, a special variant of an SCP method, is to linearize f and g_j with respect to transformed variables $(U_i^k - x_i)^{-1}$ and $(x_i - L_i^k)^{-1}$ depending on the sign of the corresponding first partial derivative. U_i^k and L_i^k are reasonable bounds and are adapted by the algorithm after each successful step. Also several other transformations have been developed in the past.

The corresponding approximating functions that define subproblem (3), are

$$\begin{aligned} f^k(y) &= \beta_{0,0}^k + \sum_{i \in I_{0k}^+} \frac{\beta_{i,0}^k(y_i)}{U_i^k - y_i} - \sum_{i \in I_k^-} \frac{\beta_{i,0}^k(y_i)}{y_i - L_i^k} \ , \\ g_j^k(y) &= \beta_{0,j}^k + \sum_{i \in I_{jk}^+} \frac{\beta_{i,j}^k}{U_i^k - y_i} - \sum_{i \in I_{jk}^-} \frac{\beta_{i,j}^k}{y_i - L_i^k} \ , \end{aligned} \tag{8}$$

$j = 1, \ldots, m$, where $y = (y_1, \ldots, y_n)^T$. The index sets are defined by

$$I_{0k}^+ = \{i : 1 \le i \le n, \frac{\partial}{\partial x_i} f(x_k) \ge 0\}$$

and

$$I_{0k}^- = \{i : 1 \le i \le n, \frac{\partial}{\partial x_i} f(x_k) < 0\} \ .$$

In a similar way, I_{jk}^+ and I_{jk}^- are defined. The coefficients $\beta_{i,j}^k$, $j = 0, \ldots, m$, $i = 1, \ldots, n$, are chosen to satisfy the requirements mentioned above, i.e., that (3) is strictly convex and smooth and that (3) is a first order approximation of (2) at x_k,

$$\beta_{0,j}^k := g_j(x_k) - \sum_{i \in I_{jk}^+} \frac{\partial g_j(x_k)}{\partial x_i} \left(U_i^k - x_{k,i}\right) + \sum_{i \in I_{jk}^-} \frac{\partial g_j(x_k)}{\partial x_i} \left(x_{k,i} - L_i^k\right) \tag{9}$$

for $j = 0, \ldots, m$ with $g_0 := f$ for simplicity, and

$$\beta_{i,j}^k := \begin{cases} \dfrac{\partial g_j(x_k)}{\partial x_i} \left(U_i^k - x_{k,i}\right)^2 , & \text{if } i \in I_{jk}^+ \\ \dfrac{\partial g_j(x_k)}{\partial x_i} \left(x_{k,i} - L_i^k\right)^2 , & \text{if } i \in I_{jk}^- \end{cases} \tag{10}$$

for $j = 1, \ldots, m$. The second subindex i of $x_{k,i}$ indicates the i-th coefficient of x_k, $i = 1, \ldots, n$.

To ensure strict convexity of the approximation of the objective function, we introduce additional positive parameters τ_i^k and get a regularized subproblem of the form

$$\beta_{i,0}^k(y_i) := \begin{cases} \dfrac{\partial f(x_k)}{\partial x_i} \left(U_i^k - x_{k,i}\right)^2 + \tau_i^k \left(y_i - x_{k,i}\right)^2 , & \text{if } i \in I_{0k}^+ \\ \dfrac{\partial f(x_k)}{\partial x_i} \left(x_{k,i} - L_i^k\right)^2 - \tau_i^k \left(y_i - x_{k,i}\right)^2 , & \text{if } i \in I_{0k}^- \end{cases} . \tag{11}$$

Regularization of the objective function guarantees strict convexity of $f^k(x)$, see Zillober [19]. As shown there, the search direction $(y_k - x_k, v_k - u_k)$ is a descent direction for the augmented Lagrangian merit function (5). The update rule for the penalty parameter r_k is the same as proposed by Schittkowski [15] for an SQP method. If the adaptation rule for the parameters L_i^k and U_i^k guarantees that the absolute value of their differences from the current iteration point x_k is uniformly bounded away from 0 and that their absolute value is bounded, global convergence can be shown.

The choice of the asymptotes L_i^k and U_i^k, is crucial for the computational behavior of the method. An efficient update scheme for the i-th coefficient, $i = 1, \ldots, n$, and the k-th iteration step is given as follows:

$$\begin{aligned} k = 0, 1 \ : \ & L_i^k = x_{k,i} - \lambda_1 (x_{u,i} - x_{l,i}) \ , \\ & U_i^k = x_{k,i} + \lambda_1 (x_{u,i} - x_{l,i}) \ . \end{aligned}$$

$$\begin{aligned} k = 2, 3, \ldots \ : \ & \text{If } \operatorname{sign}(x_{k,i} - x_{k-1,i}) = \operatorname{sign}(x_{k-1,i} - x_{k-2,i}) \ : \\ & \quad L_i^k = x_{k,i} - \lambda_2 (x_{k-1,i} - L_i^{k-1}) \ , \\ & \quad U_i^k = x_{k,i} + \lambda_2 (U_i^{k-1} - x_{k-1,i}) \ . \\ & \text{If } \operatorname{sign}(x_{k,i} - x_{k-1,i}) \neq \operatorname{sign}(x_{k-1,i} - x_{k-1,i}) \ : \\ & \quad L_i^k = x_{k,i} - \lambda_3 (x_{k-1,i} - L_i^{k-1}) \ , \\ & \quad U_i^k = x_{k,i} + \lambda_3 (U_i^{k-1} - x_{k-1,i}) \ . \end{aligned}$$

A suitable choice of the constants is $\lambda_1 = 0.5$, $\lambda_2 = 1.15$, $\lambda_3 = 0.7$. If there is no change in the sign of a component of two successive iterations, this situation is interpreted as *smooth* convergence and allows a relaxation of the asymptotes. If there are sign changes between two successive iterations, we are afraid of cycling and the asymptotes remain closer to an iteration point leading to more conservative approximations.

