Hamburger Beiträge
zur Angewandten Mathematik

Improved Smoothing-Type Methods for the Solution of Linear Programs

Stephan Engelke and Christian Kanzow

Reihe A
Preprint 156
August 2000

Hamburger Beiträge zur Angewandten Mathematik

Reihe A	Preprints
Reihe B	Berichte
Reihe C	Mathematische Modelle und Simulation
Reihe D	Elektrische Netzwerke und Bauelemente
Reihe E	Scientific Computing
Reihe F	Computational Fluid Dynamics and Data Analysis

IMPROVED SMOOTHING-TYPE METHODS FOR THE SOLUTION OF LINEAR PROGRAMS[1]

Stephan Engelke and Christian Kanzow

University of Hamburg
Department of Mathematics
Center for Optimization and Approximation
Bundesstrasse 55
20146 Hamburg
Germany
e-mail: engelke@math.uni-hamburg.de
kanzow@math.uni-hamburg.de

August 4, 2000

Abstract. We consider a smoothing-type method for the solution of linear programs. Its main idea is to reformulate the primal-dual optimality conditions as a nonlinear and nonsmooth system of equations, and to apply a Newton-type method to a smooth approximation of this nonsmooth system. The method presented here is a predictor-corrector method, and is closely related to some methods recently proposed by Burke and Xu on the one hand, and by the authors on the other hand. However, here we state stronger global and/or local convergence properties. Moreover, we present quite promising numerical results for the whole netlib test problem collection.

Key Words. Linear programs, smoothing, predictor-corrector method, Newton's method, global convergence, quadratic convergence.

[1]This research was supported by the DFG (Deutsche Forschungsgemeinschaft).

1 Introduction

In this manuscript we describe an algorithm for the solution of linear programs given either in primal form

$$\min c^T x \quad \text{s.t.} \quad Ax = b,\ x \geq 0 \tag{1}$$

or in dual form

$$\max b^T \lambda \quad \text{s.t.} \quad A^T \lambda + s = c,\ s \geq 0, \tag{2}$$

where $A \in \mathbb{R}^{m \times n}, c \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$ are the given data, and A is assumed to have full rank throughout this paper. The idea of our algorithm is to solve the corresponding optimality conditions

$$\begin{array}{rcll} A^T \lambda + s & = & c, & \\ Ax & = & b, & \\ x_i \geq 0,\ s_i \geq 0,\ x_i s_i & = & 0 & \forall i = 1, \ldots, n. \end{array} \tag{3}$$

To this end, note that (3) is totally equivalent to (1) and (2) in the sense that (3) has a solution if and only if (1) or (2) has a solution.

The widely used class of primal-dual interior-point methods follow a similar idea: They are also based on the optimality conditions (3) and introduce a certain perturbation of (3) depending on a parameter $\tau > 0$:

$$\begin{array}{rcll} A^T \lambda + s & = & c, & \\ Ax & = & b, & \\ x_i > 0,\ s_i > 0,\ x_i s_i & = & \tau^2 & \forall i = 1, \ldots, n. \end{array} \tag{4}$$

The system (4) is usually called the *central path conditions*, and is parameterized here by τ^2 instead of τ just for technical reasons which will become clear in Section 2.

Under certain assumptions, there is a unique solution $w_\tau = (x_\tau, \lambda_\tau, s_\tau)$ of (4) for each $\tau > 0$. The corresponding mapping

$$\tau \mapsto w_\tau$$

is called the *central path*, and the main idea of interior-point methods is to follow this central path numerically. This is typically done by applying Newton's method to the *equations* within the central path conditions (4), whereas a suitable stepsize rule takes care of the strict inequality constraints. In particular, all iterates generated by a primal-dual interior-point method satisfy the strict inequality constraints.

Smoothing-type methods follow a different approach. The general idea of these methods is to reformulate the optimality conditions (3) as a system of equations (not involving any inequalities). Since this system is typically nonsmooth, it then gets approximated by a smooth system of equations to which Newton's method can be applied, see [3, 5, 10, 15] and references therein for a couple of examples following this pattern.

The method to be presented here follows an idea by Jiang [11] and is based on a smooth equation reformulation of the optimality conditions (3) themselves. It is, however, closely

related to both smoothing-type methods and interior-point methods. This will be made clear as soon as we develop the algorithm in Section 2.

The algorithm we present in this paper is quite similar to the predictor-corrector method recently proposed by Burke and Xu [2], see also [1]. Our method is also a predictor-corrector method, with the corrector step being responsible for the global convergence and the predictor step guaranteeing local fast convergence under suitable assumptions. In fact, our corrector step is identical to the one by Burke and Xu [2], but we prove a different global convergence result for it using less stringent assumptions. On the other hand, the predictor step we use here is taken from [8] (and was essentially introduced by Chen, Qi and Sun [6]) and can be shown to be locally quadratically convergent under weaker assumptions than those used by Burke and Xu [2].

We therefore view our algorithm as an improved smoothing-type method due to some better theoretical properties if compared with the method by Burke and Xu [2]. In addition, it also improves the authors' previous method [8] due to some stronger global convergence properties. Furthermore, the numerical results seem to be somewhat better than for the corresponding method in [8].

This paper is organized as follows: In Section 2, we develop our algorithm and give a detailed statement. The global and local convergence properties are investigated in Section 3. Extensive numerical results are presented in Section 4, and Section 5 concludes this paper with some final remarks.

The notation used in this paper is rather standard: $\mathbb{R}^n$ denotes the n-dimensional real vector space. For $x \in \mathbb{R}^n$, we use the subscript x_i in order to indicate the ith component of x, whereas a superscript like in x^k is used to indicate that this is the kth iterate of a sequence $\{x^k\} \subseteq \mathbb{R}^n$. Quite often, we will consider a triple of the form $w = (x^T, \lambda^T, s^T)^T$, where $x \in \mathbb{R}^n, \lambda \in \mathbb{R}^m$, and $s \in \mathbb{R}^n$; of course, w is then a vector in $\mathbb{R}^{n+m+n}$. In order to simplify our notation, however, we will usually write $w = (x, \lambda, s)$ instead of using the mathematically more correct formula $w = (x^T, \lambda^T, s^T)^T$. If $x, y \in \mathbb{R}^n$ are any given vectors satisfying the inequality $x_i \geq y_i$ for all indices $i = 1, \ldots, n$, we simply write $x \geq y$. Finally, all vector norms in this manuscript are Euclidean norms.

2 Development of Algorithm

This section is devoted to the development of our algorithm. To this end, let $\varphi : \mathbb{R}^2 \to \mathbb{R}$ always denote the minimum function

$$\varphi(a, b) := 2 \min\{a, b\},$$

with the factor 2 being used here only for cosmetical reasons. Define

$$\Phi(w) := \Phi(x, \lambda, s) := \begin{pmatrix} A^T\lambda + s - c \\ Ax - b \\ \phi(x, s) \end{pmatrix},$$

where

$$\phi(x, s) := (\varphi(x_1, s_1), \ldots, \varphi(x_n, s_n))^T \in \mathbb{R}^n.$$

Since φ is an NCP-function, i.e.,

$$\varphi(a,b) = 0 \Longleftrightarrow a \geq 0, b \geq 0, ab = 0,$$

the following observation is an immediate consequence of the definition of the mapping Φ:

$$w^* = (x^*, \lambda^*, s^*) \text{ solves } (3) \Longleftrightarrow w^* \text{ solves } \Phi(w) = 0.$$

However, the system $\Phi(w) = 0$ is nonsmooth. Therefore, let $\varphi_\tau : \mathbb{R}^2 \to \mathbb{R}$ denote the smoothed minimum function

$$\varphi_\tau(a,b) := a + b - \sqrt{(a-b)^2 + 4\tau^2},$$

typically called the *Chen-Harker-Kanzow-Smale smoothing function* [4, 12, 14], where $\tau > 0$ is the smoothing parameter. Then define

$$\Phi_\tau(w) := \Phi_\tau(x, \lambda, s) := \begin{pmatrix} A^T\lambda + s - c \\ Ax - b \\ \phi_\tau(x,s) \end{pmatrix},$$

where

$$\phi_\tau(x,s) := (\varphi_\tau(x_1, s_1), \ldots, \varphi_\tau(x_n, s_n))^T \in \mathbb{R}^n.$$

It was observed in [12] that the following equivalence holds:

$$w_\tau = (x_\tau, \lambda_\tau, s_\tau) \text{ solves } (4) \Longleftrightarrow w_\tau \text{ solves } \Phi_\tau(w) = 0,$$

i.e., we obtain a reformulation of the central path conditions as a nonlinear and smooth system of equations in this way.

