Mathematical Programming 40 (1988) 59-93
North-Holland

A POLYNOMIAL-TIME ALGORITHM, BASED ON NEWTON'S METHOD, FOR LINEAR PROGRAMMING

James RENEGAR

School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853, USA

Received 4 September 1986
Revised manuscript received 15 June 1987

A new interior method for linear programming is presented and a polynomial time bound for it is proven. The proof is substantially different from those given for the ellipsoid algorithm and for Karmarkar's algorithm. Also, the algorithm is conceptually simpler than either of those algorithms.

Key words: Linear programming, interior method, computational complexity, Newton's method.

1. Introduction

The main motivation for this work was the idea that there should exist an easily understood polynomial time algorithm for linear programming, where both the algorithm and the proof of the polynomial time bound rely primarily on common ideas in the non-linear optimization literature (e.g. convergence of Newton's method). Although both the ellipsoid algorithm [7, 8] and Karmarkar's algorithm [6] are polynomial time algorithms, the main ideas behind those algorithms and the proofs of their polynomial time bounds are certainly novel as regards the optimization literature.

The algorithm presented is based on approximately following a sequence of "centers" through the interior of the feasible region. It is reminiscent of the "method of centers" of Huard [5].

The algorithm solves linear programming problems in the format

$$\begin{aligned} \max \quad & c \cdot x \\ \text{s.t.} \quad & Ax \geq b, \end{aligned} \tag{1.1}$$

where A is an $m \times n$ matrix. Assuming the coordinates of A, b and c are integers, and the sum of the bits needed to represent all entries in A, b and c is L, the algorithm solves (1.1) (i.e. determines an optimal solution, or unboundedness, or infeasibility) in $O(\sqrt{m+n}\,L)$ iterations. The work in each iteration of the algorithm is dominated by solving a system of linear equations. This requires $O((m+n)n^2)$ arithmetic operations. The equations are actually only solved approximately, this

This research was supported by an NSF Mathematical Sciences Postdoctoral Research Fellowship and by NSF Grant 8120790. The research was performed at the Mathematical Sciences Research Institute in Berkeley, California.

requiring $O((m+n)n^2L(\log L)(\log\log L))$ bit operations. The total number of arithmetic operations involved is $O((m+n)^{1.5}n^2L)$ and the total number of bit operations is $O((m+n)^{1.5}n^2L^2(\log L)(\log\log L))$.

For a linear programming problem (LPP) in the form (1.1), Karmarkar's modified algorithm first requires that LPP to be recast as an LPP in $m+n$ variables and m equations. Then Karmarkar's bound on the number of iterations required to solve the LPP is $O((m+n)L)$. On the average (where the average is over the $O((m+n)L)$ iterations), each iteration requires $O((m+n)^{2.5})$ arithmetic operations, $O((m+n)^{2.5}L(\log L)(\log\log L))$ bit operations. Thus, for Karmarkar's modified algorithm, the total number of arithmetic operations required is $O((m+n)^{3.5}L)$, and the total number of bit operations is $O((m+n)^{3.5}L^2(\log L)(\log\log L))$.

Comparing, the proven bounds for the algorithm presented herein are identical to those for Karmarkar's algorithm if m and n are of the same magnitude, and are better if $m \gg n$. The most important theoretical result in this paper is the $O(\sqrt{m+n}\, L)$ bound on the number of iterations.

The paper is organized as follows. In the next section we present the algorithm, assuming the problem to be solved fits an appropriate framework. We also state what we call the "main theorem". The main theorem does not require the entries in A, b and c to be rational, but it does assume all arithmetic operations can be performed exactly. It is dubbed the "main theorem" because it represents the ideas at the heart of the analysis.

In section three, we describe the ideas behind proving the main theorem. One of the key steps in the proof is a well-chosen change of coordinates. The new coordinates are very reminiscent of the context of Karmarkar's algorithm, a difference being that only our proofs "live" in those coordinates, not our algorithm. (Also, we do not rely on a potential function, but instead measure progress directly in terms of the objective function.)

Sections 4, 5 and 6 are devoted to proving propositions and theorems stated in Section 3.

In Section 7, we show how to recast any LPP of the form (1.1) into a format suitable for the algorithm.

Finally, in Section 8, we give a complexity analysis of the algorithm assuming the entries in A, b and c are integers. We are careful in this analysis to account for the effects of rounding. The analysis is essentially another proof of the main theorem, but the technicalities involved with accounting for the rounding obscure the central ideas that the main theorem and its proof highlight.

In the original version of this paper I wrote that I did not see how Karmarkar's algorithm could be carried out with $O(L)$ bits of accuracy (assuming the number of bits required to represent the original problem is L) as Karmarkar claimed in his paper. Subsequently, Karmarkar convinced me that this could be done if one does not rely on rank one updates, as the algorithm in the present paper does not. The argument, embedded in our complexity analysis, relies on the fact that the linear equations that need to be solved need only be solved approximately, and this can be done efficiently using Cholesky factorization and the fact that the condition number of the corresponding matrices are bounded by $2^{O(L)}$. Subsequently, Pravin Vaidya convinced me that $O(L)$ bits of accuracy also suffice if rank one updating is relied upon, which Karmarkar's modified algorithm does.

I also wrote in the original version of this paper that it was an important and open theoretical question whether or not the algorithm in this paper could be modified in a manner reminiscent of Karmarkar's modified algorithm to reduce the complexity bounds. Pravin Vaidya [15] has apparently answered this question in the affirmative. His stated bounds are $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations and $O(((m+n)n^2+(m+n)^{1.5}n)L^2(\log L)(\log\log L))$ bit operations. His analysis relies on a potential function.

Our algorithm is similar to one presented independently by Sonnevend [13, 14], who gave no complexity analysis.

There are several people I would like to thank. I would like to thank Lenore Blum [3, 4], who aroused my interest in interior methods for linear programming in talks that she gave. I would like to thank Steve Smale [11, 12] for reasons too numerous to mention. I would like to thank Jeff Lagarias [2, 9], who gave a talk that served as a catalyst for parts of the present work. I would like to thank Nimrod Megiddo and Mike Shub [10] for several interesting conversations regarding interior methods for linear programming. And I would like to thank Jim Curry, whose conversation has helped make for a pleasant year at MSRI. Finally, I would like to thank a referee for some very useful comments.

2. The algorithm

In this section we introduce the algorithm and state the "main theorem" regarding it.

Assume that we wish to solve

$$\begin{aligned} \max \quad & c\cdot x \\ \text{s.t.} \quad & Ax \geq b, \end{aligned} \tag{2.1}$$

where $z \in \mathbb{R}^n$ and A is an $m \times n$ matrix. Non-negativity constraints are not distinguished from other inequalities. We assume $c \neq 0$ and we assume none of the rows of A are the zero vector.

Let α_i denote the ith row of A. Let l be a positive integer and let $\bar{A}$ denote the $(m+l)\times n$ matrix

$$\bar{A} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \\ c \\ \vdots \\ c \end{bmatrix} \begin{matrix} \\ \\ \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} l \text{ times,} \end{matrix}$$

c being considered as a row vector. Let $k^{(0)} \in \mathbb{R}$ and let $b^{(0)} \in \mathbb{R}^{m+l}$ be the vector

$$(b^{(0)})^{\mathrm{T}} = (b_1, \ldots, b_m, k^{(0)}, \ldots, k^{(0)}),$$

where $b^{\mathrm{T}} = (b_1, \ldots, b_m)$ is the right hand side vector of (2.1). Finally, let $(\bar{A}, b^{(0)})$ denote the system of inequalities $\bar{A}x \geqslant b^{(0)}$.

The algorithm is based on approximating the "centers" of a sequence of systems $\{(\bar{A}, b^{(i)})\}$ assuming the center of the initial system, $(\bar{A}, b^{(0)})$, is given. Now we develop the notion of the center of a system of inequalities. (Apparently, Sonnevend [13, 14] and Bayer and Lagarias [2] were the first to develop the following notion of center, as I learned after I had arrived at the same notion.)

Let (A', b') be a system of linear inequalities where A' is an $m' \times n'$ matrix and $b' \in \mathbb{R}^{m'}$. We use (A', b') to denote a general system of inequalities throughout this paper, as opposed to, for example, $(\bar{A}, b^{(0)})$ which is specialized. Assume $\{x;\, A'x \geqslant b'\}$ has non-empty interior, $\mathrm{Int}(A', b')$, and let $f' : \mathrm{Int}(A', b') \to \mathbb{R}$ be defined by

$$f'(x) = \sum_{i=1}^{m'} \ln(\alpha_i' \cdot x - b_i') = \ln \prod_{i=1}^{m'} (\alpha_i' \cdot x - b_i')$$

where ln is the natural logarithm, α_i' is the ith row of A' and "$\cdot$" is the dot product. (Primes, as in f', will always imply a relation to the system (A', b') and will never be used to denote a derivative in this paper.) Note that $f'(x)$ goes to $-\infty$ as x goes to the boundary of $\mathrm{Int}(A', b')$.

We say that ξ' is a *center* of (A', b') if $\xi' \in \mathrm{Int}(A', b')$ and

$$f'(\xi') \geqslant f'(x) \quad \text{for all } x \in \mathrm{Int}(A', b').$$

(Thus, the center of the system $(\bar{A}, b^{(0)})$ is the point in $\mathrm{Int}(\bar{A}, b^{(0)})$ maximizing

$$f^{(0)}(x) = l \cdot \ln(c \cdot x - k^{(0)}) + \sum_{i=1}^{m} \ln(\alpha_i \cdot x - b_i),$$

l playing the role of a "weight".)

Proposition 2.1. *Assume* $\mathrm{Int}(A', b')$ *is non-empty and bounded. Then* f' *is strictly concave and the system* (A', b') *has a unique center* ξ'.

Proof. Using the fact that $f'(x)$ is the composition of the maps $X \mapsto \sum_{i=1}^{m'} \ln(X_i)$ and $x \mapsto (\alpha_1' \cdot x - b_1', \ldots, \alpha_{m'}' \cdot x - b_{m'}')$, it is easily shown that, for $x \in \mathrm{Int}(A', b')$,

$$\nabla_x^2 f' = -(A')^{\mathrm{T}} D_x^{-2} A', \tag{2.2}$$

where D_x is the diagonal matrix with (i, i) entry $\alpha_i' \cdot x - b_i$. However, since $\mathrm{Int}(A', b')$ is assumed non-empty and bounded, it is easily shown that A' is of full-rank, and hence, from (2.2), $\nabla_x^2 f'$ is strictly negative definite. Thus, f' is strictly concave. Since $f'(x)$ goes to $-\infty$ as x goes to the boundary of the bounded set $\mathrm{Int}(A', b')$, it follows that (A', b') has a unique center. □

Throughout the rest of this section we will assume $\mathrm{Int}(\bar{A}, b^{(0)})$ is non-empty and bounded, so that by the proposition, $(\bar{A}, b^{(0)})$ has a unique center $\xi^{(0)}$. We will also assume that we know $\xi^{(0)}$, that is, we know its coordinates. In Section 7 we will show how any linear programming problem can be put in a framework satisfying these assumptions.

It is easily shown that our assumption that $\mathrm{Int}(\bar{A}, b^{(0)})$ is bounded and non-empty implies that (2.1) has an optimal solution, but perhaps infinitely many of them. It is also easily shown that all optimal solutions for (2.1) are contained in the boundary of $\mathrm{Int}(\bar{A}, b^{(0)})$. We want to somehow move from the known point $\xi^{(0)}$ toward the optimal solutions.

To move from $\xi^{(0)}$ toward the optimal solutions, we create a new system $(\bar{A}, b^{(1)})$ whose center $\xi^{(1)}$ is known to lie closer to the optimal solutions than $\xi^{(0)}$, and then we attempt to obtain a "good" approximation to $\xi^{(1)}$. The new system $(\bar{A}, b^{(1)})$ is identical to $(\bar{A}, b^{(0)})$ except that the value $k^{(0)}$ defining $b^{(0)}$ is replaced by a larger value $k^{(1)}$. To guarantee that $\mathrm{Int}(\bar{A}, b^{(1)})$ is non-empty, it suffices to choose $k^{(1)} < c \cdot \xi^{(0)}$. Then $\xi^{(0)} \in \mathrm{Int}(\bar{A}, b^{(1)})$. Hence, $k^{(1)}$ is chosen from the range $k^{(0)} < k^{(1)} < c \cdot \xi^{(0)}$.

The new system $(\bar{A}, b^{(1)})$ is bounded and non-empty. Also, all optimal solutions to (2.1) are contained in the boundary of $\mathrm{Int}(\bar{A}, b^{(1)})$. Let $f^{(1)}: \mathrm{Int}(\bar{A}, b^{(1)}) \to \mathbb{R}$ be the function

$$f^{(1)}(x) = l \cdot \ln(c \cdot x - k^{(1)}) + \sum_{i=1}^{m} \ln(\alpha_i \cdot x - b_i).$$

Then $\xi^{(1)}$, the center of $(\bar{A}, b^{(1)})$, is, by definition, the point in $\mathrm{Int}(\bar{A}, b^{(1)})$ that maximizes $f^{(1)}$.

We want to obtain a "good" approximation to $\xi^{(1)}$. Since $\xi^{(1)}$ is defined to be the point maximizing the strictly concave function $f^{(1)}$, it is natural to use Newton's method. Since $\xi^{(0)} \in \mathrm{Int}(\bar{A}, b^{(1)})$, we can initiate Newton's method at $\xi^{(0)}$. Thus, letting #Newton be a positive integer, define recursively

$$\begin{aligned} &x[0] = \xi^{(0)}, \\ &x[i] = x[i-1] + n^{(1)}_{x[i-1]}, \quad i = 1, \ldots, \#\text{Newton}, \end{aligned} \tag{2.3}$$

where $n^{(1)}_x$, the Newton step at x, is defined to be the vector satisfying

$$(\nabla^2_x f^{(1)}) n^{(1)}_x = -(\nabla_x f^{(1)})^{\mathrm{T}},$$

that is,

$$\left(A^{\mathrm{T}} D_x^{-2} A + \frac{l}{(c \cdot x - k^{(1)})^2} c^{\mathrm{T}} c\right) n^{(1)}_x = A^{\mathrm{T}} D_x^{-1} e + \frac{l}{(c \cdot x - k^{(1)})} c^{\mathrm{T}}.$$

$e \in \mathbb{R}^m$ being the vector of all ones, D_x being the diagonal matrix with (i, i) entry $\alpha_i \cdot x - b_i$, and c being considered as a row vector.

