SIAM J. OPTIM.
Vol. 8, No. 4, pp. 1084–1107, November 1998

013

OPTIMAL TRUSS DESIGN BY INTERIOR-POINT METHODS*

FLORIAN JARRE†, MICHAL KOČVARA‡, AND JOCHEM ZOWE‡

Abstract. This article presents a primal-dual predictor-corrector interior-point method for solving quadratically constrained convex optimization problems that arise from truss design problems. We investigate certain special features of the problem, discuss fundamental differences of interior-point methods for linearly and nonlinearly constrained problems, extend Mehrotra's predictor-corrector strategy to nonlinear programs, and establish convergence of a long step method. Numerical experiments on large scale problems illustrate the surprising efficiency of the method.

Key words. interior-point methods, truss topology design, convex programming

AMS subject classifications. 90C30, 73K40

PII. S1052623496297097

1. Introduction.

1.1. The mechanical-mathematical model. In this section we briefly sketch the engineering problem which we propose to solve by interior-point methods. A more detailed introduction can be found in [14]. The reader who is not interested in the physical background can switch immediately to the final mathematical formulation (1.8).

We consider the problem of finding the stiffest truss (pin-jointed framework) with respect to a given load. This problem is studied in the so-called *ground-structure* framework [8], where we start from a dense mesh of N nodal points in $\mathbb{R}^{dim}$, dim = 2 or 3, and each two of these nodes can be connected by a bar. Thus we work with N tentative nodes and with up to $m = N(N-1)/2$ potential bars. Our optimization problem is to select from this rich network the most profitable set of nodes and bars able to carry a given load f (the thin arrows at the right of each layout in Figure 1.1).

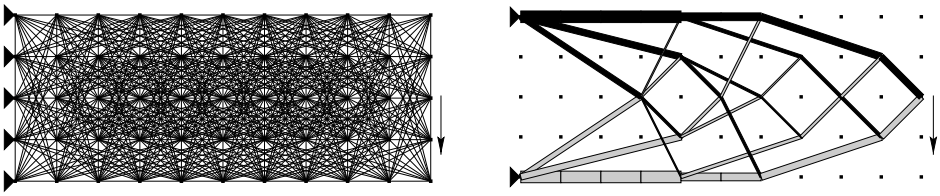

FIG. 1.1. *Initial layout and optimal truss with vertical load on the right-hand side.*

The design variables (controls) in the model are the bar volumes $t_i \geq 0, i = 1, \ldots, m$, which are aggregated in a vector t. Further let x denote the vector of nodal

*Received by the editors January 2, 1996; accepted for publication (in revised form) December 11, 1997; published electronically September 23, 1998. This research was partly supported by project 03ZO7BAY of the Federal Department of Education, Science, Research and Technology (BMBF), Germany, and grant A1075707 of the Czech Academy of Sciences.

http://www.siam.org/journals/siopt/8-4/29709.html

† and Institut für Angewandte Mathematik und Statistik, Universität Würzburg, Am Hubland, D–97074 Würzburg, Germany (jarre@mathematik.uni-wuerzburg.de).

‡Institut für Angewandte Mathematik, Universität Erlangen, Martensstr. 3, D–91058 Erlangen, Germany (kocvara@am.uni-erlangen.de, zowe@am.uni-erlangen.de).

displacements of the system under load. We assume that there are p components with prescribed fixed displacement (that are "nailed" to a solid wall, for example). These components can be omitted from the problem formulation and thus x is of dimension $n = \dim \cdot\ N - p$.

Under the assumption of linear elastic material behavior, the potential energy of such a truss structure, which is subject to an external nodal force vector $f \in \mathbb{R}^n$, is given by

$$\Pi(t,x) = \frac{1}{2} x^T \left(\sum_{i=1}^{m} t_i A_i \right) x - f^T x; \tag{1.1}$$

here

$$A_i = b_i b_i^T \qquad \text{with} \qquad b_i = \frac{\sqrt{E_i}}{l_i} \gamma_i, \qquad i = 1, 2, \ldots, m,$$

is the symmetric positive semidefinite $n \times n$ dyadic stiffness matrix of the ith bar, E_i is a material constant (Young's modulus), l_i is the length of the bar, and $\gamma_i \in \mathbb{R}^n$ is the vector of direction cosines such that $\gamma_i^T x$ measures the bar elongation.

Throughout we assume that

$$b_1, \ldots, b_m \text{ span the whole space } \mathbb{R}^n. \tag{1.2}$$

It is straightforward to show that under this assumption

$$\sum_{i=1}^{m} t_i A_i \text{ is positive definite whenever } t_i > 0 \text{ for } i = 1, \ldots, m. \tag{1.3}$$

The x that minimizes the potential energy $\Pi(t, \cdot)$ determines a state of equilibrium in which the inner and outer forces balance each other. The goal of the designer is to find some t under the (normalized) volume constraint $\sum_{i=1}^{m} t_i = 1$, for which this minimum is as close to zero as possible. Since $\min_{x \in \mathbb{R}^n} \Pi(t,x)$ is equal to or less than zero, the design problem becomes

$$\begin{aligned} \max_{t \in \mathbb{R}^m} \min_{x \in \mathbb{R}^n} \Pi(t,x) \qquad \text{subject to} \quad & t_i \geq 0, \quad i = 1, \ldots, m, \\ & \sum_{i=1}^{m} t_i = 1. \end{aligned} \tag{1.4}$$

In [14] it is shown that "max" and "min" in (1.4) and the following formulas are justified and optimal solutions exist.

Another often used formulation of the design problem is in terms of "minimization of compliance" ("maximization of global stiffness"):

$$\begin{aligned} \min_{t \in \mathbb{R}^m,\ x \in \mathbb{R}^n} \tfrac{1}{2} f^T x \qquad \text{subject to} \quad & \sum_{i=1}^{m} t_i A_i x = f, \\ & t_i \geq 0, \quad i = 1, \ldots, m, \\ & \sum_{i=1}^{m} t_i = 1. \end{aligned} \tag{1.5}$$

Formulations (1.4) and (1.5) are indeed equivalent up to a factor of -1; i.e., they yield the same optimizers (x,t), and the minimal values differ only by a factor of

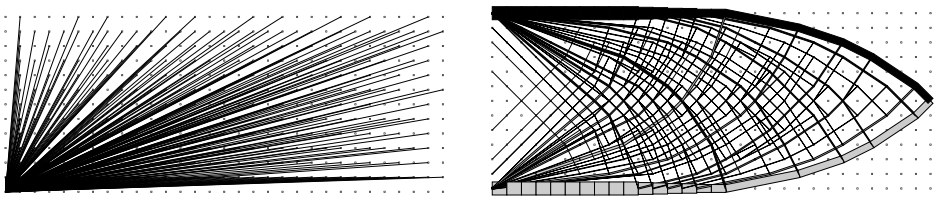

FIG. 1.2. *Refinement of example from Figure* 1.1.

-1. To realize this we note that only t with $f \in \text{range}(\sum_{i=1}^m t_i A_i)$ play a role in the minimization (1.5); such t exist due to (1.2) and (1.3). Now fix any feasible t and note that

$$f \in \text{range}\left(\sum_{i=1}^m t_i A_i\right) \text{ implies } f^T z = 0 \text{ whenever } \left(\sum_{i=1}^m t_i A_i\right) z = 0.$$

This shows that x only acts as a dummy variable in (5): each x with $(\sum_{i=1}^m t_i A_i)x = f$ yields the same value $f^T x$. Hence (1.5) reduces to

$$\begin{aligned} &\min_{t\in\mathbb{R}^m} \tfrac{1}{2} f^T x, \qquad \text{where } x \text{ solves } \quad \left(\sum_{i=1}^m t_i A_i\right) x = f \\ &\qquad\qquad \text{for } t \text{ with } \quad t_i \geq 0, \quad i = 1, \ldots, m \quad \text{and} \quad \sum_{i=1}^m t_i = 1. \end{aligned} \tag{1.6}$$

A similar arguing yields for the inner minimization in (1.4) and for each fixed t with $t_i \geq 0,\ i = 1, \ldots, m$,

$$\min_{z\in\mathbb{R}^n} \Pi(t, z) = \begin{cases} -\infty \quad \text{if } f \notin \text{range}\left(\sum_{i=1}^m t_i A_i\right), \\ -\frac{1}{2} f^T x, \quad \text{where } x \text{ solves } \left(\sum_{i=1}^m t_i A_i\right) x = f, \text{ otherwise.} \end{cases}$$

The equivalence of (1.4) and (1.5) is an immediate consequence. The philosophy of the ground-structure approach is to mimic, by a huge number of nodes in the starting layout, "moves" of the nodal positions also. Consider, e.g., refinement of the example from Figure 1.1. In each coordinate we multiply the number of nodes by three. This results in 403 nodes and 49414 potential bars. Figure 1.2 shows the initial layout (where only some of the bars originating from the south-west node are depicted) and the optimal truss. In real world applications, $n = 1000$ and $m = 100,000$ are typical numbers of nodes and bars in our context. It is well-known that standard optimization software fails to find an optimum of (1.5), even for moderate n and m. The crucial step to solve (1.5) (or, equivalently, (1.4)) lies in an appropriate reformulation. Using a classic minimax-theorem, we can switch the order of "max" and "min" in (1.4). Some further simple analysis gives

$$\min_{x\in\mathbb{R}^n} \max_{\substack{t\in\mathbb{R}^m,\, t_i\geq 0 \\ \sum_{i=1}^m t_i = 1}} \frac{1}{2} \left((b_1^T x)^2, \ldots, (b_m^T x)^2\right) \begin{pmatrix} t_1 \\ \vdots \\ t_m \end{pmatrix} - f^T x,$$

and thus by standard linear programming (LP)-theory

$$\min_{x\in\mathbb{R}^n} \max_{1\le i\le m} \left\{ \frac{1}{2}(b_i^T x)^2 - f^T x \right\}. \tag{1.7}$$

Hence we have arrived at a convex problem in x with dimension n instead of $m+n$, where m was of order n^2; cf. Ben-Tal and Bendsøe [3] or Kočvara and Zowe [14] for details.