Additional safeguards ensure the compatibility of this procedure with the overall scheme and guarantee global convergence. A small positive constant is introduced to avoid that the difference between the asymptotes and the current iteration point becomes too small. Moreover, absolute bounds are attached such that the asymptotes cannot tend to infinity. However, these safeguards are rarely used in practice, see Zillober [19] for more details. Note that the occurrence of cycling is also reduced by the line search procedure.

For the first SCP codes developed, the convex and separable subproblems are solved very efficiently by a dual approach, where dense linear systems of equations with m rows and columns are solved, see Svanberg [17] or Fleury [4]. An interior point method for the solution of the subproblems is proposed by Zillober [20]. The advantage is to be able formulate either $n \times n$ or $m \times m$ linear systems of equations leading to a more flexible treatment of large problems.

Since the interior point method is an infeasible primal-dual method, there are no further restrictions on the starting values for the SCP method. The interior point condition applies only for artificial variables, cf. Zillober [20]. The resulting algorithm is very efficient especially for large scale mechanical engineering problems, when sparsity patterns of the original problem data can be exploited. To summarize, the most important features of SCP methods are that

- very large scale problems can be solved,
- the algorithm is globally convergent,
- a large number of constraints can be treated by an active set strategy, see Zillober [21],
- sparsity of the Jacobian of the constraints can be exploited,
- bounds of variables and linear equality constraints remain satisfied.

If the constraints of (3) become inconsistent, it is possible to introduce an additional variable z and to modify objective function and constraints, for example

$$y \in \mathbb{R}^n, z \in \mathbb{R}: \quad \begin{array}{l} \min f^k(y) + \rho_k z \\ g^k(y) - z \leq 0 \ , \\ y_l^k \leq y \leq y_u^k \ , \\ z \geq 0 \end{array} \tag{12}$$

in the simplest form. The penalty term ρ_k is added to the objective function to reduce the influence of the additional variable z as much as possible. The index k implies that this parameter also needs to be updated during the algorithm. It is obvious that (12) always possesses a feasible solution.

3 Topology Optimization

A typical application of an SCP algorithm is the minimization of the weight of a mechanical structure under certain loads and constraints for admissible stresses, displacements, or dynamic responses. Highly complex industrial and academic design problems are solved today by means of nonlinear programming algorithms without any chance to get equally qualified results by traditional empirical approaches, see for example Schittkowski, Zotemantel, and Zillober [16] for comparative results and a list of 79 test problems based on a finite element formulation, confer also Kneppe, Krammer, and Winkler [8]. In Zillober and Vogel [23, 24], industrial applications of the SCP code SCPIP of Zillober ([21]),[22]) are found.

To give an impression about the capabilities of an SCP implementation for solving very large scale nonlinear programming problems, we consider now the area of mechanical topology optimization. Given a predefined domain in the 2D/3D space with boundary conditions and external load, the intention is to distribute a percentage of the initial mass on the given domain such that a global measure takes a minimum, see Bendsøe and Sigmund [2] for a broader introduction. Assuming isotropic material, the so-called power law approach, see also Bendsøe [1] or Mlejnek [11], leads to a nonlinear programming problem of the form

$$x \in I\!R^n,\ u \in I\!R^d : \quad \begin{array}{l} \min\ u^T p \\ V(x) \leq aV_0 \ , \\ K(x)u = p \ , \\ 0 < x_l \leq x \leq 1 \ , \end{array} \tag{13}$$

where $x = (x_1, \ldots, x_n)^T$ denotes the relative material densities, that are artificially introduced variables. In the final solution, we consider a small value of x_i as zero or no mass, a larger value as one or full mass. Theoretically, one is only interested in 0-1 solutions, which are not guaranteed by the continuous approach applied. $u = (u_1, \ldots, u_d)^T$ is the displacement vector computed from the linear system of equations $K(x)u = p$ with a positive definite stiffness matrix $K(x)$ and an external load vector p. d denotes the number of degrees of freedom of the structure. We assume without loss of generality that there is only one load case. The goal is to minimize the so-called compliance or, in other words, to make the structure as stiff as possible.

It is essential to understand that the system of linear equations $K(x)u = p$ can be considered as the state equations of our optimization problem. Thus, (13) is also written in the form

$$x \in I\!R^n : \quad \begin{array}{l} \min\ p^T K(x)^{-1} p \\ V(x) \leq aV_0 \ , \\ 0 < x_l \leq x \leq 1 \ , \end{array} \tag{14}$$

at least conceptually. In practice, however, we assume that finite element simulation software is available to set up the stiffness matrix and to solve the system $K(x)u = p$ internally. To indicate that u depends on the relative densities x, we use the notation $u(x)$.

