Arc-Search Path-Following Interior-Point Algorithms for Linear Programming

Yaguang Yang*

Abstract

Arc-search is developed in optimization algorithms. In this paper, simple analytic arcs are used to approximate the central path of the linear programming. The primal-dual path-following interior-point algorithms are then used to search optimizers along the arcs for linear programming. They require fewer iterations to find the optimal solutions in all the tested problems in Netlib than Matlab optimization toolbox `linprog` which implements the state-of-art Mehrotra's predictor-corrector algorithm.

Keywords: Arc-search, path-following, linear programming.

*NRC, Office of Research, 21 Church Street, Rockville, 20850. Email: yaguang.yang@verizon.net

1 Introduction

The majority of optimization algorithms that use information of derivatives use line search. However, most optimization problems are nonlinear, and the ideal search should be carried out along arcs, such as central path in linear programming, not straight lines. This paper develops arc-search techniques and applies them to primal-dual path-following interior-point method for linear programming. The proposed algorithms use both first derivative and second derivative of the iterates on the arcs that approximate the central path. Therefore, we will compare them with higher-order interior-point algorithms and other popular interior-point algorithms under commonly used criteria such as iteration count and the computational performance for standard test problems.

One of the most important theoretical criteria to estimate an algorithm's performance in the worst case is its polynomial complexity bound. This bound estimates how many iterations are needed for an algorithm to find the solution in the worst case. Intuitively, the longer step an iteration takes in the right direction, the more improvement will be obtained. Some important polynomial algorithms in early development are short-step path-following algorithm and long-step path-following algorithm proposed, for example, in [6, 7]. The main difference between these two algorthms is that they restrict iterates in different neighborhoods. The long-step path-following algorithms use larger neighborhood, therefore search step moves farther in current iteration, but may generate an iterate too close to the boundary that causes difficult in the following search steps. On the other hand, the short-step path-following algorithms use smaller neighborhood, therefore search step moves shorter in current iteration, but may generate an iterate closer to the central path that benefits the following search steps. Extensive test shows that long-step path-following algorithms perform better in numerical test. However, the short-step path-following algorithms have proved better in complexity bound $O(\sqrt{n}\log(1/\epsilon))$ which is still the best among all interior-point algorithms.

A natural idea is then to conduct the current search using a larger neighborhood, and move the iterate back to a smaller neighborhood closer to the central path. Therefore, predictor-corrector algorithm is developed [10]. The complexity of predictor-corrector algorithm is $O(\sqrt{n}\log(1/\epsilon))$ but the search is restricted in a smaller neighborhood than long-step path-following algorithms.

The third way to increase the step length is to search along an arc that approximates the central path or along a combined direction which balances two different goals: reducing duality gap and staying close to the central path. These higher-order algorithms use at least the first and second derivatives at every iteration point. Some algorithms use power series of order r at iteration points to approximate the central path and search along the arc defined by the power series, for example, [11, 5]. These algorithms use larger restricted neighborhood, the same as long-step algorithms, but do not use corrector step after a search. They all achieved $O(n^{\frac{r+1}{2r}}\log(1/\epsilon))$ complexity bound which is better than long-step algorithms but still does not reach the bound of short-step algorithms (because of using larger neighborhood without using corrector). Intuitively, the power series could approximate the central path well in a small neighborhood near the iterates but may not approximate well in a large area. To date, the most successful interior-point algorithm is MPC [14] that is a variant of Mehrotra's original algorithm [9]. It searches along a combined directions related to the first and second order derivatives. Computational test on standard Netlib problems shows that MPC is very efficient [8]. However, no global convergence or polynomial complexity are known. Actually, it is noticed [14, 1] that MPC may not converge in some cases. Recently, [13] proved that some variants of MPC have polynomial bounds of $O(n\log(1/\epsilon))$ and $O(n^2\log(1/\epsilon))$ which are much worse than the best known complexity bound even though the practical performance of these variants is unknown.

In this paper, we present two algorithms that search in a larger neighborhood. These algorithms use more accurate centering process (the iterates are close to or on the central path) than MPC and other predictor-corrector algorithms. They search along arcs that approximate the entire central path. We prove that the proposed algorithms are globally convergent. We will also provide test results on standard Netlib problems. It shows that the newly developed codes use fewer iterations in all the tested problems than Matlab code `linprog` which implements state-of-the-art MPC plus some other enhencements [16].

Throughout the paper, we will denote Hadamard (element-wise) product of two vectors x and s by $x \circ s$, the ith component of x by x_i, element-wise inverse of x by x^{-1} if $\min|s_i| > 0$, element-wise division of the two vectors by $s^{-1} \circ x$, or $x \circ s^{-1}$, or $\frac{x}{s}$ if $\min|s_i| > 0$, the inner product of two vectors x and

s by $\langle x, s \rangle$, the Euclidean norm of x by $\|x\|$, the infinite norm of x by $\|x\|_\infty$, the identity matrix of any dimension by I, the identity matrix of dimension m by I_m, the vector of all ones with appropriate dimensions by e, the transpose of matrix A by A^{T}. To make the notation simple for block column vectors, we will denote, for example, $[x^{\mathrm{T}}, s^{\mathrm{T}}]^{\mathrm{T}}$ by (x, s). For $x \in \mathbf{R}^n$, we will denote a related diagonal matrix by $X \in \mathbf{R}^{n \times n}$ whose diagonal elements are components of the vector x. Finally, we define an initial point of any algorithm by x^0, the point after the kth iteration by x^k.

2 Arc-Search in Linear Programming

Consider the Linear Programming (LP) in the standard form:

$$(LP) \quad \min \ c^T x, \quad \text{subject to} \quad Ax = b, \quad x \geq 0, \tag{1}$$

where $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$, $c \in \mathbf{R}^n$ are given, and $x \in \mathbf{R}^n$ is the vector to be optimized. Associated with the linear programming is the dual programming (DP) that is also presented in the standard form:

$$(DP) \quad \max \ b^T \lambda, \quad \text{subject to} \quad A^T \lambda + s = c, \quad s \geq 0, \tag{2}$$

where dual variable vector $\lambda \in \mathbf{R}^m$, and dual slack vector $s \in \mathbf{R}^n$. Denote the feasible set $\mathcal{F}$ as a collection of all points that meet the constraints of LP and DP,

$$\mathcal{F} = \{(x, \lambda, s) : Ax = b, A^T \lambda + s = c, (x, s) \geq 0\}, \tag{3}$$

and the strictly feasible set $\mathcal{F}^o$ as a collection of all points that meet the constraints of LP and DP and are strictly positive

$$\mathcal{F}^o = \{(x, \lambda, s) : Ax = b, A^T \lambda + s = c, (x, s) > 0\}. \tag{4}$$

Throughout the paper, we make the following assumptions.

Assumptions:

1. A is a full rank matrix.

2. $\mathcal{F}^o$ is not empty.

Assumption 1 is a standard assumption for use of KKT (Karush-Kuhn-Tucher) conditions. Assumption 2 implies existence of a central path. It is well known that $x \in \mathbf{R}^n$ is an optimal solution of (1) if and only if x, λ, and s meet the following KKT conditions

$$Ax = b \tag{5a}$$
$$A^T \lambda + s = c \tag{5b}$$
$$(x, s) \geq 0. \tag{5c}$$
$$x_i s_i = 0, \quad i = 1, \ldots, n \tag{5d}$$

The first three conditions imply that x is a feasible solution of the primal problem and (λ, s) is a feasible solution of the dual problem. The last condition implies that the duality gap is zero. Under the Assumption 2, interior-point algorithms exist. We will consider central path-following algorithms that try to search the optimizers (located at the boundary of $\mathcal{F}$) along an arc that approximates the central path $\mathcal{C} \in \mathcal{F}^o \subset \mathcal{F}$. The central path $\mathcal{C}$ is parameterized by a scalar $\tau > 0$ as follows. For each interior point $(x, \lambda, s) \in \mathcal{E}$ close to the central path, there is a $\tau > 0$ such that

$$Ax = b \tag{6a}$$
$$A^T \lambda + s = c \tag{6b}$$
$$(x, s) > 0 \tag{6c}$$
$$x_i s_i = \tau, \quad i = 1, \ldots, n. \tag{6d}$$

Therefore, the central path is an arc in $\mathbf{R}^{2n+m}$ parameterized as a function of τ and is denoted as

$$\mathcal{C} = \{(x(\tau), \lambda(\tau), s(\tau)) : \tau > 0\}. \tag{7}$$

As $\tau \to 0$, the central path $(x(\tau), \lambda(\tau), s(\tau))$ represented by (6) approaches to the solution of LP represented by (1). Theoretical analyses and computational experiments demonstrate [12] that searching along the central path is the most efficient way to find optimizers. However, there is no practical way to calculate the entire arc of the central path. All path-following algorithms try (a) to search, from the current (x, s) along certain directions related to the tangent of the central path, to a new point that reduces the value of $x^T s$ (the duality gap) and simultaneously meets (6a), (6b), and (6c), thereby moving the current point towards the solution, and (b) to stay close to the central path, thereby being able to make a good progress in the next search.

The idea of arc-search proposed in this paper is very simple. The process starts from any point (may or may not be feasible) (x^k, λ^k, s^k), finds a point $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ close to or on the central path, defines an arc passing through the point and approximating the central path, searches along the arc to a new point $(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x(\alpha), \lambda(\alpha), s(\alpha))$ that reduces the value of $x^T s$ and meets (6a), (6b), and (6c). The process is repeated by finding a better point $(\bar{x}^{k+1}, \bar{\lambda}^{k+1}, \bar{s}^{k+1})$ close to or on the central path. Intuitively, arc-search will take a longer step than line search methods, and the new point generated by the arc-search should be closer to the central path than those generated by line search methods. We will show that the proposed arc-search is simple to implement and uses fewer iterations than MPC algorithm in numerical test examples.