So far, we have viewed τ as a parameter. In the sequel, however, it will sometimes be useful to view τ as an independent variable. In order to make this different point of view clear in our notation, let us write

$$\theta(x,s,\tau) \;\; := \;\; \phi_\tau(x,s).$$

Moreover, we will exploit the mapping $\Theta : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n \times [0,\infty) \to \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}$ defined by

$$\Theta(w,\tau) := \Theta(x,\lambda,s,\tau) := \begin{pmatrix} A^T\lambda + s - c \\ Ax - b \\ \theta(x,s,\tau) \\ \tau \end{pmatrix}.$$

Note that the definition of $\Theta(w,\tau)$ is not equal to $\Phi_\tau(w)$ since we have added one more line. Since the equation $\Theta(w,\tau) = 0$ automatically implies $\tau = 0$, we obtain the equivalence

$$w^* = (x^*, \lambda^*, s^*) \text{ solves } (3) \Longleftrightarrow (w^*, 0) \text{ solves } \Theta(w,\tau) = 0.$$

In this way we therefore get a reformulation of the optimality conditions (3) with τ being viewed as an independent variable. This kind of reformulation goes back to Jiang [11].

For later reference, it will be important to exploit the relation between Newton's method applied to the system $\Phi_\tau(w) = 0$ and applied to the system $\Theta(w, \tau) = 0$. First consider the system $\Phi_\tau(w) = 0$, and assume that $w^k = (x^k, \lambda^k, s^k)$ denotes the current iterate and $\tau_k > 0$ the current value of the smoothing parameter. Then we define

$$w^{k+1} = w^k + t_k \Delta w^k$$

for a suitable stepsize $t_k > 0$, where the correction vector $\Delta w^k = (\Delta x^k, \Delta \lambda^k, \Delta s^k)$ is a solution of the linear system of equations

$$\Phi'_{\tau_k}(w^k)\Delta w^k = -\Phi_{\tau_k}(w^k).$$

Taking into account the definition of Φ_τ, this equation is equivalent to

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ D^k_{a,\tau} & 0 & D^k_{b,\tau} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = \begin{pmatrix} -A^T\lambda^k - s^k + c \\ -Ax^k + b \\ -\phi_{\tau_k}(x^k, s^k) \end{pmatrix}, \tag{5}$$

where

$$D^k_{a,\tau} := \operatorname{diag}\left(\frac{\partial \varphi_{\tau_k}}{\partial a}(x^k_1, s^k_1), \ldots, \frac{\partial \varphi_{\tau_k}}{\partial a}(x^k_n, s^k_n) \right) \in \mathbb{R}^{n \times n}$$

and, similarly,

$$D^k_{b,\tau} := \operatorname{diag}\left(\frac{\partial \varphi_{\tau_k}}{\partial b}(x^k_1, s^k_1), \ldots, \frac{\partial \varphi_{\tau_k}}{\partial b}(x^k_n, s^k_n) \right) \in \mathbb{R}^{n \times n}.$$

On the other hand, if we apply Newton's method to the system $\Theta(w, \tau) = 0$, we have to solve an equation like

$$\Theta'(w^k, \tau_k) \begin{pmatrix} \Delta w \\ \Delta \tau \end{pmatrix} = -\Theta(w^k, \tau_k)$$

at each iteration, where the derivatives are taken with respect to w and τ. Hence the above system becomes

$$\begin{pmatrix} 0 & A^T & I & 0 \\ A & 0 & 0 & 0 \\ D^k_{a,\tau} & 0 & D^k_{b,\tau} & d^k_\tau \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \\ \Delta \tau \end{pmatrix} = \begin{pmatrix} -A^T\lambda^k - s^k + c \\ -Ax^k + b \\ -\theta(x^k, s^k, \tau_k) \\ -\tau_k \end{pmatrix}, \tag{6}$$

where

$$d^k_\tau := \left(\frac{\partial \theta}{\partial \tau}(x^k_1, s^k_1, \tau_k), \ldots, \frac{\partial \theta}{\partial \tau}(x^k_n, s^k_n, \tau_k) \right)^T \in \mathbb{R}^n.$$

Motivated by similar considerations in the field of interior-point methods (see, e.g., Wright [17]), we will consider a generalization of the system (6) and replace the parameter τ_k on the last line of the right-hand side in (6) by $\sigma_k \tau_k$ for some number $\sigma_k \in [0, 1]$, i.e., we solve

$$\begin{pmatrix} 0 & A^T & I & 0 \\ A & 0 & 0 & 0 \\ D^k_{a,\tau} & 0 & D^k_{b,\tau} & d^k_\tau \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \\ \Delta \tau \end{pmatrix} = \begin{pmatrix} -A^T\lambda^k - s^k + c \\ -Ax^k + b \\ -\theta(x^k, s^k, \tau_k) \\ -\sigma_k \tau_k \end{pmatrix} \tag{7}$$

(the choice $\sigma_k = 1$ corresponds to (6)). Note, however, that we do not replace τ_k by $\sigma_k \tau_k$ in the definition of the function $\theta(x, s, \tau)$. In order to have a short-hand notation for the linear system (7), we introduce the function

$$\Theta_\sigma(w, \tau) := \begin{pmatrix} A^T\lambda + s - c \\ Ax - b \\ \theta(x, s, \tau) \\ \sigma\tau \end{pmatrix}$$

with the subscript σ indicating the dependence of Θ on the parameter σ. Then the linear system (7) can be rewritten as

$$\Theta'(w^k, \tau_k) \begin{pmatrix} \Delta w \\ \Delta\tau \end{pmatrix} = -\Theta_\sigma(w^k, \tau_k).$$

Note that this or, equivalently, (7) immediately gives

$$\Delta\tau_k = -\sigma_k \tau_k. \tag{8}$$

Replacing this expression into the remaining equations of (7), we obtain

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ D^k_{a,\tau} & 0 & D^k_{b,\tau} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{pmatrix} = \begin{pmatrix} -A^T\lambda^k - s^k + c \\ -Ax^k + b \\ -\theta(x^k, s^k, \tau_k) + \sigma_k\tau_k d^k_\tau \end{pmatrix}. \tag{9}$$

Obviously, this can be rewritten as

$$\Phi'_{\tau_k}(w^k)\Delta w = -\Phi_{\tau_k}(w^k) + \sigma_k\tau_k \begin{pmatrix} 0 \\ 0 \\ d^k_\tau \end{pmatrix}.$$

This shows that the linear system (7) can be viewed as a perturbation of the system (5), with the perturbation being active only in the third block row of the right-hand side.

Based on the notation introduced so far, we next give a precise statement of our predictor-corrector smoothing method.

Algorithm 2.1 *(Predictor-Corrector Smoothing Method)*

(S.0) (Initialization)
Choose $w^0 := (x^0, \lambda^0, s^0) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n$ *and* $\tau_0 > 0$ *such that* $A^T\lambda^0 + s^0 = c, Ax^0 = b$ *and* $\theta(x^0, s^0, \tau_0) \le 0$, *select* $\beta \ge \|\Phi_{\tau_0}(w^0)\|/\tau_0, \rho \in (0, 1), 0 < \hat{\sigma}_{\min} < \hat{\sigma}_{\max} < 1, \varepsilon \ge 0$, *and set* $k := 0$.