Problem (1.7) is a nonsmooth problem to which one can apply nonsmooth software; this has been done, e.g., in [1]. However, it proved to be more profitable to work with a reformulation of (1.7) as a constrained but smooth problem:

$$\min_{x\in\mathbb{R}^n,\ \alpha\in\mathbb{R}} \ -\alpha - f^T x \qquad \text{subject to} \quad \frac{1}{2}(b_i^T x)^2 \le -\alpha, \quad i = 1, 2, \ldots, m. \tag{1.8}$$

It is shown in [14, 3] that the optimal x^* from (1.8) together with a corresponding Lagrange vector of the constraints in (1.8) is always an optimal solution of (1.4) and (1.5). Ben-Tal and Roth [4] and Ben-Tal and Zibulevsky [5] proposed to apply modern log barrier and modified barrier methods to (1.8). Our purpose is to adapt a state-of-the-art interior-point method for linear programs to solve (1.8) (and thus (1.4) and (1.5)). We add that both these approaches provide a Lagrange multiplier (and thus an optimal design vector t^*!) "for free" when solving (1.8).

Let us note that (1.8) can even be reduced to a linear program (see Achtziger et al. [1]). On purpose we do not make use of this very special LP-feature which is lost when we switch to more general problems, like the multiload case or the optimization of material. In the multiload case, one has slightly more general constraints:

$$\frac{1}{2}\sum_{j=1}^{k}(b_{i_j}^T x)^2 \le -\alpha, \quad i = 1, 2, \ldots, m,$$

where k is the number of load vectors. In this case, an LP-reduction is no longer possible. The same is true for material optimization where the b_i's are no longer vectors but rectangular matrices. To include also these cases in our numerical approach, we stick to the formulation (1.8) which is easily generalizable to these situations.

1.2. Notation. The following notation will be used from now on. Vectors are denoted by letters like x, s, t and by $b_1, \ldots, b_m$. If the letter s denotes a vector, the diagonal matrix with diagonal s will be denoted by the capital letter S, and the diagonal matrix with diagonal t will be denoted by T. By $\|x\|$ we denote the Euclidean norm of a vector x. The iteration index k is denoted by a superscript for vectors like x^k and by a subscript for scalar quantities like α_k. Subscripts for vectors denote certain components of the vector, e.g., s_i denotes the ith component of s. Further, we write $t \ge 0$ ($t > 0$) for a vector t if $t_i \ge 0$ ($t_i > 0$) for all its components. By e we denote the vector of all ones; its dimension will always be obvious from the context.

1.3. Outline. In the next section we present an interior-point method for solving convex programs. In section 3, we discuss some special features of problem (1.8) when applying the interior-point method from section 2. In section 4 we establish global convergence of the long step method from section 2, and in section 5 we present some numerical results.

2. A primal-dual interior-point method. Polynomial time versions of interior-point methods for solving quadratically constrained convex programs like (1.8) were proposed in 1987 by Jarre [11] and independently by Mehrotra and Sun [21] in 1988. Subsequently, much more general classes of convex problems have been considered in the context of polynomial time interior-point methods in the recent past, the most general approach being that of Nesterov and Nemirovskii [23]. The papers [11, 21, 23], however, are mostly of theoretical interest, and the development of practical methods for problems like (1.8) differs from these papers.

Since the theoretical analysis of interior-point methods for problem (1.8) suggests that (1.8) is of the same complexity as a linear program (both need about the same number of iterations to reduce the "duality gap" by a factor of 2; see, e.g., [11]), one might be tempted to apply the same algorithm (suitably adapted to handle quadratic constraints) to both problems. This was done, for example, in [12], and the observation based on some preliminary numerical results was that for a primal path-following predictor-corrector method, the number of iterations for solving quadratically constrained problems was even slightly lower than the number of iterations for solving linear programs.

On the other hand, the method of choice for solving linear programs is a primal-dual version of the interior-point method, and this leads us to consider primal-dual interior-point methods for problem (1.8) as well. Our goal is to find a practical primal-dual method for the following convex program:

$$\text{minimize} \quad c^T x \quad \text{subject to} \quad g_i(x) \le 0 \quad \text{for} \quad 1 \le i \le m, \tag{2.1}$$

where $g_i : \mathbb{R}^n \to \mathbb{R}$ are C^3-smooth convex functions. Note that (1.8) is a special case of (2.1).

We use a simple modification of Newton's method for solving the KKT conditions of (2.1), much in the spirit of interior-point methods for linear programs developed, for example, in Kojima, Mizuno, and Yoshise [16], Megiddo [19], Monteiro and Adler [22], and Tanabe [26]. Our focus will be on the main differences of interior-point methods for convex problems and methods for solving linear programs as presented in the above papers. With some further modifications such methods are also generalizable to arbitrary smooth nonconvex programming problems; see, for example, the recent paper [7].

Observe that the KKT conditions for (2.1) are given by

$$\begin{cases} c + \displaystyle\sum_{i=1}^{m} t_i \nabla g_i(x) &= 0, \\ g_i(x) + s_i &= 0 \quad (1 \le i \le m), \\ s_i t_i &= 0 \quad (1 \le i \le m), \\ s_i, t_i &\ge 0. \end{cases} \tag{2.2}$$

As in most interior-point methods, the complementarity conditions $s_i t_i = 0$ are relaxed to

$$s_i t_i = r \qquad \text{for} \qquad 1 \le i \le m \tag{2.3}$$

with some small positive parameter r, and a damped version of Newton's method is applied to solve (2.2) iteratively, while maintaining the strict inequalities $s_i^k > 0$ and

$t_i^k > 0$ for the iterates s^k, t^k. Linearization of the relaxed KKT conditions results in a linear system

$$\begin{pmatrix} \sum t_i \nabla^2 g_i(x) & & \nabla g(x) \\ Dg(x) & I & \\ & T & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta t \end{pmatrix} = \begin{pmatrix} -c - \sum t_i \nabla g_i(x) \\ (-g_i(x) - s_i)_i \\ re - St \end{pmatrix}, \tag{2.4}$$

where S and T are diagonal matrices (the diagonals given by the vectors s and t, as explained in section 1.2), $g(x)$ is the vector with components $g_i(x)$, and $\nabla g(x)$ is the transpose of the Jacobian $Dg(x)$. Given a fixed starting point x^0, we define the following weighted residuals at the point (x, s, t) with $s, t > 0$:

$$\mathrm{res}_1(x,t) = \frac{1}{\|c\|}\left(c + \sum_{i=1}^{m} t_i \nabla g_i(x)\right), \tag{2.5}$$

$$\mathrm{res}_2(x,s) = \frac{1}{1 + \|g(x^0)\|}\left(g(x) + s\right), \tag{2.6}$$

$$\mathrm{res}_3(s,t) = \frac{St}{\|c\|\left(1 + \|g(x^0)\|\right)}. \tag{2.7}$$

To measure the distance from optimality of some iterate (x, s, t), we define a function $\mathbf{d}$ in terms of the above residuals as

$$\mathbf{d}(x,s,t) = \|\mathrm{res}_1(x,t)\| + \|\mathrm{res}_2(x,s)\| + \|\mathrm{res}_3(s,t)\|_1 .$$

We now outline a primal-dual interior-point algorithm.

ALGORITHM 2.1 (primal-dual interior-point algorithm).

INPUT: *An initial point x^0, $s^0 > 0$, $t^0 > 0$, and a stopping tolerance $\epsilon > 0$.*

For $k = 0, 1, \ldots,$ do :

1) *If $\mathbf{d}(x^k, s^k, t^k) \le \epsilon$, then stop.*
2) *Choose $\sigma_k \in (0, 1)$.*
3) *Compute $\Delta x, \Delta s,$ and Δt from the linear system (2.4) with*

$$x = x^k, \quad s = s^k, \quad t = t^k, \quad \text{and} \quad r = \sigma_k \frac{s^T t}{m}. \tag{2.8}$$

4) *Compute*

$$\bar{\lambda}_t^k := \max\{\, \lambda \le 1 \mid t^k + \lambda \Delta t \ge 0 \,\} \text{ and } \bar{\lambda}_s^k := \max\{\, \lambda \le 1 \mid s^k + \lambda \Delta s \ge 0 \,\}. \tag{2.9}$$

5) *Choose $\rho_k \in (0, 1)$.*
6) *Set*

$$\lambda_t^k = \rho_k \bar{\lambda}_t^k \quad \text{and} \quad \lambda_s^k = \rho_k \bar{\lambda}_s^k \quad \text{and} \quad \lambda_x^k = (\lambda_t^k + \lambda_s^k)/2.$$

7) *Set*

$$\begin{aligned} x^{k+1} &= x^k + \lambda_x^k \Delta x, \\ s^{k+1} &= s^k + \lambda_s^k \Delta s, \\ t^{k+1} &= t^k + \lambda_t^k \Delta t. \end{aligned}$$

TABLE 2.1

	$\rho \geq 0.9$ Default start	$\rho \geq 0.9$ Small s^0	$\rho \geq 0.5$ Small s^0
Corrector and diff. steps	10	19	20
Corrector and same steps	12	60	82
No corrector and diff. steps	18	$\gg 100$	49
No corrector and same steps	22	$\gg 100$	92

2.1. Differences to linear programs. The only degrees of freedom that are left in Algorithm 2.1 (apart from the choice of the starting point) are the choice of the parameter σ_k at each iteration and of the step length ρ_k that is taken along the Newton direction at each Newton step.

The choice of σ_k and ρ_k is very important for the efficiency of the above method (for the effect of ρ_k see Table 2.1). In [17], an implementation of an interior-point method for solving linear programs is discussed in detail. At each iteration, ρ_k is set to 0.9995; i.e., a step of length 99.95% to the boundary along the primal-dual Newton direction is taken. This makes sense since the residual decreases linearly and thus, the longer the step the more the infeasibility is reduced. It turns out, however, that a straightforward application of this step length strategy is not efficient (or not even convergent) for problems (1.8) or (2.1).

Why do primal-dual methods behave so differently for linear and for quadratically constrained programs while primal methods are reported to behave quite similarly for both problems (see, e.g., [12], [13])? To give a partial answer, we note that primal methods rely on the self-concordance (see [23]) of the barrier function, and self-concordance equally holds for linear and convex quadratic constraints. Primal-dual methods further exploit the structure of the problem, implicitly using the self-scaling property recently identified by Nesterov and Todd [24]. While both methods are of the same worst case complexity, primal-dual methods are reported to converge much faster for linear programs [25]. Two of the key points used in the implementation of primal-dual methods for solving linear programs are the following.