2.1 Move towards the central path

Denote $X = \mathrm{diag}(x_1, \ldots, x_n)$, $S = \mathrm{diag}(s_1, \ldots, s_n)$, and

$$\mu = \frac{x^T s}{n}. \tag{8}$$

Starting from any point (x, λ, s) with $(x, s) > 0$ that may or may not be in $\mathcal{F}^o$, to move the point to a point close to or on the central path is equivalent to solve

$$F(x(t), \lambda(t), s(t)) = \begin{pmatrix} Ax - b \\ A^T\lambda + s - c \\ XSe - t\mu e \end{pmatrix} = 0, \qquad (x, s) > 0. \tag{9}$$

(9) can be solved by repeatedly searching along Newton directions while keeping $(x, s) > 0$. In each step, the Newton direction $(dx^k, d\lambda^k, ds^k)$ can be calculated by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} dx^k \\ d\lambda^k \\ ds^k \end{bmatrix} = \begin{bmatrix} -r_B \\ -r_C \\ -r_t \end{bmatrix}, \tag{10}$$

where r_B, r_C, and r_t are defined in (11a), (11b), and (11c). r_B and r_C are zeros if a feasible point is obtained. This process is described in the following

Algorithm 2.1 (move towards the central path)
Data: A, b, c, $\epsilon > 0$, $\theta \in (0, 1)$, $\mu > 0$, *and initial point* (x^0, λ^0, s^0) *with* $(x^0, s^0) > 0$.
for *iteration* $k = 1, 2, \ldots$

Check conditions

$$\|r_B\| = \|Ax^k - b\| \le \epsilon, \tag{11a}$$

$$\|r_C\| = \|A^{\mathrm{T}}\lambda^k + s^k - c\| \le \epsilon, \tag{11b}$$

$$\|r_t\| = \|X^k S^k e - \frac{\mu}{2} e\| \le \epsilon, \tag{11c}$$

$$(x^k, s^k) > 0. \tag{11d}$$

If (11) holds, (x^k, λ^k, s^k) is the starting point on the arc which approximates the central path. Set the solution $(\bar{x}, \bar{\lambda}, \bar{s}) = (x^k, \lambda^k, s^k)$ and stop.

If (11) does not hold, calculate the Newton direction $(dx^k, d\lambda^k, ds^k)$ from (10). Carry out line search along the Newton direction

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k + \alpha dx^k, \lambda^k + \alpha d\lambda^k, s^k + \alpha ds^k) \tag{12}$$

such that the $\alpha > 0$ satisfies

$$\min_{\alpha \in (0,1], X^{k+1}s^{k+1} \geq \theta\mu e} \left(\|Ax^{k+1} - b\|^2 + \|A^{\mathrm{T}}\lambda^{k+1} + s^{k+1} - c\|^2 + \|X^{k+1}S^{k+1}e - \frac{\mu}{2}e\|^2 \right)$$

end (for) ∎

Remark 2.1 *The proposed algorithm is essentially the same as Algorithm 1 of [2] except that line search is used to replace an analytic selection of α. [2] showed that if $(x^k, s^k) > 0$ and $X^k s^k \geq \theta\mu e$, then the search along Newton direction will generate positive iterates and $X^{k+1}s^{k+1} \geq \theta\mu e$. A more versatile method of finding a relative interior close to the central path was proposed in (Algorithm 2) [2]. The global convergence and locally quadratic convergence rate are proved for both algorithms in the same paper.*

2.2 Ellipse approximation of the central path

We will use an ellipse $\mathcal{E}$ [3] in $2n + m$ dimensional space to approximate the central path described by (6), where

$$\mathcal{E} = \{(x(t), \lambda(t), s(t)) : (x(t), \lambda(t), s(t)) = \vec{a}\cos(t) + \vec{b}\sin(t) + \vec{c}\}, \tag{13}$$

$\vec{a} \in \mathbf{R}^{2n+m}$ and $\vec{b} \in \mathbf{R}^{2n+m}$ are the axes of the ellipse, and they are perpendicular to each other, $\vec{c} \in \mathbf{R}^{2n+m}$ is the center of the ellipse. Given a point $y = (x, \lambda, s) = (x(t_0), \lambda(t_0), s(t_0)) \in \mathcal{E}$ which is close to or on the central path, we will determine $\vec{a}$, $\vec{b}$, $\vec{c}$ and t_0 such that the first and second derivatives of $(x(t_0), \lambda(t_0), s(t_0))$ have the form as if they were on the central path (though they may not be on the central path). Therefore, we want the first and second derivatives at $(x(t_0), \lambda(t_0), s(t_0)) \in \mathcal{E}$ to meet

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{\lambda} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mu e \end{bmatrix}, \tag{14}$$

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\lambda} \\ \ddot{s} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -2\dot{x} \circ \dot{s} \end{bmatrix}. \tag{15}$$

It is clear that such an ellipse should approximate the central path well when $t_0 \pm \epsilon \to t_0$. To simplify the notation, let

$$y(t) = (x(t), \lambda(t), s(t)) = \vec{a}\cos(t) + \vec{b}\sin(t) + \vec{c}. \tag{16}$$

Then

$$\dot{y}(t) = (\dot{x}(t), \dot{\lambda}(t), \dot{s}(t)) = -\vec{a}\sin(t) + \vec{b}\cos(t), \tag{17}$$

$$\ddot{y}(t) = (\ddot{x}(t), \ddot{\lambda}(t), \ddot{s}(t)) = -\vec{a}\cos(t) - \vec{b}\sin(t). \tag{18}$$

It is straightforward to verify from (16), (17), and (18) that

$$\vec{a} = -\dot{y}\sin(t) - \ddot{y}\cos(t), \tag{19}$$

$$\vec{b} = \dot{y}\cos(t) - \ddot{y}\sin(t), \tag{20}$$

$$\vec{c} = y + \ddot{y}. \tag{21}$$

The search along the ellipse will be carried out on the interval $[t_0 - \alpha, t_0]$ and $\alpha \in [0, \frac{\pi}{2}]$. In the next subsection, we will show that the calculation of t_0 can be avoided.

2.3 Search along the approximate central path

Though one can search the optimizer along the ellipse defined by (16) which needs to computer $\vec{a}$, $\vec{b}$, and $\vec{c}$, we will use a simplified formula that reduces the operation counts slightly and is more convenient for convergence analysis. Denote

$$\vec{a} = \begin{bmatrix} a_x \\ a_\lambda \\ a_s \end{bmatrix} = -\dot{y}\sin(t) - \ddot{y}\cos(t) = \begin{bmatrix} -\dot{x}\sin(t) - \ddot{x}\cos(t) \\ -\dot{\lambda}\sin(t) - \ddot{\lambda}\cos(t) \\ -\dot{s}\sin(t) - \ddot{s}\cos(t) \end{bmatrix},$$

$$\vec{b} = \begin{bmatrix} b_x \\ b_\lambda \\ b_s \end{bmatrix} = \dot{y}\cos(t) - \ddot{y}\sin(t) = \begin{bmatrix} \dot{x}\cos(t) - \ddot{x}\sin(t) \\ \dot{\lambda}\cos(t) - \ddot{\lambda}\sin(t) \\ \dot{s}\cos(t) - \ddot{s}\sin(t) \end{bmatrix},$$

and

$$\vec{c} = \begin{bmatrix} c_x \\ c_\lambda \\ c_s \end{bmatrix} = y + \ddot{y} = \begin{bmatrix} x + \ddot{x} \\ \lambda + \ddot{\lambda} \\ s + \ddot{s} \end{bmatrix}.$$

Let $x(\alpha)$ and $s(\alpha)$ be the updated x and s after the search, we have

$$\begin{aligned} & x(\alpha) = a_x\cos(t_0-\alpha) + b_x\sin(t_0-\alpha) + c_x \\ = \; & a_x(\cos(t_0)\cos(\alpha) + \sin(t_0)\sin(\alpha)) + b_x(\sin(t_0)\cos(\alpha) - \cos(t_0)\sin(\alpha)) \\ & + c_x - c_x\cos(\alpha) + c_x\cos(\alpha) \\ = \; & x\cos(\alpha) + a_x\sin(t_0)\sin(\alpha) - b_x\cos(t_0)\sin(\alpha) + c_x(1-\cos(\alpha)) \\ = \; & x\cos(\alpha) - (\dot{x}\sin(t_0) + \ddot{x}\cos(t_0))\sin(t_0)\sin(\alpha) \\ & -(\dot{x}\cos(t_0) - \ddot{x}\sin(t_0))\cos(t_0)\sin(\alpha) + (x+\ddot{x})(1-\cos(\alpha)) \\ = \; & x - \dot{x}(\sin^2(t_0)\sin(\alpha) + \cos^2(t_0)\sin(\alpha)) \\ & + \ddot{x}(-\sin(t_0)\cos(t_0)\sin(\alpha) + \sin(t_0)\cos(t_0)\sin(\alpha) + (1-\cos(\alpha))) \\ = \; & x - \dot{x}\sin(\alpha) + \ddot{x}(1-\cos(\alpha)). \end{aligned} \tag{22}$$