(S.1) (Termination Criterion)
If $\|\Phi(w^k)\| \le \varepsilon$: *STOP.*

(S.2) (Predictor Step)
Compute a solution $(\Delta w^k, \Delta\tau_k) = (\Delta x^k, \Delta\lambda^k, \Delta s^k, \Delta\tau_k) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}$ *of the linear system*

$$\Theta'(w^k, \tau_k) \begin{pmatrix} \Delta w \\ \Delta\tau \end{pmatrix} = -\Theta(w^k, 0). \tag{10}$$

If $\|\Phi(w^k + \Delta w^k)\| = 0$*: STOP. Otherwise, if*

$$\|\theta(x^k + \Delta x^k, s^k + \Delta s^k, \tau_k)\| > \beta\tau_k,$$

then set

$$\hat{w}^k := w^k, \;\; \hat{\tau}_k := \tau_k, \;\; \eta_k := 1,$$

else compute $\eta_k = \rho^{\ell_k}$*, where* ℓ_k *is the nonnegative integer such that*

$$\begin{aligned} \|\theta(x^k + \Delta x^k, s^k + \Delta s^k, \rho^j\tau_k)\| &\leq \beta\rho^j\tau_k \quad \forall j = 0, 1, 2, \ldots, \ell_k \text{ and} \\ \|\theta(x^k + \Delta x^k, s^k + \Delta s^k, \rho^{\ell_k+1}\tau_k)\| &> \beta\rho^{\ell_k+1}\tau_k, \end{aligned}$$

and set $\hat{\tau}_k := \eta_k\tau_k$ *and*

$$\hat{w}^k := \begin{cases} w^k & \text{if } \ell_k = 0, \\ w^k + \Delta w^k & \text{otherwise.} \end{cases}$$

(S.3) (Corrector Step)
Choose $\hat{\sigma}_k \in [\hat{\sigma}_{\min}, \hat{\sigma}_{\max}]$*, and compute a solution* $(\Delta\hat{w}^k, \Delta\hat{\tau}_k) = (\Delta\hat{x}^k, \Delta\hat{\lambda}^k, \Delta\hat{s}^k, \Delta\hat{\tau}_k) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}$ *of the linear system*

$$\Theta'(\hat{w}^k, \hat{\tau}_k) \begin{pmatrix} \Delta\hat{w} \\ \Delta\hat{\tau} \end{pmatrix} = -\Theta_{\hat{\sigma}_k}(\hat{w}^k, \hat{\tau}_k). \tag{11}$$

Let $\hat{t}_k = \max\{\rho^\ell \,|\, \ell = 0, 1, 2, \ldots\}$ *such that*

$$\|\theta(\hat{x}^k + \hat{t}_k\Delta\hat{x}^k, \hat{s}^k + \hat{t}_k\Delta\hat{s}^k, (1 - \hat{\sigma}_k\hat{t}_k)\hat{\tau}_k)\| \leq \beta(1 - \hat{\sigma}_k\hat{t}_k)\hat{\tau}_k. \tag{12}$$

Set $w^{k+1} := \hat{w}^k + \hat{t}_k\Delta\hat{w}^k$ *and* $\tau_{k+1} := (1 - \hat{\sigma}_k\hat{t}_k)\hat{\tau}_k$.

(S.4) (Update)
Set $k \leftarrow k + 1$*, and go to Step (S.1).*

To get a better understanding of the way Algorithm 2.1 works, let us add a couple of comments. In Step (S.0), we require the starting point $w^0 = (x^0, \lambda^0, s^0)$ to be feasible with respect to the linear equations $A^T\lambda + s = c$ and $Ax = b$. Since the components x^0 and s^0 do not have to be positive (like in interior-point methods), it is relatively easy to find such a starting point. In addition, we assume that this starting point together with the initial smoothing parameter τ_0 satisfies the relation $\theta(x^0, s^0, \tau_0) \leq 0$ (this inequality will be important in order to show that the sequence $\{w^k\}$ remains bounded under certain

assumptions). Again, this can easily be done, and Section 4 gives an example of how to generate the initial data such that all requirements of Step (S.0) are satisfied.

In the predictor step, we first compute a search direction by solving the linear system (10). The interesting part about this linear system is the fact that the right-hand side of (10) is unperturbed with respect to τ, whereas we use the standard Jacobian of the perturbed function $\Theta(w, \tau)$ on the left. This may be viewed as the counterpart of the affine scaling step typically used as a predictor in primal-dual interior-point methods, see Wright [17]. Like in the interior-point setting, this predictor step will eventually guarantee local fast convergence (under suitable assumptions).

After having computed the search direction in (10), we try to reduce the smoothing parameter τ_k as much as possible with the only restriction that the full step stays within a certain neighbourhood of the central path, cf. Lemma 3.2 (c) below.

While our predictor step is different from the one used by Burke and Xu [2] in their smoothing-type method, the corrector step in (S.3) coincides with the one from [2]. Note that the linear system (11) is precisely the one from (7) and includes a perturbation on the right-hand side as well. The predictor step also contains a procedure to reduce the smoothing parameter. This procedure will guarantee global convergence in the sense that τ_k decreases to zero under mild conditions.

3 Convergence Properties

This section investigates the global and local convergence properties of Algorithm 2.1. To this end, we assume throughout this section that the termination parameter ε is equal to zero, and that Algorithm 2.1 generates an infinite number of iterates w^k, i.e., we assume that we do not stop after a finite number of iterations in a point w^k satisfying the optimality conditions (3).

We first note that Algorithm 2.1 is well-defined.

Lemma 3.1 *The following statements hold for any $k \in \mathbb{N}$:*

(a) The linear systems (10) and (11) have a unique solution.

(b) There is a unique η_k satisfying the conditions in Step (S.2).

(c) The stepsize $\hat{t}_k$ in (S.3) is uniquely defined.

Consequently, Algorithm 2.1 is well-defined.

Proof. (a) The structure of the Jacobian $\Theta'(w^k, \tau_k)$ in (6) shows that this matrix is nonsingular if and only if $\Phi'_{\tau_k}(w^k)$ is nonsingular. The latter, however, was noted in [7, Proposition 3.1].

(b) This statement follows from [8, Proposition 3.2] and is essentially due to Burke and Xu [2].

(c) This result follows from the proof of Theorem 1 in Burke and Xu [2]. □

We next state some simple properties of Algorithm 2.1 to which we will refer a couple of times in our subsequent analysis.

Lemma 3.2 *The sequences* $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ *and* $\{\tau_k\}$ *generated by Algorithm 2.1 have the following properties:*

(a) $A^T\lambda^k + s^k = c$ *and* $Ax^k = b$ *for all* $k \in \mathbb{N}$.

(b) $\tau_k = \tau_0(1 - \hat{\sigma}_0\hat{t}_0)\eta_0 \cdots (1 - \hat{\sigma}_{k-1}\hat{t}_{k-1})\eta_{k-1}$ *for all* $k \in \mathbb{N}$.

(c) $\|\theta(x^k, s^k, \tau_k)\| \leq \beta\tau_k$ *for all* $k \in \mathbb{N}$.

Proof. (a) For $k = 0$, this follows from the choice of our starting point in Step (S.0). Newton's method then guarantees that the linear equations $A^T\lambda + s = c$ and $Ax = b$ are also satisfied for all $k \geq 1$.

(b) Step (S.2) of Algorithm 2.1 implies that $\hat{\tau}_k = \eta_k\tau_k$. The updating rules in Step (S.3) therefore give

$$\tau_{k+1} = (1 - \hat{\sigma}_k\hat{t}_k)\hat{\tau}_k = (1 - \hat{\sigma}_k\hat{t}_k)\eta_k\tau_k$$

for all $k \in \mathbb{N}$. This gives the desired formula, see also [2, Theorem 2].

(c) The choice of the starting point $w^0 = (x^0, \lambda^0, s^0)$ and β in Step (S.0) guarantee that we have $\|\theta(x^k, s^k, \tau_k)\| \leq \beta\tau_k$ for $k = 0$. The updating rules in Step (S.3) show that this inequality holds for all $k \in \mathbb{N}$. □

The next result follows immediately from the definition of the smoothed minimum function.

Lemma 3.3 *Let* $0 \leq \tau_1 \leq \tau_2$ *be any given smoothing parameters. Then the inequality*

$$\theta(x, s, \tau_2) \leq \theta(x, s, \tau_1)$$

holds for all $x, s \in \mathbb{R}^n$.

