Sequential Convex Programming Methods for Solving Large Topology Optimization Problems: Implementation and Computational Results

Q. Ni[1]*, Ch. Zillober[2], K. Schittkowski[2]

[1]Nanjing University of Aeronautics and Astronautics, 210016 Nanjing, P.R. China
[2]Department of Computer Science, University of Bayreuth, D-95440 Bayreuth, Germany

Abstract

In this paper, we describe a method to solve large-scale structural optimization problems by sequential convex programming (SCP). A predictor-corrector interior point method is applied to solve the strictly convex subproblems. The SCP algorithm and the topology optimization approach are introduced. Especially, different strategies to solve certain linear systems of equations are analyzed. Numerical results are presented to show the efficiency of the proposed method for solving topology optimization problems and to compare different variants.

Keywords: large scale optimization, topology optimization, sequential convex programming method, predictor-corrector interior point method, method of moving asymptotes.

1 Introduction

The method of moving asymptotes (MMA) was introduced by Svanberg [7] in 1987. To prove global convergence and to stabilize the algorithm, Zillober [8] added a line search procedure and called it the sequential convex programming (SCP) method. Both methods are proved to be efficient tools in the context of mechanical structural optimization, see for instance the comparative study of Schittkowski et al. [3], especially since displacement dependent constraints are approximated very well. But also optimization problems from other areas can be solved very efficiently in certain situations [5]. In a recent paper of Zillober et al. [12], it is shown how very large scale optimal control problems with partial elliptic equations can be solved after a full discretization.

*This work was mainly done while the first author was visiting the University of Bayreuth, and was supported by the Chinese Scholarship Council, German Academic Exchange Service (DAAD) and the National Natural Science Foundation of China.

Zillober [9] extended the approach to a generally applicable mathematical programming framework, and in [10] the predictor-corrector interior point method for solving the convex nonlinear subproblems was introduced. Moreover, a Fortran-code with name SCPIP [11] was developed which is in practical use in many academic and commercial applications.

The main focus of this paper is to show how the SCP method can be applied to solve large-scale topology optimization problems. These problems can become extremely large and possess dense Hessians of the objective function. The mathematical structure is easily analyzed and a large number of scalable test problems is obtained in a straightforward way.

To describe the SCP method, we consider the general nonlinear programming problem

$$\begin{array}{lll} \min & f(x), & x \in \mathbb{R}^n, \\ \text{s.t.} & h_j(x) = 0, & j = 1, ..., m_{eq}, \\ & h_j(x) \leq 0, & j = m_{eq}+1, ..., m, \\ & \underline{x_i} \leq x_i \leq \overline{x_i}, & i = 1, ..., n. \end{array} \tag{1.1}$$

The functions f and h_j, $j = 1, \ldots, m$, are defined on $X := \{x \,|\, \underline{x}_i \leq x_i \leq \overline{x}_i,\ i = 1, \ldots, n\}$, are assumed to be continuous in X and at least twice continuously differentiable in the interior of X. The feasible region is assumed to be non-empty.

The objective function of (1.1) is approximated by a uniformly convex function, inequality constraints by convex functions, and equality constraints by linear functions. Thus, (1.1) is replaced by a separable, convex, and nonlinear subproblem which is much easier to solve. Numerical results show the advantages of an interior point method for solving the subproblem. It is possible to reduce the size of the internally generated linear systems, where the major part of the computing is spent, to m, which is favorable when m is small compared to n as is the case for topology optimization problems. Another possibility is to reduce the size of linear subsystems to n. A small number of variables and a large number of constraints is a typical situation for many sizing problems in structural optimization. Moreover, there is a third possibility to formulate linear systems with $n + m$ equations and variables, by which special sparsity patterns can be exploited. The first two approaches will be compared by numerical tests.

The outline of the paper is as follows. In Section 2 the SCP method is formulated and the SCPIP code is briefly introduced. Topology optimization, our main source for generating test problems, is described in Section 3. Section 4 contains numerical results.