Similarly

$$\begin{aligned} & s(\alpha) = a_s\cos(t_0-\alpha) + b_s\sin(t_0-\alpha) + c_s \\ = \; & a_s(\cos(t_0)\cos(\alpha) + \sin(t_0)\sin(\alpha)) + b_s(\sin(t_0)\cos(\alpha) - \cos(t_0)\sin(\alpha)) \\ & + c_s - c_s\cos(\alpha) + c_s\cos(\alpha) \\ = \; & s\cos(\alpha) + a_s\sin(t_0)\sin(\alpha) - b_s\cos(t_0)\sin(\alpha) + c_s(1-\cos(\alpha)) \\ = \; & s\cos(\alpha) - (\dot{s}\sin(t_0) + \ddot{s}\cos(t_0))\sin(t_0)\sin(\alpha) \\ & -(\dot{s}\cos(t_0) - \ddot{s}\sin(t_0))\cos(t_0)\sin(\alpha) + (s+\ddot{s})(1-\cos(\alpha)) \\ = \; & s - \dot{s}(\sin^2(t_0)\sin(\alpha) + \cos^2(t_0)\sin(\alpha)) \\ & + \ddot{s}(-\sin(t_0)\cos(t_0)\sin(\alpha) + \sin(t_0)\cos(t_0)\sin(\alpha) + (1-\cos(\alpha))) \\ = \; & s - \dot{s}\sin(\alpha) + \ddot{s}(1-\cos(\alpha)), \end{aligned} \tag{23}$$

and

$$\lambda(\alpha) \quad = \quad \lambda - \dot{\lambda}\sin(\alpha) + \ddot{\lambda}(1-\cos(\alpha)). \tag{24}$$

As pointed above, (22), (23), and (24) do not depend on t_0 explicitly. We summarize the above discussion as the following

Theorem 2.1 *Let $(x(t), \lambda(t), s(t))$ be an arc defined by (6) passing through a point $(x, \lambda, s) \in \mathcal{E}$, and its first and second derivatives at (x, λ, s) be $(\dot{x}, \dot{\lambda}, \dot{s})$ and $(\ddot{x}, \ddot{\lambda}, \ddot{s})$. Then an ellipse approximation of the central path is given by*

$$x(\alpha) = x - \dot{x}\sin(\alpha) + \ddot{x}(1-\cos(\alpha)). \tag{25}$$

$$\lambda(\alpha) = \lambda - \dot{\lambda}\sin(\alpha) + \ddot{\lambda}(1 - \cos(\alpha)). \tag{26}$$

$$s(\alpha) = s - \dot{s}\sin(\alpha) + \ddot{s}(1 - \cos(\alpha)). \tag{27}$$

∎

Assuming $(x, s) > 0$, one can easily see that if $\dot{x}$, $\ddot{x}$, $\dot{s}$, and $\ddot{s}$ are bounded, and if α is small enough, then $x(\alpha) > 0$ and $s(\alpha) > 0$.

Lemma 2.1 *Let $\dot{x}$, $\dot{s}$, $\ddot{x}$, and $\ddot{s}$ be the solution of (14) and (15). Then*

$$\dot{x}^{\mathrm{T}}\dot{s} = 0,$$

$$\ddot{x}^{\mathrm{T}}\dot{s} = 0,$$

$$\dot{x}^{\mathrm{T}}\ddot{s} = 0,$$

$$\ddot{x}^{\mathrm{T}}\ddot{s} = 0.$$

Proof: Pre-multiplying $\dot{x}^{\mathrm{T}}$ or $\ddot{x}^{\mathrm{T}}$ to the second rows of (14) and (15), and using the first rows of (14) and (15) gives the results. ∎

We will show that searching along this arc will reduce the duality gap, or $\mu(\alpha) = \frac{x(\alpha)^{\mathrm{T}}s(\alpha)}{n} < \mu$. If $(x(\alpha), s(\alpha)) > 0$ holds in all iterations, reducing duality gap to zero means approaching to the solution of the linear programming. Notice that

$$\mu(\alpha) = \mu(1 - \sin(\alpha)) \tag{28}$$

holds for any choice of $\alpha \in [0, \frac{\pi}{2}]$ due to the previous lemma, this means that the larger the α is, the more improvement the $\mu(\alpha)$ will be. Therefore, we will select the largest $\hat{\alpha}$ such that all $\alpha \in [0, \hat{\alpha}]$ satisafy

$$x(\alpha) = x - \dot{x}\sin(\alpha) + \ddot{x}(1 - \cos(\alpha)) \geq \sigma x, \tag{29a}$$

$$s(\alpha) = s - \dot{s}\sin(\alpha) + \ddot{s}(1 - \cos(\alpha)) \geq \sigma s, \tag{29b}$$

for some small $\sigma \in (0, 1)$. This can be done as follows. For each $i \in \{1, \ldots, n\}$, we can select the largest α_{x_i} such that for any $\alpha \in [0, \alpha_{x_i}]$, the ith inequality of (29a) holds, and the largest α_{s_i} such that for any $\alpha \in [0, \alpha_{s_i}]$ the ith inequality of (29b) holds. We then define

$$\hat{\alpha} = \min_{i \in \{1,\ldots,n\}} \{\alpha_{x_i}, \alpha_{s_i}\}. \tag{30}$$

α_{x_i} and α_{s_i} can be given in analytical forms according to the values of $\dot{x}_i$, $\ddot{x}_i$, $\dot{s}_i$, $\ddot{s}_i$. First, from (29a), we have

$$x_i - \sigma x_i + \ddot{x}_i \geq \dot{x}_i \sin(\alpha) + \ddot{x}_i \cos(\alpha). \tag{31}$$

Case 1 ($\dot{x}_i = 0$ and $\ddot{x}_i \neq 0$):

For $\ddot{x}_i \geq -(x_i - \sigma x_i)$, and for any $\alpha \in [0, \frac{\pi}{2}]$, $x_i(\alpha) \geq \sigma x_i$ holds. For $\ddot{x}_i \leq -(x_i - \sigma x_i)$, to meet (31), we must have $\cos(\alpha) \geq \frac{x_i - \sigma x_i + \ddot{x}_i}{\ddot{x}_i}$, or, $\alpha \leq \cos^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\ddot{x}_i}\right)$. Therefore,

$$\alpha_{x_i} = \begin{cases} \frac{\pi}{2} & \text{if } x_i - \sigma x_i + \ddot{x}_i \geq 0 \\ \cos^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\ddot{x}_i}\right) & \text{if } x_i - \sigma x_i + \ddot{x}_i \leq 0. \end{cases} \tag{32}$$

Case 2 ($\ddot{x}_i = 0$ and $\dot{x}_i \neq 0$):

For $\dot{x}_i \leq x_i - \sigma x_i$, and for any $\alpha \in [0, \frac{\pi}{2}]$, $x_i(\alpha) \geq \sigma x_i$ holds. For $\dot{x}_i \geq x_i - \sigma x_i$, to meet (31), we must have $\sin(\alpha) \leq \frac{x_i - \sigma x_i}{\dot{x}_i}$, or $\alpha \leq \sin^{-1}\left(\frac{x_i - \sigma x_i}{\dot{x}_i}\right)$. Therefore,

$$\alpha_{x_i} = \begin{cases} \frac{\pi}{2} & \text{if } \dot{x}_i \leq x_i - \sigma x_i \\ \sin^{-1}\left(\frac{x_i - \sigma x_i}{\dot{x}_i}\right) & \text{if } \dot{x}_i \geq x_i - \sigma x_i \end{cases} \tag{33}$$

Case 3 ($\dot{x}_i > 0$ and $\ddot{x}_i > 0$):

Let $\dot{x}_i = \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\cos(\beta)$, and $\ddot{x}_i = \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\beta)$, (31) can be rewritten as

$$x_i - \sigma x_i + \ddot{x}_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\alpha + \beta), \tag{34}$$

where

$$\beta = \sin^{-1}\left(\frac{\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right). \tag{35}$$

For $\ddot{x}_i + x_i - \sigma x_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}$, and for any $\alpha \in [0, \frac{\pi}{2}]$, $x_i(\alpha) \geq \sigma x_i$ holds. For $\ddot{x}_i + x_i - \sigma x_i \leq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}$, to meet (34), we must have $\sin(\alpha + \beta) \leq \frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}$, or $\alpha + \beta \leq \sin^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right)$. Therefore,

$$\alpha_{x_i} = \begin{cases} \frac{\pi}{2} & \text{if } x_i - \sigma x_i + \ddot{x}_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2} \\ \sin^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) - \sin^{-1}\left(\frac{\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) & \text{if } x_i - \sigma x_i + \ddot{x}_i \leq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2} \end{cases} \tag{36}$$

Case 4 ($\dot{x}_i > 0$ and $\ddot{x}_i < 0$):

Let $\dot{x}_i = \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\cos(\beta)$, and $\ddot{x}_i = -\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\beta)$, (31) can be rewritten as

$$x_i - \sigma x_i + \ddot{x}_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\alpha - \beta), \tag{37}$$

where

$$\beta = \sin^{-1}\left(\frac{-\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right). \tag{38}$$

For $\ddot{x}_i + x_i - \sigma x_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}$, and for any $\alpha \in [0, \frac{\pi}{2}]$, $x_i(\alpha) \geq \sigma x_i$ holds. For $\ddot{x}_i + x_i - \sigma x_i \leq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2}$, to meet (37), we must have $\sin(\alpha - \beta) \leq \frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}$, or $\alpha - \beta \leq \sin^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right)$. Therefore,

$$\alpha_{x_i} = \begin{cases} \frac{\pi}{2} & \text{if } x_i - \sigma x_i + \ddot{x}_i \geq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2} \\ \sin^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) + \sin^{-1}\left(\frac{-\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) & \text{if } x_i - \sigma x_i + \ddot{x}_i \leq \sqrt{\dot{x}_i^2 + \ddot{x}_i^2} \end{cases} \tag{39}$$

Case 5 ($\dot{x}_i < 0$ and $\ddot{x}_i < 0$):

Let $\dot{x}_i = -\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\cos(\beta)$, and $\ddot{x}_i = -\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\beta)$, (31) can be rewritten as

$$x_i - \sigma x_i + \ddot{x}_i \geq -\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}\sin(\alpha + \beta), \tag{40}$$

where

$$\beta = \sin^{-1}\left(\frac{-\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right). \tag{41}$$