Lemma 3.3 allows us to prove the following result which, later, will be used to show that the entire sequence $\{w^k\}$ generated by Algorithm 2.1 remains bounded under suitable assumptions. Moreover, we stress that the statement of our next result is essentially identical to a statement given in [2, Theorem 1], however, our proof is slightly different since our predictor step does not coincide with the predictor step from the algorithm in [2].

Lemma 3.4 *The sequences* $\{w^k\} = \{(x^k, \lambda^k, s^k)\}, \{\tau_k\}, \{\hat{w}^k\} = \{(\hat{x}^k, \hat{\lambda}^k, \hat{s}^k)\}$ *and* $\{\hat{\tau}_k\}$ *generated by Algorithm 2.1 have the following properties:*

(a) $\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) \leq 0$ *for all* $k \in \mathbb{N}$.

(b) $\theta(x^k, s^k, \tau_k) \leq 0$ *for all* $k \in \mathbb{N}$.

Proof. We first derive some useful inequalities, and then verify the two statements simultaneously by induction on k.

We begin with some preliminary discussions regarding statement (a). To this end, let $k \in \mathbb{N}$ be fixed for the moment, and assume that we take $\hat{w}^k = w^k + \Delta w^k$ in Step (S.2) of Algorithm 2.1. Since each component of the function θ is concave, we then obtain

$$\begin{aligned}
\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) &= \theta(x^k + \Delta x^k, s^k + \Delta s^k, \eta_k \tau_k) \\
&= \theta(x^k + \Delta x^k, s^k + \Delta s^k, \tau_k + (\eta_k - 1)\tau_k) \\
&\leq \theta(x^k, s^k, \tau_k) + \theta'(x^k, s^k, \tau_k) \begin{pmatrix} \Delta x^k \\ \Delta s^k \\ (\eta_k - 1)\tau_k \end{pmatrix} \\
&= \theta(x^k, s^k, \tau_k) + \theta'(x^k, s^k, \tau_k) \begin{pmatrix} \Delta x^k \\ \Delta s^k \\ \Delta \tau_k \end{pmatrix} + \theta'(x^k, s^k, \tau_k) \begin{pmatrix} 0 \\ 0 \\ (\eta_k - 1)\tau_k - \Delta \tau_k \end{pmatrix}.
\end{aligned} \tag{13}$$

Using (10), we have

$$\theta'(x^k, s^k, \tau_k) \begin{pmatrix} \Delta x^k \\ \Delta s^k \\ \Delta \tau_k \end{pmatrix} = -\theta(x^k, s^k, 0).$$

Furthermore, the last row from (10) implies that $\Delta \tau_k = 0$. Taking this into account, we get from (13):

$$\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) \leq \theta(x^k, s^k, \tau_k) - \theta(x^k, s^k, 0) + \theta'(x^k, s^k, \tau_k) \begin{pmatrix} 0 \\ 0 \\ (\eta_k - 1)\tau_k \end{pmatrix}.$$

Lemma 3.3 therefore gives

$$\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) \leq \theta'(x^k, s^k, \tau_k) \begin{pmatrix} 0 \\ 0 \\ (\eta_k - 1)\tau_k \end{pmatrix}. \tag{14}$$

We claim that the right-hand side of (14) is nonpositive. Indeed, since

$$\theta'(x^k, s^k, \tau_k) \begin{pmatrix} 0 \\ 0 \\ (\eta_k - 1)\tau_k \end{pmatrix} = (\eta_k - 1)\tau_k d_\tau^k$$

with

$$\begin{aligned}
d_\tau^k &= \left(\frac{\partial \theta}{\partial \tau}(x_1^k, s_1^k, \tau_k), \ldots, \frac{\partial \theta}{\partial \tau}(x_n^k, s_n^k, \tau_k) \right)^T \\
&= \left(\frac{-4\tau_k}{\sqrt{(x_1^k - s_1^k)^2 + 4\tau_k^2}}, \ldots, \frac{-4\tau_k}{\sqrt{(x_n^k - s_n^k)^2 + 4\tau_k^2}} \right) \\
&\leq 0,
\end{aligned}$$

this claim follows from the fact that $\eta_k \in (0,1]$.

We next derive some useful inequalities regarding statement (b). To this end, we still assume that $k \in \mathbb{N}$ is fixed. Using once again the fact that θ is a concave function in each component, we obtain from (11) (recall that we have $\Delta\hat{\tau}_k = -\hat{\sigma}_k\hat{\tau}_k$, cf. (8)):

$$\begin{aligned}
\theta(x^{k+1}, s^{k+1}, \tau_{k+1}) &= \theta(\hat{x}^k + \hat{t}_k\Delta\hat{x}^k, \hat{s}^k + \hat{t}_k\Delta\hat{s}^k, \hat{\tau}_k + \hat{t}_k\Delta\hat{\tau}_k) \\
&\leq \theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) + \hat{t}_k\theta'(\hat{x}^k, \hat{s}^k, \hat{\tau}_k)\begin{pmatrix} \Delta\hat{x}^k \\ \Delta\hat{s}^k \\ \Delta\hat{\tau}_k \end{pmatrix} \\
&= \theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) - \hat{t}_k\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) \\
&= (1-\hat{t}_k)\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k),
\end{aligned} \tag{15}$$

and this completes our preliminary discussions.

We now verify statements (a) and (b) by induction on k. For $k = 0$, we have $\theta(x^0, s^0, \tau_0) \leq 0$ by our choice of the starting point $w^0 = (x^0, \lambda^0, s^0)$ and the initial smoothing parameter $\tau_0 > 0$ in Step (S.0) of Algorithm 2.1. Therefore, if we set $\hat{w}^0 = w^0$ in Step (S.2) of Algorithm 2.1, we also have $\hat{\tau}_0 = \tau_0$ and therefore $\theta(\hat{x}^0, \hat{s}^0, \hat{\tau}_0) \leq 0$ as well. On the other hand, if we set $\hat{w}^0 = w^0 + \Delta w^0$ in Step (S.2), the argument used in the beginning of this proof shows that the inequality $\theta(\hat{x}^0, \hat{s}^0, \hat{\tau}_0) \leq 0$ also holds in this case.

Suppose that we have $\theta(x^k, s^k, \tau_k) \leq 0$ and $\theta(\hat{x}^k, \hat{s}^k, \hat{\tau}_k) \leq 0$ for some $k \in \mathbb{N}$. Then (15) immediately implies that we have $\theta(x^{k+1}, s^{k+1}, \tau_{k+1}) \leq 0$. Consequently, if we have $\hat{w}^{k+1} = w^{k+1}$ in Step (S.2) of Algorithm 2.1, we obviously have $\theta(\hat{x}^{k+1}, \hat{s}^{k+1}, \hat{\tau}_{k+1}) \leq 0$. Otherwise, i.e., if we set $\hat{w}^{k+1} = w^{k+1} + \Delta w^{k+1}$ in Step (S.2), the argument used in the beginning part of this proof shows that the same inequality holds. This completes the formal proof by induction. □

We next show that the sequence $\{w^k\}$ generated by Algorithm 2.1 remains bounded provided that there is a strictly feasible point for the optimality conditions (3), i.e., a vector $\hat{w} = (\hat{x}, \hat{\lambda}, \hat{s})$ satisfying $A^T\hat{\lambda} + \hat{s} = c, A\hat{x} = b$ and $\hat{x} > 0, \hat{s} > 0$. Note that this statement does not hold for the authors' previous method from [8] in case the minimum function is used in [8].

Proposition 3.5 *Assume that there is a strictly feasible point for the optimality conditions (3). Then the sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ generated by Algorithm 2.1 is bounded.*

Proof. The statement is essentially due to Burke and Xu [2], and we include a proof here only for the sake of completeness.