As our approximation to $\xi^{(1)}$ we take the point

$$x^{(1)} \doteq x[\#\text{Newton}].$$

(Newton's method can go astray in attempting to find the center of a system of linear inequalities (A', b'). In particular, even though $x \in \text{Int}(A', b')$, it is quite possible that Newton's method will assign to x a point not in $\text{Int}(A', b')$ so that further iterates of Newton's method are not even defined. These problems will be discussed more fully in section three. For now, we simply assume that the sequence (2.3), and hence $x^{(1)}$, is contained in $\text{Int}(\bar{A}, b^{(1)})$.)

Having obtained $x^{(1)}$ we begin the process again. That is, we choose $k^{(2)}$ satisfying $k^{(1)} < k^{(2)} < c \cdot x^{(1)}$ and let $(\bar{A}, b^{(2)})$ be the corresponding system of inequalities. Beginning at $x^{(1)}$ we apply #Newton iterates of Newton's method in attempting to maximize $f^{(2)}$. We let $x^{(2)}$ be the final point obtained, and so on.

Here, then, is the algorithm. Fix $0 < \delta < 1$.

Initially: $x^{(0)} = \xi^{(0)}$, $j = 1$.
Step 1: Let $k^{(j)} = \delta[c \cdot x^{(j-1)}] + (1-\delta)k^{(j-1)}$.
Step 2: Apply #Newton iterates of Newton's method, beginning at $x^{(j-1)}$, in attempting to maximize

$$f^{(j)}(x) = l \cdot \ln(c \cdot x - k^{(j)}) + \sum_{i=1}^{m} \ln(\alpha_i \cdot x - b_i).$$

Let $x^{(j)}$ be the resulting point.
Step 3: $j + 1 \to j$ and return to Step 1.

Main Theorem. Let $\delta = 1/13\sqrt{l}$ and #Newton$=1$. Then the algorithm is well-defined in the sense that Newton's method applied to maximizing $f^{(j)}$, $j = 1, 2, \ldots$, and initiated at $x^{(j-1)}$, gives points contained in $\text{Int}(\bar{A}, b^{(j)})$. Moreover, and most importantly,

$$k^{\text{opt}} - c \cdot x^{(j)} \leq \left(1 - \frac{45l}{46(m+l)}\right)\left(1 - \frac{\sqrt{l}}{14(m+l)}\right)^j (k^{\text{opt}} - k^{(0)}),$$

where k^{opt} is the optimal objective value of the LPP (2.1). (Note that $k^{\text{opt}} - c \cdot x^{(j)}$ is positive since $x^{(j)}$ is feasible for the LPP (2.1).) □

Of course the bound provided by the theorem decreases by a factor of

$$\left(1 - \frac{\sqrt{l}}{14(m+l)}\right)$$

with each iterate of the algorithm. The best factor of decrease is provided when $l = m$. This is the value of l that we will use for the complexity analysis in Section 8.

Since the theorem provides the best estimate when $l = m$, why did we bother to develop the algorithm for arbitrary l rather than just using the value $l = m$ throughout? Because the theorem is a worst case bound. In the final steps of the algorithm when an optimal solution is being "zeroed" in on, I expect the best progress will be made (with regards to the parameters l, $\delta = 1/13\sqrt{l}$, #Newton=1) if l equals the number of active constraints at that solution ($\alpha_i \cdot x \geq b_i$ is "active" at y if $\alpha_i \cdot y = b_i$). For most problems I expect that setting $l = m$ will result in a slower algorithm than, say, $l = n$, but perhaps I am wrong.

Of course the values for the parameters δ and #Newton used in the theorem should not necessarily be used in practice. The value of δ is definitely tied to a worst case analysis and is probably overly pessimistic even in that. Also, in practice it would probably be wise to choose δ "large" initially and then reduce it in later iterations if Newton's method begins having trouble in approximating centers. The main function of the parameter δ is to make sure that $x^{(j)}$ is "sufficiently close" to $\xi^{(j+1)}$ to ensure quick convergence of Newton's method.

As of this writing, no attempt at implementing the algorithm has been made.

3. Ideas behind proving the main theorem

The main theorem is proven inductively. The principal inductive hypothesis is that we begin the $(j+1)$th iteration of the algorithm with a "good" approximation $x^{(j)}$ of $\xi^{(j)}$. Then we show that Newton's method initiated at $x^{(j)}$ converges "quickly" to $\xi^{(j+1)}$. The terms "good" and "quickly" will be defined quantitatively later in this section, but to begin with we will use both terms loosely.

How astray can Newton's method go in attempting to find the center of a system of linear inequalities? Here is a simple example. Let $n = 1$, i.e., one variable, and let

$$A' = \begin{bmatrix} 1 \\ -1 \\ -1 \\ \vdots \\ -1 \end{bmatrix}, \qquad b' = \begin{bmatrix} 0 \\ -1 \\ -2 \\ \vdots \\ -m \end{bmatrix},$$

A' being an $(m+1) \times 1$ matrix and $b' \in \mathbb{R}^{m+1}$. Then $\text{Int}(A', b') = (0, 1)$, the open unit interval. For $x \in (0, 1)$, simple computations show the Newton step assigned to x is

$$n'_x = \frac{\dfrac{1}{x} - \displaystyle\sum_{i=1}^{m} \frac{1}{i-x}}{\dfrac{1}{x^2} - \displaystyle\sum_{i=1}^{m} \frac{1}{(i-x)^2}}.$$

Since $\sum_{i=1}^{\infty} 1/(i-x) = \infty$ and $\sum_{i=1}^{\infty} 1/(i-x)^2$ converges, it follows that if m is large, then $x + n'_x$ will lie far, far away from $(0, 1)$. (Of course what constitutes "large" depends on x.) In particular, as $m \to \infty$, the measure of the set of points at which we can initiate Newton's method and obtain convergence to the center goes to zero.

The above example makes it clear that $x^{(j)}$ must satisfy more than just the condition $x^{(j)} \in \text{Int}(\bar{A}, b^{(j+1)})$ if Newton's method initiated at $x^{(j)}$ is to converge to $\xi^{(j+1)}$.

Since Newton's method converges quadratically to $\xi^{(j+1)}$ if initiated sufficiently close to $\xi^{(j+1)}$, the natural approach to take in attempting to prove the main theorem inductively is to show that if δ is sufficiently small, then $\xi^{(j)}$, and hence its "good" approximation $x^{(j)}$, are "sufficiently close" to $\xi^{(j+1)}$ for quadratic convergence of Newton's method. In a sense this is what we do. The trick is describing "sufficiently close" appropriately. The notion of "sufficiently close" that we will use corresponds to the Euclidean distance defined by a new coordinate system on $\mathbb{R}^n$, the coordinates being determined by the particular system of inequalities whose center is to be approximated.

Let (A', b') be a system of m' linear inequalities in n' variables such that $\text{Int}(A', b')$ is non-empty and bounded. As will become clear, the following coordinate system is a natural coordinate system to use for examining the behavior of Newton's method applied to finding the center ξ' of (A', b'). For $x \in \mathbb{R}^{n'}$, define

$$\boldsymbol{X}'_i(x) = \frac{\alpha'_i \cdot x - b'_i}{\alpha'_i \cdot \xi' - b'_i}, \quad i = 1, \ldots, m',$$

the prime on $\boldsymbol{X}'_i(x)$ indicating these are the coordinates of x relative to (A', b'). These coordinates have a simple geometric interpretation. Defining, for $i = 1, \ldots, m'$ and $x \in \mathbb{R}^{n'}$,

$$\text{dis}_i(x) = \text{Euclidean distance from } x \text{ to the closest point on the hyperplane } \{y \in \mathbb{R}^{n'};\ \alpha'_i \cdot y = b'_i\},$$

it is easily shown that

$$\boldsymbol{X}'_i(x) = \pm \frac{\text{dis}_i(x)}{\text{dis}_i(\xi')},$$

where "+" is used if $\alpha'_i \cdot x \geq b'_i$, and "−" is used otherwise.

These coordinates are only a theoretical tool. Being able to compute these coordinates for a single point $x \in \mathbb{R}^{n'}$ is easily seen to be equivalent to knowing ξ'.

Note that in the new coordinates ξ' is assigned the vector e.

Let

$$\Delta_{m'} = \{X \in \mathbb{R}^{m'};\ X \cdot e = m' \text{ and } X_i > 0 \text{ for all } i\}.$$

The following proposition will be proven in Section 4.

Proposition 3.1. *Assume* $\text{Int}(A', b')$ *is non-empty and bounded and assume* $x \in \mathbb{R}^{n'}$ *Then* $\boldsymbol{X}'(x) \cdot e = m'$. *Moreover,* $x \in \text{Int}(A', b')$ *if and only if* $\boldsymbol{X}'(x) \in \Delta_{m'}$. □

A simple consequence of Proposition 3.1 is that if $\|\boldsymbol{X}'(x) - e\| < 1$, then $x \in \text{Int}(A', b')$. Keeping this and the fact that ξ' is assigned e in the new coordinates in mind, the following theorem is our main tool for describing the behavior of Newton's method applied to finding the center of (A', b').

Theorem 3.2. *Assume* $\operatorname{Int}(A', b')$ *is non-empty and bounded. Assume* $x \in \operatorname{Int}(A', b')$ *and* $\varepsilon \doteq \|X'(x) - e\| < 1$. *(Here,* $\|\cdot\|$ *is the Euclidean norm on* $\mathbb{R}^{m'}$.*) Let* n'_x *be the vector satisfying* $(\nabla_x^2 f')n'_x = -(\nabla_x f')^{\mathrm{T}}$, *where* $f'(z) = \sum_{i=1}^{m'} \ln(\alpha'_i \cdot z - b'_i)$. *Let* $y = x + n'_x$. *Then*

$$\|X'(y) - e\| \leq \frac{(1+\varepsilon)^2}{(1-\varepsilon)} \varepsilon^2. \qquad \square$$

Here is a simple consequence of the theorem. If $\|X'(x) - e\| \leq \frac{1}{4}$, then Newton's method initiated at x converges to ξ'. To see this, just note that if $0 < \varepsilon \leq \frac{1}{4}$, then

$$\frac{(1+\varepsilon)^2}{(1-\varepsilon)^1} \varepsilon^2 < \varepsilon.$$

The claim follows.

Theorem 3.2, which is proven in Section 5, is one of the two main tools to be used in proving the main theorem. It provides a useful description of what it means for $x^{(j)}$ to be "sufficiently close" to $\xi^{(j+1)}$ so that Newton's method will work well. The other main tool that is needed is a theorem showing that if δ is only as small as a given quantity (i.e. $1/13\sqrt{l}$ in the main theorem) and if $x^{(j)}$ is a "good" approximation to $\xi^{(j)}$, then $x^{(j)}$ will be "sufficiently close" to $\xi^{(j+1)}$ to apply Theorem 3.2. This is provided by the next theorem.

Let $X^{(j)}(x^{(j)})$ be the coordinates of $x^{(j)}$ defined by $(\bar{A}, b^{(j)})$, and let $X^{(j+1)}(x^{(j)})$ be the coordinates of $x^{(j)}$ defined by $(\bar{A}, b^{(j+1)})$. We assume $x^{(j)} \in \operatorname{Int}(\bar{A}, b^{(j)})$, i.e., thus far the algorithm has been well-defined.

Theorem 3.3. *Let* $0 < \delta < 1$, δ *being a parameter of the algorithm, i.e.,* $k^{(j+1)} = \delta[c \cdot x^{(j)}] + (1-\delta)k^{(j)}$. *Let* $\alpha = \|X^{(j)}(x^{(j)}) - e\|$ *and* $\beta = \|X^{(j+1)}(x^{(j)}) - e\|$. *Then*

$$[1 - 2\alpha]\beta^2 - \left[\frac{l\delta^2}{1-\delta} + (1 + \sqrt{l}\,\delta)\alpha\right]\beta - \left[\frac{l\delta^2}{1-\delta} + \sqrt{l}\,\delta\alpha\right] \leq 0. \qquad \square$$

Theorem 3.3 is proven in Section 6. The point of the theorem is that if δ and $\alpha < \frac{1}{2}$ are given, then β is no larger than the largest root of the resulting quadratic. We will use the following specific estimates.

Corollary 3.4. *Assume that* $\|X^{(j)}(x^{(j)}) - e\| \leq \frac{1}{46}$ *and* $0 < \delta \leq 1/13\sqrt{l}$. *Then* $\|X^{(j+1)}(x^{(j)}) - e\| < \frac{1}{9}$.

Proof. If we let $\alpha = \|X^{(j)}(x^{(j)}) - e\|$, substitution of the assumed bounds gives

$$1 - 2\alpha \geq 1 - \tfrac{2}{46} > 0.95$$

$$\frac{l\delta^2}{1-\delta} + (1 + \sqrt{l}\,\delta)\alpha \leq \frac{1}{13 \cdot 12} + (\tfrac{14}{13})(\tfrac{1}{46}) < 0.03$$

$$\frac{l\delta^2}{1-\delta} + \sqrt{l}\,\delta\alpha \leq \frac{1}{13 \cdot 12} + \frac{1}{13 \cdot 46} < 0.0081.$$

Hence, letting $\beta = \|X^{(j+1)}(x^{(j)}) - e\|$, we have, from Theorem 3.3,

$$9500\beta^2 - 300\beta - 81 \leqslant 0.$$

But for $t \geqslant \frac{1}{9}$,

$$9500t^2 - 300t - 81 \geqslant 117 - 34 - 81 > 0.$$

The corollary follows. $\square$

The final tool we need for proving the main theorem is the following proposition, which will be proven in Section 4 (it is a corollary of Proposition 3.1).

Recall that k^{opt} is defined as the optimal objective value of the linear programming problem (2.1). For $x \in \mathbb{R}^n$, let $X^{(j)}(x)$ be the coordinates of x defined by the system $(\bar{A}, b^{(j)})$.

Proposition 3.5. *Assume $x \in \mathbb{R}^n$ satisfies $c \cdot x \geqslant k^{(j)}$. Then*

$$c \cdot x - k^{(j)} \geqslant \frac{l}{m+l}\left(1 - \frac{\|X^{(j)}(x) - e\|}{\sqrt{l}}\right)(k^{\text{opt}} - k^{(j)}). \qquad \square$$

Assuming the theorems and propositions already stated in this section, we can now give the

Proof of the Main Theorem. We use the following inductive assumptions:

$$\|X^{(j)}(x^{(j)}) - e\| < \tfrac{1}{46} \tag{3.1}$$

and

$$k^{\text{opt}} - k^{(j)} \leqslant \left[1 - \frac{45\delta l}{46(m+l)}\right]^j (k^{\text{opt}} - k^{(0)}). \tag{3.2}$$

Clearly these assumptions are satisfied when $j = 0$.