For $\ddot{x}_i + (x_i - \sigma x_i) \geq 0$, for any $\alpha \in [0, \frac{\pi}{2}]$, $x_i(\alpha) \geq \sigma x_i$ holds. For $\ddot{x}_i + (x_i - \sigma x_i) \leq 0$, to meet (40), we must have $\sin(\alpha + \beta) \geq \frac{-(x_i - \sigma x_i + \ddot{x}_i)}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}$, or $\alpha + \beta \leq \pi - \sin^{-1}\left(\frac{-(x_i - \sigma x_i + \ddot{x}_i)}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right)$. Therefore,

$$\alpha_{x_i} = \begin{cases} \frac{\pi}{2} & \text{if } x_i - \sigma x_i + \ddot{x}_i \geq 0 \\ \pi - \sin^{-1}\left(\frac{x_i - \sigma x_i + \ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) - \sin^{-1}\left(\frac{-\ddot{x}_i}{\sqrt{\dot{x}_i^2 + \ddot{x}_i^2}}\right) & \text{if } x_i - \sigma x_i + \ddot{x}_i \leq 0 \end{cases} \tag{42}$$

Case 6 ($\dot{x}_i < 0$ and $\ddot{x}_i > 0$):

Clearly (31) always holds for $\alpha \in [0, \frac{\pi}{2}]$. Therefore, we can take

$$\alpha_{x_i} = \frac{\pi}{2}. \tag{43}$$

Case 7 ($\dot{x}_i = 0$ and $\ddot{x}_i = 0$):

Clearly (31) always holds for $\alpha \in [0, \frac{\pi}{2}]$. Therefore, we can take

$$\alpha_{x_i} = \frac{\pi}{2}. \tag{44}$$

Similar analysis can be performed for (29b) and similar results can be obtained for α_{s_i}. For completeness, we list the formulae without repeating the proofs.

Case 1a ($\dot{s}_i = 0$, $\ddot{s}_i \neq 0$):

$$\alpha_{s_i} = \begin{cases} \frac{\pi}{2} & \text{if } s_i - \sigma s_i + \ddot{s}_i \geq 0 \\ \cos^{-1}\left(\frac{s_i - \sigma s_i + \ddot{s}_i}{\ddot{s}_i}\right) & \text{if } s_i - \sigma s_i + \ddot{s}_i \leq 0. \end{cases} \tag{45}$$

Case 2a ($\ddot{s}_i = 0$ and $\dot{s}_i \neq 0$):

$$\alpha_{s_i} = \begin{cases} \frac{\pi}{2} & \text{if } \dot{s}_i \leq s_i - \sigma s_i \\ \sin^{-1}\left(\frac{s_i - \sigma s_i}{\dot{s}_i}\right) & \text{if } \dot{s}_i \geq s_i - \sigma s_i \end{cases} \tag{46}$$

Case 3a ($\dot{s}_i > 0$ and $\ddot{s}_i > 0$):

$$\alpha_{s_i} = \begin{cases} \frac{\pi}{2} & \text{if } s_i - \sigma s_i + \ddot{s}_i \geq \sqrt{\dot{s}_i^2 + \ddot{s}_i^2} \\ \sin^{-1}\left(\frac{s_i - \sigma s_i + \ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) - \sin^{-1}\left(\frac{\ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) & \text{if } s_i - \sigma s_i + \ddot{s}_i < \sqrt{\dot{s}_i^2 + \ddot{s}_i^2} \end{cases} \tag{47}$$

Case 4a ($\dot{s}_i > 0$ and $\ddot{s}_i < 0$):

$$\alpha_{s_i} = \begin{cases} \frac{\pi}{2} & \text{if } s_i - \sigma s_i + \ddot{s}_i \geq \sqrt{\dot{s}_i^2 + \ddot{s}_i^2} \\ \sin^{-1}\left(\frac{s_i - \sigma s_i + \ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) + \sin^{-1}\left(\frac{-\ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) & \text{if } s_i - \sigma s_i + \ddot{s}_i \leq \sqrt{\dot{s}_i^2 + \ddot{s}_i^2} \end{cases} \tag{48}$$

Case 5a ($\dot{s}_i < 0$ and $\ddot{s}_i < 0$):

$$\alpha_{s_i} = \begin{cases} \frac{\pi}{2} & \text{if } s_i - \sigma s_i + \ddot{s}_i \geq 0 \\ \pi - \sin^{-1}\left(\frac{s_i - \sigma s_i + \ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) - \sin^{-1}\left(\frac{-\ddot{s}_i}{\sqrt{\dot{s}_i^2 + \ddot{s}_i^2}}\right) & \text{if } s_i - \sigma s_i + \ddot{s}_i \leq 0 \end{cases} \tag{49}$$

Case 6a ($\dot{s}_i < 0$ and $\ddot{s}_i > 0$):

Clearly (31) always holds for $\alpha \in [0, \frac{\pi}{2}]$. Therefore, we can take

$$\alpha_{s_i} = \frac{\pi}{2}. \tag{50}$$

Case 7a ($\dot{s}_i = 0$ and $\ddot{s}_i = 0$):

Clearly (31) always holds for $\alpha \in [0, \frac{\pi}{2}]$. Therefore, we can take

$$\alpha_{s_i} = \frac{\pi}{2}. \tag{51}$$

This discussion leads to Algorithm 2.2.

Algorithm 2.2 (search along an arc)

Data: $(\dot{x}, \dot{\lambda}, \dot{s})$, $(\ddot{x}, \ddot{\lambda}, \ddot{s})$, and $\sigma > 0$.

Calculate $\hat{\alpha} > 0$ by using the following process.

$\hat{\alpha} = \frac{\pi}{2}$

for $i = 1, \ldots, n$

$\alpha_x = \frac{\pi}{2}$, $\alpha_s = \frac{\pi}{2}$

if $\dot{x}_i = 0$ and $\ddot{x}_i \neq 0$, calculate α_{x_i} using (32). if $\alpha_{x_i} < \alpha_x$, set $\alpha_x = \alpha_{x_i}$.

if $\ddot{x}_i = 0$ and $\dot{x}_i \neq 0$, calculate α_{x_i} using (33). if $\alpha_{x_i} < \alpha_x$, set $\alpha_x = \alpha_{x_i}$.

if $\dot{x}_i > 0$ *and* $\ddot{x}_i > 0$, *calculate* α_{x_i} *using (36). if* $\alpha_{x_i} < \alpha_x$, *set* $\alpha_x = \alpha_{x_i}$

if $\dot{x}_i > 0$ *and* $\ddot{x}_i < 0$, *calculate* α_{x_i} *using (39). if* $\alpha_{x_i} < \alpha_x$, *set* $\alpha_x = \alpha_{x_i}$

if $\dot{x}_i < 0$ *and* $\ddot{x}_i < 0$, *calculate* α_{x_i} *using (42). if* $\alpha_{x_i} < \alpha_x$, *set* $\alpha_x = \alpha_{x_i}$

if $\alpha_x < \hat{\alpha}$, *set* $\hat{\alpha} = \alpha_x$

if $\dot{s}_i = 0$ *and* $\ddot{s}_i \neq 0$, *calculate* α_{s_i} *using (45). if* $\alpha_{s_i} < \alpha_s$, *set* $\alpha_s = \alpha_{s_i}$.

if $\ddot{s}_i = 0$ *and* $\dot{s}_i \neq 0$, *calculate* α_{s_i} *using (46). if* $\alpha_{s_i} < \alpha_s$, *set* $\alpha_s = \alpha_{s_i}$.

if $\dot{s}_i > 0$ *and* $\ddot{s}_i > 0$, *calculate* α_{s_i} *using (47). if* $\alpha_{s_i} < \alpha_s$, *set* $\alpha_s = \alpha_{s_i}$

if $\dot{s}_i > 0$ *and* $\ddot{s}_i < 0$, *calculate* α_{s_i} *using (48). if* $\alpha_{s_i} < \alpha_s$, *set* $\alpha_s = \alpha_{s_i}$

if $\dot{s}_i < 0$ *and* $\ddot{s}_i < 0$, *calculate* α_{s_i} *using (49). if* $\alpha_{s_i} < \alpha_s$, *set* $\alpha_s = \alpha_{s_i}$

if $\alpha_s < \hat{\alpha}$, *set* $\hat{\alpha} = \alpha_s$

end (for)
Update $x(\hat{\alpha})$, $\lambda(\hat{\alpha})$, *and* $s(\hat{\alpha})$ *using (22), (24), and (23).* ∎

A complete algorithm for linear programming is therefore as follows.

Algorithm 2.3 (linear programming using arc-search)
Data: A, b, c, $\epsilon > 0$, $\sigma > 0$, *initial point* (x^0, λ^0, s^0) *with* $(x^0, s^0) = e$ *and* $\mu(\alpha) = 1$.
for *iteration* $k = 1, 2, \ldots$

Step 1: Call Algorithm 2.1 using initial point given as (x^k, λ^k, s^k) *and* $\mu = \mu(\alpha)$ *to generate* $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ *and* $\bar{\mu}^k = \frac{(\bar{x}^k)^{\mathrm{T}} \bar{s}^k}{n}$.

Step 2: If $\mu(\alpha) < 4\epsilon$ *and (11) holds, stop. Otherwise continue.*

Step 3: Set $(x, \lambda, s) = (\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$. *Calculate* $(\dot{x}, \dot{\lambda}, \dot{s})$ *and* $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ *by solving (14) and (15).*

Step 4: Call Algorithm 2.2 to generate $(x(\alpha), \lambda(\alpha), s(\alpha))$, *and set*

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x(\alpha), \lambda(\alpha), s(\alpha)).$$

$$\mu^{k+1} = \mu(\alpha) = \bar{\mu}^k (1 - \sin(\alpha)). \tag{52}$$

end (for) ∎

Remark 2.2 *The proposed algorithm is similar to predictor-corrector method [10]. It is different from the predictor-corrector method in (a) algorithm 2.1 finds a point with controlled accuracy to the central path rather than a point loosely near the central path and (b) algorithm 2.1 searches along an ellipse rather than a straight line.*

One of our main tasks in the rest of the section is to show that Algorithm 2.2 always finds a new point $(x(\alpha), \lambda(\alpha), s(\alpha)) \in \mathcal{F}^o$, and the new point always reduces the duality gap. Moreover, (29a) and (29b) are always achievable with a constant $\bar{\alpha}$ independent of iterations, a conservative estimation of minimum improvement in duality reduction. In the rest of the paper, for the sake of simplicity, we frequently use (x, λ, s) for $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ which is the starting point on the ellipse $\mathcal{E}$ at iteration k. When iteration number is important, we use $(x^{k+1}, \lambda^{k+1}, s^{k+1})$ for $(x(\alpha), \lambda(\alpha), s(\alpha))$ and μ^{k+1} for $\mu(\alpha)$.