Assume that the sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ generated by Algorithm 2.1 is unbounded. Since $\{\tau_k\}$ is monotonically decreasing, it follows from Lemma 3.2 (c) that

$$\|\theta(x^k, s^k, \tau_k)\| \leq \beta\tau_k \leq \beta\tau_0 \tag{16}$$

for all $k \in \mathbb{N}$. The definition of the (smoothed) minimum function therefore implies that there is no index $i \in \{1, \ldots, n\}$ such that $x_i^k \to -\infty$ or $s_i^k \to -\infty$ on a subsequence, since otherwise we would have $\varphi_{\tau_k}(x_i^k, s_i^k) \to -\infty$ which, in turn, would imply $\|\theta(x^k, s^k, \tau_k)\| \to$

$+\infty$ on a subsequence in contrast to (16). Therefore, all components of the two sequences $\{x^k\}$ and $\{s^k\}$ are bounded from below, i.e.,

$$x_i^k \geq \gamma \quad \text{and} \quad s_i^k \geq \gamma \quad \forall i = 1, \ldots, n, \ \forall k \in \mathbb{N}, \tag{17}$$

where $\gamma \in \mathbb{R}$ denotes a suitable (possibly negative) constant.

On the other hand, the sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ is unbounded by assumption. This implies that there is at least one component $i \in \{1, \ldots, n\}$ such that $x_i^k \to +\infty$ or $s_i^k \to +\infty$ on a subsequence since otherwise the two sequences $\{x^k\}$ and $\{s^k\}$ would be bounded which, in turn, would imply the boundedness of the sequence $\{\lambda^k\}$ as well because we have $A^T\lambda^k + s^k = c$ for all $k \in \mathbb{N}$ (cf. Lemma 3.2 (a)) and because A is assumed to have full rank.

Now let $\hat{w} = (\hat{x}, \hat{\lambda}, \hat{s}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n$ be a strictly feasible point for (3) whose existence is guaranteed by our assumption. Then, in particular, we have

$$A^T\hat{\lambda} + \hat{s} = c \quad \text{and} \quad A\hat{x} = b.$$

Since we also have

$$A^T\lambda^k + s^k = c \quad \text{and} \quad Ax^k = b$$

for all $k \in \mathbb{N}$ by Lemma 3.2 (a), we get

$$A^T(\hat{\lambda} - \lambda^k) + (\hat{s} - s^k) = 0 \quad \text{and} \quad A(\hat{x} - x^k) = 0 \tag{18}$$

by subtracting these equations. Premultiplying the first equation in (18) with $(\hat{x} - x^k)^T$ and taking into account the second equation in (18) gives

$$(\hat{x} - x^k)^T(\hat{s} - s^k) = 0.$$

Reordering this equation, we obtain

$$\hat{s}^T x^k + \hat{x}^T s^k = (x^k)^T s^k + \hat{x}^T \hat{s} \tag{19}$$

for all $k \in \mathbb{N}$. Using (17) as well as $\hat{x} > 0$ and $\hat{s} > 0$ in view of the strict feasibility of the vector $\hat{w} = (\hat{x}, \hat{\lambda}, \hat{s})$, it follows from (19) and the fact that $x_i^k \to +\infty$ or $s_i^k \to +\infty$ on a subsequence for at least one index $i \in \{1, \ldots, n\}$ that

$$(x^k)^T s^k \to +\infty.$$

Hence there exists a component $j \in \{1, \ldots, n\}$ (independent of k) such that

$$x_j^k s_j^k \to +\infty \tag{20}$$

on a suitable subsequence.

Now, using Lemma 3.4 (b), we have

$$\theta(x^k, s^k, \tau_k) \leq 0$$

for all $k \in \mathbb{N}$. Taking into account the definition of θ, this implies

$$x_j^k + s_j^k \leq \sqrt{(x_j^k - s_j^k)^2 + 4\tau_k^2} \tag{21}$$

for all $k \in \mathbb{N}$. Using (20) and (17), we see that we necessarily have $x_j^k > 0$ and $s_j^k > 0$ for all those k belonging to the subsequence for which (20) holds. Therefore, taking the square in (21), we obtain

$$4x_j^k s_j^k \leq 4\tau_k^2$$

after some simplifications. However, since the right-hand side of this expression is bounded by $4\tau_0^2$, this gives a contradiction to (20). □

We next give a global convergence result for Algorithm 2.1. Note that this result is different from the one provided by Burke and Xu [2]. (They use an assumption which is even stronger than the one we use for our local convergence result in Theorem 3.9; on the other hand, the main emphasis in [2] was to prove a global linear rate of convergence result.)

Theorem 3.6 *Assume that the sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ generated by Algorithm 2.1 has at least one accumulation point. Then $\{\tau_k\}$ converges to zero.*

Proof. Since the sequence $\{\tau_k\}$ is monotonically decreasing and bounded from below by zero, it converges to a number $\tau_* \geq 0$. If $\tau_* = 0$, we are done.

So assume that $\tau_* > 0$. Then the updating rules in Step (S.2) of Algorithm 2.1 immediately give

$$\hat{w}^k = w^k, \quad \hat{\tau}_k = \tau_k, \quad \text{and} \quad \eta_k = 1 \tag{22}$$

for all $k \in \mathbb{N}$ sufficiently large. Subsequencing if necessary, we assume without loss of generality that (22) holds for all $k \in \mathbb{N}$. Then Lemma 3.2 (b) and $\hat{\sigma}_k \geq \hat{\sigma}_{\min}$ yield

$$\tau_k = \tau_0 \prod_{j=0}^{k-1}(1 - \hat{\sigma}_j \hat{t}_j) \leq \tau_0 \prod_{j=0}^{k-1}(1 - \hat{\sigma}_{\min} \hat{t}_j). \tag{23}$$

Since $\tau_k \to \tau_* > 0$ by assumption, it follows from (23) that $\lim_{k\to\infty} \hat{t}_k = 0$. Therefore, the stepsize $\hat{\alpha}_k := \hat{t}_k/\rho$ does not satisfy the line search criterion (12) for all $k \in \mathbb{N}$ sufficiently large. Hence we have

$$\|\theta(\hat{x}^k + \hat{\alpha}_k \Delta\hat{x}^k, \hat{s}^k + \hat{\alpha}_k \Delta\hat{s}^k, (1 - \hat{\sigma}_k\hat{\alpha}_k)\hat{\tau}_k)\| > \beta(1 - \hat{\sigma}_k\hat{\alpha}_k)\hat{\tau}_k \tag{24}$$

for all these $k \in \mathbb{N}$.

Now let $w^* = (x^*, \lambda^*, s^*)$ be an accumulation point of the sequence $\{w^k\}$, and let $\{w^k\}_K$ be a subsequence converging to w^*. Since $\hat{\sigma}_k \in [\hat{\sigma}_{\min}, \hat{\sigma}_{\max}]$ for all $k \in \mathbb{N}$, we can assume without loss of generality that the subsequence $\{\hat{\sigma}_k\}_K$ converges to some number $\hat{\sigma}_* \in [\hat{\sigma}_{\min}, \hat{\sigma}_{\max}]$. Furthermore, since $\tau_* > 0$, it follows from Lemma 3.1 (a) that the corresponding

subsequence $\{(\Delta\hat{w}^k, \Delta\hat{\tau}_k)\}_K$ converges to a vector $(\Delta\hat{w}^*, \Delta\hat{\tau}_*) = (\Delta\hat{x}^*, \Delta\hat{\lambda}^*, \Delta\hat{s}^*, \Delta\hat{\tau}_*)$, where $(\Delta\hat{w}^*, \Delta\hat{\tau}_*)$ is the unique solution of the linear equation

$$\Theta'(w^*, \tau_*) \begin{pmatrix} \Delta\hat{w} \\ \Delta\hat{\tau} \end{pmatrix} = -\Theta_{\hat{\sigma}_*}(w^*, \tau_*), \tag{25}$$

cf. (11). Using $\{\hat{\alpha}_k\}_K \to 0$ and taking the limit $k \to \infty$ on the subset K, we then obtain from (22) and (24) that

$$\|\theta(x^*, s^*, \tau_*)\| \geq \beta\tau_* > 0. \tag{26}$$

On the other hand, we get from (24), (22), Lemma 3.2 (c), and $\hat{\sigma}_k \leq \hat{\sigma}_{\max}$ that

$$\begin{aligned} \|\theta(\hat{x}^k + \hat{\alpha}_k\Delta\hat{x}^k, \hat{s}^k + \hat{\alpha}_k\Delta\hat{s}^k, (1 - \hat{\sigma}_k\hat{\alpha}_k)\hat{\tau}_k\| &> (1 - \hat{\sigma}_k\hat{\alpha}_k)\beta\hat{\tau}_k \\ &= (1 - \hat{\sigma}_k\hat{\alpha}_k)\beta\tau_k \\ &\geq (1 - \hat{\sigma}_k\hat{\alpha}_k)\|\theta(x^k, s^k, \tau_k)\| \\ &\geq (1 - \hat{\sigma}_{\max}\hat{\alpha}_k)\|\theta(x^k, s^k, \tau_k)\| \end{aligned}$$

for all $k \in \mathbb{N}$ sufficiently large. Using (22) and $\Delta\hat{\tau}_k = -\hat{\sigma}_k\hat{\tau}_k$ (cf. (8)), this implies

$$\frac{\|\theta(x^k + \hat{\alpha}_k\Delta\hat{x}^k, s^k + \hat{\alpha}_k\Delta\hat{s}^k, \tau_k + \hat{\alpha}_k\Delta\hat{\tau}_k)\| - \|\theta(x^k, s^k, \tau_k)\|}{\hat{\alpha}_k} \geq -\hat{\sigma}_{\max}\|\theta(x^k, s^k, \tau_k)\|.$$