2 The Sequential Convex Programming Method

Similar to most other nonlinear programming algorithms, the SCP method replaces problem (1.1) in the k-th step by a subproblem

$$\begin{array}{lll} \min & f^k(x), & x \in \mathbb{R}^n, \\ \text{s.t.} & h_j^k(x) = 0, & j = 1, ..., m_{eq}, \\ & h_j^k(x) \leq 0, & j = m_{eq}+1, ..., m, \\ & \underline{x_i}' \leq x_i \leq \overline{x_i}', & i = 1, ..., n. \end{array} \tag{2.1}$$

If we define

$$\phi_i(g, z, y, x) = \frac{\partial g(y)}{\partial x_i} \left(\frac{(z_i - y_i)^2}{z_i - x_i} - (z_i - y_i) \right), \tag{2.2}$$

where $g : \mathbb{R}^n \longrightarrow \mathbb{R}$, $x, y, z \in \mathbb{R}^n$ with $x = (x_1, \ldots, x_n)^T$, $y = (y_1, \ldots, y_n)^T$, and $z = (z_1, \ldots, z_n)^T$, the approximation of the objective function f^k in the $k-$th step is defined by

$$f^k(x) = f(x^k) + \sum_{i \in I_+^k} \left(\phi_i(f, U^k, x^k, x) + \tau_i^k \frac{(x_i - x_i^k)^2}{U_i^k - x_i} \right) + \sum_{i \in I_-^k} \left(\phi_i(f, L^k, x^k, x) + \tau_i^k \frac{(x_i - x_i^k)^2}{x_i - L_i^k} \right),$$

where $I_+^k = \{i : \frac{\partial f(x^k)}{\partial x_i} \geq 0\}$, $I_-^k = \{1, \ldots, n\} \setminus I_+^k$. Inequality constraints are approximated by $h_j^k(x)$, $j = m_{eq} + 1$, ..., m,

$$h_j^k(x) = h_j(x^k) + \sum_{i \in J_{j,+}^k} \phi_i(h_j, U^k, x^k, x) + \sum_{i \in J_{j,-}^k} \phi_i(h_j, L^k, x^k, x),$$

where $J_{j,+}^k = \{i : \frac{\partial h_j(x^k)}{\partial x_i} \geq 0\}$, $J_{j,-}^k = \{1, \ldots, n\} \setminus J_{j,+}^k$. Equality constraints are replaced by linear functions $h_j^k(x)$, $j = 1, \ldots, m_{eq}$,

$$h_j^k(x) = h_j(x^k) + \sum_{i=1}^{n} \frac{\partial h_j(x^k)}{\partial x_i}(x_i - x_i^k).$$

All functions f^k and h_j^k, $j = m_{eq} + 1, \ldots, m$, are defined on

$$D^k = \{x \mid L_i^k < x_i < U_i^k,\ i = 1, 2, \ldots, n\}.$$

L_i^k and U_i^k are so-called *moving asymptotes* with $L_i^k < x_i^k < U_i^k$, τ_i^k is a positive parameter, and

$$\underline{x}_i' = \max\{\underline{x}_i, x_i^k - \omega(x_i^k - L_i^k)\},\ \overline{x}_i' = \min\{\overline{x}_i, x_i^k + \omega(U_i^k - x_i^k),\ \omega \in (0,1),\ i = 1, 2, \ldots, n.$$

In principle, the solution of (2.1) is the new iteration point. But to ensure a globally convergent algorithm, an augmented Lagrangian merit function is introduced,

$$\Phi_r(x, y) = f(x) + \sum_{j=1}^{m_{eq}} (y_j h_j(x) + \frac{r_j}{2} h_j^2(x)) + \sum_{j=m_{eq}+1}^{\hat{m}} \kappa_j(x, y) \tag{2.3}$$

where

$$\kappa_j(x, y) = \begin{cases} y_j h_j(x) + \frac{r_j}{2} h_j^2(x), & \text{if } -\frac{y_j}{r_j} \leq h_j(x) \\ -\frac{y_j^2}{2r_j}, & \text{otherwise} \end{cases},$$

$\hat{m} := m + 2n$, $h_{m+i} := \underline{x}_i - x_i$, $h_{m+n+i} := x_i - \overline{x}_i$, $i = 1$, ..., n. After solving subproblem (2.1), a descent direction is defined by the difference of the solution of (2.1) and the previous iteration point. Then a line search is performed along this direction with respect to the the merit function (2.3). For a detailed outline of the individual steps of the SCP method, we refer to Zillober [9].