Lemma 2.2 *For every iteration* k *for which the stopping rule in Step 2 is not met, after moving towards the central line (Step 1), the following relation holds.*

$$\frac{1}{4}\mu I \leq \bar{X}^k \bar{S}^k \leq \frac{3}{4}\mu I. \tag{53}$$

Proof: From (11c) and the stopping criterion in Step 2, we have $|\bar{x}_i{}^k \bar{s}_i^k - \frac{\mu}{2}| \le \|\bar{X}\bar{s} - \frac{\mu}{2}e\|_\infty \le \|\bar{X}^k\bar{s}^k - \frac{\mu}{2}e\| \le \epsilon \le \frac{1}{4}\mu$. This gives

$$\frac{1}{4}\mu \le \bar{x}_i^k \bar{s}_i^k \le \frac{3}{4}\mu, \tag{54}$$

and the result follows. ∎

Lemma 2.3 *Let $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ and $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ meet (14) and (15), $(x(\alpha), \lambda(\alpha), s(\alpha))$ be calculated using (22), (23), and (24), then the following conditions hold.*

$$Ax(\alpha) = b, \quad A^{\mathrm{T}}\lambda(\alpha) + s(\alpha) = c.$$

Proof: Since $(x, \lambda, s) \in \mathcal{E}$, direct calculation verifies the result. ∎

Assume that $(x, \lambda, s) \in \mathcal{E}$, we show that there exist a constant $\bar{\alpha}$ independent of iteration index k such that for all $0 < \alpha \le \bar{\alpha}$, $(x(\alpha), s(\alpha)) > 0$. I.e., $(x(\alpha), \lambda(\alpha), s(\alpha)) \in \mathcal{F}^o$.

Lemma 2.4 *Let $\hat{A} \in \mathbf{R}^{n \times (n-m)}$ be a base of the null space of A. Let $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ and $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ meet (14) and (15). Then, $\frac{\dot{x}}{x}$ is the projection of $x^{-1} \circ s^{-1} \circ \mu e$ on to the subspace spanned by the columns of $(X^{-1}\hat{A})$ and $\frac{\dot{s}}{s}$ is the projection of $x^{-1} \circ s^{-1} \circ \mu e$ on to the subspace spanned by the columns of $(S^{-1}A^{\mathrm{T}})$ i.e.,*

$$\begin{aligned} \frac{\dot{x}}{x} &= X^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}Sx^{-1} \circ s^{-1} \circ \mu e \\ &= \left[I - S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1}AX\right] x^{-1} \circ s^{-1} \circ \mu e, \end{aligned} \tag{55}$$

and

$$\begin{aligned} \frac{\dot{s}}{s} &= S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1}AXx^{-1} \circ s^{-1} \circ \mu e \\ &= \left[I - X^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}S\right] x^{-1} \circ s^{-1} \circ \mu e. \end{aligned} \tag{56}$$

Moreover,

$$\dot{\lambda} = -(AA^{\mathrm{T}})^{-1}A\dot{s}. \tag{57}$$

Proof: Since $A\dot{x} = 0$, we have $AX\frac{\dot{x}}{x} = 0$, this means that there exists a vector v such that $X\frac{\dot{x}}{x} = \hat{A}v$, i.e.,

$$\frac{\dot{x}}{x} = X^{-1}\hat{A}v. \tag{58}$$

Since $(x, \lambda, s) \in \mathcal{E}$, from the last row of (14)

$$\frac{\dot{s}}{s} = x^{-1} \circ s^{-1} \circ \mu e - \frac{\dot{x}}{x} = x^{-1} \circ s^{-1} \circ \mu e - X^{-1}\hat{A}v. \tag{59}$$

Similarly, $A^{\mathrm{T}}\dot{\lambda} + \dot{s} = 0$ is equivalent to

$$S^{-1}A^{\mathrm{T}}\dot{\lambda} + \frac{\dot{s}}{s} = 0. \tag{60}$$

Substituting (59) into (60) gives $S^{-1}A^{\mathrm{T}}\dot{\lambda} + x^{-1} \circ s^{-1} \circ \mu e - X^{-1}\hat{A}v = 0$, or in matrix form

$$\left[X^{-1}\hat{A}, -S^{-1}A^{\mathrm{T}}\right] \begin{bmatrix} v \\ \dot{\lambda} \end{bmatrix} = x^{-1} \circ s^{-1} \circ \mu e. \tag{61}$$

Since A is full rank, it is easy to verify that

$$\left[\begin{array}{c} (\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}S \\ -(AXS^{-1}A^{\mathrm{T}})^{-1}AX \end{array}\right]\left[X^{-1}\hat{A}, -S^{-1}A^{\mathrm{T}}\right] = I.$$

Taking the inverse in (61) gives

$$\left[\begin{array}{c} v \\ \dot{\lambda} \end{array}\right] = \left[\begin{array}{c} (\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}S \\ -(AXS^{-1}A^{\mathrm{T}})^{-1}AX \end{array}\right] x^{-1}\circ s^{-1}\circ \mu e. \tag{62}$$

Substituting (62) into (58), (59), and (60) finishes the proof. ∎

Corollary 2.1 *Let $\hat{A} \in \mathbf{R}^{n\times(n-m)}$ be a base of the null space of A. Let $(x,\lambda,s)\in\mathcal{E}$, $(\dot{x},\dot{\lambda},\dot{s})$ and $(\ddot{x},\ddot{\lambda},\ddot{s})$ meet (14) and (15). Then, $\frac{\ddot{x}}{x}$ is the projection of $-2\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right)$ on to the subspace spanned by the columns of $(X^{-1}\hat{A})$ and $\frac{\ddot{s}}{s}$ is the projection of $-2\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right)$ on to the subspace spanned by the columns of $(S^{-1}A^{\mathrm{T}})$, i.e.,*

$$\begin{aligned} \frac{\ddot{x}}{x} &= -2X^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}S\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right) \\ &= -2\left[I - S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1}AX\right]\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right), \end{aligned} \tag{63}$$

and

$$\begin{aligned} \frac{\ddot{s}}{s} &= -2S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1}AX\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right) \\ &= -2\left[I - X^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}S\right]\left(\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right). \end{aligned} \tag{64}$$

Moreover,

$$\ddot{\lambda} = -(AA^{\mathrm{T}})^{-1}A\ddot{s}. \tag{65}$$

∎

We will repeatedly use a simple lemma.

Lemma 2.5 *Let $p>0$, $q>0$, and $r>0$. If $p+q\le r$, then $pq\le\frac{r^2}{4}$.* ∎

Lemma 2.6 *Let $(x,\lambda,s)\in\mathcal{E}$, $(\dot{x},\dot{\lambda},\dot{s})$ and $(\ddot{x},\ddot{\lambda},\ddot{s})$ meet (14) and (15). Then*

$$\left\|\frac{\dot{x}}{x}\right\|^2 + \left\|\frac{\dot{s}}{s}\right\|^2 \le 96n, \tag{66}$$

$$\left\|\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}\right\|^2 \le (48n)^2. \tag{67}$$

Proof: Repeatedly using Lemma 2.2, $x^{-1}\circ s^{-1}\circ\mu e\le 4e$, we have

$$\begin{aligned} \left\|\frac{\dot{x}}{x}\right\|^2 &= e^{\mathrm{T}}S^{\mathrm{T}}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}X^{-1}X^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}Se(4)^2 \\ &\le (\frac{1}{4}\mu)^{-1}e^{\mathrm{T}}S^{\mathrm{T}}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}SX^{-1}\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}Se(4)^2 \\ &= (\frac{1}{4}\mu)^{-1}e^{\mathrm{T}}S\hat{A}(\hat{A}^{\mathrm{T}}SX^{-1}\hat{A})^{-1}\hat{A}^{\mathrm{T}}Se(4)^2 \\ &\le \frac{3/4}{1/4}e^{\mathrm{T}}S\hat{A}(\hat{A}^{\mathrm{T}}S^2\hat{A})^{-1}\hat{A}^{\mathrm{T}}Se(4)^2 \\ &\le 48e^{\mathrm{T}}S\hat{A}(\hat{A}^{\mathrm{T}}S^2\hat{A})^{-1}\hat{A}^{\mathrm{T}}Se. \end{aligned} \tag{68}$$

Using QR decomposition

$$S\hat{A} = Q \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = [Q_1, Q_2] \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1,$$

where Q_1 and Q_2 are orthonormal matrices and they are orthogonal to each other, we have

$$\left\| \frac{\dot{x}}{x} \right\|^2 \le 48 e^{\mathrm{T}} Q_1 Q_1^{\mathrm{T}} e \le 48 \|e\|^2 = 48n. \tag{69}$$

Similarly,

$$\left\| \frac{\dot{s}}{s} \right\|^2 \le 48 e^{\mathrm{T}} X A^{\mathrm{T}} (A X^2 A^{\mathrm{T}})^{-1} A X e \le 48n. \tag{70}$$