Since $\|\theta(\cdot, \cdot, \cdot)\|$ is a continuously differentiable function at (x^*, s^*, τ_*) due to (26), taking the limit $k \to \infty$ for $k \in K$ then gives

$$\frac{\theta(x^*, s^*, \tau_*)^T}{\|\theta(x^*, s^*, \tau_*)\|}\theta'(x^*, s^*, \tau_*) \begin{pmatrix} \Delta\hat{x}^* \\ \Delta\hat{s}^* \\ \Delta\hat{\tau}_* \end{pmatrix} \geq -\hat{\sigma}_{\max}\|\theta(x^*, s^*, \tau_*)\|,$$

where $(\Delta\hat{x}^*, \Delta\hat{\lambda}^*, \Delta\hat{s}^*, \Delta\hat{\tau}_*)$ denotes the solution of the linear system (25). Using (25) then gives

$$-\|\theta(x^*, s^*, \tau_*)\| \geq -\hat{\sigma}_{\max}\|\theta(x^*, s^*, \tau_*)\|.$$

Since $\hat{\sigma}_{\max} \in (0, 1)$, this implies $\|\theta(x^*, s^*, \tau_*)\| = 0$, a contradiction to (26). □

Due to Proposition 3.5, the assumed existence of an accumulation point in Theorem 3.6 is automatically satisfied if there is a strictly feasible point for the optimality conditions (3). — An immediate consequence of Theorem 3.6 is the following result.

Corollary 3.7 *Every accumulation point of a sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$ generated by Algorithm 2.1 is a solution of the optimality conditions (3).*

Proof. Let $w^* = (x^*, \lambda^*, s^*)$ be an accumulation point of the sequence $\{w^k\} = \{(x^k, \lambda^k, s^k)\}$, and let $\{w^k\}_K$ denote a subsequence converging to w^*. Then we have $\tau_k \to 0$ in view of Theorem 3.6. Hence Lemma 3.2 (c) implies

$$\|\theta(x^*, s^*, 0)\| = \lim_{k \in K} \|\theta(x^k, s^k, \tau_k)\| \leq \beta \lim_{k \in K} \tau_k = 0,$$

i.e., we have $x^* \geq 0, s^* \geq 0$ and $x_i^* s_i^* = 0$ for $i = 1, \ldots, n$ due to the definition of θ. Since Lemma 3.2 (a) also shows that we have $A^T \lambda^* + s^* = c$ and $Ax^* = b$, we see that $w^* = (x^*, \lambda^*, s^*)$ is indeed a solution of the optimality conditions (3). □

We next want to give a local convergence result. To this end, we first note that the search direction we obtain in our predictor step is identical to the one obtained in the predictor step of the method from [8].

Lemma 3.8 *The vector $(\Delta w^k, \Delta \tau_k)$ is a solution of the linear system (10) if and only if Δw^k solves the system*

$$\Phi'_{\tau_k}(w^k)\Delta w = -\Phi_0(w^k),$$

and $\Delta \tau_k = 0$.

Proof. Since the smoothing parameter on the right-hand side of the linear system (10) is equal to zero, the assertion follows immediately from the discussion following (6). □

The previous result implies that we can apply the local rate of convergence analysis from [8]. Hence we obtain the following result from [8] (see also Tseng [16]).

Theorem 3.9 *Assume that the sequence $\{w^k\}$ generated by Algorithm 2.1 converges to a strictly complementary solution of the optimality conditions (2.1). Suppose further that the parameter β from Step (S.0) of Algorithm 2.1 is chosen sufficiently large such that $\beta > 2\sqrt{n}$. Then the predictor step is eventually accepted, and we have*

$$\tau_{k+1} = O(\tau_k^2)$$

for all $k \in \mathbb{N}$ sufficiently large, i.e., the smoothing parameter converges locally Q-quadratically to zero.

Note that a typical interior-point method can guarantee the convergence of the corresponding iteration sequence to a strictly complementary solution, so from this point of view, the assumptions we use in Theorem 3.9 are stronger. However, this is basically the only difference, in particular, we stress that the assumptions used in Theorem 3.9 do not necessarily imply that the solution set of the optimality conditions (3) reduces to a singleton.

We close this section by noting that all results (with the possible exception of Theorem 3.9) would still be true if φ would denote the Fischer-Burmeister function

$$\varphi(a, b) := a + b - \sqrt{a^2 + b^2}$$

from [9] together with its smooth counterpart

$$\varphi_\tau(a, b) := a + b - \sqrt{a^2 + b^2 + 2\tau^2}$$

from [12]; this can be seen by an easy inspection of the previous proofs. On the other hand, it is currently an open question whether or not Theorem 3.9 also holds for the Fischer-Burmeister function.

4 Numerical Results

We implemented Algorithm 2.1 in MATLAB by modifying the LIPSOL code from Zhang [18, 19]. LIPSOL is a primal-dual interior-point solver for linear programs, written in MATLAB and calling a FORTRAN subroutine in order to solve certain linear systems using the sparse Cholesky method by Ng and Peyton [13]. Since the linear systems occuring in Algorithm 2.1 have essentially the same structure as those arising in primal-dual interior-point methods, it was possible to use the numerical linear algebra part from LIPSOL for our implementation of Algorithm 2.1.

The starting point $w^0 = (x^0, \lambda^0, s^0)$ was constructed in the following way:

(a) Solve $AA^T y = b$ using a sparse Cholesky code in order to compute $y^0 \in \mathbb{R}^m$.

(b) Set $x^0 := A^T y^0$.

(c) Solve $AA^T \lambda = Ac$ using a sparse Cholesky code to compute $\lambda^0 \in \mathbb{R}^m$.

(d) Set $s^0 := c - A^T \lambda^0$.

This is exactly the starting point used in [8]. One arrives at this starting point by solving the two simple programs

$$\min \frac{1}{2}\|x\|^2 \quad \text{s.t.} \quad Ax = b$$

for x^0 and

$$\min \frac{1}{2}\|s\|^2 \quad \text{s.t.} \quad A^T\lambda + s = c$$

for λ^0 and s^0. The construction of the starting point guarantees that the two linear systems $Ax = b$ and $A^T\lambda + s = c$ are satisfied in $w^0 = (x^0, \lambda^0, s^0)$. Furthermore, an easy calculation shows that we also have $\theta(x^0, s^0, \tau_0) \leq 0$ (both for the minimum and the Fischer-Burmeister function) provided that we take

$$\tau_0 \geq \sqrt{x_i^0 s_i^0} \quad \forall i \in \{1, \ldots, n\} \text{ with } x_i^0 > 0, s_i^0 > 0.$$

This choice implies that our starting point satisfies all the requirements from Step (S.0) of Algorithm 2.1.

We terminate our iteration if one of the following conditions hold:

(a) $\tau_k < 10^{-4}$ or

(b) $\|\Phi(w^k)\|_\infty < 10^{-4}$ or

(c) $\|\Phi(w^k)\|_\infty < 10^{-3}$ and $\|\Phi(w^k)\|_\infty / \|\Phi(w^0)\|_\infty < 10^{-6}$.