The relative densities and the elementary stiffness matrices K_i define $K(x)$ by

$$K(x) = \sum_{i=1}^{n} x_i^q\ K_i \ .$$

$V(x)$ is the volume of the structure, usually a linear function of the design variables,

$$V(x) = \sum_{i=1}^{n} x_i V_i \quad ,$$

where V_i is the volume of the i-th finite element. V_0 is the available volume, $V_0 = \sum_{i=1}^{n} V_i$, and a with $0 < a < 1$ the given fraction of the full volume to distribute the available mass.

x_l is a vector of small positive numbers for avoiding singularities. The nonlinearity x_i^q in the state equation is found heuristically and usually applied in practice with $q = 3$. Its role is to penalize intermediate values between the lower bound and 1.

The partial derivatives of the objective function of problem (13) or (14), respectively, are computed from

$$\frac{\partial}{\partial x_j}\Big(u(x)^T p\Big) \quad = \quad -q x_j^{q-1} u(x)^T K_j u(x) \tag{15}$$

for $j = 1, \ldots, n$. Since the elementary stiffness matrices K_j are very sparse, for example containing only non-zero entries on an $8 \times 8-$submatrix in case of the rectangular elements used in this section, the j-th partial derivative is computed very efficiently as soon as the displacement vector $u(x)$ is available. (15) follows from the identity $K(x)u(x) = p$ with a constant external force p, and

$$\begin{aligned} \frac{\partial}{\partial x_j}\Big(K(x)u(x)\Big) \quad &= \quad 0 \\ &= \quad q x_j^{q-1} K_j u(x) + K(x) \frac{\partial}{\partial x_j} u(x) \end{aligned} \tag{16}$$

leading to

$$\begin{aligned} p^T \frac{\partial}{\partial x_j} u(x) \quad &= \quad -q x_j^{q-1} p^T K(x)^{-1} K_j u(x) \\ &= \quad -q x_j^{q-1} u(x)^T K_j u(x) \end{aligned} \tag{17}$$

In addition to the standard topology optimization problem (13) or (14) with only one constraint for the volume, we consider also more realistic problems containing bounds for the local compliances

$$c_i(x) := x_i^q u(x)^T K_i u(x) \quad , \tag{18}$$

$i = 1, \ldots, n$. If bounds are given by some suitable values $\overline{c}_i$, we obtain constrained nonlinear programs of the form

$$x \in I\!\!R^n : \quad \begin{array}{l} \min \; p^T K(x)^{-1} p \\ c_i(x) \leq \overline{c}_i \;, \quad i = 1, \ldots, n \quad , \\ V(x) \leq a V_0 \quad , \\ 0 < x_l \leq x \leq 1 \quad , \end{array} \tag{19}$$

Derivatives for local compliances are obtained from

$$\begin{aligned}\frac{\partial}{\partial x_j} c_i(x) &= \frac{\partial}{\partial x_j}\left(x_i^q u(x)^T K_i u(x)\right) \\ &= \delta_{ij} q x_j^{q-1} u(x)^T K_j u(x) + 2x_i^q u(x)^T K_i \frac{\partial}{\partial x_j} u(x) \\ &= \delta_{ij} q x_j^{q-1} u(x)^T K_j u(x) - 2q x_j^{q-1} x_i^q u(x)^T K_i K(x)^{-1} K_j u(x) \ .\end{aligned} \tag{20}$$

Here, δ_{ij} represents the Kronecker symbol, and the last equation follows from (17). Since $K_j u(x)$ is a very sparse vector and can be computed in advance for all j, and since a Cholesky decomposition of $K(x)$ is known from the solution of $K(x)u = p$, the calculation of the j-th partial derivative of the local compliance can be organized in an efficient way, but still requires substantial amount of computational work.

The solution of topology optimization problems easily leads to very large scale, highly nonlinear programs. The probably most simple example is a beam, which is loaded in the middle and supported at the two lower vertices. The design region is a rectangular plate, see Figure 2, discretized by rectangular finite elements. For symmetry reasons, we consider only one half of the beam for our calculations. The number of horizontal grid lines is denoted by n_x, the number of vertical grid lines by n_y.

The solution of topology optimization problems as outlined so far, produces a strange phenomenon, so-called checkerboards. In some areas, the 0-1 distribution is structured like a checkerboard in certain areas. Thus, an additional filter is applied by which partial derivatives are modified by certain weights depending on a given radius around the considered element, see for example Bendsøe and Sigmund [2]. Moreover, the final structure depends heavily on the volume fraction a used to restrict the mass distribution, see (14). In Figure 3, a series of final beam structures is listed depending on increasing filter size ranging from 0, 4, 8, 12, and 16 in rows and increasing mass distribution values for a in columns from $a = 0.30$ to $a = 0.70$, where 150 elements are defined in x- and 100 elements in y-direction, altogether 15,000 2D-elements or relative densities, respectively. We start with a maximum feasible distribution of mass, i.e. $x_i = a$ for all elements. The numerical calculations are performed on a SUN Fire V880 with eight processors running under 750 MHz and 16 GB memory. More detailed results are shown in Table 1, where the following abbreviations are used,

n_x, n_y	-	number of elements in x- and y-direction,
n	-	number of optimization variables,
n_{it}	-	number of iterations,
$f(x)$	-	final objective function value,
$\|\nabla_x L(x,u)\|$	-	norm of final gradient of Lagrangian function,
r_f	-	filter radius.