Corollary 3.4 and (3.1) imply

$$\|X^{(j+1)}(x^{(j)}) - e\| < \tfrac{1}{9}.$$

Consequently, Theorem 3.2 shows that one iteration of Newton's method, applied to finding the center of $(\bar{A}, b^{(j+1)})$ and initiated at $x^{(j)}$, gives a point $x^{(j+1)}$ satisfying

$$\|X^{(j+1)}(x^{(j+1)}) - e\| < \tfrac{1}{46},$$

thus establishing the inductive assumption (3.1). This also shows, by Proposition 3.1, that $x^{(j+1)} \in \text{Int}(\bar{A}, b^{(j+1)})$. Moreover, using $k^{(j+1)} = \delta[c \cdot x^{(j)}] + (1 - \delta)k^{(j)}$, we have

$$\begin{aligned}
k^{\text{opt}} - k^{(j+1)} &= k^{\text{opt}} - k^{(j)} - \delta[c \cdot x^{(j)} - k^{(j)}] \\
&\leqslant k^{\text{opt}} - k^{(j)} - \delta\left[\frac{l}{m+l}\left(1 - \frac{1}{46}\right)(k^{\text{opt}} - k^{(j)})\right] \\
&= \left(1 - \frac{45\delta l}{46(m+l)}\right)(k^{\text{opt}} - k^{(j)}) \\
&\leqslant \left(1 - \frac{45\delta l}{46(m+l)}\right)^{j+1}(k^{\text{opt}} - k^{(0)}),
\end{aligned}$$

the first inequality being implied by (3.1) and Proposition 3.5 and the second inequality being implied by (3.2). We have now established the inductive assumption (3.2).

Now we establish the bound on $k^{\text{opt}} - c \cdot x^{(j)}$ that is stated in the main theorem. We have

$$\begin{aligned}
k^{\text{opt}} - c \cdot x^{(j)} &= k^{\text{opt}} - k^{(j)} - [c \cdot x^{(j)} - k^{(j)}] \\
&\leqslant k^{\text{opt}} - k^{(j)} - \left[\frac{l}{m+l}\left(1 - \frac{1}{46}\right)(k^{\text{opt}} - k^{(j)})\right] \\
&= \left(1 - \frac{45 l}{46(m+l)}\right)(k^{\text{opt}} - k^{(j)}) \\
&\leqslant \left(1 - \frac{45 l}{46(m+l)}\right)\left(1 - \frac{45\delta l}{46(m+l)}\right)^{j}(k^{\text{opt}} - k^{(0)}) \\
&< \left(1 - \frac{45 l}{46(m+l)}\right)\left(1 - \frac{\sqrt{l}}{14(m+l)}\right)^{j}(k^{\text{opt}} - k^{(0)}),
\end{aligned}$$

the first inequality being implied by (3.1) and Proposition 3.5, the second inequality being implied by (3.2), and the third inequality being obtained by substitution of $\delta = 1/13\sqrt{l}$. This concludes the proof of the main theorem.

4. Proofs of propositions

In this section we prove the two propositions stated in the previous section. For the reader's convenience, we restate these propositions before proving them.

Recall that

$$\Delta_{m'} = \{X \in \mathbb{R}^{m'};\ X \cdot e = m' \text{ and } X_i > 0 \text{ for all } i\}$$

and that for $x \in \mathbb{R}^{n'}$, $\boldsymbol{X}'(x)$ are the coordinates assigned to x with respect to the system (A', b').

Proposition 3.1. *Assume* $x \in \mathbb{R}^{n'}$. *Then* $\boldsymbol{X}'(x) \cdot e = m'$. *Moreover,* $x \in \operatorname{Int}(A', b')$ *if and only if* $\boldsymbol{X}'(x) \in \Delta_{m'}$.

Proof. To show $\boldsymbol{X}'(x) \cdot e = m'$ for all $x \in \mathbb{R}^{n'}$, it suffices to show that $(\boldsymbol{X}'(x) - e) \cdot e = 0$ for all $x \in \mathbb{R}^{n'}$. But, by definition of $\boldsymbol{X}'(x)$,

$$
\begin{aligned}
(\boldsymbol{X}'(x) - e) \cdot e &= \sum_{i=1}^{m'} \left(\frac{\alpha_i' \cdot x - b_i'}{\alpha_i' \cdot \xi' - b_i'} - 1 \right) \\
&= \sum_{i=1}^{m'} \frac{\alpha_i'(x - \xi')}{\alpha_i' \cdot \xi' - b_i'} \\
&= (\nabla_{\xi'} f')(x - \xi') \\
&= 0,
\end{aligned}
$$

the last equality since ξ' maximizes f', and hence $\nabla_{\xi'} f' = 0$.

The claim that $x \in \operatorname{Int}(A', b')$ if and only if $\boldsymbol{X}'(x) \in \Delta_{m'}$ now follows immediately from the definition of $\boldsymbol{X}'(x)$. □

Proposition 3.5. *Assume* $x \in \mathbb{R}^n$ *satisfies* $c \cdot x \geq k^{(j)}$. *Then*

$$
c \cdot x - k^{(j)} \geq \frac{l}{m+l} \left(1 - \frac{\|\boldsymbol{X}^{(j)}(x) - e\|}{\sqrt{l}} \right) (k^{\text{opt}} - k^{(j)}).
$$

Proof. We begin by showing that

$$
c \cdot \xi^{(j)} - k^{(j)} \geq \frac{l}{m+l} (k^{\text{opt}} - k^{(j)}). \tag{4.1}
$$

Let x^{opt} be an optimal solution to the LPP (2.1). (Under our assumptions that $\operatorname{Int}(\bar{A}, b^{(0)})$ is non-empty and bounded, an optimal solution exists.) Let $\boldsymbol{X}_i^{(j)}(x^{\text{opt}})$ denote the ith coordinate of x^{opt} with respect to the system $(\bar{A}, b^{(j)})$. By definition of the coordinates,

$$
k^{\text{opt}} - k^{(j)} = \boldsymbol{X}_i^{(j)}(x^{\text{opt}})(c \cdot \xi^{(j)} - k^{(j)}), \quad i = m+1, \ldots, m+l. \tag{4.2}
$$

On the other hand, by Proposition 3.1, proven above,

$$
\sum_{i=1}^{m+l} \boldsymbol{X}_i^{(j)}(x^{\text{opt}}) = m + l
$$

and

$$
\boldsymbol{X}_i^{(j)}(x^{\text{opt}}) \geq 0 \quad \text{for all } i.
$$

Since $\boldsymbol{X}_{m+1}^{(j)}(x^{\text{opt}}) = \cdots = \boldsymbol{X}_{m+l}^{(j)}(x^{\text{opt}})$, it follows that

$$
\boldsymbol{X}_i^{(j)}(x^{\text{opt}}) \leq \frac{m+l}{l} \quad \text{for } i = m+1, \ldots, m+l. \tag{4.3}
$$

Together, (4.2) and (4.3) imply (4.1).

Now consider arbitrary $x \in \mathbb{R}^n$. Then for $i = m+1, \ldots, m+l$,

$$\begin{aligned} c \cdot x - k^{(j)} &= \boldsymbol{X}_i^{(j)}(x)(c \cdot \xi^{(j)} - k^{(j)}) \\ &\geqslant \frac{l}{m+l} \boldsymbol{X}_i^{(j)}(x)(k^{\text{opt}} - k^{(j)}) \\ &\geqslant \frac{l}{m+l}(1 - |\boldsymbol{X}_i^{(j)}(x) - 1|)(k^{\text{opt}} - k^{(j)}), \end{aligned} \tag{4.4}$$

the first inequality by (4.1). However, since $\boldsymbol{X}_{m+1}^{(j)}(x) = \cdots = \boldsymbol{X}_{m+l}^{(j)}(x)$, we have

$$\|\boldsymbol{X}^{(j)}(x) - e\| \geqslant \sqrt{l}|\boldsymbol{X}_i^{(j)}(x) - 1| \quad \text{for } i = m+1, \ldots, m+l. \tag{4.5}$$

Substituting (4.5) into (4.4) concludes the proof of the proposition. □

5. Proof of Theorem 3.2

Theorem 3.2. *Assume* $\text{Int}(A', b')$ *is non-empty and bounded. Assume* $x \in \text{Int}(A', b')$ *and let* $\varepsilon = \|\boldsymbol{X}'(x) - e\| < 1$. *Let* n'_x *be the vector satisfying* $(\nabla_x^2 f')n'_x = -(\nabla_x f')^{\text{T}}$ *and let* $y = x + n'_x$. *Then,*

$$\|\boldsymbol{X}'(y) - e\| \leqslant \frac{(1+\varepsilon)^2}{(1-\varepsilon)} \varepsilon^2. \quad \square$$

The proof of this theorem in the original manuscript was very long and messy. I wrote: "Surely a shorter proof must exist!" The following proof relies on a few key observations provided by Jeff Lagarias.

Let $\mathbb{R}_+^{m'}$ denote the strictly positive orthant in $\mathbb{R}^{m'}$ and let $F : \mathbb{R}^{m'} \to \mathbb{R}$ be the function

$$F(X) = \sum_{i=1}^{m'} \ln(X_i).$$

We begin the proof with the following lemma.

Lemma 5.1. *If* $x \in \text{Int}(A', b')$, *then*

$$f'(x) = F(\boldsymbol{X}'(x)) + \sum_{i=1}^{m'} \ln(\alpha'_i \cdot \xi' - b'_i).$$

Proof

$$\begin{aligned} f'(x) &= \sum_{i=1}^{m'} \ln(\alpha'_i \cdot x - b'_i) \\ &= \sum_{i=1}^{m'} \ln\left(\frac{\alpha'_i \cdot x - b'_i}{\alpha'_i \cdot \xi' - b'_i}\right) + \sum_{i=1}^{m'} \ln(\alpha'_i \cdot \xi' - b'_i) \\ &= F(\boldsymbol{X}'(x)) + \sum_{i=1}^{m'} \ln(\alpha'_i \cdot \xi' - b'_i). \quad \square \end{aligned}$$

Let $L : \mathbb{R}^{n'} \to \mathbb{R}^{m'}$ denote the affine map $L(x) = \boldsymbol{X}'(x)$.

Since y is, by definition, the point in $\mathbb{R}^{n'}$ maximizing

$$q(z) = (\nabla_x f')(z-x) + \tfrac{1}{2}(z-x)^{\mathrm{T}}(\nabla_x^2 f')(z-x),$$

it follows from Lemma 5.1 that $\boldsymbol{X}'(y)$ is the point in the image of L maximizing

$$Q(X) = e^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-1}(X - \boldsymbol{X}'(x)) - \tfrac{1}{2}(X - \boldsymbol{X}'(x))^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-2}(X - \boldsymbol{X}'(x)),$$

where D_X is the diagonal matrix with ith diagonal entry X_i.

Let $\mathbb{Z}'$ denote the point in $\{X \in \mathbb{R}^{m'};\, X \cdot e = m'\}$ that maximizes $Q(X)$. Then, assuming $X \cdot e = m'$, we have

$$Q(X) = Q(\mathbb{Z}') - \tfrac{1}{2}(X - \mathbb{Z}')^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-2}(X - \mathbb{Z}'). \tag{5.1}$$

Since $\boldsymbol{X}'(y)$ maximizes Q over the image of L, since the image of L is contained in $\{X \in \mathbb{R}^{m'};\, X \cdot e = m'\}$, and since the image of L is an affine space containing e, we find from (5.1) that $\boldsymbol{X}'(y) - e$ must be perpendicular to $D_{\boldsymbol{X}'(x)}^{-2}(\mathbb{Z}' - \boldsymbol{X}'(y))$, that is,

$$(\boldsymbol{X}'(y) - e)^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-2}(\mathbb{Z}' - \boldsymbol{X}'(y)) = 0.$$

Hence, $D_{\boldsymbol{X}'(x)}^{-1}(\boldsymbol{X}'(y) - e)$ is perpendicular to $D_{\boldsymbol{X}'(x)}^{-1}(\mathbb{Z}' - \boldsymbol{X}'(y))$. Consequently,

$$\|D_{\boldsymbol{X}'(x)}^{-1}(\boldsymbol{X}'(y) - e)\|^2 + \|D_{\boldsymbol{X}'(x)}^{-1}(\mathbb{Z}' - \boldsymbol{X}'(y))\|^2 = \|D_{\boldsymbol{X}'(x)}^{-1}(\mathbb{Z}' - e)\|^2$$

and thus

$$\frac{1}{\max(\boldsymbol{X}'(x))}\|\boldsymbol{X}'(y) - e\| \leqslant \frac{1}{\min(\boldsymbol{X}'(x))}\|\mathbb{Z}' - e\|$$

where

$$\max(X) \doteq \max_i\{X_i\}, \qquad \min(X) \doteq \min_i\{X_i\}.$$

We have proven

Proposition 5.2. *Let $\mathbb{Z}'$ denote the optimal solution of the problem*

$$\begin{aligned} \max \quad & e^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-1}(X - \boldsymbol{X}'(x)) - \tfrac{1}{2}(X - \boldsymbol{X}'(x))^{\mathrm{T}} D_{\boldsymbol{X}'(x)}^{-2}(X - \boldsymbol{X}'(x)) \\ \text{s.t.} \quad & X \cdot e = m'. \end{aligned}$$

Then

$$\|\boldsymbol{X}'(y) - e\| \leqslant \frac{\max(\boldsymbol{X}'(x))}{\min(\boldsymbol{X}'(x))}\|\mathbb{Z}' - e\|. \qquad \square$$

Now we obtain a bound on $\|\mathbb{Z}' - e\|$.