Therefore,

$$\left\| \frac{\dot{x}}{x} \right\|^2 + \left\| \frac{\dot{s}}{s} \right\|^2 \le 96n,$$

that is (66). Combining (66) and Lemma 2.5 yields

$$\left\| \frac{\dot{x}}{x} \right\| \left\| \frac{\dot{s}}{s} \right\| \le 48n. \tag{71}$$

This leads to,

$$\left\| \frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right\|^2 \le \left\| \frac{\dot{x}}{x} \right\|^2 \left\| \frac{\dot{s}}{s} \right\|^2 \le (48n)^2.$$

∎

Lemma 2.7 *Let $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ and $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ meet (14) and (15). Then*

$$\left\| \frac{\ddot{x}}{x} \right\|^2 \le (3)\,(96n)^2, \qquad \left\| \frac{\ddot{s}}{s} \right\|^2 \le (3)\,(96n)^2. \tag{72}$$

Proof: Let $\phi = -2\left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s}\right)$. Similar to the proof of Lemma 2.6, from Corollary 2.1,

$$\left\| \frac{\ddot{x}}{x} \right\|^2 \le 3\|\phi\|^2 \le (3)\,(96n)^2, \qquad \left\| \frac{\ddot{s}}{s} \right\|^2 \le 3\|\phi\|^2 \le (3)\,(96n)^2.$$

This finishes the proof. ∎

From Lemma 2.6, $\left|\frac{\dot{x}_i}{x_i}\right| \le C_1 \sqrt{n}$, where $C_1 = (48)^{\frac{1}{2}}$. From Lemma 2.7, $\left|\frac{\ddot{x}_i}{x_i}\right| \le C_2 n$, where $C_2 = 96\,(3)^{\frac{1}{2}}$.

Lemma 2.8 *For $\alpha \in [0, \frac{\pi}{2}]$,*

$$\sin(\alpha) \ge 1 - \cos(\alpha).$$

Proof: Since $\alpha \in [0, \frac{\pi}{2}]$, $\sin^2(\alpha) + 2\sin(\alpha)\cos(\alpha) + \cos^2(\alpha) \ge 1$, hence for $\forall \alpha \in [0, \frac{\pi}{2}]$, $\sin(\alpha) + \cos(\alpha) \ge 1$, i.e., $\sin(\alpha) \ge 1 - \cos(\alpha)$. ∎

Lemma 2.9 *Let $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ and $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ meet (14) and (15). Then there exists a constant $\bar{\alpha} = \sin^{-1}(\frac{1-\sigma}{2(C_1+C_2)n})$ independent of iteration index k such that for $\alpha \in [0, \bar{\alpha}]$, $(x(\alpha), s(\alpha)) > 0$.*

Proof: From (22), (23), and Lemma 2.8, for $\forall \alpha \in [0, \frac{\pi}{2}]$

$$\begin{aligned} \frac{x_i(\alpha)}{x_i} &= 1 - \frac{\dot{x}_i}{x_i}\sin(\alpha) + \frac{\ddot{x}_i}{x_i}(1 - \cos(\alpha)) \\ &\ge 1 - \left|\frac{\dot{x}_i}{x_i}\right| \sin(\alpha) - \left|\frac{\ddot{x}_i}{x_i}\right| (1 - \cos(\alpha)) \\ &\ge 1 - (C_1\sqrt{n} + C_2 n)\sin(\alpha) \ge 1 - (C_1 + C_2) n \sin(\alpha) \ge \sigma. \end{aligned}$$

Select $\sin(\bar{\alpha}) = \frac{1-\sigma}{2(C_1+C_2)n}$, we have $x_i(\alpha) \ge \sigma x_i$. Similar argument can show $s_i(\alpha) \ge \sigma s_i$ for $\forall i$. ∎

Remark 2.3 *As it will be clear that if* $\left|\frac{\dot{x}_i}{x_i}\right|$, $\left|\frac{\ddot{x}_i}{x_i}\right|$, $\left|\frac{\dot{s}_i}{s_i}\right|$, $\left|\frac{\ddot{s}_i}{s_i}\right|$ *are smaller than some constant independent of* m *and* n, *arc-search would produce an improvement independent of* m *and* n. *In randomly generated tens of thousands problems, we observed that these variables are smaller than 2. But we cannot prove that this is true in general case.*

Lemma 2.10 *Let* $x(\alpha)$ *and* $s(\alpha)$ *are defined by (22) and (23) and* $\bar{\alpha}$ *is defined in Lemma 2.9, then,*

$$\bar{\mu}^{k+1} < \bar{\mu}^{k}\left(1-\frac{1}{4}\right) \tag{73}$$

Proof: Using (22), (23), Lemma 2.1 and $\bar{\alpha}$ defined in Lemma 2.9, it is easy to check that the duality gap improvement in arc-search is

$$\mu(\bar{\alpha}) = \bar{\mu}(1-\sin(\bar{\alpha})) = \bar{\mu}\left(1-\frac{1-\sigma}{2(C_1+C_2)n}\right). \tag{74}$$

From the stopping criterion (Step 2) and (52), for all iterations k before the algorithm stops, $\epsilon \leq \frac{\mu(\alpha)}{4} = \frac{\mu^{k+1}}{4}$. After the step of moving towards the central path, from (11c) and (52), we have

$$\bar{\mu}^{k+1} \leq \frac{\mu^{k+1}}{2} + \epsilon \leq \frac{\mu^{k+1}}{2} + \frac{\mu^{k+1}}{4} \leq \frac{3}{4}\bar{\mu}^{k}\left(1-\frac{1-\sigma}{2(C_1+C_2)n}\right) < \bar{\mu}^{k}\left(1-\frac{1}{4}\right).$$

∎

Remark 2.4 *The convergent analysis may lead to believe that Step 1 produces more improvement in reducing duality gap than Step 4. Actually, our extensive experience shows that the arc-search in Step 4 always produces more significant improvement than Step 1. For all the tested Netlib problems, this algorithm and the algorithm to be proposed in the next require fewer than 10 iterations to converge, which means that the arc-search is very efficient in reducing the duality gep. Another concern about the new algorithm could be that the centering process in Step 1 may be expensive. Our experience shows that once we find a point close to the central path, then after searching along the arc, it just takes a few Newton steps to come back to a point close enough to the central path which meets the requirement of Algorithm 2.1. A predictor-corrector algorithm proposed by Mehrotra [9] is similar to the method proposed in this paper in that they both use the information of the second derivatives. While the convergence property of Mehrotra's algorithm is poorly understood, we have shown that the algorithms proposed here are globally convergent.*

Now we can state one of the main Theorem of this paper.

Theorem 2.2 *Assume that* $\bar{\mu}^0$ *is bounded. Let* $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ *be the sequence generated by Algorithm 2.3, then* $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k) \in \mathcal{F}^o$, *and* $\bar{\mu}^k \to 0$.

Proof: The result directly follows from Lemmas 2.3, 2.9, and 2.10. ∎

2.4 Alternative approximation of the central path

Though the experiment tests demonstrate that the arc-search in Algorithm 2.3 is very efficient, we would like to find an algorithm that has better estimated arc-search efficiency. We replace (15) by

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\lambda} \\ \ddot{s} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\frac{2\dot{x}\circ\dot{s}}{\sqrt{n}} \end{bmatrix}. \tag{75}$$

With the proposed change, most results developed earlier still hold except Corollary 2.1, Lemmas 2.7, 2.9, 2.10, and Theorem 2.2. We give modified version for these Corollary, Lemmas, and Theorem, and then claim the modified algorithm has better estimated arc-search efficiency in the worst case. We will omit the proofs because they just repeat steps by using $\frac{\dot{x}\circ\dot{s}}{\sqrt{n}}$ and $\frac{1}{\sqrt{n}}\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}$ to replace $\dot{x}\circ\dot{s}$ and $\frac{\dot{x}}{x}\circ\frac{\dot{s}}{s}$. An updated complete algorithm for linear programming is therefore as follows.

Algorithm 2.4 (linear programming using arc-search)
Data: A, b, c, $\epsilon > 0$, $\sigma > 0$, *initial point* (x^0, λ^0, s^0) *with* $(x^0, s^0) = e$ *and* $\mu(\alpha) = 1$.
for *iteration* $k = 1, 2, \ldots$

Step 1: Call Algorithm 2.1 using initial point given as (x^k, λ^k, s^k) *and* $\mu = \mu(\alpha)$ *to generate* $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ *and* $\bar{\mu}^k = \frac{(\bar{x}^k)^{\mathrm{T}} \bar{s}^k}{n}$.

Step 2: If $\mu(\alpha) < 4\epsilon$ *and (11) holds, stop. Otherwise continue.*

Step 3: Set $(x, \lambda, s) = (\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$. *Calculate* $(\dot{x}, \dot{\lambda}, \dot{s})$ *and* $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ *by solving (14) and (75).*

Step 4: Call Algorithm 2.2 to generate $(x(\alpha), \lambda(\alpha), s(\alpha))$, *and set*

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x(\alpha), \lambda(\alpha), s(\alpha)).$$

$$\mu^{k+1} = \mu(\alpha) = \bar{\mu}^k (1 - \sin(\alpha)). \tag{76}$$

end (for) ∎

Corollary 2.2 *Let* $\hat{A} \in \mathbf{R}^{n \times (n-m)}$ *be a base of the null space of* A. *Let* $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ *and* $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ *meet (14) and (15). Then,* $\frac{\ddot{x}}{x}$ *is the projection of* $-\frac{2}{\sqrt{n}} \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right)$ *on to the subspace spanned by the columns of* $(X^{-1}\hat{A})$ *and* $\frac{\ddot{s}}{s}$ *is the projection of* $-\frac{2}{\sqrt{n}} \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right)$ *on to the subspace spanned by the columns of* $(S^{-1}A^{\mathrm{T}})$, *i.e.,*

$$\begin{aligned} \frac{\ddot{x}}{x} &= -\frac{2}{\sqrt{n}} X^{-1}\hat{A}(\hat{A}^{\mathrm{T}} S X^{-1} \hat{A})^{-1} \hat{A}^{\mathrm{T}} S \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right) \\ &= -\frac{2}{\sqrt{n}} \left[I - S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1} AX \right] \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right), \end{aligned} \tag{77}$$

and

$$\begin{aligned} \frac{\ddot{s}}{s} &= -\frac{2}{\sqrt{n}} S^{-1}A^{\mathrm{T}}(AXS^{-1}A^{\mathrm{T}})^{-1} AX \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right) \\ &= -\frac{2}{\sqrt{n}} \left[I - X^{-1}\hat{A}(\hat{A}^{\mathrm{T}} S X^{-1} \hat{A})^{-1} \hat{A}^{\mathrm{T}} S \right] \left(\frac{\dot{x}}{x} \circ \frac{\dot{s}}{s} \right). \end{aligned} \tag{78}$$

Moreover,

$$\ddot{\lambda} = -(AA^{\mathrm{T}})^{-1} A\ddot{s}. \tag{79}$$

∎

Lemma 2.7 is updated for Algorithm 2.4.