As an example for an optimization problem of the form (19) with a large number of constraints, we consider the beam structure of Figure 4. The struc-

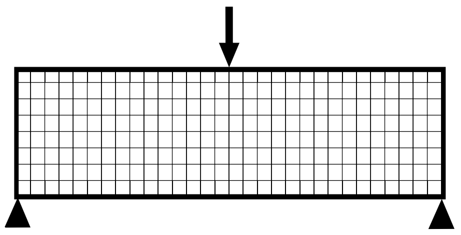

Figure 2: Design Region of Beam

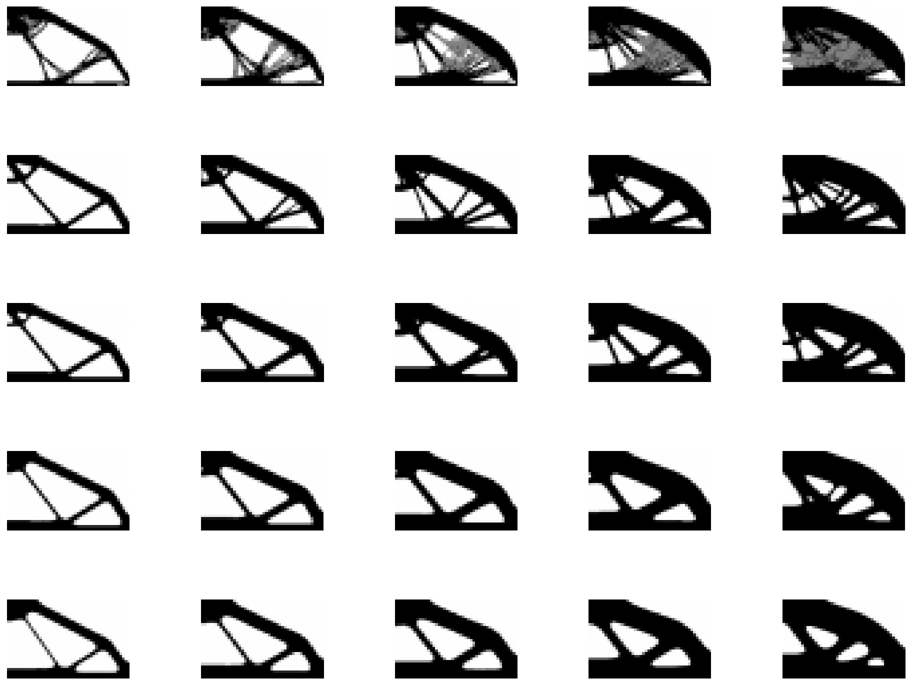

Figure 3: Half Beam with Varying Volume Restrictions and Filter Radii

n_x	n_y	n	n_{it}	$f(x)$	$\|\nabla_x L(x,u)\|$	r_f
600	400	240,000	22	52.63	1.3E-3	8
600	400	240,000	26	54.25	6.5E-4	0
1,050	700	735,000	38	54.39	4.6E-4	10
1,260	840	1,058,400	43	56.55	1.3E-5	0

Table 1: Numerical Results of SCPIP for the Half Beam

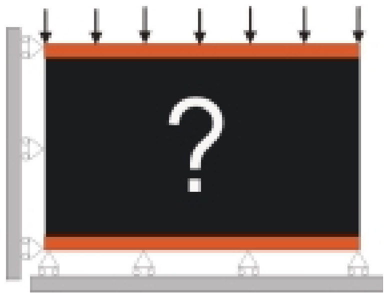

Figure 4: Design Region of Locally Constrained Beam

n_x	n_y	n	n_{it}	$f(x)$	$\|\nabla_x L(x,u)\|$	r_f
270	180	48600	40	1590.2	7.7E-2	9

Table 2: Numerical Results of SCPIP for the Beam with Local Constraints

ture is fixed in $y-$direction at the left edge and in $x-$direction at the lower right vertex. At the top and at the bottom, there is a horizontal region of fixed material which is not allowed to be changed. The load applies equally distributed at the top of the structure. We use a discretization of 270×180 finite elements such that the optimization problem (19) has 48,600 variables and 48,601 constraints besides box-constraints. The parameter a in (19) is 0.6 in this case which includes 10% of material for the fixed regions. Starting point is a region of full material, i.e. $x_i = 1$ for all finite elements, which violates the volume constraint but ensures that all local constraints are fulfilled as long as the corresponding bounds $\overline{c_i}$ are suitable.

Active constraints occur locally in regions of high stresses. Application of a simple active set strategy reduces the number of constraints that must be considered in the convex subproblems. In our example, we do not get more than 69 constraints that are considered to be active. The idea is to define all constraints as active ones that are active or violated in (19) subject to a small tolerance. Moreover, constraints are considered to become active when the corresponding Lagrangian multiplier is positive.

Some numerical results for the constrained case are shown in Figure 5 and in Table 2. It is worthwhile to mention that the final convergence of the method is quite slow in this case. In a practical situation, relaxed convergence checks are applied. The iteration cycle is stopped for example, if the relative progress in the objective function is less than 1% while retaining feasibility. A weaker criterion would reduce the number of iterations, but has not been applied.

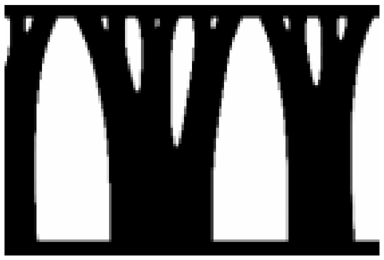

Figure 5: Optimal Topology of Locally Constrained Beam

4 Optimal Control of Semilinear Elliptic Partial Differential Equations

The intention behind the numerical tests of this section is to show that SCP methods can be applied also to optimization problems which are completely different from the original mechanical engineering applications. We consider a series of test problems investigated by Maurer and Mittelmann [9, 10] when studying necessary optimality conditions for optimal control of elliptic partial differential equations with state and control constraints.