In order to use a predictor-corrector interior point method, (2.1) is modified,

$$\begin{array}{lll} \min & f^k(x), & x, s, t \in \mathbb{R}^n, \quad c, r \in \mathbb{R}^{m - m_{eq}}, \\ s.t. & h_j^k(x) = 0, & j = 1, \ldots, m_{eq}, \\ & h_j^k(x) + c_{j - m_{eq}} = 0, & j = m_{eq} + 1, \ldots, m, \\ & -c_j + r_j = 0, & j = 1, \ldots, m - m_{eq}, \\ & \underline{x}_i' - x_i + s_i = 0, & i = 1, \ldots, n, \\ & x_i - \overline{x}_i' + t_i = 0, & i = 1, \ldots, n, \\ & r, s, t \geq 0. & \end{array}$$

c, s, and t are introduced as slack variables to transform inequality constraints and bounds to equality constraints. Another artificial variable r allows, that c becomes exactly 0 also in the beginning of the iterations which is of some practical advantage.

The lower-bound constraints are added to the Lagrangian function of the subproblem by logarithmic barrier terms with a positive barrier parameter μ,

$$L_\mu(x,y,c,r,s,t,d_r,d_s,d_t) = f^k(x) - \mu \sum_{j=1}^{m-m_{eq}} \ln r_i - \mu \sum_{i=1}^{n} \ln s_i - \mu \sum_{i=1}^{n} \ln t_i + \sum_{j=1}^{m_{eq}} y_j h_j^k(x)$$
$$+ \sum_{j=m_{eq}+1}^{m} y_j (h_j^k(x) + c_{j-m_{eq}}) + d_r^T(-c+r) + d_s^T(\underline{x}' - x + s) + d_t^T(x - \overline{x}' + t),$$

where y, d_r, d_s, d_t are the dual variable vectors to the corresponding constraints. From the necessary optimality condition $\nabla L_\mu = 0$, we obtain the following linear system to compute a Newton step,

$$\begin{pmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & 0 \end{pmatrix} \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta x_3 \end{pmatrix} = -\nabla L_\mu \tag{2.4}$$

where

$$A_{11} = \begin{pmatrix} \nabla_{xx} L & J_{eq} & J_{ie} & \\ J_{eq}^T & & & \\ J_{ie}^T & & & I \\ & & I & \end{pmatrix}, \; A_{13} = \begin{pmatrix} 0 & -I & I \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -I & 0 & 0 \end{pmatrix}, \; A_{31} = \begin{pmatrix} 0 & 0 & 0 & -I \\ -I & 0 & 0 & 0 \\ I & 0 & 0 & 0 \end{pmatrix},$$

$$\Delta x_1 = \begin{pmatrix} \Delta x \\ \Delta y_{eq} \\ \Delta y_{ie} \\ \Delta c \end{pmatrix}, \; \Delta x_2 = \begin{pmatrix} \Delta r \\ \Delta s \\ \Delta t \end{pmatrix}, \; \Delta x_3 = \begin{pmatrix} \Delta d_r \\ \Delta d_s \\ \Delta d_t \end{pmatrix},$$

$A_{22} = \mathrm{diag}(D_r, D_s, D_t)$, $A_{23} = \mathrm{diag}(R, S, T)$, $A_{32} = \mathrm{diag}(I, I, I)$. $J_{eq} \in R^{n \times m_{eq}}$ and $J_{ie} \in R^{n \times (m-m_{eq})}$ are the transposed Jacobian matrices of the constraints. $\nabla_{xx} L = \nabla^2 f^k(x) + \frac{d}{dx}(J_{eq} y_{eq} + J_{ie} y_{ie})$, $y_{eq} = (y_1, \ldots, y_{m_{eq}})^T$, $y_{ie} = (y_{m_{eq}+1}, \ldots, y_m)^T$. $R = \mathrm{diag}(r_1, \ldots, r_{m-m_{eq}})$, S, T, D_r, D_s and D_t, respectively, are diagonal matrices in the appropriate dimensions.