Proposition 5.3. *Letting $\varepsilon \doteq \|\boldsymbol{X}'(x) - e\|$, we have that*

$$\|\mathbb{Z}' - e\| \leqslant (1+\varepsilon)\varepsilon^2.$$

Proof. It is not difficult to show that

$$\mathbb{Z}' = 2\boldsymbol{X}'(x) - \frac{m'}{\|\boldsymbol{X}'(x)\|^2} D^2_{\boldsymbol{X}'(x)} e. \tag{5.2}$$

Let $V = \boldsymbol{X}'(x) - e$. Substituting $\boldsymbol{X}'_i(x) = 1 + V_i$ and

$$\|\boldsymbol{X}'(x)\|^2 = \|e\|^2 + \|\boldsymbol{X}'(x) - e\|^2 = m' + \varepsilon^2$$

into (5.2) gives, with a little rearrangement,

$$\mathbb{Z}'_i = 1 + \frac{1}{m' + \varepsilon^2}(\varepsilon^2 - m' V_i^2 + 2\varepsilon^2 V_i).$$

Consequently,

$$\begin{aligned}\|\mathbb{Z}' - e\| &= \frac{1}{m' + \varepsilon^2}\left[\sum_{i=1}^{m'} (\varepsilon^2 - m' V_i^2 + 2\varepsilon^2 V_i)^2\right]^{1/2} \\ &\leqslant \frac{1}{m' + \varepsilon^2}\left(\left[\sum_{i=1}^{m'} (\varepsilon^2 - m' V_i^2)^2\right]^{1/2} + \left[\sum_{i=1}^{m'} (2\varepsilon^2 V_i)^2\right]^{1/2}\right).\end{aligned} \tag{5.3}$$

However, expanding and making use of the substitution $\sum_{i=1}^{m'} V_i^2 = \varepsilon^2$ shows

$$\begin{aligned}\sum_{i=1}^{m'} (\varepsilon^2 - m' V_i^2)^2 &= m'\left[-\varepsilon^4 + m' \sum_{i=1}^{m'} V_i^4\right] \\ &\leqslant m'\left[-\varepsilon^4 + m'\left(\sum_{i=1}^{m'} V_i^2\right)^2\right] \\ &= m'(m'-1)\varepsilon^4.\end{aligned}$$

Also,

$$\sum_{i=1}^{m'} (2\varepsilon^2 V_i)^2 = 4\varepsilon^6.$$

Hence, by (5.3),

$$\begin{aligned}\|\mathbb{Z}' - e\| &\leqslant \frac{[m'(m'-1)]^{1/2}\varepsilon^2 + 2\varepsilon^3}{m' + \varepsilon^2} \\ &\leqslant \frac{[m'(m'-1)]^{1/2}\varepsilon^2 + 2\varepsilon^3}{m'} \\ &< \varepsilon^2 + \frac{2}{m'}\varepsilon^3 \leqslant \varepsilon^2(1+\varepsilon),\end{aligned}$$

the last inequality using $m' \geqslant 2$, which follows from the assumption that Int(A', b') is bounded. This concludes the proof of the proposition. □

Combining Propositions 5.2 and 5.3 and using the easily proven fact that

$$\frac{\max(\boldsymbol{X}'(x))}{\min(\boldsymbol{X}'(x))} \leqslant \frac{1+\varepsilon}{1-\varepsilon}$$

concludes the proof of Theorem 3.2.

6. Proof of Theorem 3.3

Theorem 3.3. *Let* $0<\delta<1$, δ *being a parameter of the algorithm, i.e.,* $k^{(j+1)}=\delta[c\cdot x^{(j)}]+(1-\delta)k^{(j)}$, *where* $x^{(j)}\in \operatorname{Int}(\bar{A}, b^{(j)})$. *Let* $\alpha=\|\boldsymbol{X}^{(j)}(x^{(j)})-e\|$ *and* $\beta=\|\boldsymbol{X}^{(j+1)}(x^{(j)})-e\|$. *Then*

$$[1-2\alpha]\beta^2-\left[\frac{l\delta^2}{1-\delta}+(1+\sqrt{l}\,\delta)\alpha\right]\beta-\left[\frac{l\delta^2}{1-\delta}+\sqrt{l}\,\delta\alpha\right]\leqslant 0. \qquad \square$$

The only assumptions used in the proof of Theorem 3.3 are that $\operatorname{Int}(\bar{A}, b^{(j)})$ is non-empty and bounded and $x^{(j)}\in \operatorname{Int}(\bar{A}, b^{(j)})$.

To simplify notation throughout this section, we let $x=x^{(j)}$, $\boldsymbol{X}^{(j)}=\boldsymbol{X}^{(j)}(x^{(j)})$ *and* $\boldsymbol{X}^{(j+1)}=\boldsymbol{X}^{(j+1)}(x^{(j)})$.

Lemma 6.1. *For* $i=m+1,\dots,m+l$,

$$\boldsymbol{X}_i^{(j+1)}\geqslant(1-\delta)\boldsymbol{X}_i^{(j)}.$$

Proof. Since $k^{(j+1)}=\delta(c\cdot x)+(1-\delta)k^{(j)}$ it follows that

$$\frac{c\cdot x-k^{(j+1)}}{c\cdot x-k^{(j)}}=(1-\delta), \tag{6.1}$$

and hence, for $i=m+1,\dots,m+l$,

$$\begin{aligned}\frac{\boldsymbol{X}_i^{(j+1)}}{\boldsymbol{X}_i^{(j)}}&=\frac{(c\cdot x-k^{(j+1)})/(c\cdot\xi^{(j+1)}-k^{(j+1)})}{(c\cdot x-k^{(j)}/(c\cdot\xi^{(j)}-k^{(j)})}\\&=(1-\delta)(c\cdot\xi^{(j)}-k^{(j)})/(c\cdot\xi^{(j+1)}-k^{(j+1)}).\end{aligned}$$

Consequently, to prove the lemma it suffices to show that

$$c\cdot\xi^{(j)}-k^{(j)}\geqslant c\cdot\xi^{(j+1)}-k^{(j+1)}. \tag{6.2}$$

Let $\varphi:[0,1]\to[k^{(j)},k^{(j+1)}]$ be the map

$$\varphi(t)=k^{(j)}+t(k^{(j+1)}-k^{(j)}),$$

and let $\xi(t)$ be the center of the system $(\bar{A}, b(t))$, where $(b(t))^{\mathrm{T}}=(b_1,\dots,b_m,\varphi(t),\dots,\varphi(t))$. To show (6.2), it suffices to show that

$$c\cdot\left(\frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\right)\leqslant k^{(j+1)}-k^{(j)} \tag{6.3}$$

since

$$c \cdot \xi^{(j+1)} - k^{(j+1)} = c \cdot [\xi^{(j)} + \int_0^1 \frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\mathrm{d}t] - k^{(j)} + (k^{(j)} - k^{(j+1)})$$

$$= c \cdot \xi^{(j)} - k^{(j)} + \int_0^1 c \cdot \frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\,\mathrm{d}t + (k^{(j)} - k^{(j+1)}).$$

Since $\xi(t)$ maximizes the function

$$y \mapsto l \cdot \ln(c \cdot y - \varphi(t)) + \sum_{i=1}^{m} \ln(\alpha_i \cdot y - b_i),$$

evaluation of the gradient of this function at $\xi(t)$ gives

$$\frac{l}{c \cdot \xi(t) - \varphi(t)} c + \sum_{i=1}^{m} \frac{1}{\alpha_i \cdot \xi(t) - b_i} \alpha_i = 0.$$

Taking the derivatives of both sides of the last equation with respect to t, and rearranging slightly, gives

$$\left[-\sum_{i=1}^{m} \frac{1}{(\alpha_i \cdot \xi(t) - b_i)^2} \alpha_i^{\mathrm{T}} \alpha_i\right]\left(\frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\right)$$

$$= \frac{1}{(c \cdot \xi(t) - \varphi(t))^2}\left[c^{\mathrm{T}}c\left(\frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\right) - c^{\mathrm{T}}\left(\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t)\right)\right].$$

recalling α_i and c to be row vectors. Now the dot product of the left side of this equation with $((\mathrm{d}/\mathrm{d}t)\xi(t))$ is non-positive (because the matrix in the bracket is negative definite), and thus, from the right side we obtain

$$\left[c \cdot \left(\frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\right)\right]^2 \leqslant \left[c \cdot \left(\frac{\mathrm{d}}{\mathrm{d}t}\xi(t)\right)\right]\left(\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t)\right).$$

Since

$$\frac{\mathrm{d}}{\mathrm{d}t}\varphi(t) = k^{(j+1)} - k^{(j)} > 0,$$

this immediately yields (6.3). Thus the proposition. □

Lemma 6.2

$$\sum_{i=1}^{m+l} \frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}} = m + l + \sum_{i=m+1}^{m+l} \frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}} \delta(1 - \boldsymbol{X}_i^{(j+1)}).$$

Proof. Using (6.1), it is easily shown that for any $y \in \mathbb{R}^n$,

$$\frac{c \cdot y - k^{(j)}}{c \cdot x - k^{(j)}} = 1 + (1 - \delta)\left(\frac{c \cdot y - k^{(j+1)}}{c \cdot x - k^{(j+1)}} - 1\right).$$

In particular, letting $y=\xi^{(j+1)}$, we have, for $i=m+1,\ldots,m+l$,

$$\frac{c\cdot\xi^{(j+1)}-k^{(j)}}{c\cdot x-k^{(j)}}=1+(1-\delta)\left(\frac{1}{\boldsymbol{X}_i^{(j+1)}}-1\right)=\frac{1}{\boldsymbol{X}_i^{(j+1)}}[1-\delta(1-\boldsymbol{X}_i^{(j+1)})].$$

Multiplying both sides by

$$\boldsymbol{X}_i^{(j)}=\frac{c\cdot x-k^{(j)}}{c\cdot\xi^{(j)}-k^{(j)}},$$

we find that for $i=m+1,\ldots,m+l$,

$$\frac{c\cdot\xi^{(j+1)}-k^{(j)}}{c\cdot\xi^{(j)}-k^{(j)}}=\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}[1-\delta(1-\boldsymbol{X}_i^{(j+1)})]. \tag{6.4}$$

Also, for $i=1,\ldots,m$,

$$\frac{\alpha_i\cdot\xi^{(j+1)}-b_i}{\alpha_i\cdot\xi^{(j)}-b_i}=\frac{\dfrac{\alpha_i\cdot x-b_i}{\alpha_i\cdot\xi^{(j)}-b_i}}{\dfrac{\alpha_i\cdot x-b_i}{\alpha_i\cdot\xi^{(j+1)}-b_i}}=\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}. \tag{6.5}$$

But now note that the quantities on the left of (6.4) and (6.5) are the coordinates of $\xi^{(j+1)}$ with respect to $(\bar{A},b^{(j)})$. In particular, by Proposition 3.1, they sum to $m+l$. Setting the sum of the terms on the right of (6.4) and (6.5) equal to $m+l$, and rearranging, gives the identity stated in the lemma. □

Now we can prove Theorem 3.3.

Let $U=\boldsymbol{X}^{(j)}-e$ and $V=\boldsymbol{X}^{(j+1)}-e$. Of course $\boldsymbol{X}_i^{(j+1)}>0$, for all i, since $x\in\operatorname{Int}(\bar{A},b^{(j+1)})$. Thus, expanding $r\mapsto 1/(1+rV_i)$ around $r=0$, we have, for all i, the Taylor series with remainder integral

$$\frac{1}{\boldsymbol{X}_i^{(j+1)}}=\frac{1}{1+V_i}=1-V_i+\int_0^1(1-t)\frac{2V_i^2}{(1+tV_i)^3}\,\mathrm{d}t.$$

Consequently,

$$\sum_{i=1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}=m+l-V\cdot e+U\cdot e-V\cdot U+\sum_{i=1}^{m+l}(1+U_i)\int_0^1(1-t)\frac{V_i^2}{(1+tV_i)^3}\,\mathrm{d}t.$$

Substituting $V\cdot e=U\cdot e=0$, $1+U_i\geqslant 1-\alpha$ and $0<1+tV_i\leqslant 1+t\beta$ for $0\leqslant t\leqslant 1$ (recalling $\alpha=\|\boldsymbol{X}^{(j)}-e\|$, $\beta=\|\boldsymbol{X}^{(j+1)}-e\|$), we find

$$\sum_{i=1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}\geqslant m+l-\alpha\beta+(1-\alpha)\sum_{i=1}^{m+l}\int_0^1(1-t)\frac{2V_i^2}{(1+t\beta)^3}\,\mathrm{d}t$$

$$=m+l-\alpha\beta+(1-\alpha)\int_0^1(1-t)\frac{2\beta^2}{(1+t\beta)^3}\,\mathrm{d}t. \tag{6.6}$$

Using the Taylor expression

$$\frac{1}{1+\beta}=1-\beta+\int_0^1 (1-t)\frac{2\beta^2}{(1+t\beta)^3}\,\mathrm{d}t,$$

(6.6) reduces to

$$\sum_{i=1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}\geq m+l-\alpha\beta+\frac{1-\alpha}{1+\beta}\beta^2. \tag{6.7}$$

On the other hand, we know from Lemma 6.2 that

$$\sum_{i=1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}=m+l+\sum_{i=m+1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}\delta(1-\boldsymbol{X}_i^{(j+1)})$$

which combined with Lemma 6.1 implies

$$\begin{aligned}\sum_{i=1}^{m+l}\frac{\boldsymbol{X}_i^{(j)}}{\boldsymbol{X}_i^{(j+1)}}&\leq m+l+\sum_{i=m+1}^{m+l}\frac{1}{1-\delta}\delta[1-(1-\delta)\boldsymbol{X}_i^{(j)}]\\&=m+l+\frac{l\delta^2}{1-\delta}+\delta\sum_{i=m+1}^{m+l}(1-\boldsymbol{X}_i^{(j)})\\&\leq m+l+\frac{l\delta^2}{1-\delta}+\sqrt{l}\,\delta\alpha,\end{aligned} \tag{6.8}$$

where we used $\sum_i (1-\boldsymbol{X}_i^{(j)})^2\leq\alpha$ to infer that $\sum_{i=m+1}^{m+l}(1-\boldsymbol{X}_i^{(j)})\leq\sqrt{l}\,\alpha$. Combining (6.7) and (6.8), clearing the denominator $1+\beta$ and rearranging, gives the inequality in the theorem.

7. Recasting an LPP into the proper format

In this section we show how any linear programming problem can be recast into a problem fitting the framework of the algorithm. Such a recasting can be done in several ways. The method we develop was chosen primarily for its expository simplicity.

For the complexity analysis to be performed in the next section we need to relate complexity measures of the original problem (e.g. number of bits in the representation) to complexity measures of the recasted problem, and we need to bound the number of arithmetic and bit operations required for the recasting.

Assume that we are interested in solving

$$\begin{aligned}&\max\quad \hat{c}\cdot x\\&\text{s.t.}\quad \hat{A}x\geq\hat{b}\end{aligned} \tag{7.1}$$

where $\hat{c}\in\mathbb{R}^{\hat{n}}$, $\hat{b}\in\mathbb{R}^{\hat{m}}$ and $\hat{A}$ is an $\hat{m}\times\hat{n}$ matrix. (We reserve A, b and c to refer to an LPP already fitting the framework of the algorithm.)