- The use of different step lengths in the primal and the dual space due to the fact that the primal and dual residuals depend *only* on the primal and dual variables, respectively.
- The use of *very long steps* due to the fact that both primal and dual residuals are linear functions of the step length, and a step of length 1 reduces both residuals to 0.[1]

Both of these key points no longer hold true for problem (2.1). The primal and dual residuals res_1 and res_2 both depend on x, and when different steps in s and t are taken, it is not clear what step length to choose for x. (Our choice of the arithmetic mean above has no theoretical foundation.) Furthermore, both residuals are nonlinear. In particular, in our implementation a step of length 1 was possible at certain iterations, but if a step of length 1 was blindly taken, then the norm of the residuals could increase by a factor of over 100 (due to the nonlinearity) instead of going to 0 as in the case of a linear program. Sometimes it may even be necessary to increase the norm of the residuals along the search direction. If both residuals, res_1 and res_2, happen to be very small, for example, but the parameter $r = r_k$ is large,

[1] In analogy to the commonly used terminology in interior-point methods, by "very long" we denote a step of length ≤ 1 that brings the iterate close to the boundary of the feasible orthant $s \geq 0$, $t \geq 0$. (A step of length 1 typically violates the nonnegativity requirements $s \geq 0$ or $t \geq 0$.)

then only very short steps can be taken when reducing r to 0 if one does not allow for an increase of the residuals. (Loosely speaking, this is the case since for large r the current iterate is still "far away" from an optimal solution, and when taking a long Newton step toward the optimal solution, the residuals will necessarily increase because of their nonlinearity.) For problem (1.8), a much more careful choice of the step length is therefore necessary than it is for linear programs. Similarly, for linear programs it is reported that a rather small choice of σ in Algorithm 2.1 yields an efficient method, while our observation for problem (1.8) was that for most parts of the algorithm, large values $\sigma \geq 0.5$ were more efficient than smaller values. Only in the final iterations our algorithm "chose" small values of $\sigma \leq 0.01$ and showed a fast rate of convergence (in all examples that we tested).

2.2. Adaptation of Mehrotra's predictor-corrector method. The most surprising experience, when testing different versions of our primal-dual method, was the observation that inclusion of a corrector step similar to the one used by Mehrotra [20] for linear programs along with different step lengths in x, s, and t greatly reduced the number of iterations. Mehrotra's predictor step is based on a fixed point iteration derived from (2.2)–(2.3). Relations (2.2)–(2.3) can be written as

$$\Psi(z) = 0,$$

where $z = (x, s, t)$ and the function Ψ computes the left-hand sides of the equations in (2.2)–(2.3). Mehrotra's approach can be motivated as follows: first, Newton's method computes a direction (prediction) Δz^p from the relation

$$0 \stackrel{!}{=} \Psi(z + \Delta z^p) = \Psi(z) + D\Psi(z)\Delta z^p + R_z(\Delta z^p) \tag{2.10}$$

by ignoring the remainder term R_z. Hence Newton direction satisfies the linear approximation to (2.10)

$$0 = \Psi(z) + D\Psi(z)\Delta z^p.$$

(This is exactly (2.4).) We note that $R_z(\Delta z^p)$ is easily computable for a given function Ψ and a given Δz^p by

$$R_z(\Delta z^p) = \Psi(z + \Delta z^p) - \Psi(z) - D\Psi(z)\Delta z^p.$$

Moreover, in our particular example (1.8), Ψ is quadratic, so that

$$R_z(\Delta z^p) = \frac{1}{2} D^2\Psi(z)[\Delta z^p, \Delta z^p]. \tag{2.11}$$

Mehrotra's corrector defines Δz as

$$\Delta z = -D\Psi(z)^{-1}(\Psi(z) + R_z(\Delta z^p)). \tag{2.12}$$

Therefore, Δz satisfies the relation

$$0 = \Psi(z) + D\Psi(z)\Delta z + R_z(\Delta z^p),$$

and in particular, if Δz^p happened to be (almost) equal to Δz, then the relation (2.10) was satisfied (almost) exactly.

We call the solution of (2.12) *Mehrotra's corrector*—even though (2.12) differs slightly from the scheme used by Mehrotra [20]—and we use this corrector by replacing the Newton direction $\Delta x, \Delta s, \Delta t$ of (2.4), in Algorithm 2.1 with the direction $\Delta x, \Delta s, \Delta t$ from (2.12). Variants of Algorithm 2.1 based on (2.4) (but using different step lengths) have been analyzed in a general framework in [23], and more recently in [27]. At this point we are not able to prove convergence for Algorithm 2.1 when the search direction is obtained by Mehrotra's corrector, nor can we prove convergence when different steps in the variables s and t are taken at each iteration. In our program we therefore included a safeguard. When the corrected Newton step does not result in a certain decrease of the residuals in (2.4), the program temporarily reverts to the plain primal-dual method with equal step sizes (and such that the step length minimizes a certain measure for the residual corresponding to $s_i t_i = r$). In our experiments the corrector step proved to be very efficient, and the safeguard of reverting to the plain primal-dual method was necessary only when the starting point was deliberately poorly chosen (very small s^0 in Table 2.1). Table 2.1 illustrates the importance of the corrector step in different versions of the algorithm. It refers to a truss topology design problem with $n = 144$, and $m = 2040$. We list the number of iterations for the method with and without corrector step as well as with and without the use of different step lengths in s and t. The first column refers to the default starting point of section 2.5, and the second column to the default starting point but with s^0 divided by 1000. In both columns, we force a step of at least 90% to the boundary of the positive orthant at each iteration, that is, $\rho_k \geq 0.9$ in the description of Algorithm 2.1. The third column refers to the same (poor) starting point as column 2 but with shorter steps ρ_k along the Newton direction.

Concerning the starting point, when one of the variables is poorly chosen (in columns 2 and 3, s^0 is closer by a factor of 1000 to the "boundary $s = 0$" than t^0), then the table indicates that different steps in s and t are meaningful. Column 2 of the table also shows the intuitively obvious fact that the method does not seem to be globally convergent when long steps towards the boundary are taken at each iteration regardless of the behavior of the norms of the residuals.

A qualitative analysis of Mehrotra's corrector step reveals that a slight damping of his corrector might be useful, and this observation was indeed confirmed by our experiments. In Algorithm 2.1, due to the positivity requirement, only a step of length $\rho < 1$ along Δz is taken at each iteration. For a step of length less than 1, however, it may be more efficient to define Δz as

$$\Delta z = -D\Psi(z)^{-1}(\Psi(z) + \lambda R_z(\Delta z^p))$$

for some value of $\lambda \in [0, 1]$. The motivation for considering values of $\lambda < 1$ arose as in one of our test runs we observed that for $\lambda = 1$ the reduction of the step length ρ from 0.99 to 0.5 significantly increased the dual residual res_2. In this particular situation, the iterate was in the domain of superlinear convergence[2] of the interior-point method, and the scaled norm of the dual residual res_2 was in the order of 10^{-9} for $\rho = 0$ (i.e., the current iterate), as well as for $\rho = 1$ (the full step), but it was in the order of 10^{-5} for $\rho = 0.5$. Due to the nonlinearity, Δx happened to be a good approximation for the full step, but not for a step of length 0.5. Note that this could not happen if res_2 was linear!

[2]We do not prove superlinear convergence of our algorithm, but our numerical experiments unanimously reconfirm it.

We examine which values of λ may be most efficient when a step of length ρ is taken. A step with parameters λ and ρ results in the following function value for Ψ:

$$\begin{aligned}&\Psi(z-\rho D\Psi(z)^{-1}(\Psi(z)+\lambda R_z(\Delta z^p)))\\&\quad=(1-\rho)\Psi(z)-\lambda\rho R_z(\Delta z^p)+R_z(\rho D\Psi(z)^{-1}(\Psi(z)+\lambda R_z(\Delta z^p)))\end{aligned}$$

(by using (2.10)). Applying (2.11) and the definition of Δz^p, this can further be reduced to

$$\begin{aligned}&=(1-\rho)\Psi(z)+(\rho^2-\lambda\rho)R_z(\Delta z^p)+\lambda\rho^2 D^2\Psi(z)[\Delta z^p, D\Psi(z)^{-1}R_z(\Delta z^p)]\\&\qquad+\rho^2\lambda^2 R_z(D\Psi(z)^{-1}R_z(\Delta z^p)).\end{aligned}\tag{2.13}$$

For $\lambda=\rho=1$, this becomes

$$D^2\Psi(z)[\Delta z^p, D\Psi(z)^{-1}R_z(\Delta z^p)]+R_z(D\Psi(z)^{-1}R_z(\Delta z^p)),$$

which is a cubic term in Δz^p. Based on this relation, it is straightforward to show that Mehrotra's corrector step is locally cubically convergent for fixed $r_k>0$.

When taking a step of length $\rho<1$, we cannot expect to reduce Ψ to less than $(1-\rho)\Psi(z)$. The remaining terms in (2.13) are cubic in Δz^p if and only if λ is chosen as $\lambda=\rho$. In this case the remaining terms reduce to

$$\lambda^3 D^2\Psi(z)[\Delta z^p, D\Psi(z)^{-1}R_z(\Delta z^p)]+\lambda^4 R_z(D\Psi(z)^{-1}R_z(\Delta z^p)).$$

We point out, however, that the value of the step length ρ in our algorithm depends on the prior choice of λ (determining the search direction), and in our experiments, larger values of λ resulted in larger values of ρ and thus in faster convergence. In some examples, setting $\lambda=0.5(1+\rho^p)$, where ρ^p is determined from the predictor step Δx^p, improved the number of iterations by one. Overall, however, the value $\lambda=1$ generated steps that allowed larger values of ρ without violating the positivity requirements $s>0, t>0$, and therefore, the optimal choice of λ might be close or equal to one.

Another option, recently proposed by Andersen and Ye [2], is to apply the Mehrotra corrector only to the complementarity condition, but not to the nonlinear equalities. This approach somewhat simulates the strategy of SQP methods that solve the complementarity condition exactly for each subproblem. In the future we plan to use a hybrid approach and to use the full Mehrotra corrector for the complementarity condition and a damped corrector term for the nonlinear equalities.