Proceeding from the integration area

$$\Omega := \{x = (x_1, x_2) \in \mathbb{R}^2 : 0 < x_1, x_2 < 1\} \quad ,$$

i.e., the unit square, and the corresponding boundary

$$\Gamma := \{x = (x_1, x_2) \in \mathbb{R}^2 : x_1 = 0 \text{ or } x_1 = 1 \text{ or } x_2 = 0 \text{ or } x_2 = 1\} \quad ,$$

the optimal control problem is

$$u \in L^\infty(\overline{\Omega}), y \in C^2(\overline{\Omega}) : \begin{array}{l} \min \ \frac{1}{2} \displaystyle\int_\Omega (y(x) - y_d(x))^2 dx + \frac{\alpha}{2} \int_\Lambda u(x)^2 dx \\ -\Delta y + d(x, y, u) = 0 \ , \ \ x \in \Omega \ \ , \\ \ \ b(x, \partial_\nu y, y, u) = 0 \ , \ \ x \in \Gamma \ \ , \\ \ \ y \leq \overline{y} \ , \ \ \underline{u} \leq u \leq \overline{u} \ \ , \end{array} \tag{21}$$

$\overline{\Omega} = \Omega \cup \Gamma$. Here, u is the control function we want to compute subject to constant lower and upper bounds $\underline{u}$, $\overline{u}$, and y denotes the state variable, i.e., the solution of the semilinear elliptic partial differential equation $\Delta y = d(x, y, u)$ subject to either a Dirichlet or Neumann boundary condition of the form $b(x, \partial_\nu y, y, u) = 0$. $\partial_\nu y$ denotes the outward unit normal along the boundary Γ. The solution of the state equations depends on the control function u and a state constraint for y is given in form of upper constant bounds $\overline{y}$. The

name	$d(y,u)$	$y_d(x)$	$b(x, \partial_\nu y, y, u)$
ELL_1	-20	$3 + 5x_1(x_1-1)x_2(x_2-1)$	$y - u$
ELL_2	-20	$3 + 5x_1(x_1-1)x_2(x_2-1)$	$y - u$
ELL_3	-20	$3 + 5x_1(x_1-1)x_2(x_2-1)$	$y - u$
ELL_4	-20	$3 + 5x_1(x_1-1)x_2(x_2-1)$	$y - u$
ELL_5	0	$2 - 2(x_1(x_1-1) + x_2(x_2-1))$	$\partial_\nu y - u + y^2$
ELL_6	0	$2 - 2(x_1(x_1-1) + x_2(x_2-1))$	$\partial_\nu y - u + y^2$
ELL_7	$y^3 - y$	$2 - 2(x_1(x_1-1) + x_2(x_2-1))$	$\partial_\nu y - u$
ELL_8	$y^3 - y$	$2 - 2(x_1(x_1-1) + x_2(x_2-1))$	$\partial_\nu y - u$
ELL_9	$y^3 - y - u$	$1 + 2(x_1(x_1-1) + x_2(x_2-1))$	y
ELL_10	$y^3 - y - u$	$1 + 2(x_1(x_1-1) + x_2(x_2-1))$	y
ELL_11	$-e^y - u$	$\sin(2\pi x_1)\sin(2\pi x_2)$	y
ELL_12	$-e^y - u$	$\sin(2\pi x_1)\sin(2\pi x_2)$	$\partial_\nu y + y$
ELL_13	$-e^y - u$	$\sin(2\pi x_1)\sin(2\pi x_2)$	$\partial_\nu y + y$

Table 3: Elliptic Control Problems: Functions

cost function is of tracking type, see Ito and Kunisch [7], and given by the space-dependent function $y_d(x)$. α is a suitable weighting factor leading to a bang-bang control if set to zero. In case of distributed control, the second integral is taken over the whole unit interval, i.e., $\Lambda = \Omega$, and in case of boundary control, integration is performed over the boundary, i.e., $\Lambda = \Gamma$.

Our set of test problems is completely described by the functions $y_d(x)$, $d(x, y, u)$, and $b(x, \partial_\nu y, y, u)$, see Table 3, and the data α, $\overline{y}$, $\underline{u}$, and $\overline{u}$, see Table 4. Note that Maurer and Mittelmann [9, 10] study a more general class of optimal control problems, but all of their test problems are of the type (21). The first eight problems possess boundary control, the subsequent ones distributed control functions. Example ELL_1 is taken from Bergounioux and Kunisch [3], and examples ELL_9 and ELL_10 are related to a simplified Ginzburg-Landau equation, see Ito and Kunisch [7].