For the complexity analysis, we will assume the entries of $\hat{A}$, $\hat{b}$ and $\hat{c}$ are integers and we will let

$$\hat{L} = \text{sum of bits needed to represent all entries in } \hat{A}, \hat{b} \text{ and } \hat{c},$$

in particular, $\hat{L} \geq \hat{m}\hat{n}$.

In what follows, only those statements referring specifically to the complexity analysis are based on the assumption that the entries in $\hat{A}$, $\hat{b}$ and $\hat{c}$ are integers. Also, for the complexity analysis we will employ "O($\hat{L}$)" notation in a manner that we now elucidate. Our recasting of an LPP into a format fitting the framework of the algorithm will require certain quantities to be specified, for example, upper and lower bounds on the coordinates of optimal solutions. Using Cramer's rule and the fact that all determinants of $[\hat{A}|\hat{b}]$ (i.e. $\hat{b}$ appended to $\hat{A}$) have absolute value bounded by $2^{\hat{L}}$, such upper and lower bounds are provided by $2^{\hat{L}}$ and $2^{-\hat{L}}$. Rather than giving specific bounds such as these, we will simply say that such bounds can be chosen of the form $2^{O(\hat{L})}$ and $-2^{O(\hat{L})}$, the intended implication being that specific values (involving $\hat{L}$ linearly) *can* be determined by a more lengthy analysis (*and would need to be determined to carry out the actual recasting of the LPP*). In practice, such values would be guessed at. Values like $2^{\hat{L}}$ would be ridiculously larger than what is generally needed.

We assume that we know integer values low_j, up_j, $j = 1, \ldots, \hat{n}$, such that (i) if the LPP (7.1) has an optimal solution, then it has an optimal solution x^{opt} satisfying $\text{low}_j \leq x_j^{\text{opt}} \leq \text{up}_j$ for all j, and (ii) if the LPP (7.1) is unbounded, then it has a feasible solution x satisfying $\text{low}_j \leq x_j \leq \text{up}_j$ for all j. By standard arguments using Cramer's rule, we may choose the low_j and up_j so that $-2^{O(\hat{L})} \leq \text{low}_j \leq \text{up}_j \leq 2^{O(\hat{L})}$ for all j.

To recast (7.1) into the form required for our algorithm, we create a closely related LPP

$$\begin{aligned} \max \quad & c \cdot (x, t) \\ \max \quad & A(x, t) \geq b \end{aligned} \tag{7.2}$$

where A is an $(\hat{m} + 2\hat{n} + 3) \times (\hat{n} + 1)$ matrix, and $(x, t) \in \mathbb{R}^{\hat{n}} \times \mathbb{R}$. The "artificial variable" t is added to guarantee feasibility. We will see that the number of arithmetic operations required to convert (7.1) to (7.2) is $O(\hat{m}\hat{n})$, and the number of bit operations is $O(\hat{m}\hat{n}\hat{L}(\log \hat{L})(\log\log \hat{L}))$, the factor $\hat{L}(\log \hat{L})(\log\log \hat{L})$ arising from the multiplication of $O(\hat{L})$ bit numbers. Although the number of bits needed to represent the LPP (7.2) may not be $O(\hat{L})$, we will see that the bit lengths of the determinants of all square submatrices of $[A|b]$ are $O(\hat{L})$, as the entries in c. These are the quantities that will be focused on in the complexity analysis contained in the next section.

Let $v = \sum_{i=1}^{\hat{m}} \hat{a}_i^{\mathrm{T}}$, where $\hat{a}_i$ is the ith row of $\hat{A}$, and let $\text{Pos} = \{j;\, v_j > 0\}$ and $\text{Neg} = \{j;\, v_j < 0\}$.

For $s \in \mathbb{R}$, define

$$T(s) = s(\hat{m} + 2\hat{n} + 1) - \left[-\sum_{i=1}^{\hat{m}} b_i + \sum_{j=1}^{\hat{n}} (\text{up}_j - \text{low}_j) + \sum_{j \in \text{Pos}} v_j \cdot \text{up}_j + \sum_{j \in \text{Neg}} v_j \cdot \text{low}_j \right].$$

Choose $s^*>0$ sufficiently large so that $T(s^*)>0$. For the complexity analysis, we may choose s^* to be an integer bounded by $2^{O(\hat{L})}$. Then $T(s^*)$ is also an integer, bounded in absolute value by $2^{O(\hat{L})}$.

For $\hat{c}_{\hat{n}+1} \in \mathbb{R}$ (to be specified shortly), consider the LPP in $\hat{n}+1$ variables $(x, t) \in \mathbb{R}^{\hat{n}} \times \mathbb{R}$,

$$\begin{array}{lrl} \max & \hat{c}\cdot x + \hat{c}_{\hat{n}+1} t & \\ \text{s.t.} & \hat{a}_i \cdot x \quad -(s^*+\hat{b}_i)t \geq -s^*, & i=1,\ldots,m, \\ & -(1+v_j)x_j \; -[s^*-(1+v_j)\,\mathrm{up}_j]t \geq -s^*, & \text{for } j\in \mathrm{Pos}, \\ & -x_j \quad -[s^*-\mathrm{up}_j]t \geq -s^*, & \text{for } j\notin \mathrm{Pos}, \\ & (1-v_j)x_j - [s^*+(1-v_j)\,\mathrm{low}_j]t \geq -s^*, & \text{for } j\in \mathrm{Neg}, \\ & x_j \quad -[s^*+\mathrm{low}_j]t \geq -s^*, & \text{for } j\notin \mathrm{Neg}, \\ & -t \geq -1, & \\ & T(s^*)t \geq -s^*. & \end{array} \tag{7.3}$$

The LPP (7.2) will be the above LPP with one additional constraint.

The main relation between the feasible region of the above LPP and that of (7.1) is the following. A point $(x, 1)$ is feasible for the above LPP if and only if $\hat{A}x \geq \hat{b}$ and $\mathrm{low}_j \leq x_j \leq \mathrm{up}_j$, $j=1,\ldots,\hat{n}$. The value $\hat{c}_{\hat{n}+1}$ serves as a "big M" quantity, and will be chosen to force $t=1$ in the optimal solution (assuming feasibility).

The last two inequalities in the above LPP bound t above and below. In turn, these bounds and the inequalities involving $j\in \mathrm{Pos}$ and $j\notin \mathrm{Pos}$ bound all variables x_j from above. Similarly, the inequalities involving $j\in \mathrm{Neg}$ and $j\notin \mathrm{Neg}$ bound all variables x_j from below. Hence the feasible region for this LPP is bounded. Also, it is easily verified that 0 is in the interior of this LPP.

Lemma 7.1. *Let $\tilde{A}$ denote the constraint matrix for* (7.3) *and let $\tilde{b}$ denote the right hand side vector. Then the absolute values of the determinants of all square submatrices of $[\tilde{A}\,|\,\tilde{b}]$ are bounded by $2^{O(\hat{L})}$.*

Proof. Rearranging slightly, we may assume

$$[\tilde{A}\,|\,\tilde{b}] = \begin{bmatrix} \hat{A} & & \\ D_1 & u & \tilde{b} \\ D_2 & & \\ 0 & & \end{bmatrix}$$

where $u \in \mathbb{R}^{\hat{m}+2\hat{n}+2}$ and D_1 and D_2 are $\hat{n}\times\hat{n}$ diagonal matrices (e.g. the (j,j) entry of D_1 is $-(1+v_j)$ if $j\in\mathrm{Pos}$, and is -1 if $j\notin\mathrm{Pos}$). Noting that the absolute value of the determinants of all square submatrices of $\hat{A}$, D_1 and D_2 are bounded by $2^{O(\hat{L})}$, as are the absolute values of the entries in u and $\tilde{b}$, the lemma follows from performing cofactor expansion along the last two columns of $[\tilde{A}\,|\,\tilde{b}]$ and making use of the diagonal structure of D_1 and D_2. $\square$

Using the fact that all feasible points for the LPP (7.3) have last coordinate $t \leq 1$, choose $\hat{c}_{\hat{n}+1}$ sufficiently large so that if there exists a feasible point for (7.3) with $t = 1$, then all optimal solutions have $t = 1$. For the complexity analysis, $\hat{c}_{\hat{n}+1}$ can be chosen to be a positive integer bounded by $2^{O(\hat{L})}$. This is a consequence of (i) the feasible region for (7.3) is bounded; (ii) by Lemma 7.1 and Cramer's rule, the absolute value of the coordinates of all extreme points of (7.3) are bounded by $2^{O(\hat{L})}$; (iii) by Lemma 7.1 and Cramer's rule, all extreme points for (7.3) with last coordinate $t < 1$ actually have $t < 1 - 2^{-O(\hat{L})}$; and (iv) the values $|\hat{c}_j|$, $j = 1, \ldots, \hat{n}$, are bounded by $2^{O(\hat{L})}$.

Now choose K sufficiently large so that $|\hat{c} \cdot x + \hat{c}_{\hat{n}+1} t| \leq K$ for all feasible points of (7.3). Using Lemma 7.1 and our previous bound on $\hat{c}_{\hat{n}+1}$ it is easily seen that K can be chosen not to exceed $2^{O(\hat{L})}$.

Finally, the LPP (7.2) (which is the recast version of the LPP (7.1)), is obtained by appending to (7.3) the additional constraint

$$-c \cdot (x, t) \geq -K,$$

c being the vector $(\hat{c}, \hat{c}_{\hat{n}+1})$. This constraint is added to help make 0 the center of the polytope. It is easily shown, using Lemma 7.1, that the absolute values of the determinants of all square submatrices of $[A|b]$ (A and b in the recasted problem (7.2)) are bounded by $2^{O(\hat{L})}$.

Since $(x, 1)$ is feasible for the LPP (7.2) if and only if $\hat{A}x \geq \hat{b}$ and $\text{low}_j \leq x_j \leq \text{up}_j$ for all j, the following proposition is easily proven (relying on the definitions of $\hat{c}_{\hat{n}+1}$, low_j and up_j).

Proposition 7.2. *Assume (x, t) is an optimal solution for the LPP (7.2). If $t < 1$, then the LPP (7.1) is infeasible. If $t = 1$ and $\text{low}_j < x_j < \text{up}_j$ for all j, then (x, t) is an optimal solution for the LPP (7.1). If $t = 1$ and $x_j = \text{low}_j$ or $x_j = \text{up}_j$ for some j, then either (x, t) is an optimal solution for the LPP (7.1) or that LPP is unbounded.* □

For l a positive integer, let $k^{(0)} = -lK$. By definition of K, $k^{(0)}$ is a lower bound on the optimal objective value of the LPP (7.2). As in Section 2, let $(\bar{A}, b^{(0)})$ be the corresponding system of inequalities where $c \cdot (x, t) \geq k^{(0)}$ occurs l times.

Proposition 7.3. *The center of $(\bar{A}, b^{(0)})$ is 0.*

Proof. It is easily seen that $0 \in \text{Int}(\bar{A}, b^{(0)})$. Also, we know $\text{Int}(\bar{A}, b^{(0)})$ is bounded. Thus, to prove the proposition it suffices to show that the gradient of

$$f^{(0)}(x, t) = l \cdot \ln[c \cdot (x, t) - k^{(0)}] + \sum_{i=1}^{\hat{m}+2\hat{n}+3} \ln[\alpha_i \cdot (x, t) - b_i]$$

evaluated at 0 is 0, where α_i is the ith row of A.

Letting e^j denote the jth unit vector in $\mathbb{R}^{\hat{n}}$, the first $\hat{n}$ coordinates of $(\nabla_0 f^{(0)})$ are

$$\sum_{i=1}^{\hat{m}} \frac{1}{s^*}\alpha_i - \sum_{j\in\text{Pos}} \left(\frac{1+v_j}{s^*}\right)e^j - \sum_{j\notin\text{Pos}} \frac{1}{s^*}e^j$$

$$+ \sum_{j\in\text{Neg}} \left(\frac{1-v_j}{s^*}\right)e^j + \sum_{j\notin\text{Neg}} \frac{1}{s^*}e^j - \frac{1}{K}c + \frac{l}{lK}c$$

$$= \frac{1}{s^*}\left[v - \sum_{j\in\text{Pos}} v_j e^j - \sum_{j\in\text{Neg}} v_j e^j\right] = 0.$$

The last coordinate of $(\nabla_0 f^{(0)})$ is

$$\sum_{i=1}^{\hat{m}} \frac{-(s^*+\hat{b}_i)}{s^*} - \sum_{j\in\text{Pos}} \frac{s^*-(1+v_j)\,\text{up}_j}{s^*} - \sum_{j\notin\text{Pos}} \frac{s^*-\text{up}_j}{s^*} - \sum_{j\in\text{Neg}} \frac{s^*+(1-v_j)\,\text{low}_j}{s^*}$$

$$- \sum_{j\notin\text{Neg}} \frac{s^*+\text{low}_j}{s^*} - 1 + \frac{T(s^*)}{s^*} - \frac{c_{n+1}}{K} + \frac{lc_{n+1}}{lK}$$

$$= -\hat{m} - 2\hat{n} - 1 + \frac{1}{s^*}\left[-\sum_{i=1}^{\hat{m}} \hat{b}_i + \sum_{j=1}^{\hat{n}} (\text{up}_j - \text{low}_j) + \sum_{j\in\text{Pos}} v_j\,\text{up}_j\right.$$

$$\left. + \sum_{j\in\text{Neg}} v_j\,\text{low}_j\right] + \frac{T(s^*)}{s^*}$$

$$= 0$$

by definition of $T(s^*)$. Thus, the proposition. □

In closing this section we remark that if the algorithm is applied to finding an optimal solution of the LPP (7.2) and the resulting point (x, t) satisfies $t = 1$ and either $x_j = \text{low}_j$ or $x_j = \text{up}_j$ for some j, so that either x is an optimal solution for the LPP (7.1) or that LPP is unbounded (Proposition 7.2), one can determine which of these is indeed the case by replacing all low_j by $\text{low}_j - 1$ and all up_j by $\text{up}_j + 1$ and running the algorithm again. If the resulting point $(x', 1)$ satisfies $\hat{c}\cdot x' > \hat{c}\cdot x$, then the LPP (7.1) is unbounded, whereas if $\hat{c}\cdot x' = \hat{c}\cdot x$, then x is an optimal solution.