From (2.4), we deduce three different transformations which can be used alternatively depending on the problem structure and size. First, we get a linear system with $n+m$ equations and variables by

$$\begin{pmatrix} \nabla_{xx} L + S^{-1} D_s + T^{-1} D_t & J_{eq} & J_{ie} \\ J_{eq}^T & & \\ J_{ie}^T & & -D_r^{-1} R \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y_{eq} \\ \Delta y_{ie} \end{pmatrix} = \begin{pmatrix} a_1 \\ -h_{eq}^k(x) \\ a_2 \end{pmatrix} \tag{2.5}$$

where

$$\begin{aligned} a_1 &= -\nabla f^k(x) - J_{eq} y_{eq} - J_{ie} y_{ie} + d_s - d_t - S^{-1} D_s (x - \underline{x}') + T^{-1} D_t (\overline{x}' - x), \\ a_2 &= -h_{ie}^k - D_r^{-1} R(-y_{ie} + d_r), \\ h_{eq}^k(x) &= (h_1^k(x), \ldots, h_{m_{eq}}^k(x))^T, \\ h_{ie}^k(x) &= (h_{m_{eq}+1}^k(x), \ldots, h_m^k(x))^T. \end{aligned}$$

Alternatively, it is possible to get a linear system of size m of the form

$$\begin{pmatrix} J_{eq}^T Q^{-1} J_{eq} & J_{eq}^T Q^{-1} J_{ie} \\ J_{ie}^T Q^{-1} J_{eq} & J_{ie}^T Q^{-1} J_{ie} + D_r^{-1} R \end{pmatrix} \begin{pmatrix} \Delta y_{eq} \\ \Delta y_{ie} \end{pmatrix} = \begin{pmatrix} J_{eq}^T Q^{-1} a_1 + h_{eq}^k(x) \\ J_{ie}^T Q^{-1} a_1 - a_2 \end{pmatrix} \tag{2.6}$$

where

$$Q = \nabla_{xx} L + S^{-1} D_s + T^{-1} D_t.$$

If $m_{eq} = 0$ as in case for all our examples of Section 4, (2.6) reduces to

$$(J_{ie}^T Q^{-1} J_{ie} + D_r^{-1} R) \Delta y_{ie} = J_{ie}^T Q^{-1} a_1 - a_2 \tag{2.7}$$

If $m_{eq} = 0$ it is possible to reduce (2.6) to the n by n system

$$(Q + J_{ie} R^{-1} D_r J_{ie}^T) \Delta x = a_1 + J_{ie} R^{-1} D_r a_2. \tag{2.8}$$

In the corrector step, these three systems are similar, only the right-hand sides change. Hence, the linear system (2.4) can be solved by using any of the three different approaches, i.e., a Newton step is obtained by solving (2.5), (2.6), or (2.8), as implemented in a code called SCPIP. For a detailed description we refer to Zillober [10, 11]. In this paper, subroutine SCPIP is used to solve large-scale topology optimization problems which are described in the following section.

3 Topology Optimization

Topology optimization generates the optimal shape of a mechanical structure. Given a predefined domain in the 2D/3D space with boundary conditions and external loads, 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. Without any further decisions and guidance of the user, the method will form the structural shape thus providing a first idea of an efficient geometry.

To represent the mass distribution respectively the form of the structure and at the same time the structural behavior, the design space is discretized by the finite element method. Assuming isotropic material, the so-called power law approach, see Bendsøe [1] or Mlejnek [4], leads to a frequently used formulation

$$\begin{aligned} \min \quad & c(x) = u^T K(x) u = \sum_{i=1}^{n} x_i^p u_i^T K_i(x) u_i, \\ s.t. \quad & V(x) \le a V_0, \\ & K(x) u = f, \\ & 0 < x_{min} \le x \le 1. \end{aligned} \tag{3.1}$$

Here $x \in \mathbb{R}^n$ is the vector of variables called the relative densities of material in the finite elements. u and f are the global displacement and force vectors, respectively, and K is the global stiffness matrix which is defined by the element stiffness matrices K_i and displacement vectors u_i, respectively, $i = 1, \ldots, n$. $x_{min} > 0$ is a vector of lower bounds for the relative densities to avoid singularities, n is the number of elements used to discretize the design domain, p is a penalization power, $V(x)$ and V_0 are the material volume and the design domain volume, respectively, and a is the prescribed volume fraction to be used for the final design.