The elliptic control problem (21) is fully discretized subject to the control and state variables, and we apply the same discretization technique proposed by Maurer and Mittelmann [9, 10]. First, the unit square is discretized by a uniform mesh of size N, to get mesh points

$$x_{i,j} := (ih, jh) \quad , i = 0, \ldots, N+1 \ , \quad j = 0, \ldots, N+1$$

with $h := \frac{1}{N+1}$. For each grid point, we get a discretized control variable u_{ij} and a discretized state variable y_{ij}, where either $i = 0, \ldots, N+1$, $j = 0, \ldots, N+1$, or $i = 1, \ldots, N$, $j = 1, \ldots, N$ depending on the type of the boundary values, see below. The Laplace operator is discretized by the five-point-star leading to

name	α	$\overline{y}$	$\underline{u}$	$\overline{u}$
ELL_1	0.01	-	0.0	10.0
ELL_2	0.0	-	0.0	10.0
ELL_3	0.01	3.2	1.6	2.3
ELL_4	0.0	3.2	1.6	2.3
ELL_5	0.01	2.071	3.7	4.5
ELL_6	0.0	2.835	6.0	9.0
ELL_7	0.01	2.7	1.8	2.5
ELL_8	0.0	2.7	1.8	2.5
ELL_9	0.001	0.185	1.5	4.5
ELL_10	0.0	0.185	1.5	4.5
ELL_11	0.001	0.11	-5.0	5.0
ELL_12	0.001	0.371	-8.0	9.0
ELL_13	0.0	0.371	-8.0	9.0

Table 4: Elliptic Control Problems: Data

	$N+1=100$		$N+1=200$	
name	n	m	n	m
ELL_1	10197	9801	40397	39601
ELL_2	10197	9801	40397	39601
ELL_3	10197	9801	40397	39601
ELL_4	10197	9801	40397	39601
ELL_5	10593	10197	41193	40397
ELL_6	10593	10197	41193	40397
ELL_7	10593	10197	41193	40397
ELL_8	10593	10197	41193	40397
ELL_9	19602	9801	79202	39601
ELL_10	19602	9801	79202	39601
ELL_11	19602	9801	79202	39601
ELL_12	19998	10197	79998	40397
ELL_13	19998	10197	79998	40397

Table 5: Elliptic Control Problems: Dimensions

a set of N^2 equality constraints

$$4y_{ij} - y_{i,j-1} - y_{i-1,j} - y_{i,j+1} - y_{i+1,j} + h^2 d(x_{ij}, y_{ij}, u_{ij}) = 0 \ . \tag{22}$$

First derivatives in Neumann boundary conditions are approximated by forward or backward differences, respectively, and Dirichlet boundary conditions are directly inserted either by a constant value or the control variable. Thus, the number of variables, n, and the number of equality constraints, m, depend on the structure of the boundary values and are on the order of N^2. The size of the discretized state equations, m, is identical to the number of discretized state variables and is also on the order of N^2. Table 5 contains the problem dimensions of the resulting nonlinear programming problems, see also Maurer and Mittelmann [9, 10] for details. To compute the objective function or cost functional, respectively, we replace the integrals in (21) by straightforward approximations at the grid points, i.e., by

$$\begin{aligned} f^h(u,y) \ &:= \ \frac{h^2}{2} \sum_{i,j=1}^{N} (y_{ij} - y_d(ih, jh))^2 \\ &\quad + \frac{h\alpha}{2} \left(\sum_{i=1}^{N} u_{i0}^2 + \sum_{i=1}^{N} u_{i,N+1}^2 + \sum_{j=1}^{N} u_{0j}^2 + \sum_{j=1}^{N} u_{N+1,j}^2 \right) \end{aligned} \tag{23}$$

in case of boundary control and

$$f^h(u,y) := \frac{h^2}{2} \sum_{i,j=1}^{N} (y_{ij} - y_d(ih, jh))^2 + \frac{h^2\alpha}{2} \sum_{i,j=1}^{N} u_{ij}^2 \tag{24}$$

in case of distributed control.

After applying the discretization procedure, the nonlinear program is of the form

$$\begin{array}{rl} & \min f^h(u,y) \\ u \in I\!R^n, y \in I\!R^m : & g^h(u,y) = 0 \ , \\ & \underline{u} \le u \le \overline{u} \ , \ \ y \le \overline{y} \ , \end{array} \tag{25}$$

with distributed parameters u and y and m nonlinear equality constraints, $g^h(u,y) = (g_1^h(u,y), \ldots, g_m^h(u,y))^T$. It is important to know that SCP methods were not invented and investigated before to solve equality constrained problems. Convex approximation cannot be applied to equality constraints, which are linearized by SCPIP internally, see Zillober [20].

Problem (25) is solved by the SCP code SCPIP with termination accuracy $\epsilon = 10^{-7}$ for KKT condition and $\epsilon = 10^{-10}$ for maximum constraint violation. Starting values are $u_0 = 0$ and $y_0 = 0$ for all test runs. For our numerical experiments, we use the Compaq Visual Fortran Optimizing Compiler, Version 6.5, with double precision arithmetic. The numerical results are obtained on a PC running under Windows NT 4.0 with a Pentium II processor (450 MHz) which is comparable to the hardware that has been used by Maurer and Mittelmann [9, 10] for their tests. Tables 6 and 7 summarize the obtained performance data, where we use the notation

N	-	number of mesh intervals,
n_{it}	-	number of iterations until convergence,
$f(u,y)$	-	final objective function value.
CPU	-	CPU time until termination.
n_{it}^L	-	number of LOQO iterations until convergence,
$f(u,y)^L$	-	final objective function value obtained by LOQO.
CPU^L	-	CPU time of LOQO until termination.