8. Complexity analysis

In this section we examine the computational complexity of the algorithm when applied to solving

$$\begin{aligned} \max \quad & c\cdot x \\ \text{s.t.} \quad & Ax \geq b, \end{aligned} \tag{8.1}$$

A being an $m \times n$ matrix and where the entries of A, b and c are integers. We will assume $\mathrm{Int}(A, b)$ is bounded and non-empty, and we will assume that we know an integer lower bound $k^{(0)}$ of k^{opt}. As in section two, we define

$$\bar{A} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \\ c \\ \vdots \\ c \end{bmatrix} \left.\vphantom{\begin{matrix} c \\ \vdots \\ c \end{matrix}}\right\} m \text{ times}, \qquad b^{(0)} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \\ k^{(0)} \\ \vdots \\ k^{(0)} \end{bmatrix}.$$

We restrict l to equal m in this section.

We will also assume that the center of $(\bar{A}, b^{(0)})$ is $\xi^{(0)} = 0$.

We will use L^* to denote a known (i.e. efficiently computed) positive integer such that 2^{L^*} is an upper bound on (i) the absolute values of the determinants of all square submatrices of $[A \mid b]$, (ii) the absolute values of the coordinates of c, and (iii) the quantity $n \cdot \log_2(m)$.

We will assume that the known lower bound $k^{(0)}$ of k^{opt} is an integer and that

$$1 \leq k^{\mathrm{opt}} - k^{(0)} \leq 2^{HL^*}, \tag{8.2}$$

H representing some constant independent of L^*. The lower bound of 1 on $k^{\mathrm{opt}} - k$ is assumed in order to simplify certain aspects of the analysis.

In the last section we saw that any LPP

$$\begin{aligned} \max \quad & \hat{c} \cdot x \\ \text{s.t.} \quad & \hat{A}x \geq \hat{b} \end{aligned} \tag{8.3}$$

where $\hat{A}$ is an $\hat{m} \times \hat{n}$ matrix and the entries of $\hat{A}$, $\hat{b}$ and $\hat{c}$ are integers, can be solved by solving a related LPP (8.1), with $m = \hat{m} + 2\hat{n} + 3$ and $n = \hat{n} + 1$, which satisfies the above assumptions, and where L^* of the related problem is $O(\hat{L})$, $\hat{L}$ being the number of bits required to represent (8.3). The constant H in (8.2) can then be determined independently of $\hat{L}$. Moreover, the number of arithmetic operations needed to construct the LPP (8.1) related to the LPP (8.3) is $O(\hat{m}\hat{n})$, and the number of bit operations required is $O(\hat{m}\hat{n}\hat{L}(\log \hat{L})(\log\log \hat{L}))$.

Now we describe the slightly modified algorithm that we will study in this section. The modifications involve rounding, a stopping criterion, and a method for moving to an optimal solution from an "almost" optimal solution.

Each step of the algorithm involves an unspecified positive integer constant, e.g., the first step of the algorithm rounds the current value $k^{(j)}$ upward to a fraction with denominator $2^{K_1 L^*}$, the value K_1 being the unspecified constant. By carefully proceeding through the subsequent analysis one could determine exact values for these constants, independent of L^*, that would suffice to guarantee that the algorithm returns an optimal solution, but I am sure that the values thus obtained would be too large to be of use in practice. Leaving the constants unspecified results in a shorter and more easily read analysis. (An earlier draft of this section, with constants specified, was unwieldy.)

Here is the modified version of the algorithm that we consider.

Initially: $x^{(0)} = \xi^{(0)} = 0$, $j = 1$.

Step 1: Let $k^{(j)}$ be the least fraction, with denominator $2^{K_1 L^*}$, greater than or equal to

$$\frac{1}{14\sqrt{m}}[c \cdot x^{(j-1)}] + \left(1 - \frac{1}{14\sqrt{m}}\right) k^{(j-1)}.$$

Step 2: For $i = 1, \ldots, m$, let $d_i^{(j)}$ be obtained by rounding $1/[\alpha_i \cdot x^{(j-1)} - b_i]$ to the nearest fraction with denominator $2^{K_2 L^*}$, and for $i = m+1, \ldots, 2m$, let $d_i^{(j)}$ be obtained by rounding $1/[c \cdot x^{(j-1)} - k^{(j)}]$ to the nearest fraction with denominator $2^{K_2 L^*}$. Let $D^{(j)}$ be the $2m \times 2m$ diagonal matrix with ith diagonal entry $d_i^{(j)}$. Compute the integer matrix and integer vector

$$B^{(j)} = 2^{2K_2 L^*} \bar{A}^{\mathrm{T}} (D^{(j)})^2 \bar{A}, \qquad b^{(j)} = 2^{2K_2 L^*} \bar{A}^{\mathrm{T}} D^{(j)} e.$$

Step 3: Obtain, using Cholesky decomposition, a vector $\eta^{(j)}$ of fractions with denominators $2^{K_3 L^*}$ where $\eta^{(j)}$ satisfies the condition that

$$\max_i |\eta_i^{(j)} - ([B^{(j)}]^{-1} b^{(j)})_i| \leq 1/2^{K_3 L^*},$$

(i.e., $\eta^{(j)}$ approximately solves $B^{(j)} v = b^{(j)}$). Let $x^{(j)} = x^{(j-1)} + \eta^{(j)}$.

Step 4: If $j < K_4 \lceil \sqrt{m} \rceil L^*$, let $j + 1 \mapsto j$ and return to Step 1.

Step 5: Let $S \subseteq \{1, \ldots, m\}$ denote the set of indices i for which $\alpha_i \cdot x^{(J)} - b_i \leq 2^{-K_5 L^*}$, where $J = K_4 \lceil \sqrt{m} \rceil L^*$. Then (as will be proven), the set $\{y;\ \alpha_i \cdot y = b_i$ for all $i \in S\}$ is non-empty. Compute the orthogonal projection of $x^{(J)}$ onto this set. This projection is an optimal solution for the LPP (8.1) (as will be proven).

Assuming the validity of the algorithm, let us examine its computational complexity. For each iteration $j < K_4 \lceil \sqrt{m} \rceil L^*$, the work in that iteration is dominated by approximately solving the system of linear equations with integer coefficients in Step 3, that is, $B^{(j)} v = b^{(j)}$. The coefficients in this system are bounded by $2^{\mathrm{O}(L^*)}$. This is a consequence of (i) the absolute value of the determinants of square submatrices of A being bounded by $2^{\mathrm{O}(L^*)}$; (ii) the fact (as will be proven) that $x^{(j-1)} \in \mathrm{Int}(\bar{A}, b^{(j)})$ from which it follows using (i) and Cramer's rule that the coordinates of $x^{(j-1)}$ are bounded by $2^{\mathrm{O}(L^*)}$; (iii) the coordinates of $x^{(j-1)}$ being fractions with *common* denominator $2^{K_3 L^*}$, and hence, using (ii), $\alpha_i \cdot x^{(j-1)} - b_i \geq 1/2^{\mathrm{O}(L^*)}$ and $c \cdot x^{(j-1)} - k^{(j)} \geq 1/2^{\mathrm{O}(L^*)}$; and (iv) the entries in $D^{(j)}$ being obtained by rounding the values $1/[\alpha_i \cdot x^{(j-1)} - b_i]$ and $1/[c \cdot x^{(j-1)} - k^{(j)}]$ to a fraction with denominator $2^{K_2 L^*}$—so assume K_2 is sufficiently large. It is now also easily seen that all entries of $2^{2K_2 L^*} \bar{A}^{\mathrm{T}} (D^{(j)})^2 \bar{A}$ and $2^{2K_2 L^*} \bar{A}^{\mathrm{T}} D^{(j)} e$ can be computed with $\mathrm{O}(mn^2)$ arithmetic operations on $\mathrm{O}(L^*)$ bit numbers, hence with $\mathrm{O}(mn^2 L^* (\log L^*)(\log \log L^*))$ bit operations.

As will be shown in Lemma 8.3, $\|(\nabla_x^2 f^{(j)})^{-1}v\| \leq 2^{O(L^*)}\|v\|$ for all v, where $x = x^{(j)}$. Here, the constants in $O(L^*)$ are dependent on K_1 and K_3. It follows (as in the proof of Lemma 8.4) that if K_2 is sufficiently large, then $B^{(j)}$ is invertible and $\|[B^{(j)}]^{-1}v\| \leq 2^{O(L^*)}\|v\|$ for all v (we have implicitly used $L^* \geq \log n$ here.) Using this and the $2^{O(L^*)}$ bounds on the absolute values of the coefficients in $B^{(j)}$ and $b^{(j)}$, Wilkinson's analysis of Cholesky decomposition [16; Section 44] shows that $\eta^{(j)}$ as in Step 3 can be obtained by $O(n^3)$ arithmetic operations where each arithmetic operation is carried out to $O(L^*)$ bits of accuracy. Since multiplication and division require $O(L^*(\log L^*)(\log\log L^*))$ bit operations to be carried out to accuracy $O(L^*)$ (e.g. see [1, Corollary to Theorem 8.5]), the total number of bit operations required in Step 3 is $O(n^3 L^*(\log L^*)(\log\log L^*))$.

The projection $P(x)$ of x in Step 5 of the algorithm can be computed by first determining a maximal linearly independent subset of $\{\alpha_i\}_{i\in S}$, letting $\tilde{A}$ be the matrix whose rows are this subset and $\tilde{b}$ the corresponding right-hand side vector, computing $(\tilde{A}\tilde{A}^T)^{-1}$ and finally $P(x) = x - \tilde{A}^T(\tilde{A}\tilde{A}^T)^{-1}(\tilde{A}x - \tilde{b})$. A maximal linearly independent subset of $\{\alpha_i\}_{i\in S}$ can be computed using Edmond's variation on Gaussian elimination (this is discussed in the appendix) with $O(mn^2)$ arithmetic operations on $O(L^*)$ bit numbers, hence with $O(mn^2L^*(\log L^*)(\log\log L^*))$ bit operations. By the Binet–Cauchy theorem (e.g. see [10]), the absolute value of the determinants of all square submatrices of $\tilde{A}\tilde{A}^T$ are bounded by $2^{O(L^*)}$. Hence, the inverse of $\tilde{A}\tilde{A}^T$ can be computed using Edmond's algorithm with $O(n^3)$ arithmetic operations on $O(L^*)$ bit numbers, hence with $O(n^3L^*(\log L^*)(\log\log L^*))$ bit operations. Finally, it is now easily seen that $O((m+n)n^2)$ arithmetic operations and $O((m+n)n^2L^*(\log L^*)(\log\log L^*))$ bit operations suffice to compute the projection in Step 5 of the algorithm.

Since $O(\sqrt{m}\,L^*)$ iterations of the algorithm are performed, overall the algorithm requires $O((m+n)n^2\sqrt{m}\,L^*)$ arithmetic operations and $O((m+n)\times n^2\sqrt{m}\,(L^*)^2(\log L^*)(\log\log L^*))$ bit operations. Consequently, using the results of the last section, any LPP of the form (8.3) can be solved (or determined infeasible, or determined unbounded) by the algorithm with $O((m+n)n^2\sqrt{m}\,\hat{L})$ arithmetic operations and $O((m+n)n^2\sqrt{m}(\hat{L})^2(\log\hat{L})(\log\log\hat{L}))$ bit operations.

Now we begin proving that the final point returned by the algorithm is an optimal solution for the LPP (8.1). We begin with four lemmas. The constants $K_1, \ldots, K_5$ of the algorithm appear in these lemmas, but the proofs of the lemmas are based only on the assumption that these are *positive integers* which satisfy whatever conditions are stated in the lemmas. The constant H of (8.2) appears likewise in the lemmas.

Besides $K_1, \ldots, K_5$ and H, additional *positive* constants $H_1, \ldots, H_6$ also appear in the lemmas. Such constants H_i appearing in the assumptions of a lemma can be taken to be any positive constants. The constants H_i appearing in the conclusions of a lemma are implied to exist, and specific values for them could be determined by a more lengthy analysis. Although constants H_i appearing in the conclusion of a lemma may depend on H, *they do not depend on* $K_1, \ldots, K_5$, a fact that the reader should keep in mind. Also, in the proofs of the lemmas we employ "$\mathrm{O}(L^*)$" notation frequently, where the constants in $\mathrm{O}(L^*)$ may depend on H and the "H_i's" (as will be clear from the context), *but will never depend on* $K_1, \ldots, K_5$.

The format in which the lemmas are presented was chosen to make for ease of application in the subsequent analysis.

Lemma 8.1. *Assume* $0 < k^{\mathrm{opt}} - k^{(j)} \leq 2^{H_1 L^* - K_4 H_2 L^*}$. *Assume* $x \in \mathrm{Int}(\bar{A}, b^{(j)})$ *satisfies* $\|X^{(j)} - e\| < \frac{1}{46}$. *Let S be the set of indices i for which* $\alpha_i \cdot x - b_i \leq 2^{-K_5 L^*}$. *If* K_4 *and* K_5 *are sufficiently large (where what constitutes sufficiently large for* K_4 *depends in part on* K_5, *but not vice versa), then the set* $\{y;\ \alpha_i \cdot y = b_i \text{ for all } i \in S\}$ *is non-empty, and the projection of x onto this set is an optimal solution for the LPP* (8.1).

Proof. Let $v^1, \ldots, v^Q$ denote the distinct extreme points of $\mathrm{Int}(A, b)$ and assume $v^1, \ldots, v^q$ are the optimal extreme points. Let T be the set of indices i for which $\alpha_i \cdot v^h = b_i$ for *all* of $h = 1, \ldots, q$. We begin by showing that if K_4 and K_5 are sufficiently large, then $T = S$, where S is as in the statement of the lemma.