2.3. The duality parameter. As mentioned, apart from the choice of the step length, the only degree of freedom in Algorithm 2.1 is the choice of σ at each iteration. We use an adaptive strategy that depends on a number of parameters specified in some input parameter file. To simplify the presentation, we denote some of the constants by their default values rather than giving them an extra name. The value of σ at iteration k is defined as

$$\sigma_k=\begin{cases}\frac{3}{4} & \text{if } \|\text{res}_3\|_1<(\|\text{res}_1\|+\|\text{res}_2\|)/1000,\\ \frac{1}{2} & \text{if } \|\text{res}_3\|_1\geq(\|\text{res}_1\|+\|\text{res}_2\|)/1000 \text{ and } \beta>30,\\ \min\{0.6,(1-\tilde{\rho}_{k-1})_1^{\text{const}}\} & \text{else.}\end{cases}\tag{2.14}$$

In the above expression, the number $\tilde{\rho}_{k-1}$ is the actual step length taken along Δx in the previous iteration $k-1$, the number $\beta = \beta(s,t)$,

$$\beta(s,t) = \frac{s^T t}{m \min_i s_i t_i} - 1, \tag{2.15}$$

measures the size of the residual (2.3), and the quantities $\mathrm{res}_1, \mathrm{res}_2, \mathrm{res}_3$ are evaluated at x^k, s^k, t^k. The constant const_1 was set to 3 in our experiments. The remaining constants used in (2.14), i.e., the numbers $\frac{3}{4}$, 1000, $\frac{1}{2}$, 30, and 0.6, are arbitrary in

$$(\mathrm{const}_2, 1) \times [1, \infty) \times (0, \mathrm{const}_2] \times (1, \infty) \times (0, \mathrm{const}_2]$$

for some $\mathrm{const}_2 \in (0,1)$ and can also be changed with a parameter input file; the given numbers are merely the default values. While Mehrotra's corrector was quite insensitive to the reduction σ of r, a stronger reduction of r like $\sigma \le 0.1$ increased the number of iterations for the plain primal-dual method. For the example in Table 2.1 of section 2.1, the method with $\sigma \le 0.1$ took 28 iterations compared to 18 with $\sigma \le 0.6$. The strategy of determining σ_k from $\tilde{\rho}_{k-1}$ is motivated by the observation that the convergence behavior of Algorithm 2.1 was fairly monotone, in the sense that when a long step was possible at iteration $k-1$, then, typically, the same was true for iteration k. Since long steps are taken only when σ is small, the above strategy is an adaptive rule based on the previous iteration. In the final iterations this strategy always resulted in a fast (superlinear) convergence in all our experiments.

2.4. The line search. The most delicate detail of the implementation of Algorithm 2.1 turned out to be the choice of an appropriate step length ρ at each iteration. The line search forms the longest subroutine in our program, and this fact is also reflected by the somewhat lengthy description below, where we summarize the principles used during the line search.

At first, we compute some auxiliary quantities.

- Determine $\bar{\lambda}_s$ and $\bar{\lambda}_t$ as the largest numbers that are less or equal to 1 and such that $s + \bar{\lambda}_s \Delta s \ge 0$ and $t + \bar{\lambda}_t \Delta t \ge 0$.
- Choose a constant $\mathrm{const}_3 \in [0,1]$ and
  multiply Δs by $(\bar{\lambda}_s)^{\mathrm{const}_3}$,
  Δt by $(\bar{\lambda}_t)^{\mathrm{const}_3}$, and
  Δx by $((\bar{\lambda}_s)^{\mathrm{const}_3} + (\bar{\lambda}_t)^{\mathrm{const}_3})/2$.
  Define
  $$\hat{\lambda}_s = (\bar{\lambda}_s)^{1-\mathrm{const}_3}, \qquad \hat{\lambda}_t = (\bar{\lambda}_t)^{1-\mathrm{const}_3}.$$
  Here, $\hat{\lambda}_s$ and $\hat{\lambda}_t$ are the maximum feasible steps along (the new) Δs and Δt. We use this rescaling of the search direction to be able to take different steps in the variables s and t (namely, when $\mathrm{const}_3 > 0$), while formally still taking a step of the same length in all variables. Only in the case that $\beta > 30$ (see (2.15)), we restrict ourselves to equal steps in x and s ($\mathrm{const}_3 = 0$). In our test runs, the choice $\mathrm{const}_3 = 1$ overall resulted in the fastest algorithm.
- Next, investigate the step $(x^{k+1}, s^{k+1}, t^{k+1}) = (x^k, s^k, t^k) + \rho(\Delta x^k, \Delta s^k, \Delta t^k)$, and find the smallest step length $\rho = \hat{\rho} \in (0,1]$ that is a local minimizer of
  $$\delta(\rho) := \sqrt{\mathrm{res}_1(x + \rho\Delta x,\ t + \rho\Delta t)^2 + \mathrm{res}_2(x + \rho\Delta x,\ s + \rho\Delta s)^2}. \tag{2.16}$$
  Define the reduction of the residuals resulting from a step of length $\hat{\rho}$ by
  $$\bar{\sigma} = \delta(\hat{\rho})/\delta(0). \tag{2.17}$$

By our choice of the Euclidean norm, finding $\hat{\rho}$ is cheap. If all constraints are quadratic, it amounts to minimizing just one scalar polynomial of degree 4. Above, the quantity $\hat{\rho}$ refers to the same step length in Δs and Δt. Note, however, that Δs and Δt may have been rescaled if $\text{const}_3 > 0$.

The step length $\tilde{\rho}$ used for the update

$$(x^{k+1}, s^{k+1}, t^{k+1}) = (x^k, s^k, t^k) + \tilde{\rho}(\Delta x^k, \Delta s^k, \Delta t^k)$$

is defined as the maximum step that satisfies the following restrictions.

1. Allow long steps only in the case when the primal and dual residuals are reduced significantly. More precisely, restrict $\tilde{\rho} \le \hat{\rho}$ (see (2.16)) and $\tilde{\rho} \le \bar{\sigma} \min\{\hat{\lambda}_t,\ \hat{\lambda}_s\}$ unless the norm of $\text{res}_3(s,t)$ happens to be larger than the sum of the norms of $\text{res}_1(x,t)$ and $\text{res}_2(x,s)$. In the latter case allow for a moderate[3] increase of the norms of $\text{res}_1(x,t)$ and $\text{res}_2(x,s)$.
2. Restrict $\tilde{\rho} \le (1-\sigma)\min\{\hat{\lambda}_t,\ \hat{\lambda}_s\}$; i.e., allow long steps only in the case when the duality parameter r has been significantly reduced.
3. Restrict $\tilde{\rho} \le \bar{\rho}$, where $\bar{\rho}$ is the largest number in $[0, \min\{\hat{\lambda}_s, \hat{\lambda}_t\}]$ such that $\beta(s + \rho\Delta s, t + \rho\Delta t) \le 60$ for all $\rho \in [0, \bar{\rho}]$. Here, 60 is a default value larger than the value 30 in the definition (2.14) of σ_k, and β is as in (2.15).
4. If $\|\text{res}_3\|_1 < (\|\text{res}_1\| + \|\text{res}_2\|)/1000$, use the following safeguard: set $\text{const}_3 = 0$ and restrict $\tilde{\rho} \le \breve{\rho}$, where $\breve{\rho}$ is the largest number in $[0,1]$ with

$$\frac{(s+\rho\Delta s)^T(t+\rho\Delta t)}{s^T t} \ge \frac{\delta(\rho)}{\delta(0)} \qquad \text{for all} \quad \rho \in [0, \breve{\rho}].$$

The number 1000 is the same default value as in (2.14).

2.5. The starting point. We conclude this section by outlining our procedure for finding a starting point. We focus our attention to the special case of problem (1.8) and point out some extensions to the general case of (2.1). First, fix x^0 (e.g., $x^0 = 0$) and set

$$H = \sum_{i=1}^{m} \nabla^2 g_i(x^0) + \nabla g_i(x^0) \nabla g_i(x^0)^T \qquad \text{and} \qquad \nu = \sqrt{c^T H^{-1} c}.$$

In the notation of (1.8) this is

$$H = \sum_{i=1}^{m} b_i b_i^T \qquad \text{and} \qquad \nu = \sqrt{f^T H^{-1} f}.$$

Note that H is nonsingular because of (1.2) and thus ν is well defined. Then define $\hat{r} = \sqrt{m}\nu$, $t^0 = 3e/n$, and $r_0 = \left\|t^0\right\|_\infty \hat{r}$. Finally, set $s^0 = \hat{r}e$. Obviously, $(t^0)^T s^0/m = r_0$.

In the above procedure, the factorization of H used to compute ν can be used in the first iteration of the algorithm so that the cost of finding the starting point is essentially $O(m)$.

We note that the seemingly better choice of $t^0 = e/m$, satisfying relation (3.1) and preserving $r_0 = \hat{r}$, turned out to be slightly less efficient in all tested examples.

[3] We suppress the details of how to interpret "moderate"; we assume, however, that the norms of res_1 and res_2 remain bounded.

Our choice above was motivated by the idea that, at the optimal solution, the largest components of t are at least of the size $1/n$, and that it is "better" to start with a too large starting point rather than with a too small one. The constant 3 in the above definition of t is the default value used in our numerical experiments and can be changed by an input parameter file.

The choice of the initial duality parameter $\hat{r}$ is motivated by the primal problem. For problem (1.8) the initial point $x^0 = 0$ is the analytic center of the (primal) feasible set. Thus, if $\hat{r}$ was set to $\hat{r} = m\nu$, the H^{-1}-norm of the initial gradient of the primal problem was equal to one. If the H^{-1}-norm was less than $1/4$, for example, the theory of self-concordant functions would imply quadratic convergence of Newton's method to the analytic center. However, since the theory is overly pessimistic in general, we reduce $\hat{r}$ by a factor of $\sqrt{m}$. Again, the factor of $\sqrt{m}$ is an input parameter and can be changed to any power of m. For a general problem (2.1) for which $x^0 = 0$ is not a central point or not even feasible, it seems meaningful to add a term to $\hat{r}$ that models the infeasibility and to define, for example,

$$\hat{r} = \sqrt{m}\nu + \max\{0, \max_i\{g_i(x^0) + \sqrt{m}\nu\}\}.$$

(For (1.8) this is the same value as before.) The initial value of t for a general problem (2.1) can be chosen, for example, as a multiple of the all-ones-vector that minimizes the square norm of $\mathrm{res}_1(x^0, s^0, t^0)$ plus a multiple of the barrier term $-\sum \ln t_i$. (If the multiple is chosen as $18/n^2 - 6/(nm)$ we obtain the starting vector $t^0 = 3e/n$ for problem (1.8) above.)

The numerical example in section 2.2 referred to the above starting point and to the same starting point with r_0 and s^0 divided by 1000.

3. The special structure of problem (1.8). The KKT conditions for problem (1.8) can be written as follows: there exist vectors $s, t \in \mathbb{R}^m$, $s, t \geq 0$ such that

$$\sum_{i=1}^{m} t_i = 1, \tag{3.1}$$

$$\sum_{i=1}^{m} t_i (b_i^T x) b_i = f, \tag{3.2}$$

$$\frac{1}{2}(b_i^T x)^2 + \alpha + s_i = 0 \quad \text{for} \quad 1 \leq i \leq m, \tag{3.3}$$

$$s_i t_i = 0 \quad \text{for} \quad 1 \leq i \leq m. \tag{3.4}$$

Note that Slater's condition is trivially satisfied for (1.8), and hence the above conditions are necessary and sufficient for optimality.