Figure 1: TOP1, ground structure

Figure 2: TOP2, ground structure

In problem (3.1), vector u depends on vector x by the condition of mechanical equilibrium,

$$K(x)u = f. \tag{3.2}$$

This condition is usually solved in an outer loop. Hence, problem (3.1) becomes

$$\begin{aligned} \min \quad & c(x) = u(x)^T K(x) u(x) = \sum_{i=1}^{n} x_i^p u_i(x)^T K_i(x) u_i(x), \\ s.t. \quad & V(x) \le a V_0, \\ & 0 < x_{min} \le x \le 1, \end{aligned} \tag{3.3}$$

where $u(x)$ is implicitly obtained from (3.2), and $V(x)$ depends linearly on the relative densities x_i, $i = 1, \ldots, n$. (3.1) describes how an optimal material distribution of a given ground structure can be computed such that its compliance is minimized. Here, only one load case is considered. For the more general situation of l load cases, there are l force and l displacement vectors. Hence, the objective function is now the sum of l compliances

$$c(x) = \sum_{j=1}^{l} u_j(x)^T K(x) u_j(x), \tag{3.4}$$

where $u_1(x)$, ..., $u_l(x)$ are obtained by solving the linear systems of equations

$$K(x)u_j = f_j, \quad j = 1, \ldots, l. \tag{3.5}$$

Note that for large design structures (3.1) and a sufficiently fine grid, the stiffness matrix $K(x)$ is a large sparse matrix.

4 Numerical Test Results

4.1 Topology Optimization Problems

In this section, we use the same test problems that were derived by Sigmund [6]. The first one, TOP1, is the so-called MBB-halfbeam, see Figure 1 for the ground structure. TOP2 is a cantilever beam, see Figure 2, TOP3 a cantilever beam with two load cases, i.e., $l = 2$, see also (3.4,3.5) and Figure 3, and TOP4 is a cantilever beam with a fixed circular hole, i.e., with passive elements fixed at the lower bound, see Figure 4.

The direct solution of the optimization problem as outlined above leads to unsatisfactory results. The so-called checkerboard phenomenon, a certain type of mesh-dependency, would

Figure 3: TOP3, ground structure

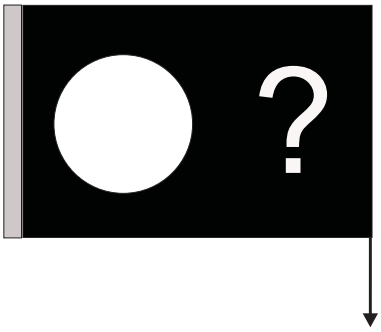

Figure 4: TOP4, ground structure

very likely appear, see Bendsøe and Sigmund [2] for an overview on these difficulties and some remedies. Typically, a filter technique is added, i.e., a certain manipulation of the gradient of the objective function to avoid black and white patterns of neighbored elements. Thus, we will always work with perturbed gradients. The first consequence is that we will never fulfill a strong convergence criterion based on the norm of the gradient of the Lagrangian. Moreover, the application of the line-search technique (2.3) becomes too erroneous and destroys fast numerical convergence speed. The perturbed gradients are passed to the subproblem formulation, cf. (2.2), leading to perturbed iterates.