By the representation theorem for linear programming, since $\mathrm{Int}(A, b)$ is bounded, there exist $\varepsilon_h \geq 0$, $h = 1, \ldots, Q$, where $\sum_{h=1}^{Q} \varepsilon_h = 1$ and $\sum_{h=1}^{Q} \varepsilon_h v^h = x$. Hence,

$$c \cdot x = \left(\sum_{h=1}^{q} \varepsilon_h\right) k^{\mathrm{opt}} + \sum_{h=q+1}^{Q} \varepsilon_h (c \cdot v^h),$$

from which, using $\sum_{h=1}^{q} \varepsilon_h = 1 - \sum_{h=q+1}^{Q} \varepsilon_h$, we find

$$k^{\mathrm{opt}} - c \cdot x = \sum_{h=q+1}^{Q} \varepsilon_h (k^{\mathrm{opt}} - c \cdot v^h). \tag{8.4}$$

However, it can be easily shown that $k^{\mathrm{opt}} - c \cdot v^h \geq 2^{-\mathrm{O}(L^*)}$ for $h = q+1, \ldots, Q$. Consequently, since $c \cdot x \geq k^{(j)}$, our assumed bound on $k^{\mathrm{opt}} - k^{(j)}$ and (8.4) together imply

$$\sum_{h=q+1}^{Q} \varepsilon_h \leq 2^{\mathrm{O}(L^*) - K_4 H_2 L^*}. \tag{8.5}$$

Now assume $i \in T$. Then

$$\alpha_i \cdot x - b_i = \sum_{h=1}^{Q} \varepsilon_h (\alpha_i \cdot v^h - b_i)$$

$$= \sum_{h=q+1}^{Q} \varepsilon_h (\alpha_i \cdot v^h - b_i).$$

Since $0 \leq \alpha_i \cdot v^h - b_i \leq 2^{\mathrm{O}(L^*)}$ for all h (as is easily shown), we thus have, using (8.5), that

$$\alpha_i \cdot x - b_i \leq 2^{\mathrm{O}(L^*) - K_4 H_2 L^*}.$$

Hence, if K_4 is sufficiently large (as determined in part by K_5), then $\alpha_i \cdot x - b_i \leq 2^{-K_5 L^*}$, proving $i \in S$.

Now assume $i \notin T$. For simplicity we may assume $\alpha_i \cdot v^1 \neq b_i$. Then, as is easily shown, $\alpha_i \cdot v^1 \geq b_i + 2^{-O(L^*)}$. In particular, since v^1 is an extreme point of $\text{Int}(\bar{A}, b^{(j)})$, we have

$$\sup_i \doteq \sup\{\alpha_i \cdot y;\, y \in \text{Int}(\bar{A}, b^{(j)})\} \geq b_i + 2^{-O(L^*)}. \tag{8.6}$$

However, repeating the proof of Proposition 3.5 verbatim, substituting α_i for c, $\sup_i$ for k^{opt}, b_i for $k^{(j)}$, and using the fact that $\alpha_i \cdot x \geq b_i$ may occur only once among the $2m$ inequalities in the system $(\bar{A}, b^{(j)})$ (rather than l times as was used for $c \cdot x - k^{(j)}$ in the proof of Proposition 3.5), one can show that for any $x \in \mathbb{R}^n$,

$$\alpha_i \cdot x - b_i \geq \frac{1}{2m}(1 - \|\boldsymbol{X}^{(j)}(x) - e\|)\left(\sup_i - b_i\right).$$

Hence, using (8.6) and our assumption that $\|\boldsymbol{X}^{(j)}(x) - e\| < \frac{1}{46}$, we have $\alpha_i \cdot x - b_i \geq 2^{-O(L^*)}$. Choosing K_5 sufficiently large then implies $\alpha_i \cdot x - b_i > 2^{-K_5 L^*}$, proving $i \notin S$.

It is easily shown, using the definition of T, that if y satisfies $Ay \geq b$ and $\alpha_i \cdot y = b_i$ for all $i \in T$, then y is an optimal point for the LPP (8.1). Hence, since $S = T$, to prove the lemma it suffices to show that the projection $P(x)$ of x onto the set $\{y;\, \alpha_i \cdot y = b_i \text{ for all } i \in S\}$ satisfies $AP(x) \geq b$.

Using (8.5) it is easily shown that

$$\left\| x - \sum_{h=1}^{q} \varepsilon_h v^h \right\| = \left\| \sum_{h=q+1}^{Q} \varepsilon_h v^h \right\| \leq \left(\sum_{h=q+1}^{Q} \varepsilon_h \right) \max_h \|v_h\| \leq 2^{O(L^*) - K_4 H_2 L^*},$$

from which it follows, since $\alpha_i \cdot (\sum_{h=1}^{q} \varepsilon_h v^h) - b_i = 0$ for $i \in S$ $(= T)$, that

$$\|x - P(x)\| \leq 2^{O(L^*) - K_4 H_2 L^*}.$$

Hence, for all $i = 1, \ldots, m$ we have

$$\alpha_i \cdot P(x) - b_i \geq \alpha_i \cdot x - b_i - 2^{O(L^*) - K_4 H_2 L^*}.$$

Since $\alpha_i \cdot x - b_i \geq 2^{-K_5 L^*}$ if $i \notin S$, it follows that if K_4 is sufficiently large and $i \notin S$, then $\alpha_i \cdot P(x) - b_i > 0$. On the other hand, of course $\alpha_i \cdot P(x) - b_i = 0$ if $i \in S$. In all, $AP(x) \geq b$. This completes the proof of the lemma. □

Lemma 8.2. *Assume that $k^{(j)}$ is a fraction, with denominator $2^{K_1 L^*}$, satisfying $0 < k^{\text{opt}} - k^{(j)} < 2^{HL^*}$. Then for any $x, y \in \mathbb{R}^n$,*

$$\|\boldsymbol{X}^{(j)}(x) - \boldsymbol{X}^{(j)}(y)\| \leq 2^{K_1 H_3 L^*} \|x - y\|$$

(*where H_3 is dependent on H*).

Proof. We will show that $\xi^{(j)}$, the center of $(\bar{A}, b^{(j)})$, satisfies

$$\alpha_i \cdot \xi^{(j)} - b_i \geq 1/2^{K_1 \cdot O(L^*)} \tag{8.7}$$

for all i, and

$$c \cdot \xi^{(j)} - k^{(j)} \geq 1/2^{K_1 \cdot O(L^*)}. \tag{8.8}$$

These inequalities immediately imply the lemma since

$$\|X^{(j)}(x)-X^{(j)}(y)\|^2=m\left[\frac{c\cdot(x-y)}{c\cdot\xi^{(j)}-k^{(j)}}\right]^2+\sum_{i=1}^{m}\left[\frac{\alpha_i\cdot(x-y)}{\alpha_i\cdot\xi^{(j)}-b_i}\right]^2.$$

To prove (8.7), it suffices to show that for each i there exists z satisfying $Az\geqslant b$, $c\cdot x\geqslant k^{(j)}$ and

$$\alpha_i\cdot z-b_i\geqslant 1/2^{K_1\cdot O(L^*)}, \tag{8.9}$$

because, by Proposition 3.1,

$$X^{(j)}(z)=\frac{\alpha_i\cdot z-b_i}{\alpha_i\cdot\xi^{(j)}-b_i}\leqslant 2m.$$

However, the existence of z satisfying (8.9) is an immediate consequence of $\operatorname{Int}(\bar{A}, b^{(j)})$ being non-empty and bounded, the fact that the absolute values of the determinants of all square submatrices of $[\bar{A}\,|\,b^{(j)})$ are fractions with numerators and denominators bounded by $2^{K_1\cdot O(L^*)}$, and Cramer's rule. (Choose z to be a vertex of $\operatorname{Int}(\bar{A}, b^{(j)})$ such that $\alpha_i\cdot z-b_i\neq 0$.)

The inequality (8.8) is proven similarly. □

Lemma 8.3. *Assume that $k^{(j)}$ is a fraction, with denominator $2^{K_1L^*}$, satisfying $0\leqslant k^{\text{opt}}-k^{(j)}\leqslant 2^{HL^*}$. Also assume that the coordinates of $x\in\operatorname{Int}(\bar{A}, b^{(j)})$ can be expressed as fractions with common denominator $2^{K_3L^*}$. Then, for all $v\in\mathbb{R}^n$,*

$$\|(\nabla_x^2 f^{(j)})^{-1}v\|\leqslant 2^{(K_1+K_3)H_4L^*}\|v\|$$

(where H_4 is dependent on H).

Proof. We first note that the determinant of every square submatrix of $\bar{A}^{\mathrm{T}}\bar{A}$ is bounded in absolute value by $m^n 2^{L^*}$, and hence, $2^{O(L^*)}$. This is a simple consequence of the Binet–Cauchy theorem. Thus, by Cramer's rule, each entry of $(\bar{A}^{\mathrm{T}}\bar{A})^{-1}$ is bounded in absolute value by $2^{O(L^*)}$. It follows that for any $w\in\mathbb{R}^n$ we have $w^{\mathrm{T}}\bar{A}^{\mathrm{T}}\bar{A}w\geqslant\|w\|^2/2^{O(L^*)}$, and hence, since $\bar{A}^{\mathrm{T}}\bar{A}=mc^{\mathrm{T}}c+\sum_{i=1}^{m}\alpha_i^{\mathrm{T}}\alpha_i$, for any $w\in\mathbb{R}^n$ there exists some α_i (or c) such that $w^{\mathrm{T}}\alpha_i^{\mathrm{T}}\alpha_i w\geqslant\|w\|^2/2^{O(L^*)}$ (or $w^{\mathrm{T}}c^{\mathrm{T}}cw\geqslant\|w\|^2/2^{O(L^*)}$). Consequently, since the assumption of the lemma implies $\alpha_i\cdot x-b_i\geqslant 1/2^{K_3\cdot O(L^*)}$ (and $c\cdot x-k^{(j)}\geqslant 1/2^{(K_1+K_3)O(L^*)}$), we must have for the same α_i (or for c) that $w^{\mathrm{T}}\alpha_i^{\mathrm{T}}\alpha_i w/(\alpha_i\cdot x-b_i)^2\geqslant\|w\|^2/2^{K_3\cdot O(L^*)}$ (or $w^{\mathrm{T}}c^{\mathrm{T}}cw/(c\cdot x-k^{(j)})^2\geqslant\|w\|^2/2^{(K_1+K_3)O(L^*)}$). So $w^{\mathrm{T}}(\nabla_x^2 f^{(j)})w\geqslant\|w\|^2/2^{(K_1+K_3)O(L^*)}$ for any $w\in\mathbb{R}^n$, from which it follows that $\|(\nabla_x^2 f^{(j)})w\|\geqslant\|w\|/2^{(K_1+K_3)O(L^*)}$, and thus $\|(\nabla_x^2 f^{(j)})^{-1}v\|\leqslant 2^{(K_1+K_3)O(L^*)}\|v\|$ for all $v\in\mathbb{R}^n$. □

Lemma 8.4. *Assume the same assumptions as in Lemma 8.3 for $k^{(j)}$ and x. Let $D^{(j)}$ be obtained as in Step 2 of the algorithm by rounding the quantities $1/[\alpha_i\cdot x-b_i]$ and $1/[c\cdot x-k^{(j)}]$ to fractions with denominator $2^{K_2L^*}$. If K_2 is sufficiently large, then*

$\bar{A}^{\mathrm{T}}(D^{(j)})^2\bar{A}$ is invertible. Moreover, letting $\eta^{(j)}$ be as in Step (3) of the algorithm and letting $n_x^{(j)}$ satisfy

$$(\nabla_x^2 f^{(j)})n_x^{(j)} = -(\nabla_x f^{(j)})^{\mathrm{T}},$$

we have

$$\|\eta^{(j)} - n_x^{(j)}\| \leq 2^{(K_1+K_3)H_5L^* - K_2H_6L^*} + \sqrt{n}\, 2^{-2K_3L^*}$$

(*where H_5 and H_6 depend on H*).

Proof. To ease notation, let

$$M = \nabla_x^2 f^{(j)}, \qquad \mathcal{M} = -\bar{A}^{\mathrm{T}}(D^{(j)})^2\bar{A} = -2^{-2K_2L^*}B^{(j)}.$$

It is easily verified that for all $v \in \mathbb{R}^n$,

$$\|(\mathcal{M} - M)v\| \leq 2^{\mathrm{O}(L^*) - K_2H_7L^*}\|v\|$$

where H_7 is a positive constant dependent on H. This, together with Lemma 8.3, implies that for all $v \in \mathbb{R}^n$,

$$\|M^{-1}(\mathcal{M} - M)v\| \leq 2^{(K_1+K_3)\mathrm{O}(L^*) - K_2H_7L^*}\|v\|. \tag{8.10}$$

Hence, if K_2 is sufficiently large, the matrix limit $\sum_{i=0}^{\infty}(-1)^i[M^{-1}(\mathcal{M}-M)]^i$ exists and is the inverse of $I + M^{-1}(\mathcal{M} - M) = M^{-1}\mathcal{M}$. Consequently, $\mathcal{M}$ is invertible and

$$\mathcal{M}^{-1} = \left(\sum_{i=0}^{\infty}(-1)^i[M^{-1}(\mathcal{M}-M)]^i\right)M^{-1}$$

$$= M^{-1} + \left(\sum_{i=1}^{\infty}(-1)^i[M^{-1}(\mathcal{M}-M)]^i\right]M^{-1}.$$

Thus, (8.10) and Lemma 8.3 imply that if K_2 is sufficiently large, then for any $v \in \mathbb{R}^n$,

$$\|(M^{-1} - \mathcal{M}^{-1})v\| \leq 2^{(K_1+K_3)\mathrm{O}(L^*) - K_2H_7L^*}\|v\|. \tag{8.11}$$

Also, as can be easily verified,

$$\|(\nabla_x f^{(j)})^{\mathrm{T}} - \bar{A}^{\mathrm{T}}D^{(j)}e\| \leq 2^{\mathrm{O}(L^*) - K_2H_8L^*}, \tag{8.12}$$

where H_8 is a positive constant dependent on H. Letting $N^{(j)}$ be the exact solution to $B^{(j)}w = b^{(j)}$, the lemma follows from (8.11), (8.12), Lemma 8.3 and the inequality

$$\|n_x^{(j)} - \eta_x^{(j)}\| \leq \|n_x^{(j)} - N^{(j)}\| + \|N^{(j)} - \eta^{(j)}\|$$

$$\leq \|(M^{-1} - \mathcal{M}^{-1})\tilde{A}^{\mathrm{T}}D^{(j)}e\| + \|M^{-1}[(\nabla_x f^{(j)})^{\mathrm{T}} - \bar{A}^{\mathrm{T}}D^{(j)}e]\|$$

$$+ \|N^{(j)} - \eta^{(j)}\|. \qquad \square$$

We are finally ready to prove that the algorithm is well-defined and that the final point returned by the algorithm is an optimal solution for the LPP (8.1). We will show that for any fixed positive integer K_4 we can choose K_1, K_2 and K_3 such that for any $j = 0, 1, \ldots, K_4\lceil\sqrt{m}\rceil L^*$ the following inequalities hold;

$$\|\boldsymbol{X}^{(j)}(x^{(j)}) - e\| < \tfrac{1}{46}, \tag{8.13}$$

$$k^{\text{opt}} - k^{(j)} \leq \left[1 - \frac{1}{29\sqrt{m}}\right]^j (k^{\text{opt}} - k^{(0)}) \tag{8.14}$$

and

$$k^{\text{opt}} - k^{(j)} \geq \left[1 - \frac{1}{13\sqrt{m}}\right]^j. \tag{8.15}$$

The most important of the above inequalities is (8.14), although (8.13) does imply that the algorithm is well-defined (i.e. can be used to guarantee that $x^{(j)} \in \text{Int}(\bar{A}, b^{(j+1)})$ so that the algorithm can continue). Inequalities (8.13) and (8.15) will be used in the inductive proof of (8.14).