3.1. Scaling the problem. Let us now analyze the properties that follow from the special structure of our problem and, in particular, its sensitivity to scaling of the data f, b_i. We identify linear transformations of the data f and b_i $(1 \leq i \leq m)$ in (1.8), such that the transformed problem is of the same structure as the original problem, and such that the solution of the original problem can immediately be recovered from the solution of the transformed problem. Such problems are called "equivalent." The transformations will also affect the quantities involved in the KKT conditions (3.1)–(3.4).

A natural choice to measure the closeness of some tuple $\alpha, x, s \geq 0, t \geq 0$ to an optimal solution of (1.8) might seem to require that conditions (3.1)–(3.4) hold up to

some tolerance ϵ. This, of course, is meaningful only if we can somehow eliminate the dependence of (3.1)–(3.4) on linear transformations that lead to equivalent problems. For this purpose we will introduce in the next section a "distance" function which is the balanced sum of the residuals of (3.1)–(3.4) and which is in some sense balanced under the following linear transformation leading to equivalent variants of (1.8).

In general, we may consider an arbitrary linear transformation of the variables x, α in (1.8). Transformations, however, that maintain the special structure of (1.8), namely, separability of x and α in all constraints, are of the form $\hat{x} := U^{-T}x$ and $\hat{\alpha} := \rho_1^{-1}\alpha$ with some nonsingular matrix U and some positive scalar ρ_1. Since the solution of the problem does not change when the objective function is multiplied by a positive scalar ρ_2 and the constraints are multiplied by positive scalars d_i, we obtain the following class of problems that are equivalent to (1.8) and depend on ρ_1, U, and on additional parameters $\rho_2 > 0$, $d_i > 0$ $(1 \le i \le m)$,

$$\min_{\hat{x}\in\mathbb{R}^n,\ \hat{\alpha}\in\mathbb{R}} \left\{ -\rho_2\rho_1\hat{\alpha} - \rho_2(Uf)^T\hat{x} \mid d_i \left(\frac{1}{2}\left((Ub_i)^T\hat{x}\right)^2 + \rho_1\hat{\alpha} \right) \le 0 \right\}. \tag{3.5}$$

For a solution $\hat{x}, \hat{\alpha}$ of (3.5), a solution to (1.8) is given by $x = U^T\hat{x}$ and $\alpha = \rho_1\hat{\alpha}$. The KKT conditions for (3.5) are

$$\begin{aligned} \sum_{i=1}^{m} \rho_1 d_i \hat{t}_i &= \rho_1\rho_2, \\ \sum_{i=1}^{m} d_i\hat{t}_i((Ub_i)^T\hat{x})Ub_i &= \rho_2 Uf, \\ d_i\left(\frac{1}{2}((Ub_i)^T\hat{x})^2 + \rho_1\hat{\alpha}\right) + \hat{s}_i &= 0 \quad \text{for} \quad 1 \le i \le m, \\ \hat{s}_i\hat{t}_i &= 0 \quad \text{for} \quad 1 \le i \le m. \end{aligned} \tag{3.6}$$

Clearly, for a given (x, α, s, t) and its transformed counterpart

$$(\hat{x}, \hat{\alpha}, \hat{s}, \hat{t}) = (U^{-T}x, \rho_1^{-1}\alpha, Ds, \rho_2 D^{-1}t),$$

the norms of the residuals (3.1)–(3.4) may differ a lot depending on the choice of ρ_1, ρ_2, U and $D = \mathrm{diag}(d_i)$.

3.2. Measuring closeness to optimality. Next we discuss suitable proximity measures for the "distance" from some point x, α, s, and t to the optimal solution of (1.8). A desirable property for the measure $\mathbf{d}$ is certainly that it should not change under the above linear transformations. On the other hand, we search for a simple measure based on some weighted norms of the residuals of (3.2)–(3.4). Here, the residual of (3.1) is neglected since (3.1) is the only linear equation among (3.1)–(3.4) and its residual is reduced the fastest in our interior-point method below. (Note that, given an approximate solution (x, α, s, t) of (3.1)–(3.4), by setting $\rho_2 = \sum t_i$, $\rho_1 = 1$, $U = I$, and $D = I$, we obtain an equivalent problem (3.5) for which $(\rho_2 x, \rho_2^2\alpha, \rho_2^2 s, t)$ is an approximate solution and the residual of (3.1) (resp., (3.6)) is zero while the residuals of (3.2)–(3.4) are multiplied by ρ_2 (resp., ρ_2^2).) Since Algorithm 2.1 generates only iterates that satisfy $s, t > 0$, we also restrict our measure to points with

$$s, t > 0,$$

and try to identify suitable weights and norms for the residuals of (3.2)–(3.4).

As we will use the measure in a line search, norms induced by some simple scalar product are most suitable. We assume that $\|f\| \neq 0$, and measure (3.2) by $\frac{1}{\|f\|}$ times the Euclidean norm which effects that the right-hand side of (3.2) has size 1. We would like to choose the remaining norms such that the weights of (3.2), (3.3), and (3.4) are "balanced" in some sense. Rather than normalizing the residual of (3.3) as in (2.6), we use the Euclidean norm of (3.3) divided by $\|s\|$ (note that $s > 0$, and hence $\|s\| \neq 0$). Since s_i and t_i are assumed to be positive and since $s^T t$ measures the duality gap (if the other residuals are zero), a suitably scaled multiple of $s^T t$ seems appropriate as a measure for (3.4). Choosing $\frac{s^T t}{\|s\|\|t\|}$, i.e., the cosine of the angle between s and t, results in a measure that is at most 1 and is thus[4] consistent with the measures for (3.2) and (3.3). For problem (1.8) we use the normalization of res_2 by $\frac{1}{\|s\|}$ in (3.7) instead of the one in (2.6) since it does not depend on the (arbitrary) choice of a starting point. For a general convergence analysis, however, the definition (2.6) is more appropriate. Likewise for the normalization of res_3. By further neglecting α due to its special role, the resulting distance function used in our stopping test is

$$\begin{aligned} \mathbf{d} &= \mathbf{d}(\alpha, x, s, t) \\ &:= \frac{1}{\|f\|} \left\| f - \sum_{i=1}^{m} t_i (b_i^T x) b_i \right\| + \frac{1}{\|s\|} \left\| \begin{bmatrix} \vdots \\ s_i + \frac{1}{2}(b_i^T x)^2 + \alpha \\ \vdots \end{bmatrix} \right\| + \frac{s^T t}{\|s\|\,\|t\|}. \end{aligned} \tag{3.7}$$

It is easy to verify that $\mathbf{d}$ is invariant under the above linear transformations if U in (3.5) is unitary and D is a positive multiple of the identity matrix. (The more ambitious goal to find a measure, that is invariant for all invertible U and diagonal $D > 0$ is not satisfied by the above simple measure $\mathbf{d}$.)

3.3. The linear systems for problem (1.8). To simplify the description of Newton's method, we use the notation

$$\tilde{x} = \begin{bmatrix} \alpha \\ x \end{bmatrix}, \qquad \Delta\tilde{x} = \begin{bmatrix} \Delta\alpha \\ \Delta x \end{bmatrix}$$

from now on. Given some iterate $\tilde{x}$, $s > 0$, $t > 0$ and some "centering parameter" $r = r_k$, it is straightforward to derive that the Newton direction for (3.1)–(3.3), (2.3) is given by the solution $\Delta\tilde{x}, \Delta t, \Delta s$ of

$$\begin{array}{rcrcrcl} Q\Delta\tilde{x} & + & A\Delta t & & & = & \mathrm{Res}^{(1)}, \\ A^T\Delta\tilde{x} & & & + & \Delta s & = & \mathrm{Res}^{(2)}, \\ & & S\Delta t & + & T\Delta s & = & \mathrm{Res}^{(3)}, \end{array} \tag{3.8}$$

where S and T are diagonal matrices (the diagonals given by the vectors s and t, as explained in section 1.2), and

$$Q = Q(t) = \sum_{i=1}^{m} t_i \begin{bmatrix} 0 \\ b_i \end{bmatrix} \begin{bmatrix} 0 \\ b_i \end{bmatrix}^T, \qquad A = A(x) = \left[\begin{bmatrix} 1 \\ (b_1^T x) b_1 \end{bmatrix} \cdots \begin{bmatrix} 1 \\ (b_m^T x) b_m \end{bmatrix} \right],$$

[4] It is somewhat disturbing that the function $\cos(s,t)$ has slope zero at the (standard) initial values $s = s^0 = \rho_1 e$, $t = t^0 = \rho_2 e$ for some $\rho_1, \rho_2 > 0$. To "regularize" this behavior for parallel s, t, one might therefore consider "$1 - \frac{2}{\pi}\arccos\frac{s^T t}{\|s\|\|t\|}$" instead of $\frac{s^T t}{\|s\|\|t\|}$.

$$\mathrm{Res}^{(1)} = \begin{bmatrix} \mathrm{Res}^{(1,1)} \\ \mathrm{Res}^{(1,2)} \end{bmatrix} = \begin{bmatrix} 1 - \sum_{i=1}^{m} t_i \\ f - \sum_{i=1}^{m} t_i (b_i^T x) b_i \end{bmatrix},$$

$$\mathrm{Res}_i^{(2)} = -s_i - \alpha - \frac{1}{2}(b_i^T x)^2 \quad \text{for} \quad 1 \le i \le m, \tag{3.9}$$

and

$$\mathrm{Res}_i^{(3)} = r - s_i t_i \quad \text{for} \quad 1 \le i \le m.$$

Since T and S are diagonal, it is efficient to reduce (3.8) to the smaller system

$$(Q + ATS^{-1}A^T)\Delta\tilde{x} = \mathrm{Res}^{(1)} + AS^{-1}(T\mathrm{Res}^{(2)} - \mathrm{Res}^{(3)}). \tag{3.10}$$

Note that due to (1.3) this is a positive definite system. Recovering the vectors Δs and Δt from $\Delta\tilde{x}$ can be done by using the *stable reduction* to the Schur complement introduced in [9]. The systems matrix in (3.8) is very large and sparse, while the matrix in (3.10) is of moderate dimension but rather dense. We therefore used sparse data structures for A and a full matrix for $Q + ATS^{-1}A^T$ and its Cholesky decomposition.

4. Convergence analysis. A proof of global convergence of a potential reduction interior-point method for convex programs (and certain classes of constrained equations) is given, for example, in [27]. The proof below is intended to provide a guarantee that the "technical details" of this paper are consistent, in particular, to show that the line search in section 2.4 allows a proof of global convergence contrary to the numerical example in section 2.2, where the result of the line search was overruled by the demand to go at least 90% towards the boundary, a practice that is efficient for linear programs [18].