The filter technique works as follows, see Sigmund [6] or Bendsøe and Sigmund [2]. Instead of the original gradient $\nabla f(x^k)$ of the objective function at a given iteration point x^k we use a weighted sum of the original components in a certain neighborhood for a particular component. We apply the formula

$$\frac{\partial \hat{f}}{\partial x_i} = \frac{1}{x_i \sum\limits_{j=1}^{n} w_{ij}} \sum_{j=1}^{n} w_{ij} x_j \frac{\partial f}{\partial x_j},$$

for approximating the gradient, where the weight factors are defined as $w_{ij} = r_{\text{filter}} - \text{dist}(i,j)$, $i = 1, \ldots, n$ and j is chosen so that $\text{dist}(i,j) \leq r_{\text{filter}}$. The weight factors are set to 0 outside a circle with radius r_{filter} around the current element. The function dist measures the distance of the centers between two elements.

Numerical tests were performed on a PC running under Windows XP and a Pentium 4 processor with 2.6 GHz and 512 MB RAM. We used the Compaq Visual Fortran 6.6 Compiler. All computations are performed in double precision Fortran77. For our numerical tests, the linear systems (3.2) are solved by a minimum degree ordering algorithm and a Cholesky decomposition.

The initial point to start the SCP code, is always set to $x_i^0 = a$ for all $i = 1, \ldots, n$, leading to an active volume constraint. The termination criterion for SCPIP is

$$\|f(x^k) - f(x^{k-1})\| \leq 10^{-4}$$

and feasibility subject to the given tolerance. There are alternative stopping criteria implemented in SCPIP, but under the test conditions outlined above, this criterion seems to be the most reliable one. To give an example, a criterion based on the change of design variables as used by Sigmund [6] leads to too many iterations near the solution because some cycling of a few variables at the border of the structure.

				(2.7)		(2.8)	
Problem	r_{filter}	$n_x \times n_y$	n	IT	CPU	IT	CPU
TOP1	1.5	60×20	1200	52	3.3	54	1,155.0
TOP2	1.2	32×20	640	46	1.4	92	316.3
TOP3	1.2	30×30	900	65	3.6	35	292.1
TOP4	1.5	45×30	1350	30	2.2	26	623.8

Table 1: Results for SCPIP, small-scale problems

Problem	r_{filter}	$n_x \times n_y$	n	IT	CPU
TOP1	10.5	420×140	58,800	47	773.0
TOP2	12.0	320×200	64,000	48	1,033.4
TOP3	9.6	240×240	57,600	49	1,819.2
TOP4	10.5	315×210	66,150	33	805.4

Table 2: Results for SCPIP, large-scale problems

Table 1 contains some performance results for TOP1 to TOP4 obtained by SCPIP. We used the same data as Sigmund [6]. We report the number of main iterations (IT) and the CPU-time in seconds (CPU) for two different versions differing by the way the linear subproblems of the Newton step are solved, i.e., formulation (2.7) and formulation (2.8). n_x and n_y denote the number of used finite elements in $x-$direction and $y-$direction, respectively.

Although the linear systems (2.7) and (2.8) are algebraically equivalent, we observe differences in the number of iterations caused by round-off errors. As expected, SCPIP with (2.7) performs much better than the version with (2.8), since there is only one constraint. The reported figures show the impact of choosing an adequate subproblem. It is emphasized that SCPIP can choose the most adequate variant automatically, a somewhat tricky procedure in case of balanced n and m.

We do not report any comparisons with the Matlab code of Sigmund [6] because of obvious differences in CPU times. We also tested SCPIP for larger instances of the same problems. For that purpose we applied a scaling factor of 7 to TOP1 and TOP4, of 8 to TOP3 and of 10 to TOP2, respectively, to the discretization parameters n_x and n_y. The filter size had also been multiplied by the scaling factors. Numerical results are shown in Table 2, where we omitted version (2.8) because of too excessive calculation times. Final designs are shown in Figures 5 and 6 for TOP1, Figures 7 and 8 for TOP2, Figures 9 and 10 for TOP3, and Figures 11 and 12 for TOP4.