As far as available, we present also results published by Maurer and Mittelmann [9, 10]. The entry *n.r.* stands for *not reported.* We show number of iterations, final objective function value and CPU-time only for the code LOQO, version 4.01, since this code turned out to be the most reliable and efficient one in the comparative evaluation of the authors. LOQO is an infeasible primal-dual interior point algorithm, see Vanderbei and Shanno [18]. Essentially, the KKT optimality conditions are formulated and solved by a path-following variant of Newton's method. The starting values are the same used for SCPIP, $u_0 = 0$ and $y_0 = 0$. In the study of Maurer and Mittelmann [9, 10], LOQO is stopped if there are at least eight correct digits in the objective function value. Thus, we are unable to compare also the convergence accuracy of both methods, since the termination criteria of SCPIP are based on the KKT-conditions.

The results show that the calculation times of SCPIP are higher than those of LOQO for problems ELL_1 to ELL_8 and significantly lower for problems ELL_9 to ELL_13. However, problem functions for LOQO are provided in the AMPL modeling language, see Fourer et al. [6], automatic differentiation is applied for computing derivatives, and internal numerical calculations are quite different, in particular the solution of systems of linear equations. For SCPIP functions and gradients are coded directly in Fortran. Thus, a direct comparison of calculation times must be done very carefully.

A star indicates that the corresponding problem could not be solved by the default parameters. By adapting solution tolerances, also these problems can be solved successfully and the performance results are reported. The objective function values are equal or better than the values reported for LOQO. Note that one iteration corresponds to one evaluation of all gradients. Additional function evaluations are only needed in case of performing a linesearch, but appear so seldom that we omit listing them.

In the discretized problem formulation (25), problems ELL_1 to ELL_4 possess strictly convex quadratic objective functions and linear constraints. It should be stressed that the authors would never recommend SCPIP for this situation. Problems ELL_5 and ELL_6 have additionally convex quadratic constraints. Additional nonlinear and nonquadratic components are added with increasing problem number. Thus, from the point of view of optimization, the

name	n_{it}	$f(u,y)$	CPU		n_{it}^L	$f(u,y)^L$	CPU^L
ELL_1	20	0.196525	476		26	0.196525	96
ELL_2	23	0.096695	600		*n.r.*	0.096695	78
ELL_3	43	0.321010	940		*n.r.*	0.321010	103
ELL_4	20	0.249178	471		*n.r.*	0.249178	116
ELL_5	23	0.552246	531	$\star$	*n.r.*	0.553324	494
ELL_6	17	0.015079	318	$\star$	*n.r.*	0.015078	864
ELL_7	123	0.263910	2173	$\star$	*n.r.*	0.264163	317
ELL_8	103	0.161664	1963	$\star$	*n.r.*	0.165531	570
ELL_9	8	0.0621646	152		30	0.0621615	1897
ELL_10	4	0.0564540	91		37	0.0564474	2169
ELL_11	11	0.110266	212		32	0.110263	2257
ELL_12	26	0.0780640	519		23	0.0780638	1325
ELL_13	20	0.0527913	494		60	0.0544745	5735

Table 6: Elliptic Control Problems: Numerical Results for $N+1 = 100$

higher the number of the problem, the harder the problem is to solve. Compared to LOQO, SCPIP performs better and better with increasing complexity of the problems. A possible reason is the specific way how sparse Jacobians are handled inside the algorithm.

Table 8 contains some results of SCPIP when applied to problem ELL_13 with increasing mesh refinement, i.e., increasing dimensions of the optimization problem. We observe that the total number of iterations does not increase with the number of variables. These results are obtained on a PC running under Windows 2000 with a Pentium III processor (750 MHz). Since no results are reported for more than 200 mesh intervals for LOQO we omit CPU-times in this table.

5 Conclusions

We introduce sequential convex programming methods, which do not update any second-order information. Strictly convex and separable subproblems are formulated from which a suitable search direction with respect to the design variables and multiplier estimates are computed by an interior point method. A subsequent line search based on the augmented Lagrangian merit function stabilizes the optimization algorithm and allows to prove global convergence. Starting from an arbitrary initial design, a stationary point satisfying the necessary Karush-Kuhn-Tucker conditions is approximated.

To show that these methods have the potential to solve very large scale optimization problems, we apply the implementation SCPIP of Zillober ([21],[22]) to topology optimization problems and to optimal control problems for semilinear elliptic partial differential equations. It is shown that an SCP method is capable of solving dense optimization problems with more than 10^6 variables.

name	n_{it}	$f(u,y)$	CPU		n_{it}^L	$f(u,y)^L$	CPU^L
ELL_1	16	0.2007716	3751		29	0.200772	1477
ELL_2	24	0.1004422	6164		*n.r.*	*n.r.*	*n.r.*
ELL_3	63	0.3281215	12220		*n.r.*	*n.r.*	*n.r.*
ELL_4	26	0.2558766	7246		*n.r.*	*n.r.*	*n.r.*
ELL_5	27	0.5543687	5367	$\star$	*n.r.*	*n.r.*	*n.r.*
ELL_6	24	0.0156019	4403	$\star$	*n.r.*	*n.r.*	*n.r.*
ELL_7	103	0.2671222	20163	$\star$	*n.r.*	*n.r.*	*n.r.*
ELL_8	153	0.1657739	31122	$\star$	*n.r.*	*n.r.*	*n.r.*
ELL_9	5	0.0645308	675		35	0.0644259	54831
ELL_10	4	0.0587030	762		45	0.0596968	67769
ELL_11	9	0.1103035	1286		31	0.1102690	42644
ELL_12	23	0.0784266	3591		25	0.0784259	43640
ELL_13	22	0.0529923	4479		84	0.0547818	137514