Lemma 2.11 *Let* $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ *and* $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ *be calculated by using (14) and (75). Then*

$$\left\| \frac{\ddot{x}}{x} \right\|^2 \le (3)\,(96)^2 n, \qquad \left\| \frac{\ddot{s}}{s} \right\|^2 \le (3)\,(96)^2 n. \tag{80}$$

∎

Lemma 2.11 is used to prove the following

Lemma 2.12 *Let* $(x, \lambda, s) \in \mathcal{E}$, $(\dot{x}, \dot{\lambda}, \dot{s})$ *and* $(\ddot{x}, \ddot{\lambda}, \ddot{s})$ *be calculated by using (14) and (75). Then there exists a constant* $\bar{\alpha} = \sin^{-1}\left(\frac{1-\sigma}{2(C_1+C_2)\sqrt{n}}\right)$ *such that for* $\alpha \in [0, \bar{\alpha}]$, $(x(\alpha), s(\alpha)) > 0$. ∎

From Lemma 2.12, it is easy to see that the duality gap improvement in the new algorithm is

$$\mu(\bar{\alpha}) = \bar{\mu}(1 - \sin(\bar{\alpha})) = \bar{\mu} \left(1 - \frac{1-\sigma}{2(C_1 + C_2)\sqrt{n}} \right), \tag{81}$$

which is better than (74).

Lemma 2.13 *Let $x(\alpha)$ and $s(\alpha)$ be defined by (22) and (23) and $\bar{\alpha}$ be defined in Lemma 2.12, then,*

$$\bar{\mu}^{k+1} < \bar{\mu}^{k} \left(1 - \frac{1}{4}\right). \tag{82}$$

∎

This leads to another main result which directly follows from Lemmas 2.3, 2.12, and 2.13..

Theorem 2.3 *Assume that $\bar{\mu}^0$ is bounded. Let $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ be the sequence generated by Algorithm 2.4, then $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k) \in \mathcal{F}^o$, and $\bar{\mu}^k \to 0$.* ∎

3 Convergence Analysis

Let (x^*, λ^*, s^*) be any solution of (5). Let index sets $\mathcal{B}$, $\mathcal{N}$ be defined as

$$\mathcal{B} = \{j \in \{1, \ldots, n\} \mid x_j^* \neq 0\}. \tag{83}$$

$$\mathcal{N} = \{j \in \{1, \ldots, n\} \mid s_j^* \neq 0\}. \tag{84}$$

From Goldman-Tucker theorem [4], it can be shown [14] that there always exist a solution (x^*, λ^*, s^*) of (5), where x^* is a solution of the primary linear programming and (λ^*, s^*) is a solution of the dual linear programming, such that $\mathcal{B} \cap \mathcal{N} = \emptyset$ and $\mathcal{B} \cup \mathcal{N} = \{1, \ldots, n\}$. I.e., $x^* \circ s^* = 0$, and $x^* + s^* > 0$. An optimal solution with this property is called strictly complementary. The following Lemma, independent of any algorithm, is given in [14].

Lemma 3.1 *Let $\mu^0 > 0$, and $\gamma \in (0, 1)$, Then for all points (x, λ, s) with $(x, \lambda, s) \in \mathcal{F}^o$, $x_i s_i > \gamma\mu$ (where $\mu = \frac{x^{\mathrm{T}} s}{n}$), and $\mu < \mu^0$, there are constants M, C_1, and C_2 such that*

$$\|(x, s)\| \le M, \tag{85}$$

$$0 < x_i \le \mu / C_1 \quad (i \in \mathcal{N}), \qquad 0 < s_i \le \mu / C_1 \quad (i \in \mathcal{B}). \tag{86}$$

$$s_i \ge C_2 \gamma \quad (i \in \mathcal{N}), \qquad x_i \ge C_2 \gamma \quad (i \in \mathcal{B}). \tag{87}$$

∎

Lemma 3.2 *Let $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ be generated by Algorithms 2.3 and 2.4. Then, $(\bar{x}^k, \bar{s}^k)$ has at least one limit point. Moreover, every limit point is a strictly complementary primary-dual solution of the linear programming, i.e.,*

$$s_i^* \ge C_2 \gamma \quad (i \in \mathcal{N}), \qquad x_i^* \ge C_2 \gamma \quad (i \in \mathcal{B}). \tag{88}$$

Proof: First, we show that after moving towards the central line (Step 1 in both algorithms), $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k) \in \mathcal{E}$, and $\bar{x}_i^k \bar{s}_i^k \ge \gamma \bar{\mu}^k$. By taking summation in (54), we have $\frac{1}{4}\mu(\alpha) \le \bar{\mu}^k \le \frac{3}{4}\mu(\alpha)$. The claim follows from $\bar{x}_i^k \bar{s}_i^k \ge \frac{1}{4}\mu(\alpha) \ge \frac{1}{4}\frac{4}{3}\bar{\mu}^k = \frac{1}{3}\bar{\mu}^k$. Hence, $(\bar{x}^k, \bar{s}^k)$ meet the condition of Lemma 3.1. From Lemma 3.1, $(\bar{x}^k, \bar{s}^k)$ is bounded, therefore there is at least one limit point (x^*, s^*). Without loss of generality, assume $(\bar{x}^k, \bar{s}^k) \to (x^*, s^*)$. Since,

$$\bar{s}_i^k \ge C_2 \gamma \quad (i \in \mathcal{N}), \qquad \bar{x}_i^k \ge C_2 \gamma \quad (i \in \mathcal{B}).$$

(88) follows from the fact that $C_2 \gamma$ is a constant. ∎

Therefore, in view of Theorems 2.2, 2.3, and Lemma 3.2, we have

Theorem 3.1 *The sequence $(\bar{x}^k, \bar{\lambda}^k, \bar{s}^k)$ generated by Algorithms 2.3 or by Algorithms 2.4 globally converges to a set of limit points (x^*, λ^*, s^*). For every limit point (x^*, λ^*, s^*), x^* is the optimal solution of the primal problem, (λ^*, s^*) is the optimal solution of the dual problem, and (x^*, s^*) is strictly complementary. Moreover, the global convergence rate of Algorithms 2.3 and Algorithms 2.4 is at least $\left(1 - \frac{1}{4}\right)$.*

∎

4 Numerical Tests

In this section, we first use a simple problem as an example to show the central path, the ellipse approximation, and the arc-search in every iteration by plots. From these plots, we can intuitively see that searching along an ellipse is more attractive than searching along a straight line. We then provide the numerical test results of larger scale test problems.

4.1 A simple illustrative example

Let us consider

Example 4.1

$$\min x_1, \quad s.t. \;\; x_1 + x_2 = 5, \quad x_1 \geq 0, \quad x_2 \geq 0.$$

The central path (x, s) meets the following conditions.

$$x_1 + x_2 = 5,$$
$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \lambda + \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$
$$x_1 s_1 = \mu, \quad x_2 s_2 = \mu.$$

The optimizer is given by $x_1 = 0$, $x_2 = 5$, $\lambda = 0$, $s_1 = 1$, and $s_2 = 0$. The central path for this problem is given analytically as

$$\lambda = \frac{5 - 2\mu - \sqrt{(5-2\mu)^2 + 20\mu}}{10},$$
$$s_1 = 1 - \lambda, \quad s_2 = -\lambda, \quad x_1 s_1 = \mu, \quad x_2 s_2 = \mu.$$

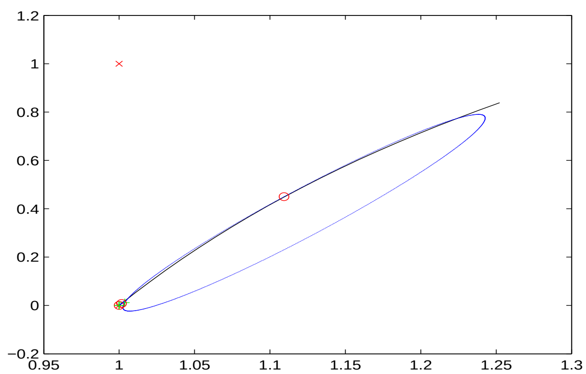

Figure 1: Arc-search for the simple example.