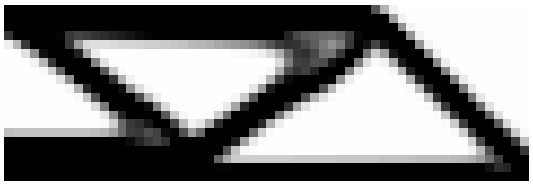

Figure 5: TOP1, 60×20

Figure 6: TOP1, 420×140

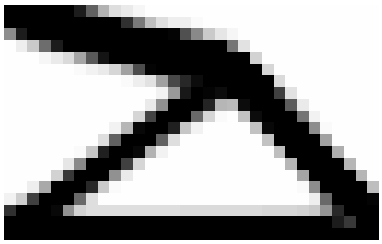

Figure 7: TOP2, 32×20

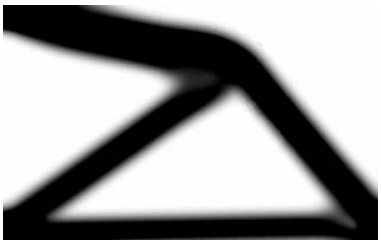

Figure 8: TOP2, 320×200

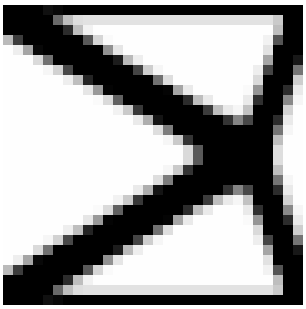

Figure 9: TOP3, 30×30

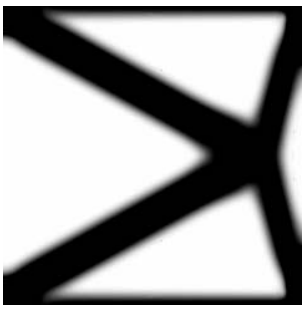

Figure 10: TOP3, 240×240

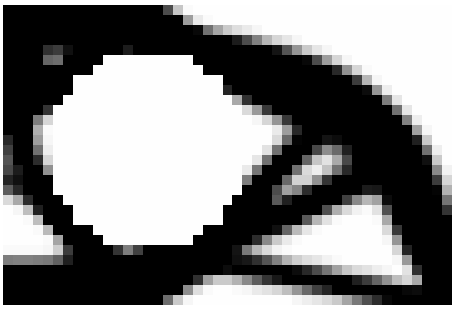

Figure 11: TOP4, 45×30

Figure 12: TOP4, 315×210

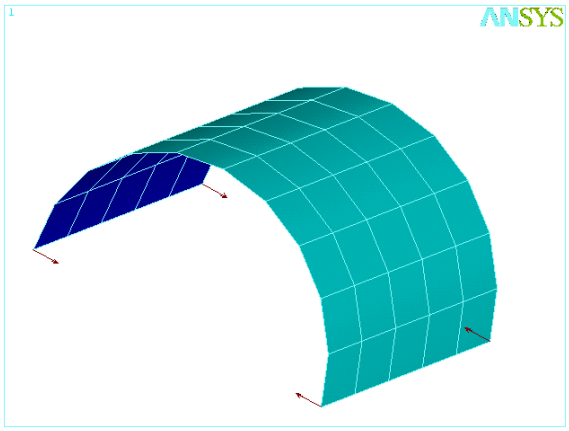

Figure 13: Arc structure

4.2 Comparison Between SCP and MMA

As outlined in Section 2, SCP is considered as an extension of MMA obtained by adding a linea search procedure subject to an augmented Lagrangian merit function. In general, we do not observe great differences of the performance between SCP and MMA in practice. Usually, structural design problems are provided with good starting points, such that the steplength one is preferred in most iterations. In other words, both methods coincide in most iterations.

However, there are exceptions by which the impact of the globalizing strategy can be shown. The example under consideration is the arc structure of Figure 13, which is fixed in two nodes at the border of the top. The bottom nodes are restricted through the ground. Loads, indicated by arrows, are applied at four nodes at the edges of the bottom part. Objective function to be minimized is the total volume. Design variables are the thicknesses of elements and stresses of elements are to be constrained. We get an optimization problem with 50 variables. i.e., 50 finite elements, and 100 constraints, i.e. two stress bounds in each element. The finite element structure is set up by ANSYS[1].