Table 7: Elliptic Control Problems: Numerical Results for $N + 1 = 200$

name	$N+1$	n	m	n_{it}	$f(u,y)$	n_{it}^L	$f(u,y)^L$
ELL_13	100	19,998	10,197	20	0.0528	60	0.0545
	200	79,998	40,397	22	0.0530	84	0.0548
	300	179,998	90,597	16	0.0535	*n.r.*	*n.r.*
	400	319,998	160,797	18	0.0534	*n.r.*	*n.r.*
	500	499,998	250,997	16	0.0536	*n.r.*	*n.r.*
	600	719,998	361,197	16	0.0537	*n.r.*	*n.r.*

Table 8: Elliptic Control Problems: Numerical Results for ELL_13

In the second case, we get a large number of equality constraints in the discretized nonlinear program. Although SCP methods are not particularly tuned to this situation, the code solves problems with about 720,000 variables and about 360,000 equality constraints on a standard PC. It is shown that the code SCPIP is at least as efficient as the code LOQO of Vanderbei and Shanno [18], which turned out to be the best one in the comparative studies of Maurer and Mittelmann [9, 10].

References

[1] Bendsøe M.P. (1989): *Optimal shape design as a material distribution problem*, Structural Optimization, Vol. 1, 193-202

[2] Bendsøe M.P., Sigmund, O. (2003): *Topology Optimization — Theory, Methods and Applications*, Springer, Heidelberg

[3] Bergounioux M., Kunisch K. (1997): *Augmented Lagrangian techniques for elliptic state constrained optimal control problems*, SIAM Journal on Optimization, Vol. 35, 1524-1543

[4] Fleury C. (1989): *An efficient dual optimizer based on convex approximation concepts*, Structural Optimization, Vol. 1, 81-89

[5] Fleury C., Braibant V. (1986): *Structural Optimization – a new dual method using mixed variables*, International Journal for Numerical Methods in Engineering, Vol. 23, 409-428

[6] Fourer R., Gay D.M., Kernighan B.W. (1993): *AMPL: A Modelling Language for Mathematical Programming*, Duxbury Press, Brooks-Cole Publishing Company

[7] Ito K., Kunisch K. (1996): *Augmented Lagrangian-SQP methods for nonlinear optimal control problems of tracking type*, SIAM Journal on Optimization, Vol. 6, 96-125

[8] Kneppe G., Krammer J., Winkler E. (1987): *Structural optimization of large scale problems using MBB-LAGRANGE*, Report MBB-S-PUB-305, Messerschmitt-Bölkow-Blohm, Munich

[9] Maurer H., Mittelmann H.D. (2000): *Optimization techniques for solving elliptic control problems with control and state constraints: Part I. Boundary control*, Computational Optimization and Applications, Vol. 16, 29-55

[10] Maurer H., Mittelmann H.D. (2001): *Optimization techniques for solving elliptic control problems with control and state constraints: Part II. Distributed control*, Computational Optimization and Applications, Vol. 18, 141-160

[11] Mlejnek H.P. (1992): *Some aspects of the genesis of structures*, Structural Optimization, Vol. 5, 64-69

[12] Nguyen V.H., Strodiot J.J., Fleury C. (1987): *A mathematical convergence analysis for the convex linearization method for engineering design optimization*, Engineering Optimization, Vol. 11, 195-216

[13] Ortega J.M., Rheinboldt W.C. (1970): *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York-San Francisco-London

[14] Rockafellar R.T. (1974): *Augmented Lagrange multiplier functions and duality in non-convex programming*, Journal on Control, Vol. 12, 268-285

[15] Schittkowski K. (1983): *On the convergence of a sequential quadratic programming method with an augmented Lagrangian search direction,* Optimization, Vol. 14, 197-216

[16] Schittkowski K., Zillober C., Zotemantel R. (1994): *Numerical comparison of nonlinear programming algorithms for structural optimization*, Structural Optimization, Vol. 7, No. 1, 1-28

[17] Svanberg K. (1987): *The Method of Moving Asymptotes – a new method for Structural Optimization*, International Journal for Numerical Methods in Engineering, Vol. 24, 359-373

[18] Vanderbei R.J., Shanno D.F. (1999): *An interior point algorithm for nonconvex nonlinear programming*, Computational Optimization and Applications, Vol. 13, 231-252

[19] Zillober Ch. (2001a): *Global convergence of a nonlinear programming method using convex approximations*, Numerical Algorithms, Vol. 27, 256-289

[20] Zillober Ch. (2001b): *A combined convex approximation – interior point approach for large scale nonlinear programming*, Optimization and Engineering, Vol. 2, 51-73

[21] Zillober Ch. (2002a): *Software manual for SCPIP 2.3*, Report, Department of Mathematics, University of Bayreuth

[22] Zillober Ch. (2002b): *SCPIP – an efficient software tool for the solution of structural optimization problems*, Structural and Multidisciplinary Optimization, Vol. 24, No. 5, 362-371

[23] Zillober Ch., Vogel F. (2000a): *Solving large scale structural optimization problems*, in: Proceedings of the 2nd ASMO UK/ISSMO Conference on Engineering Design Optimization, J. Sienz ed., University of Swansea, Wales, 273-280

[24] Zillober Ch., Vogel F. (2000b): *Adaptive strategies for large scale optimization problems in mechanical engineering*, in: Recent Advances in Applied and Theoretical Mathematics, N. Mastorakis ed., World Scientific and Engineering Society Press, 156-161