Letting $H_1 = H$, where H is the constant in (8.2), and letting $H_2 > 0$ satisfy

$$\left[1 - \frac{1}{29\sqrt{m}}\right]^{\lceil\sqrt{m}\rceil} \leq 2^{-H_2} \quad \text{for all } m = 1, 2, \ldots,$$

(such an H_2 exists since $\lim_{t\to\infty}(1 - 1/t)^t = e^{-1}$), note that (8.14) implies for $J = K_4\lceil\sqrt{m}\rceil L^*$ that

$$k^{\text{opt}} - k^{(J)} \leq 2^{H_1 L^* - K_4 H_2 L^*}.$$

Hence, assuming the validity of (8.14) and choosing K_4 and K_5 sufficiently large, Proposition 8.1 shows the final point returned by the algorithm is an optimal solution for the LPP (8.1).

Now we establish the inequalities (8.13)-(8.15) inductively, considering K_4 and K_5 as being fixed. Of course (8.14) holds when $j = 0$, as does (8.13) since $x^{(0)} = \xi^{(0)} = 0$. Also, (8.15) holds when $j = 0$ by assumption (8.2).

Choose K_1, sufficiently large (independently of L^*) so that

$$\frac{1}{13\sqrt{m}} - \frac{1}{14\sqrt{m}} \geq \frac{92}{45}\left(1 - \frac{1}{13\sqrt{m}}\right)^{-K_4\lceil m\rceil L^*} 2^{-K_1 L^*}. \tag{8.16}$$

This can be accomplished since $(1 - 1/13\sqrt{m})^{-\lceil m\rceil}$ is bounded above independently of m and since $L^* > \log_2 m$.

Next, choose K_3 sufficiently large so that

$$\sqrt{n}\, 2^{K_1 H_3 L^* - K_3 L^*} < \tfrac{1}{2}\left(\tfrac{1}{46} - \tfrac{1000}{46656}\right), \tag{8.17}$$

where H_3 is the constant in Lemma 8.2.

Finally, choose K_2 sufficiently large so that

$$2^{K_1 H_3 L^* + (K_1 + K_3) H_5 L^* - K_2 H_6 L^*} < \tfrac{1}{2}\left(\tfrac{1}{46} - \tfrac{1000}{46656}\right), \tag{8.18}$$

where H_5 and H_6 are as in Lemma 8.4.

For the above values of K_1, K_2 and K_3, we show that if (8.13)–(8.15) are valid for $j < K_4\lceil\sqrt{m}\rceil L^*$, then they are also valid if j is replaced by $j+1$.

Let

$$\delta^{(j+1)} = \frac{k^{(j+1)} - k^{(j)}}{c \cdot x^{(j)} - k^{(j)}}, \tag{8.19}$$

and

$$\begin{aligned}\tilde{k}^{(j+1)} &= \frac{1}{14\sqrt{m}}[c \cdot x^{(j)}] + \left(1 - \frac{1}{14\sqrt{m}}\right)k^{(j)} \\ &= \frac{1}{14\sqrt{m}}[c \cdot x^{(j)} - k^{(j)}] + k^{(j)}.\end{aligned} \tag{8.20}$$

Since $k^{(j+1)}$ is obtained by rounding $\tilde{k}^{(j+1)}$ upwards to a fraction with denominator $2^{K_1 L^*}$, it follows from (8.19) and (8.20) that

$$\frac{1}{14\sqrt{m}} \leq \delta^{(j+1)} \leq \frac{1}{14\sqrt{m}} + \frac{1}{[c \cdot x^{(j)} - k^{(j)}]2^{K_1 L^*}}. \tag{8.21}$$

However, Proposition 3.5, (8.13), (8.15) and $j < K_4\lceil\sqrt{m}\rceil L^*$ imply

$$\begin{aligned}c \cdot x^{(j)} - k^{(j)} &\geq \tfrac{45}{92}(k^{\text{opt}} - k^{(j)}) \\ &\geq \tfrac{45}{92}\left(1 - \frac{1}{13\sqrt{m}}\right)^{K_4\lceil\sqrt{m}\rceil L^*}.\end{aligned} \tag{8.22}$$

Together, (8.16), (8.21) and (8.22) imply

$$\frac{1}{14\sqrt{m}} \leq \delta^{(j+1)} \leq \frac{1}{13\sqrt{m}}. \tag{8.23}$$

We will use these inequalities in several ways.

Using the definition (8.19), the fact that $c \cdot x^{(j)} \leq k^{\text{opt}}$, the upper bound given by (8.23), and the inductive assumption (8.15), we have

$$\begin{aligned}k^{\text{opt}} - k^{(j+1)} &= k^{\text{opt}} - k^{(j)} - [k^{(j+1)} + k^{(j)}] \\ &= k^{\text{opt}} - k^{(j)} - \delta^{(j+1)}[c \cdot x^{(j)} - k^{(j)}] \\ &\geq (1 - \delta^{(j+1)})(k^{\text{opt}} - k^{(j)}) \\ &\geq \left(1 - \frac{1}{13\sqrt{m}}\right)^{j+1},\end{aligned}$$

establishing the inductive assumption (8.15).

On the other hand, since Proposition 3.5 and (8.13) imply $c \cdot x^{(j)} - k^{(j)} \geq \frac{45}{92}(k^{\text{opt}} - k^{(j)})$, using (8.13) and the lower bound in (8.23) we have

$$\begin{aligned}k^{\text{opt}} - k^{(j+1)} &= k^{\text{opt}} - k^{(j)} - \delta^{(j+1)}[c \cdot x^{(j)} - k^{(j)}] \\ &\leq \left(1 - \frac{45\delta^{(j+1)}}{92}\right)(k^{\text{opt}} - k^{(j)})\end{aligned}$$

$$\leqslant \left(1-\frac{1}{29\sqrt{m}}\right)^{j+1}(k^{\text{opt}}-k^{(0)}),$$

establishing the inductive assumption (8.14).

Together, (8.13), the upper bound in (8.23) and Corollary 3.4 (using $\delta=\delta^{(j+1)}$) show $\|X^{(j+1)}(x^{(j)})-e\|<\frac{1}{9}$. Consequently, Theorem 3.2 shows that one iteration of Newton's method, applied to finding the center of $(\bar{A}, b^{(j+1)})$, gives a point $\tilde{y}$ satisfying $\|X^{(j+1)}(\tilde{y})-e\|<\frac{1000}{46656}$. Hence, Lemmas 8.2 and 8.4 show that the vector $x^{(j+1)}=x^{(j)}+\eta^{(j)}$ computed in Step 3 of the algorithm satisfies

$$\|X^{(j+1)}(x^{(j+1)})-e\|<\tfrac{1000}{46656}+2^{K_1H_3L^*+(K_1+K_3)H_5L^*-K_2H_6L^*}+\sqrt{n}\,2^{K_1H_3L^*-K_3L^*}.$$

Together with (8.17) and (8.18), this establishes the inductive assumption (8.13).

This concludes the complexity analysis.

9. Appendix

The results in this section are certainly "common knowledge", but I do not know of a reference where they are carefully proven.

Here we describe the variation of Gaussian elimination due to Edmonds [8]. Actually, what we describe is itself a slight variation of Edmonds' work. Edmonds' motivation was to compute the rank of a matrix efficiently. But, with only a little care, his algorithm can be used to solve linear equations efficiently.

Assume we wish to solve a system of equations $Mx=y$, where M is an $n\times n$ integer matrix (we assume M is a square matrix for simplicity) and y is an integer vector. As in standard Gaussian elimination, but with a slight modification to be described, through row operations we convert $[M\,|\,y]$ to a matrix ready for back substitution. We will also assume, for simplicity, that M is of full rank.

Assume that at the present stage the algorithm has converted $[M\,|\,y]$ to the matrix $[m'_{ij}]$, where $m'_{ij}=0$ if $i>j$, $j<j_0\leqslant n$, and where $m'_{ij}\neq 0$ if $i=j$, $j\leqslant j_0$. (This includes the simplifying assumption that $m_{j_0,j_0}\neq 0$-row permutations might be required.) Now we want to pivot on m'_{j_0,j_0} to obtain a new matrix with all zeros under the (j_0,j_0) entry.

Just as in standard Gaussian elimination, to obtain a zero in the (i,j_0) entry, where $i>j_0$, multiply the ith row of $[m'_{ij}]$ by m'_{j_0,j_0} and subtract from the resulting vector the vector obtained by multiplying the j_0th row by m'_{ij_0}. *Then divide the difference by* m'_{j_0-1,j_0-1} (defining $m'_{j_0-1,j_0-1}=1$ if $j_0=0$). Doing this for each row $i>j_0$, we obtain a new matrix $[m''_{ij}]$ all of whose entries (as Edmonds shows) are determinants of square submatrices of $[M\,|\,y]$, in particular, they are integers with apriori bounds on their bit lengths. If one does not perform the above division, then examples can be constructed where the bit lengths of the resulting entries "blow-up".

Proceeding with the above algorithm, we obtain a matrix, say $[\hat{M}\,|\,\hat{y}]$, ready for back substitution. Now we need to show that the back substitutions can be accomplished without intermediate fractions occurring whose numerators or denominators are "too" large. Although by Cramer's rule, one can show that in *reduced form* the numerators and denominators can be bounded in terms of the determinants of $[M|y]$, blind back substitution does not necessarily produce fractions in reduced form. Moreover, if at some stage of the back substitution the coordinates (of the solution) obtained thus far do not have a common denominator, then clearing the denominators in the next equation to be solved results in the next coordinate being expressed as a fraction with even larger denominator. If the fractions are not somehow reduced (which adds to the complexity), the resulting bit lengths can grow exponentially with the number of coordinates solved for thus far.

Let $\mathscr{D}$ denote the determinant of M. We can efficiently determine the coordinates of the solution to $Mx = y$ through back substitution in $[\hat{M}|\hat{y}]$ by making use of the fact that all coordinates of the solution can be expressed as fractions with *common* denominator $\mathscr{D}$.

Edmonds shows the (n, n) entry in $[\hat{M}|\hat{y}]$ is precisely $\mathscr{D}$. Thus, assuming $Mx = y$, we have $x_n = \hat{y}_n/\mathscr{D}$. Substituting this for x_n in the next to last equation, substituting $z_{n-1}/\mathscr{D}$ for x_{n-1}, clearing the common denominator $\mathscr{D}$ and solving for z_{n-1} algebraically (the division by $\hat{m}_{n-1,n-1}$ that occurs will be exact), we are able to obtain x_{n-1} as a fraction with denominator $\mathscr{D}$, where the bit lengths of the intermediate numbers occurring during the solution process remain "nicely" bounded. Proceeding through all the variables in this manner (i.e., substituting $x_i = z_i/\mathscr{D}$), we find that the back substitution process can be carried out in $O(mn)$ arithmetic operations on numbers whose bit lengths are of order equal to the bit lengths of the determinants of square submatrices of $[M|y]$, plus $\log_2 m$ (to allow for addition of m numbers).

This concludes our discussion of the bit complexity of solving linear equations.

References

[1] A. Aho, J. Hopcroft and J. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Reading, MA, 1974).

[2] D.A. Bayer and J.C. Lagarias, "The non-linear geometry of linear programming, I. Affine and projective scaling trajectories, II. Legendre transform coordinates, III. Central trajectories," preprints, AT&T Bell Laboratories (Murray Hill, NJ, 1986).

[3] L. Blum, talk at Workshop on Problems Relating Numerical Analysis to Computer Science, Mathematical Sciences Research Institute, Berkeley, California (January 1986).

[4] L. Blum, "Towards an asymptotic analysis of Karmarkar's algorithm," *Information Processing Letters* 23 (1986) 189–194.

[5] P. Huard, "Resolution of mathematical programming with non-linear constraints by the method of centers," in: J. Abadie, ed., *Non-Linear Programming* (North-Holland, Amsterdam, 1967) pp. 207–219.

[6] N. Karmarkar, "A new polynomial-time algorithm for linear programming," in: *Proceedings of the 16th Annual ACM Symposium on Theory of Computing* (1984), ACM, New York, 1984, pp. 302–311; revised version: *Combinatorica* 4 (1984) pp. 373–395.

[7] L.G. Khachiyan, "A polynomial algorithm in linear programming," *Soviet Mathematics Doklady* 20 (1979) pp. 191–194.

[8] L.G. Khachiyan, "Polynomial algorithms in linear programming," *USSR Computational Mathematics and Mathematical Physics* 20 (1980) pp. 53–72.

[9] J. Lagarias, talk at Mathematical Sciences Research Institute (Berkeley, California, December, 1985).

[10] N. Megiddo and M. Shub, "Boundary behavior of interior point algorithms in linear programming," Research Report RJ5319, IBM Thomas J. Watson Research Center (1986).

[11] S. Smale, "On the efficiency of algorithms of analysis," *Bulletin of the American Mathematical Society* 13 (1985) pp. 87–121.

[12] S. Smale, "Algorithms for solving equations," to appear in: *Proceedings, International Congress of Mathematicians* (Berkeley, 1986).

[13] Gy. Sonnevend, "A new method for solving a set of linear (convex) inequalities and its applications for identification and optimization," preprint, Department of Numerical Analysis, Institute of Mathematics, Eötvös University, 1088, Budapest, Muzeum Körut 6–8.

[14] Gy. Sonnevend, "An analytical centre' for polyhedrons and new classes of global algorithms for linear (smooth convex) programming," preprint, Department of Numerical Analysis, Institute of Mathematics, Eötvös University, 1088, Budapest, Muzeum Körut 6–8.

[15] P. Vaidya, "An algorithm for linear programming which requires $O((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations," AT&T Bell Laboratories, Murray Hill, NJ (1987).

[16] J.H. Wilkinson, *The algebraic Eigenvalue Problem* (Oxford University Press, Oxford, 1965).