Criterion (a) was used in [8] and is motivated by the fact that the square of τ does, more or less, play the role of the duality gap in interior-point methods (cf. (4)) for which 10^{-8} is a typical value for the stopping parameter. Criterion (b) is an absolute error measuring the total residual $\|\Phi(w^k)\|_\infty$, whereas (c) is a mixture between a weakened form of this absolute

error and a relative error comparing the kth residual $\|\Phi(w^k)\|_\infty$ with the initial residual $\|\Phi(w^0)\|_\infty$.

The remaining parameters from Step (S.0) of Algorithm 2.1 were chosen as follows:

$$\rho = 0.9,\ \beta := \|\Phi_{\tau_0}(w^0)\|/\tau_0$$

and φ being the Fischer-Burmeister function (according to our experience, the Fischer-Burmeister function gives better results than the minimum function, at least within the framework of Algorithm 2.1). Finally, the parameter $\hat{\sigma}_k$ from Step (S.3) of Algorithm 2.1 was always taken to be 0.5.

All test runs were done on a SUN Ultra 2 with 300 MHz, and Table 1 contains the corresponding results, with the columns of Table 1 having the following meanings:

problem:	name of the test problem in the netlib collection,
m:	number of equality constraints (after preprocessing),
n:	number of variables (after preprocessing),
k:	number of iterations until termination,
P:	number of accepted predictor steps,
τ_f:	value of τ_k at the final iterate,
$\|\Phi(w^f)\|_\infty$:	value of $\|\Phi(w^k)\|_\infty$ at the final iterate,
primal objective:	value of the primal objective function at final iterate.

Table 1: Numerical results for Algorithm 2.1

problem	m	n	k	P	τ_f	$\|\Phi(w^f)\|_\infty$	primal objective
25fv47	798	1854	34	17	6.0250e−04	2.9537e−04	5.5018459053e+03
80bau3b	2235	11516	29	23	1.0987e−03	7.1465e−04	9.8722419211e+05
adlittle	55	137	15	15	2.1737e−02	8.7395e−05	2.2549496391e+05
afiro	27	51	10	10	4.7999e−02	3.2452e−05	−4.6474687177e+02
agg	488	615	23	20	1.3406e−02	6.9052e−04	−3.5991767286e+07
agg2	516	758	25	18	7.6969e−03	4.7607e−04	−2.0239252355e+07
agg3	516	758	30	14	7.0617e−03	1.1522e−04	1.0312115936e+07
bandm	269	436	20	19	4.9684e−04	9.3110e−05	−1.5862801756e+02
beaconfd	148	270	18	15	2.7426e−03	5.8563e−04	3.3592485986e+04
blend	74	114	13	12	2.7468e−03	2.9106e−06	−3.0812134385e+01
bnl1	632	1576	26	16	3.5355e−04	7.0895e−05	1.9776295617e+03
bnl2	2268	4430	26	14	9.5617e−04	4.8772e−04	1.8112367543e+03
boeing1	347	722	26	16	2.2843e−03	4.9528e−04	−3.3521310546e+02
boeing2	140	279	16	15	5.4054e−03	9.8945e−04	−3.1500732408e+02
bore3d	199	300	28	22	1.3979e−03	4.1970e−05	1.3730804026e+03
brandy	149	259	19	15	1.1040e−03	7.8205e−05	1.5185099104e+03
capri	267	476	20	19	9.5525e−03	6.4732e−04	2.6900133856e+03
cycle	1801	3305	39	19	9.4566e−05	1.0106e−02	−5.2249915841e+00
czprob	737	3141	22	19	1.1163e−02	2.8069e−04	2.1851966995e+06
d2q06c	2171	5831	57	19	8.3779e−05	4.0045e−05	1.2278421095e+05

Table 1 (continued): Numerical results for Algorithm 2.1

problem	m	n	k	P	τ_f	$\|\Phi(w^f)\|_\infty$	primal objective
d6cube	404	6184	25	21	1.7077e−03	2.6014e−05	3.1549167161e+02
degen2	444	757	23	23	2.3842e−03	9.9759e−05	−1.4351779632e+03
degen3	1503	2604	16	16	7.9692e−04	5.7716e−05	−9.8729398786e+02
dfl001	6071	12230	—	—	—	—	
e226	220	469	27	25	2.4792e−04	5.3902e−05	−1.8751928739e+01
etamacro	357	692	26	13	1.5436e−04	7.3792e−05	−7.5571522983e+02
fffff800	501	1005	36	14	6.2879e−03	8.5460e−04	5.5567957590e+05
finnis	492	1014	31	20	1.3882e−03	2.9195e−04	1.7279127031e+05
fit1d	24	1049	20	18	5.7480e−04	4.0534e−05	−9.1463780917e+03
fit1p	627	1677	19	19	1.7472e−03	7.3692e−06	9.1463780936e+03
fit2d	25	10524	22	20	6.0248e−04	8.2675e−05	−6.8464293289e+04
fit2p	3000	13525	20	20	1.2942e−03	3.0570e−04	6.8464293283e+04
forplan	135	463	28	17	4.7384e−03	9.3267e−04	−6.6421820761e+02
ganges	1137	1534	25	20	3.0644e−03	6.4436e−04	−1.0958573612e+05
gfrd-pnc	600	1144	23	16	1.2681e−02	2.4942e−04	6.9022360024e+06
greenbea	2318	5424	25	20	5.7593e−03	8.5643e−04	−7.2462520306e+07
greenbeb	2317	5415	35	15	2.2551e−03	4.7930e−04	−4.3022602607e+06
grow15	300	645	37	18	2.4283e−02	3.7293e−06	−1.0687094129e+08
grow22	440	946	37	15	1.0882e−01	7.5577e−06	−1.6083433646e+08
grow7	140	301	34	19	3.2322e−02	2.2135e−05	−4.7787811813e+07
israel	174	316	27	17	3.9926e−03	2.8436e−04	−8.9664482178e+05
kb2	43	68	32	10	3.3160e−03	9.8866e−05	−1.7499000911e+03
lotfi	151	364	35	16	3.7591e−03	8.0923e−04	−2.5263066012e+01
maros	835	1921	37	12	3.1239e−03	7.1513e−04	−5.8063742927e+04
maros-r7	3136	9408	22	22	4.7684e−03	9.7763e−04	1.4971851671e+06
modszk1	686	1622	26	17	1.1499e−02	8.3608e−04	3.2061981508e+02
nesm	654	2922	52	14	2.9135e−04	2.4794e−04	1.4076036489e+07
perold	625	1530	33	14	3.3115e−03	7.3875e−04	−9.3805322461e+03
pilot	1441	4657	81	14	1.1707e−04	1.4059e−04	−5.5748445014e+02
pilotja	924	2044	76	14	9.6009e−05	4.3618e−03	−6.1130052461e+03
pilotwe	722	2930	61	13	5.8582e−04	3.7472e−04	−2.7201075333e+06
pilot4	402	1173	132	13	9.5377e−05	6.9395e−03	−2.5810606602e+03
pilot87	2030	6460	63	14	8.3070e−05	8.8733e−03	3.0173031374e+02
pilotnov	951	2242	27	22	2.5409e−03	3.0714e−04	−4.4972761773e+03
recipe	85	177	14	14	1.2207e−03	3.5042e−05	−2.6661598322e+02
sc105	105	163	19	13	1.4451e−03	7.7940e−05	−5.2202033312e+01
sc205	205	317	22	19	7.5991e−04	1.3473e−04	−5.2202035425e+01
sc50a	49	77	15	10	3.6860e−03	4.9576e−05	−6.4575009902e+01
sc50b	48	76	14	11	6.6556e−03	7.6186e−06	−6.9999776566e+01
scagr25	471	671	19	17	2.3406e−02	2.9238e−04	−1.4753433056e+07
scagr7	129	185	19	18	3.4159e−03	3.3656e−04	−2.3313898243e+06
scfxm1	322	592	20	19	5.9750e−03	6.4703e−04	1.8416759818e+04