For simplicity of presentation, we restrict our analysis to the case of equal step lengths in the variables s and t ($\mathrm{const}_3 = 0$) and assume that the safeguard rule (4.1) below is applied. The formulation of the safeguard is more complicated for the case of different steps in Algorithm 2.1 and is therefore omitted in section 2.4. Nevertheless, our proof below can be extended to this situation analogously as in (5b), (6b) of Kojima, Megiddo, and Mizuno [15].

We further restrict our analysis to the case where the following assumptions are satisfied.

(A) For $s, t > 0$ the linear systems (2.4) are nonsingular.

(B) The iterates generated by Algorithm 2.1 are bounded.

Assumption (B) is the weak point in our analysis; in general it will be difficult to verify this assumption. Nevertheless, as far as we know, all proofs of convergence of long step interior-point methods in the literature use Assumption (B) (and provide sufficient criteria as to when (B) is satisfied). Also in Lemma 4.1 below, we will show that both assumptions (A) and (B) are indeed satisfied for problem (1.8).

In addition to the line search requirements in section 2.4, we further require the descent condition (5b) of Kojima, Megiddo, and Mizuno [15] for the line search[5]:

- Let $\tilde{\rho}$ be the max. number $\le \min\{\bar{\rho},\ \hat{\rho},\ (1-\sigma)\min\{\hat{\lambda}_t, \hat{\lambda}_s\},\ \bar{\sigma}\min\{\hat{\lambda}_t, \hat{\lambda}_s\}\}$ such that

$$(s + \rho\Delta s)^T(t + \rho\Delta t) \le \left(1 - \rho\left(1 - \frac{1+\sigma_k}{2}\right)\right) s^T t \tag{4.1}$$

[5]The paper [15] allows for different steps in s and t (only for linear programs) and therefore complements (5b) by a second condition (6b).

for all $\rho \in [0, \tilde{\rho}]$.

LEMMA 4.1. *Assumptions* (A) *and* (B) *are satisfied for problem* (1.8) *if the set of* b_i $(1 \le i \le m)$ *spans all of* $\mathbb{R}^n$.

Proof. Regularity of the systems defining the Newton direction follows from the formulation (3.10) and the fact that the b_i span all of $\mathbb{R}^n$.

Since condition (3.1) is linear, it follows from Newton's method that

$$\sum_{i=1}^{m} t_i^k \in \left[\min\left\{1, \sum_{i=1}^{m} t_i^0\right\},\ \max\left\{1, \sum_{i=1}^{m} t_i^0\right\}\right] \tag{4.2}$$

for all $k \ge 0$ and, by nonnegativity, the multipliers t^k are bounded. For simplicity of notation, we tighten relation (4.2) and assume from now on $\sum_{i=1}^{m} t_i^k = 1$ for all k.

By (4.1), $(s^k)^T t^k$ is decreasing, $(s^k)^T t^k \le M$ for all k and some sufficiently large fixed value M. Likewise, by the choice of $\hat{\rho}$ in the line search, the Euclidean norm of $\begin{pmatrix} \mathrm{res}_1 \\ \mathrm{res}_2 \end{pmatrix}$ is bounded for all iterations k. By enlarging M, we may assume without loss of generality that M is also an upper bound for the norms of the residuals of (3.2) and (3.3).

If α_k is bounded below, then s^k is bounded (else res_2 in (3.3), (2.6) became unbounded) and, likewise, x^k is bounded because else $(b_i^T x)^2$ was unbounded for at least one index i and again res_2 became unbounded.

If α_k is unbounded below, we will derive a contradiction. This contradiction is somewhat tedious to develop and fills the remainder of this proof: let k_j be a subsequence such that $\alpha_{k_j} \le -j^2$. For $j > 2M$ it follows that

$$\frac{1}{2}(b_i^T x^{k_j})^2 \le -\left(1 + \frac{1}{j}\right)\alpha_{k_j} \qquad \text{for all } i, \tag{4.3}$$

because if there existed $\bar{i}$ violating (4.3), then

$$\|\mathrm{res}_2\| = \left\| \left(\frac{1}{2}(b_i^T x^{k_j})^2 + \alpha_{k_j} + s_i^{k_j} \right)_{i=1\ldots m} \right\| \ge \frac{1}{2}(b_{\bar{i}}^T x^{k_j})^2 + \alpha_{k_j} \ge j > 2M, \tag{4.4}$$

which is a contradiction. Tightening (4.3) to the stronger inequality

$$\frac{1}{2}(b_i^T x^{k_j})^2 \le -\left(1 - \frac{1}{j}\right)\alpha_{k_j} \qquad \text{for all } i,$$

however, is wrong. If it was true and

1. if $s_i^{k_j} \ge \frac{1}{2} j$ for all i, then from $\sum t_i = 1$ it follows that $(s^{k_j})^T t^{k_j} \ge \frac{1}{2} j > M$, which is a contradiction, and
2. if $s_{\bar{i}}^{k_j} \le \frac{1}{2} j$ for some $\bar{i}$, then as in (4.4),

$$\left\| \left(\frac{1}{2}(b_i^T x^{k_j})^2 + \alpha_{k_j} + s_i^{k_j} \right)_{i=1\ldots m} \right\| \ge \frac{1}{2} j > M \tag{4.5}$$

is a contradiction.

Hence there exists a nonempty set $I_j \subset \{1, \ldots, n\}$ with

$$\frac{1}{2}(b_i^T x^{k_j})^2 \in \left[-\left(1 - \frac{1}{j}\right)\alpha_{k_j},\ -\left(1 + \frac{1}{j}\right)\alpha_{k_j} \right] \qquad \text{for } \ i \in I_j, \tag{4.6}$$

and $\frac{1}{2}(b_i^T x^{k_j})^2 < -(1-\frac{1}{j})\alpha_{k_j}$ for $i \notin I_j$. Since there are only finitely many different index sets I_j, there is a subsequence j_l for which $I_{j_l} \equiv \bar{I}$ is constant. By taking another subsequence, if necessary, we may assume that for fixed $i \in \bar{I}$ the sign of $b_i^T x^{k_j}$ is constant for all $j = j_l$. When changing from b_i to $-b_i$, the problem (1.8) does not change, and without loss of generality we therefore assume $b_i^T x^{k_{j_l}} \geq 0$ for all $i \in \bar{I}$ and all natural numbers l.

To simplify the notation, we omit the index j and write l in place of j_l from now on. For $i \notin \bar{I}$ follows $s_i^{k_l} \geq l/2$ by (4.5). From the bound on res_3 it follows that $t_i^{k_l} \leq 2M/l$ for $i \notin \bar{I}$. We obtain

$$\begin{aligned} M &\geq \left\| f - \sum_{i\in\bar{I}} t_i^{k_l}(b_i^T x^{k_l}) b_i - \sum_{i\notin\bar{I}} \underbrace{t_i^{k_l}}_{\leq 2M/l} \underbrace{(b_i^T x^{k_l})}_{\leq\sqrt{2|\alpha_{k_l}|}} b_i \right\| \\ &\geq \left\| \sum_{i\in\bar{I}} \underbrace{t_i^{k_l}(b_i^T x^{k_l})}_{=:\nu_i>0} b_i \right\| - \|f\| - m\frac{2M}{l}\sqrt{2|\alpha_{k_l}|}\max_i \|b_i\| \\ &\geq \|\vec{\nu}\|\,\sigma_{min}\left((b_i)_{i\in\bar{I}}\right) - \|f\| - m\frac{2M}{l}\sqrt{2|\alpha_{k_l}|}\max_i \|b_i\|, \end{aligned}$$

where σ_{min} denotes the minimal singular value of a matrix, and $\vec{\nu}$ is the vector with components ν_i for $i \in \bar{I}$. Since $b_i^T x^{k_l} \geq \sqrt{|\alpha_{k_l}|}$ and

$$\max_{i\in\bar{I}} t_i^{k_l} \geq \left(\underbrace{\sum_{i=1}^{m} t_i^{k_l}}_{=1} - \ (m - |\bar{I}|) \underbrace{\frac{2M}{l}}_{\text{bound for } t_i,\ i\notin\bar{I}} \right) / |\bar{I}| \geq \epsilon > 0$$

for sufficiently large l, the norm $\|\vec{\nu}\| \geq \epsilon\sqrt{|\alpha_{k_l}|}$ grows faster (when $l \to \infty$) than

$$m\frac{2M}{l}\sqrt{2|\alpha_{k_l}|}\max_i \|b_i\|,$$

and we conclude that $\sigma_{min}\left((b_i)_{i\in\bar{I}}\right) = 0$; since $\nu_i > 0$, there exists a nonnegative vector $\vec{\lambda} \neq 0$ such that $\sum_{i\in\bar{I}} \lambda_i b_i = 0$. Now,

$$\sum_{i\in\bar{I}} \lambda_i \underbrace{b_i^T x^{k_l}}_{\in\left[\sqrt{2(1-\frac{1}{l})|\alpha_{k_l}|},\ \sqrt{2(1+\frac{1}{l})|\alpha_{k_l}|}\ \right]} = 0$$

yields $\sum \lambda_i = 0$ (when $l \to \infty$) in contradiction to $\lambda_i \geq 0$ and $\vec{\lambda} \neq 0$. This completes our proof. ∎

Under the assumptions (A) and (B) we now show the following theorem.

THEOREM 4.2. *The sequence of iterates* (x^k, s^k, t^k) *generated by Algorithm* 2.1 *and parameter updates as in sections* 2.3 *and* 2.4 *has accumulation points. Each accumulation point is an optimal solution of* (1.8).

(We do not explicitly assume Slater's condition. We believe, however, that assumption (A) might not be satisfied if Slater's condition is violated.)

Proof. For simplicity we use the default values of the parameters from sections 2.3 and 2.4. Since (x^k, s^k, t^k) are bounded, the existence of accumulation points is evident.