The central path is an arc in 5-dimensional space $(\lambda, x_1, s_1, x_2, s_2)$. If we project the central path to 2-dimensional space spanned by (s_1, x_1), it is an arc in 2-dimensional space. Similarly, we can project the ellipse in 5-dimensional space to the same 2-dimensional space spanned by (s_1, x_1). Figure 1 shows all iterations in the two dimmensional subspace spanned by (s_1, x_1). The first iteration moves the iterate very close to the solution, and the remaining iterations have to be rescaled to show the details. In Figure 1, $(\dot{x}, \dot{s}, \dot{\lambda})$ is calculated by using (14), $(\ddot{x}, \ddot{s}, \ddot{\lambda})$ is calculated by using (75), the projected central path is the continuous line in black, the projected ellipse approximations are the dotted lines in blue in every iteration (they may look like continuous line some times because many dots are used), the initial point (s_1^0, x_1^0) is marked by 'x' in red, after moving 'x' towards the central line, $(\bar{s}_1^k, \bar{x}_1^k)$ is marked by 'o' in red, after arc-search, the point (s_1^k, x_1^k) on the ellipse is marked by '+' in green, the optimal solution (s^*, x^*) is marked by '*' in red. More detailed information in the second iteration, the third iteration, and the

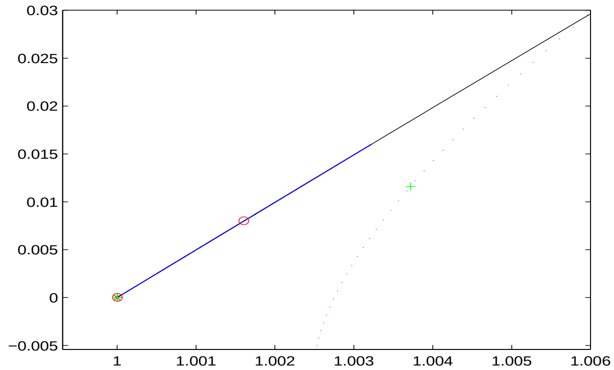

Figure 2: Arc-search of the second iteration for the simple example.

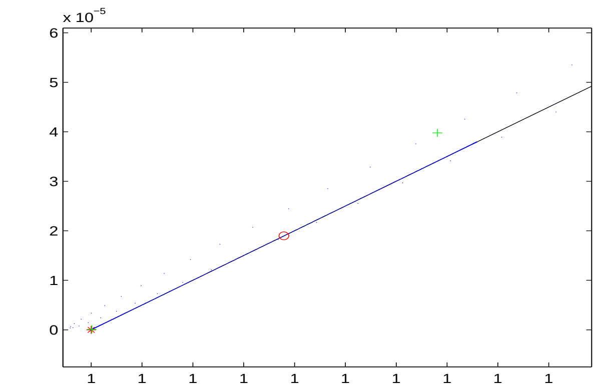

Figure 3: Arc-search of the third iteration for the simple example.

final result are presented in Figures 2, 3, and 4 which are amplified parts of Figure 1. It is clear that after the first iteration, the central path is close to a straight line, and the ellipses approximate the central path very well no matter if the central path is close to a straight line or not. We expect that the newly developed algorithms are more efficient in the early stage than MPC and other interior-point algorithms when the central path is not close to the straight line as shown in Figure 1.

4.2 Large scale test examples

The algorithms developed in this paper are implemented in Matlab functions. Function `arc1` implements Algorithm 2.3 and `arc2` implements Algorithm 2.4. Numerical tests have been performed for linear programming problems in Netlib LP library. For Netlib LP problems, [2] has classified these problems into two categories: problems with strict interior-point and problems without strict interior-point. Though the newly developed Matlab codes and other existing codes can solve problems without strict interior-point, we are most interested in the problems with strict interior-point that is assumed by all interior-point methods. Among these problems, we choose only problems presented in standard form and the A are full rank matrices. The selected problems are solved by Matlab functions `arc1`, `arc2`, along with function `linprog` in Matlab optimization toolBox. The iterations used to solve these problems are compared and the iteration numbers are listed in table 1. Only two Netlib problems that are classified as problems with strict interior-point and are presented in standard form are not included in the table because the PC computer used in the test does not have enough memory to handle problems of this size.

In our implementation, the parameters in the proposed algorithms are chosen as follows. To move to

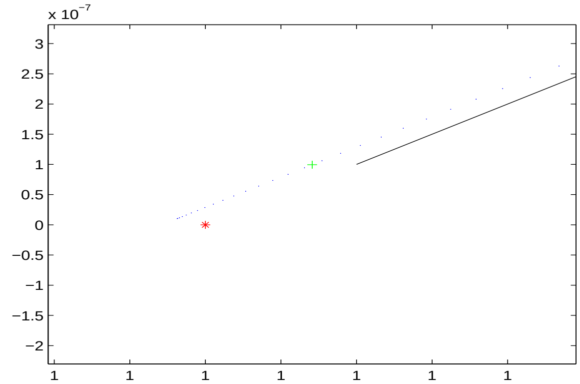

Figure 4: Arc-search of the final result for the simple example.

Table 1: Iteration counts for test problems in Netlib and Matlab

	iterations used by different algorithms			
Problem	arc1	arc2	linprog	source
AFIRO	4	4	7	netlib
blend	5	6	8	netlib
SCAGR25	4	4	16	netlib
SCAGR7	4	5	12	netlib
SCSD1	6	7	10	netlib
SCSD6	7	9	12	netlib
SCSD8	6	7	11	netlib
SCTAP1	6	8	17	netlib
SCTAP2	6	8	18	netlib
SCTAP3	6	8	18	netlib
SHARE1B	5	7	22	netlib

the center of the central path, $\epsilon = 0.0001$ is used in (11a) and (11b). (11c) is replaced by $(\max(X^kS^ke - \mu^ke) - \min(X^kS^ke - \mu^ke))/\max(X^kS^ke - \mu^ke) \leq 0.01$. The stopping ceriterion used in outloop in Algorithm 2.3 and 2.4 is the same as `linprog` [16]

$$\frac{\|r_B\|}{\max\{1, \|b\|\}} + \frac{\|r_C\|}{\max\{1, \|c\|\}} + \frac{\|\mu\|}{\max\{1, \|c^{\rm T}x\|, \|b^{\rm T}\lambda\|\}} < 10^{-8}.$$

$\sigma = 0.0001$ is used in arc-search. For all problems, the initial point is set to $x = s = e$. The initial point used in `linprog` is essentially the same as used in [8] with some minor modificatins (see [16]).

It is clear that the implementations of the newly proposed algorithms use fewer iterations than `linprog` for all tested Netlib problems. For all the tested problems, once a point close to the central path is found, and $\mathcal{E}$ is obtained from this point, after searching along the ellipse, moving the iterate from the ellipse back to a point close to the central path is very efficient. It takes a few Newton steps in all tested problems. This number can be reduced if we relax the ϵ in (11a) and (11b). Relaxing ϵ in (11a) and (11b) does not affect much the error in the linear constraints at convergence (it affects the constraint error in the early stage). The majority computational burden is to move the initial point towards the central path. Little serious efforts were made to this important "Phase 1" step with very few exceptions, for example, [2]. More investigation in this direction should be made.

5 Conclusions

This paper proposes two interior-point path-following algorithms that search the optimizers along the ellipses that approximate central path. Both algorithms are proved to be globally convergent. Numerical test results show that both algorithms need fewer iterations to converge than Matlab code `linprog` in all tested Netlib problems.

6 Acknowledgement

The author wishes to thank Professor Andre L. Tits and Professor T. Terlaky for their valuable comments on the earlier versions of this paper. The author is grateful to Professor Pierre-Antoine Absil, Dr. Simon Schurr, and Mr. Luke Winternitz for providing information in connection with the test problems. The author would like to thank Professor Timothy A. Davis for providing his Matlab code `spqr` and his explanation on how to use `spqr` to solve sparse QR decomposition problem.

References

[1] C. Cartis, *Some disadvantages of a mehrotra-type primal-dual corrector interior point algorithm for linear programming*, Applied Numerical Mathematics, 59 (2009), pp. 1110–1119.

[2] C. Cartis and N. I. M. Gould, *Finding a point in the relative interior of a polyhedren*, Technical Report NA-07/01, Computing Laboratory, Oxford University, Oxford, UK, 2007.

[3] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice-Hall, New Jersey, 1976.

[4] A. Goldman and A. Tucker, *Theory of linear programming*, in Linear Equalities and Related Systems, H. Kuhn and Tucker, eds., Princeton University Press, Princeton, N.J, 1956, pp. 53–97.

[5] B. Jansen, C. Roos, T. Terlaky, and Y. Ye, *Improved complexity using higher-order correctors for primal-dual dikin affine scaling*, Mathematical Programming, 76 (1996), pp. 117–130.

[6] M. Kojima, S. Mizuno, and A. Yoshise, *A polyomial-time algorithm for a class of linear complementarity problems*, Mathematical Programming, 44 (1989), pp. 1–26.

[7] ——, *A primal-dual interior point algorithm for linear programming*, in Progress in Mathematical Programming: Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 29–47.

[8] I. Lustig, R. Marsten, and D. Shanno, *On implementing mehrotra's predictor-corrector interior point method for linear programming*, SIAM Journal on Optimization, 2 (1992), pp. 432–449.

[9] S. Mehrotra, *On the implementation of a primal-dual interior point method*, SIAM Journal on Optimization, 2 (1992), pp. 575–601.

[10] S. Mizuno, M. Todd, and Y. Ye, *On adaptive step primal-dual interior-point algorithms for linear programming*, Mathematics of Operartions Research, 18 (1993), pp. 964–981.

[11] R. Monteiro, I. Adler, and M. Resende, *A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension*, Mathematics of Operations Research, 15 (1990), pp. 191–214.

[12] Y. Nesterov, *Long-step strategies in interior-point primal-dual methods*, Mathematical Programming, 76 (1996), pp. 47–94.

[13] M. Salahi, J. Peng, and T. Terlaky, *On mehrotra-type predictor-corrector algorithms*, SIAM Journal on Optimization, 18 (2007), pp. 1377–1397.

[14] S. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.

[15] Y. Ye, *Interior Point Algorithms: Theory and Analysis*, John Wiley & Son, Inc, New York, 1997.

[16] Y. Zhang, *Solving large-scale linear programs by interior-point methods under the matlab environment*, Technical Report TR96-01, Department of Mathematics and Statistics, University of Maryland, Baltimore County, Marland, 1996.