The subsequent table uses the following abbreviations:

k	-	iteration number,
$\|h(x_k)^+\|_\infty$	-	maximum constraint violation,
$f(x_k)$	-	objective function value,
α_k	-	steplength,
$\|\nabla L(x_k, u_k)\|_\infty$	-	gradient norm of Lagrangian function.

For this example we used the stopping criterion

$$\|\nabla L(x_k, u_k)\|_\infty \leq 10^{-5} \text{ and } \|h(x_k)^+\|_\infty \leq 10^{-5}.$$

All computations have been performed in double precision Fortran 77.

SCP terminates after iteration 29 with a total of 38 function and 30 gradient evaluations. MMA does not converge within the given maximum number of iterations (100). It is evident that a line search handles large initial infeasibility much better. During the first forty iterations, MMA cycles forth and back, and shows slow monotone convergence afterwards.

[1]ANSYS is a product of ANSYS Inc., Southpointe, 275 Technology Drive, Canonsburg, PA 15317, USA

	SCP				MMA		
k	$\|h(x_k)^+\|_\infty$	$f(x_k)$	α_k	$\|\nabla L(x_k,u_k)\|_\infty$	$\|h(x_k)^+\|_\infty$	$f(x_k)$	$\|\nabla L(x_k,u_k)\|_\infty$
0	81.08	4.693D-2	–	6.6D-2	81.08	4.693D-2	6.6D-2
1	67.24	5.768D-2	1	6.4D-2	67.24	5.768D-2	6.4D-2
2	17.16	9.964D-2	1	0.50	17.16	9.964D-2	0.50
3	7.76	0.1935	1	0.89	7.76	0.1935	0.89
4	5.26	0.3120	1	232.1	5.26	0.3120	232.1
5	2.90	0.4380	1	230.2	2.90	0.4380	230.2
6	2.55	0.4716	0.42	249.3	1.85	0.5188	752.6
7	0.42	0.6187	1	18.3	0.45	0.6082	510.8
8	0.16	0.5190	0.36	145.2	35.5	0.3907	2902.3
...			...			...	
28	1.5D-7	0.3477	1	2.4D-5	1982.2	0.4936	2.17D+4
29	2.5D-9	0.3477	1	6.1D-6	388.1	0.5126	663.8
...			...			...	
100	–	–	–	–	1.3D-6	0.3587	4.8D-5

Table 3: Iteration history of the arc example

5 Conclusion

In this paper, we describe the usage of sequential convex programming methods to solve topology optimization problems arising in structural mechanical engineering. The numerical results show that SCPIP is an efficient and robust tool for solving these problems, in particular in case of higher dimensions. At least for solving standard topology optimization problems, a specific version based on the internal formulation of linear systems of m equations, see (2.7), is recommended.

6 Acknowledgements

It is our pleasure to thank Prof. Sigmund for providing the Matlab files of the topology optimization problems. The support of the Chinese Scholarship Council, German Academic Exchange Service (DAAD) and the National Natural Science Foundation of China is appreciated. We thank ANSYS Inc., Canonsburg, for providing a test version of ANSYS.

References

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

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

[3] K. Schittkowski, Ch. Zillober and R. Zotemantel (1994). Numerical comparison of nonlinear programming algorithms for structural optimzation, Structural Optimization, 7, 1-19.

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

[5] Q. Ni (2003). A globally convergent method of moving asymptotes with trust region technique, Optimization Methods and Software, 18, 283-297.

[6] O. Sigmund (2001). A 99 line topology optimization code written in Matlab, Structural and Multidisciplinary Optimization, 21, 120-127.

[7] K. Svanberg (1987). The method of moving asymptotes – a new method for structural optimization, International Journal for Numerical Methods in Engineering, 24, 359-373.

[8] Ch. Zillober (1993). A globally convergent version of the method of moving asymptotes, Structural Optimzation, 6, 166-174.

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

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

[11] Ch. Zillober (2002). SCPIP - an efficient software tool for the solution of structural optimization problems, Structural and Multidisciplinary Optimization, 24, 362-371.

[12] Ch. Zillober, K. Schittkowski and K. Moritzen (2004). Very large scale optimization by sequential convex programming, to appear in Optimization Methods and Software.