Let $(\bar{x}, \bar{s}, \bar{t})$ be an accumulation point and assume by contradiction that $\mathbf{d}(\bar{x}, \bar{s}, \bar{t}) > 0$. By considering a subsequence if necessary, we assume that the sequence of iterates $(x^k, s^k, t^k) \to (\bar{x}, \bar{s}, \bar{t})$ converges. By the safeguard in Step 4 of the line search it follows from $\mathbf{d}(\bar{x}, \bar{s}, \bar{t}) > 0$ that $\bar{s}^T \bar{t} > 0$. Let $\bar{s}^T \bar{t} = \epsilon_1$. Further, from the bound on β (2.15) guaranteed by Step 3 of the line search it follows that

$$\frac{(s^k)^T t^k}{m \min s_i^k t_i^k} - 1 \leq 60$$

for all k, and therefore $\min \bar{s}_i \bar{t}_i \geq \frac{\bar{s}^T \bar{t}}{61m}$. It follows that there is some $\widetilde{M} > 0$ such that $\bar{s}_i,\ \bar{t}_i \in [\frac{1}{\widetilde{M}},\ \widetilde{M}]$ for all $i \in \{1, \ldots, m\}$. Since x^k is bounded and s^k, t^k converge, we may assume, without loss of generality,

$$s_i^k,\ t_i^k \in \left[\frac{1}{2\widetilde{M}},\ 2\widetilde{M}\right], \qquad \left\|x^k\right\| \leq \widetilde{M} \qquad \text{for all } k. \tag{4.7}$$

On the compact set (4.7), the Jacobian in (2.4) is nonsingular, and hence its inverse is bounded. Boundedness of the right-hand side in (2.4) implies a uniform bound on the corrections,

$$\left\|(\Delta x^k,\ \Delta s^k,\ \Delta t^k)\right\| \leq \widehat{M}$$

for some fixed $\widehat{M} > 0$. Hence, and by (4.7), there is some $\epsilon_2 > 0$ such that $\bar{\lambda}_s,\ \bar{\lambda}_t \geq \epsilon_2$ for all iterations k.

If the safeguard in Step 4 of the line search is active, then $\sigma_k = 3/4$, and from the third block row of (2.4) with $r = \frac{3}{4}\frac{s^T t}{m}$ follows $s^T \Delta t + t^T \Delta s = -\frac{1}{4} s^T t$. Hence,

$$\frac{(s + \rho \Delta s)^T (t + \rho \Delta t)}{s^T t} = 1 - \rho/4 + \rho^2 \frac{\Delta s^T \Delta t}{s^T t}, \tag{4.8}$$

while

$$\frac{\delta(\rho)}{\delta(0)} = 1 + \rho \frac{\delta'(0)}{\delta(0)} + \rho^2 \frac{\delta''(\xi)}{\delta(0)}. \tag{4.9}$$

Here,

$$\delta(\rho) = \left(\frac{1}{\|c\|^2} \|c + \sum (t_i + \rho \Delta t_i) \nabla g_i (x + \rho \Delta x)\|^2 + \frac{1}{(1 + \|g(x^0)\|)^2} \|g(x + \Delta x) + s + \rho \Delta s\|^2 \right)^{1/2}$$

and ξ is some intermediate point $\xi \in (0, \rho)$. Note that $\delta'(0) = -\delta(0)$ by the definition of $(\Delta x, \Delta s, \Delta t)$ in (2.4). The denominator $\delta^{3/2}(\xi)$ of the second derivative $\delta''(\xi)$ is bounded below by $\epsilon_1^{3/2}$, and the numerator is uniformly bounded by continuity of g and its derivatives, and the bounds on the iterates and the search directions. Hence, the coefficients of the ρ^2-terms in (4.8) and (4.9) are bounded and there exists $\epsilon_3 > 0$ independent of k with $\breve{\rho} \geq \epsilon_3$.

From C^3-smoothness of the constraints also follows that there is some $\epsilon_4 > 0$ such that $\hat{\rho} \geq \epsilon_4$ and $\bar{\sigma} \geq \epsilon_4$; both numbers equal 1 if g_i are linear. If $\delta(0) \geq \epsilon_1$ the existence of a uniform lower bound $\hat{\rho} > \epsilon_4 > 0$ follows from the same argument as

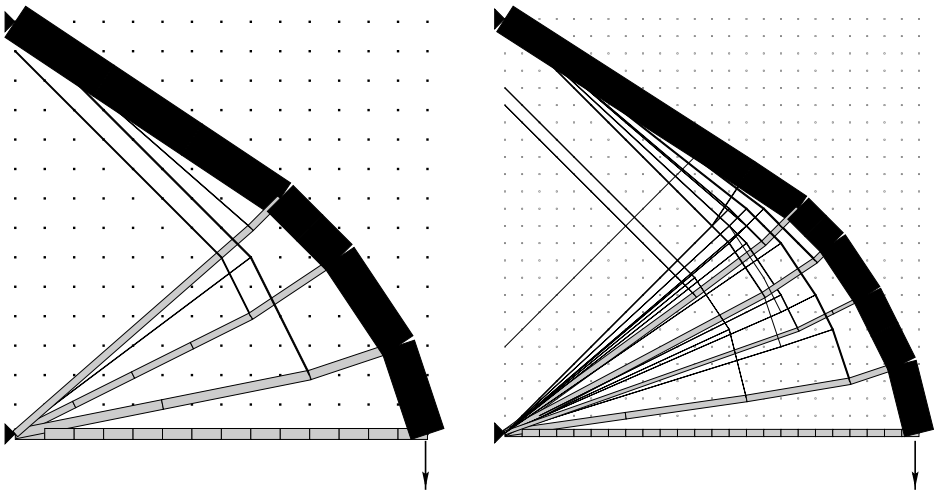

FIG. 5.1. *Optimal topologies for the smallest and largest example.*

used in (4.9). Since the line search allows a "moderate" increase in the norm of the residuals, the analogous bound follows when $\delta(0) < \epsilon_1$.

Finally, $(1 - \sigma_k) \geq 1/4$, and analogously to condition (12) in [15], there exists $\epsilon_5 > 0$ such that $\tilde{\rho}$ in (4.1) satisfies $\tilde{\rho} > \epsilon_5$. Likewise, since $\sigma_k = 1/2$ (bounded away from zero) when $\beta > 30$, it follows as for condition (13a) in [15] that $\bar{\rho} > \epsilon_5$ as well.

Summarizing, independently of k, the step taken along $(\Delta x^k, \Delta s^k, \Delta t^k)$ has length at least $\min\{\epsilon_2, \epsilon_3, \epsilon_4, \epsilon_5\} > 0$. Thus, since $\sigma_k \leq 3/4$, (4.1) implies that $s^T t$ is reduced by a fixed fraction (independent of k) at each iteration in contradiction to $\bar{s}^T \bar{t} = \epsilon_1 > 0$.

Hence, $\mathbf{d}(\bar{x}, \bar{s}, \bar{t}) = 0$, and $(\bar{x}, \bar{s}, \bar{t})$ satisfies the first-order sufficient conditions for optimality. □

Remark. The above proof of convergence extends in a straightforward manner to the case when Mehrotra's corrector is used for the complementarity condition only (as suggested by Andersen and Ye [2]), and a safeguard is used to guarantee the line search conditions in section 2.4. The extension of the preceding rules to the Mehrotra corrector as used in our program is not possible. As pointed out, the Mehrotra corrector may locally increase the residuals res_1 and res_2, and $\hat{\rho}$ may be zero. In the program for the numerical experiments below we used the Mehrotra corrector whenever it resulted in a reduction of $\mathbf{d}$ without violating the constraint $\beta \leq 60$. Else we reverted to a plain primal-dual method as analyzed above. In such very rare cases where the plain primal-dual method was used, however, convergence became slow.

Not all the rules of the line search in section 2.4 are needed to establish convergence. For example, the bounds based on $\bar{\sigma}$ or the restriction in Step 2 of the line search only complicate the above proof but are never really used. In practice, these rules help prevent that the bounds in Steps 3 and 4 become activated.

5. Numerical experiments. An implementation based on Algorithm 2.1 was tested by us on a number of examples derived from the design of cantilever trusses. We consider examples with the same geometry and physical data and with regularly increasing numbers of nodes (unknowns) and potential bars (constraints). In Figure 5.1 we show the optimal topologies for our smallest and largest example. The

TABLE 5.1

Example		Ben-Tal – Zibulevsky				Algorithm 2.1			
n	m	CPU	CPU-FAC	ITER	NEWT	CPU	CPU-FAC	ITER	FAC
450	15556	140		18	36	31		18	4.5
578	25456	262	1.9	16	31	86	2.8	23	3.0
722	39724	677	2.6	18	44	220	2.6	31	3.1
882	59456	1235	1.8	17	46	384	1.7	29	3.2
1058	85252	2390	1.9	19	50	817	2.1	36	2.9
1250	119040	4387	1.8	18	63	1410	1.7	38	3.1
1458	161932	6840	1.6	20	60	2580	1.8	46	2.7
1682	215136					4203	1.6	51	
1922	280916					7497	1.8	54	
2178	361328					13400	1.8	66	

growth of the number of iterations and the growth in computation time is monitored as a function of the number of constraints and number of unknowns.

We compare our results based on Algorithm 2.1 with those obtained by a penalty/barrier algorithm due to Ben-Tal and Zibulevsky [5]. The results are summarized in Table 5.1. In the first two columns we see the data of the examples: number n of unknowns and number m of quadratic constraints. The next columns contain the results obtained by the code due to Ben-Tal and Zibulevsky and by Algorithm 2.1. Column CPU refers to the CPU time (in seconds) on a workstation HP9000-735. CPU-FAC is the ratio of CPU times for two subsequent examples. ITER refers to the number of iterations while NEWT refers to the global number of Newton steps. Recall that in Algorithm 2.1 these two numbers are equal. Let us add that the work for a Newton step is about the same in both algorithms. The last column in Table 5.1 shows the ratio of CPU times for the two algorithms. We remark that the comparison of CPU times should not be taken too seriously since both algorithms are in a "research" stage and are not tuned with respect to CPU time. The last three examples were only computed with Algorithm 2.1.

The last example with about 361328 nonlinear constraints is certainly large scale. We stress again that, even though the test problems of this paper can be reformulated as linear programs, our algorithm did not use such a reformulation but entirely relied on techniques devised for nonlinear programs.

The cantilever truss seems to be "difficult" for both codes, which might be due to the distribution of forces and boundary conditions. We have computed another series of examples for a structure carrying a bridge, which is fixed at its left and right endpoints and whose street-nodes are under vertical loads. Figure 5.2 shows as an example a ground-structure mesh of 7×5 nodes and 386 bars. Note that we do not make any a priori assumptions on the shape of the bridge; the shape itself should be the outcome of the optimization process. For this type of truss, the ratio between the two codes was about the same as for cantilever truss. For example, for a ground-structure with $n = 506$ and $m = 19500$, the code due to Ben-Tal and Zibulevsky needed 167 seconds, 15 iterations, and 31 Newton steps, while Algorithm 2.1 needed 16 iterations in 41 seconds. And for the largest example we have computed ($n = 1408, m = 150031$) Algorithm 2.1 needed only 20 iterations (compare to 46 iterations for the cantilever truss of about the same dimension!). The optimal design for the largest example is shown in Figure 5.3. Obviously, an "arch" seems to be the optimal structure for carrying the street.