Table 1 (continued): Numerical results for Algorithm 2.1

problem	m	n	k	P	τ_f	$\|\Phi(w^f)\|_\infty$	primal objective
scfxm2	644	1184	26	18	4.3503e−03	8.7736e−04	3.6660262213e+04
scfxm3	966	1776	26	21	4.9124e−03	9.6075e−04	5.4901255716e+04
scorpion	375	453	21	20	2.7373e−04	1.8825e−05	1.8781248227e+03
scrs8	485	1270	21	19	6.6791e−04	4.5185e−05	9.0429695560e+02
scsd1	77	760	22	22	4.7684e−03	9.5696e−06	8.6666991041e+00
scsd6	147	1350	15	15	4.0199e−04	1.1125e−06	5.0500000067e+01
scsd8	397	2750	13	13	1.1068e−02	4.6656e−05	9.0500023711e+02
sctap1	300	660	24	23	4.0019e−03	5.1964e−05	1.4122500207e+03
sctap2	1090	2500	18	16	1.5629e−03	1.4780e−05	1.7248071430e+03
sctap3	1480	3340	18	17	2.1396e−03	9.2047e−05	1.4240000008e+03
seba	515	1036	23	15	3.1158e−03	9.3931e−05	1.5711600096e+04
share1b	112	248	43	14	3.0326e−03	9.3991e−04	−7.6589318369e+04
share2b	96	162	16	16	3.0518e−04	4.4814e−07	−4.1573224024e+02
shell	496	1487	22	14	1.7418e−01	1.6486e−05	1.2088253461e+09
ship04l	356	2162	20	20	1.6465e−02	8.0608e−04	1.7933245380e+06
ship04s	268	1414	20	20	1.2018e−02	6.5490e−05	1.7987147004e+06
ship08l	688	4339	21	20	1.0490e−02	3.7678e−04	1.9090552114e+06
ship08s	416	2171	20	20	1.8916e−02	1.4499e−04	1.9200982105e+06
ship12l	838	5329	21	20	8.4547e−03	5.6258e−04	1.4701879193e+06
ship12s	466	2293	20	19	8.9471e−03	6.2617e−04	1.4892361344e+06
sierra	1222	2715	22	19	1.9638e−02	9.5043e−04	1.5394362263e+07
stair	356	538	19	18	2.3050e−03	1.6815e−04	−2.5126689656e+02
standata	359	1258	13	12	7.2079e−02	9.5241e−06	1.2577586668e+03
standgub	361	1366	13	12	7.2079e−02	9.5241e−06	1.2577586668e+03
standmps	467	1258	18	14	9.0258e−03	1.1631e−05	1.4060176463e+03
stocfor1	109	157	16	11	3.3401e−02	1.2536e−04	−4.1131976111e+04
stocfor2	2157	3045	29	16	8.3230e−04	2.4732e−05	−3.9024408532e+04
stocfor3	16675	23541	63	17	1.8715e−04	2.8605e−04	−3.9976784284e+04
stocfor3old	16675	23541	70	13	8.7370e−05	6.0456e−04	−3.9976783942e+04
truss	1000	8806	19	18	6.3715e−03	5.6972e−05	4.5881584778e+05
tuff	292	617	32	16	3.2629e−04	1.5624e−04	2.9216987102e−01
vtpbase	194	325	19	19	3.8147e−02	3.2374e−05	1.2983146617e+05
wood1p	244	2595	13	13	3.3617e−04	6.1025e−05	1.4429024524e+00
woodw	1098	8418	34	22	2.2390e−04	9.7122e−05	1.3044869516e+00

The overall results are quite good and seem to be better than the corresponding results from the three-step method described in [8]. The method has only one failure on problem `dfl001` (interestingly, LIPSOL also produces an error for this example, at least on our machine), and most test problems can be solved in less than 20–30 iterations. Although interior-point methods are still more efficient on most examples, the numerical behaviour of our smoothing-type method is getting pretty close to the one of interior-point methods, and is definitely approaching an area where it may be viewed as a possible alternative to

interior-point methods.

5 Concluding Remarks

In this paper, we modified the recently proposed smoothing-type methods from [2, 8]. The modified method has some stronger global and/or local convergence properties than the methods from [2, 8], and the numerical results indicate that the method works very well on the netlib test problem collection. Since these results were obtained by using the Fischer-Burmeister function (rather than the minimum function) and since the local convergence result from Theorem 3.9 does not necessarily hold for the Fischer-Burmeister function, this function certainly deserves further investigation. In fact, this is part of our future research, and we hope that this, in turn, will have a positive influence on our implementation of the predictor step.

References

[1] J.V. Burke and S. Xu: *A non-interior predictor-corrector path-following method for LCP.* In: M. Fukushima and L. Qi (eds.): *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods.* Kluwer Academic Publishers, 1999.

[2] J.V. Burke and S. Xu: *A non-interior-predictor-corrector path following algorithm for the monotone linear complementarity problem.* Mathematical Programming 87, 2000, pp. 113–130.

[3] B. Chen and X. Chen: *A global and local superlinear continuation-smoothing method for P_0 and R_0 NCP or monotone NCP.* SIAM Journal on Optimization 9, 1999, pp. 624–645.

[4] B. Chen and P.T. Harker: *A non-interior-point continuation method for linear complementarity problems.* SIAM Journal on Matrix Analysis and Applications 14, 1993, pp. 1168–1190

[5] B. Chen and N. Xiu: *A global linear and local quadratic noninterior continuation method for nonlinear complementarity problems based on Chen-Mangasarian smoothing functions.* SIAM Journal on Optimization 9, 1999, pp. 605–623.

[6] X. Chen, L. Qi and D. Sun: *Global and superlinear convergence of the smoothing Newton method and its application to general box constrained variational inequalities.* Mathematics of Computation 67, 1998, pp. 519–540.

[7] S. Engelke and C. Kanzow: *On the solution of linear programs by Jacobian smoothing methods.* Annals of Operations Research, to appear.

[8] S. Engelke and C. Kanzow: *Predictor-corrector smoothing methods for the solution of linear programs.* Preprint 153, Institute of Applied Mathematics, University of Hamburg, Hamburg, March 2000.

[9] A. Fischer: *A special Newton-type optimization method.* Optimization 24, 1992, pp. 269–284.

[10] K. Hotta and A. Yoshise: *Global convergence of a class of non-interior point algorithms using Chen-Harker-Kanzow-Smale functions for nonlinear complementarity problems.* Mathematical Programming 86, 1999, pp. 105–133.

[11] H. Jiang: *Smoothed Fischer-Burmeister equation methods for the complementarity problem.* Technical Report, Department of Mathematics, University of Melbourne, Melbourne, Australia, June 1997.

[12] C. Kanzow: *Some noninterior continuation methods for linear complementarity problems.* SIAM Journal on Matrix Analysis and Applications 17, 1996, pp. 851–868.

[13] E. Ng and B.W. Peyton: *Block sparse Cholesky algorithm on advanced uniprocessor computers.* SIAM Journal on Scientific Computing 14, 1993, pp. 1034–1056.

[14] S. Smale: *Algorithms for solving equations.* In *Proceedings of the International Congress of Mathematicians.* AMS, Providence, 1987, pp. 172–195.

[15] P. Tseng: *Analysis of a non-interior continuation method based on Chen-Mangasarian smoothing functions for complementarity problems.* In: M. Fukushima and L. Qi (eds.): *Reformuation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods.* Kluwer Academic Publishers, 1998, pp. 381–404.

[16] P. Tseng: *Error bounds and superlinear convergence analysis of some Newton-type methods in optimization.* In: G. Di Pillo and F. Giannessi (eds.): *Nonlinear Optimization and Related Topics.* Kluwer Academic Publishers, to appear.

[17] S.J. Wright: *Primal-Dual Interior-Point Methods.* SIAM, Philadelphia, PA, 1997.

[18] Y. Zhang: *Solving large-scale linear programs by interior-point methods under the MATLAB environment.* Optimization Methods and Software 10, 1998, pp. 1–31.

[19] Y. Zhang: *User's guide to LIPSOL: Linear programming interior point solver v0.4.* Optimization Methods and Software 11 & 12, 1999, pp. 385–396.