Finally, we present two results on design of three-dimensional cantilever trusses. Figure 5.4 shows the initial data: the geometry (a cube), the load, and the fixed nodes

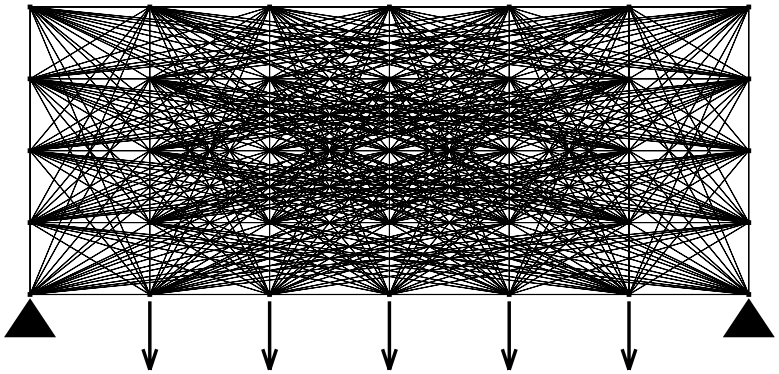

FIG. 5.2. *Initial layout for a bridge fixed at its end-points.*

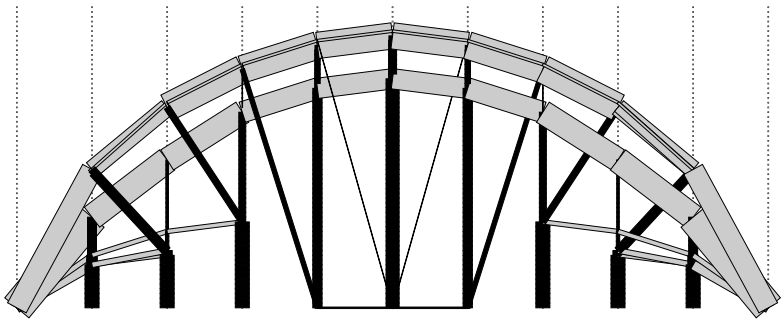

FIG. 5.3. *Bridge fixed at its end-points.*

(on the shaded surface). Obviously, the complexity of three-dimensional problems is much higher. Table 5.2 presents the size, the number of iterations, and CPU time for two examples with $7 \times 7 \times 7$ and $9 \times 9 \times 9$ nodes, respectively. The optimal structures are depicted in Figure 5.5. Note that the size (number of variables) of the second example is about the same as the size of the largest two-dimensional example (Table 5.1). However, the $9 \times 9 \times 9$ "discretization" is much coarser than even the smallest two-dimensional example (15×15); cf. Figures 5.1 and 5.5. This explains the small number of iterations needed to solve the (large) three-dimensional examples: for a fine grid, there are many nearly optimal designs, and the interior-point method needs more iterations to identify the true one. If (for a coarse mesh) all other design candidates are far from optimality, then the interior-point method will find out and converges quadratically near the optimal solution.

TABLE 5.2

Example		Algorithm 2.1	
n	m	CPU	ITER
1029	48426	450	18
2187	220280	3131	16

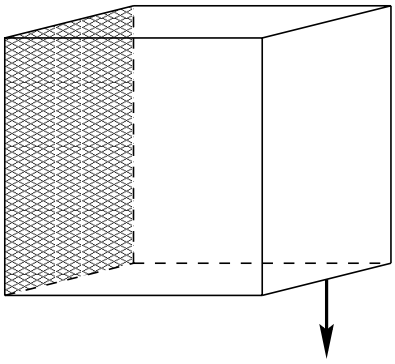

FIG. 5.4. *Initial data for three-dimensional cantilever truss.*

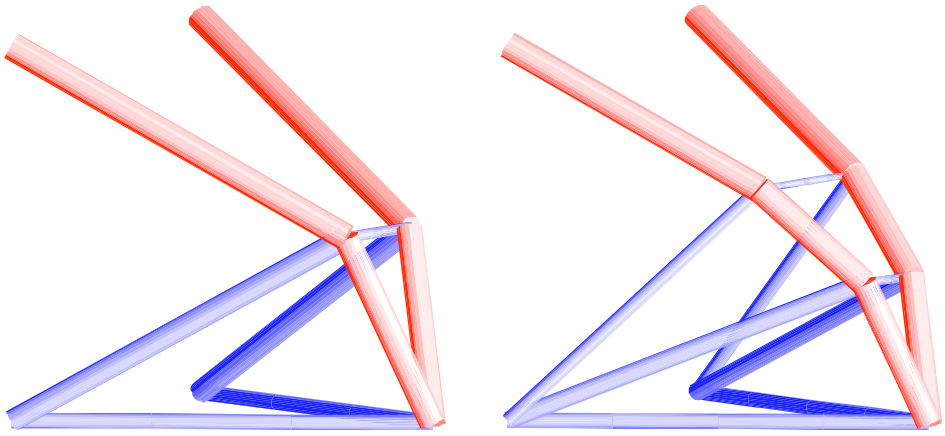

FIG. 5.5. *Optimal designs for two three-dimensional cantilever trusses.*

Acknowledgments. The authors would like to thank Aharon Ben-Tal and Michael Zibulevsky from Technion, Israel, for making their code available to them.

Part of this work was done while the first author was visiting the Institute of Statistical Mathematics and the Tokyo Institute of Technology in Tokyo, Japan, and he would like to thank Professor Shinji Mizuno and Professor Kojima for their warm hospitality and Professor Takashi Tsuchiya for many helpful discussions about this article.

REFERENCES

[1] W. ACHTZIGER, M. P. BENDSØE, A. BEN-TAL, AND J. ZOWE, *Equivalent displacement based formulations for maximum strength truss topology design*, Impact Comput. Sci. Engrg., 4 (1992), pp. 315–345.

[2] E. ANDERSEN AND Y. YE, *On a Homogeneous Algorithm for a Monotone Complementarity Problem with Nonlinear Equality Constraints*, Working paper, Department of Management Sciences, The University of Iowa, Iowa City, IA, January, 1996.

[3] A. BEN-TAL AND M. P. BENDSØE *A new method for optimal truss topology design*, SIAM J. Optim., 3 (1993), pp. 322–358.

[4] A. Ben-Tal and G. Roth, *A truncated log barrier algorithm for large scale convex programming and minimax problems implementation and computational results*, Optimization Methods Software, 6 (1996), pp. 283–312.
[5] A. Ben-Tal and M. Zibulevsky, *Penalty/barrier multiplier methods for convex programming problems*, SIAM J. Optim., 7 (1997), pp. 347–366.
[6] E. Blum and W. Oettli, *Mathematische Optimierung*, Springer-Verlag, Berlin, Germany, 1975.
[7] R. Byrd, M. B. Hribar, and J. Nocedal, *An Interior Point Algorithm for Large Scale Nonlinear Programming*, Report OTC 97/05, Optimization Technology Center, Northwestern University, Evanston, IL, July, 1997.
[8] W. S. Dorn, R. E. Gomory, and H. J. Greenberg, *Automatic design of optimal structures*, J. Mec., 3 (1964), pp. 25–52.
[9] R. W. Freund and F. Jarre, *A QMR-based interior-point algorithm for solving linear programs*, Math. Programming Ser. B, 76 (1996), pp. 183–210.
[10] D. den Hertog, *Interior Point Approach to Linear, Quadratic and Convex Programming*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.
[11] F. Jarre, *On the convergence of the method of analytic centers when applied to convex quadratic programs*, Math. Programming, 49 (1991), pp. 341–358.
[12] F. Jarre, *The Method of Analytic Centers for Smooth Convex Programs*, dissertation, Universität Würzburg, Würzburg, Germany, 1989.
[13] F. Jarre, *Interior-Point Methods via Self-Concordance or Relative Lipschitz Condition*, Habilitation thesis, Universität Würzburg, Würzburg, Germany, 1994.
[14] M. Kočvara and J. Zowe, *How mathematics can help in design of mechanical structures*, in Proc. of the 16th Biennial Conf. on Numer. Analysis, D. Griffiths and G. Watson, eds., Longman Scientific and Technical, Harlow, 1996, pp. 76–93.
[15] M. Kojima, N. Megiddo, and S. Mizuno, *A primal-dual infeasible-interior-point algorithm for linear programming*, Math. Programming, 61 (1993), pp. 263–280.
[16] M. Kojima, S. Mizuno, and A. Yoshise, *A primal-dual interior point algorithm for linear programming*, Progr. Math. Programming, Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 29–47.
[17] I. J. Lustig, R. E. Marsten, and D. F. Shanno, *Computational experience with a primal-dual interior point method for linear programming*, Linear Algebra Appl., 152 (1991), pp. 191–222.
[18] I. J. Lustig, R. E. Marsten, and D. F. Shanno, *On implementing Mehrotra's predictor-corrector interior-point method for linear programming*, SIAM J. Optim., 2 (1992), pp. 435–449.
[19] N. Megiddo, *Pathways to the optimal set in linear programming*, Progr. Math. Programming, Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 131–158.
[20] S. Mehrotra, *On the Implementation of a (Primal-Dual) Interior-Point Method*, Technical report 90-03, Dept. of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1990.
[21] S. Mehrotra and J. Sun, *An interior-point algorithm for solving smooth convex programs based on Newton's method*, in Mathematical Developments Arising from Linear Programming, Contemp. Math. 114, Amer. Math. Soc., Providence, RI, 1990, pp. 265–284.
[22] R. D. C. Monteiro and I. Adler, *Interior path following primal-dual algorithms, Part* I: *Linear programming*, Math. Programming, 44 (1989), pp. 27–41.
[23] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Methods in Convex Programming*, SIAM, Philadelphia, 1994.
[24] Y. Nesterov and M. J. Todd, *Self-scaled cones and interior-point methods in nonlinear programming*, Math. Oper. Res., 22 (1997), pp. 1–42.
[25] D. Shanno, *private communication.*
[26] K. Tanabe, *Centered Newton method for mathematical programming*, in System Modelling and Optimization, M. Iri and K. Yajima, eds., Springer-Verlag, New York, 1988, pp. 197–206.
[27] T. Wang, R. D C. Monteiro, and J.-S. Pang, *An interior point potential reduction method for constrained equations*, Math. Programming, 74 (1996) pp. 159–195.