An Oblivious Ellipsoid Algorithm for Solving a System of (In)Feasible Linear Inequalities[*]

Jourdain Lamperski[†] Robert M. Freund[‡] Michael J. Todd[§]

December 23, 2020

Abstract

The ellipsoid algorithm is a fundamental algorithm for computing a solution to the system of m linear inequalities in n variables $(P) : A^\top x \leq u$ when its set of solutions has positive volume. However, when (P) is infeasible, the ellipsoid algorithm has no mechanism for proving that (P) is infeasible. This is in contrast to the other two fundamental algorithms for tackling (P), namely the simplex method and interior-point methods, each of which can be easily implemented in a way that either produces a solution of (P) or proves that (P) is infeasible by producing a solution to the alternative system $(Alt) : A\lambda = 0,\ u^\top \lambda < 0,\ \lambda \geq 0$. This paper develops an Oblivious Ellipsoid Algorithm (OEA) that either produces a solution of (P) or produces a solution of (Alt). Depending on the dimensions and on other natural condition measures, the computational complexity of the basic OEA may be worse than, the same as, or better than that of the standard ellipsoid algorithm. We also present two modified versions of OEA, whose computational complexity is superior to that of OEA when $n \ll m$. This is achieved in the first modified version by proving infeasibility without actually producing a solution of (Alt), and in the second modified version by using more memory.

1 Introduction, preliminaries, and summary of results

Given data $(A, u) \in \mathbb{R}^{n \times m} \times \mathbb{R}^m$, the ellipsoid algorithm is a fundamental algorithm for computing a solution to the system of linear inequalities

$$(P) : \quad A^\top x \leq u$$

[*] The authors thank the associate editor and two anonymous referees for their diligent efforts and for their constructive and very helpful comments.

[†] MIT Operations Research Center, 77 Massachusetts Avenue, Cambridge, MA 02139 (mailto: jourdain@mit.edu). This author's research is supported by AFOSR Grant No. FA9550-19-1-0240.

[‡] MIT Sloan School of Management, 77 Massachusetts Avenue, Cambridge, MA 02139 (mailto: rfreund@mit.edu). This author's research is supported by AFOSR Grant No. FA9550-19-1-0240.

[§] Cornell University, mjt7@cornell.edu when the set of solutions $\mathcal{P} := \{x \in \mathbb{R}^n : A^\top x \le u\}$ has positive volume. However, when (P) is infeasible, existing versions of the ellipsoid algorithm have no mechanism for deciding if (P) is infeasible. (We use the real number model of computation throughout this paper. In the bit model of computation the ellipsoid method will correctly decide infeasibility even though it will not produce a solution of a dual/alternative system – instead a volume argument is used to prove infeasibility; see [6].) By a *certificate of infeasibility* we informally mean a mathematical object that yields a proof that (P) is infeasible. For example, when and only when (P) is infeasible, there exists a solution $\lambda \in \mathbb{R}^m$ to the alternative system (Alt) below, which we formally call a *type-L* certificate of infeasibility:

Type-L Certificate of Infeasibility. If $\lambda \in \mathbb{R}^m$ satisfies:

$$(Alt): \quad \begin{cases} A\lambda &= & 0 \\ \lambda &\ge & 0 \\ u^\top \lambda &< & 0 \ , \end{cases}$$

then it is simple to demonstrate that (P) is infeasible. We refer to a solution to (Alt) as a *type-L* certificate of infeasibility, where L stands for *linear* because the certificate is identified with a linear inequality system, and in order to distinguish it from two other types of certificates of infeasibility for (P) that will be developed herein. We view a type-L certificate of infeasibility as special because – like a solution to (P) – it is a solution to a particular linear inequality system (namely (Alt)), it does not require excessive storage (m coefficients), and the computation involved in verifying (Alt) is not excessive ($O(mn)$ operations).

The two other fundamental algorithms for tackling (P), namely the simplex algorithm and interior-point methods, each can be implemented in a way that either produces a solution to (P) or certifies that (P) is infeasible by producing a type-L certificate of infeasibility. This has begged the question of whether such a version of the ellipsoid method can be developed [14], that is, can one develop an *oblivious ellipsoid algorithm* that produces a solution to (P) or (Alt), *without knowing* which system is feasible? Accordingly, we consider the following two challenges, the *oblivious linear certification challenge* and the *oblivious determination challenge*:

Challenge I (Oblivious Linear Certification). Develop a version of the ellipsoid algorithm that produces a feasible solution of (P) when (P) is feasible, and produces a type-L certificate of infeasibility, i.e., a solution of (Alt), when (P) is infeasible.

Challenge II (Oblivious Determination). Develop a version of the ellipsoid algorithm that produces a feasible solution of (P) when (P) is feasible, and proves that (P) is infeasible when (P) is infeasible.

When (P) is feasible, both Challenges I and II require producing a solution of (P). But when (P) is not feasible, Challenge I requires producing a type-L certificate of infeasibility, whereas Challenge II only requires proving infeasibility – though not necessarily producing a type-L certificate. It follows that any resolution of Challenge I is also a resolution of Challenge II.

Of course, one could address Challenge I or Challenge II by running the standard ellipsoid method in parallel simultaneously on (P) and (Alt). That is, one could perform (one arithmetic operation at a time) one operation of the ellipsoid algorithm applied to (P) followed by one operation of the ellipsoid algorithm applied to (Alt), and then stop when one of the two algorithms produces a solution. (Equivalently, one could run each algorithm on a separate machine.) However, there is an aesthetic interest in developing a single oblivious ellipsoid algorithm (which we call OEA) that will either produce a solution of (P) or prove that (P) is infeasible by producing a solution of (Alt). Such a version would elevate the ellipsoid algorithm to be "on par" with the other two fundamental algorithms for solving (P) in this regard, namely the simplex method and interior-point methods.

Before presenting a schematic of OEA and stating our main results, we first need to develop some relevant concepts and related notation. We will make the following assumption about the data throughout this paper:

Assumption 1.1. The conic hull of the columns of A is equal to $\mathbb{R}^n$, namely $\{A\lambda : \lambda \geq 0\} = \mathbb{R}^n$, and each of the columns $a_1, ..., a_m$ of A has unit Euclidean norm.

The first part of Assumption 1.1 ensures that (P) is bounded if (P) is feasible. Note that Assumption 1.1 implies that $m > n$ and that A has rank n. The second part of Assumption 1.1 is without loss of generality because feasible solutions of (P) do not change under positive rescaling of the constraints of (P), and any zero a_j's either yield redundant constraints or immediate proofs of infeasibility.

We will suppose that for each $i \in \{1, ..., m\}$ we know a *lower bound* $\ell_i \in \mathbb{R}$ that satisfies $x \in \mathcal{P} \Rightarrow a_i^\top x \geq \ell_i$. (When (P) is infeasible, any $\ell_i \in \mathbb{R}$ satisfies this implication vacuously.) Accordingly, if (P) is feasible and we know lower bounds $\ell_1, \ldots, \ell_m$, then we know how to bound $a_i^\top x$ for $x \in \mathcal{P}$ since u_i is an upper bound for $a_i^\top x$ over all $x \in \mathcal{P}$); see Figure 1.

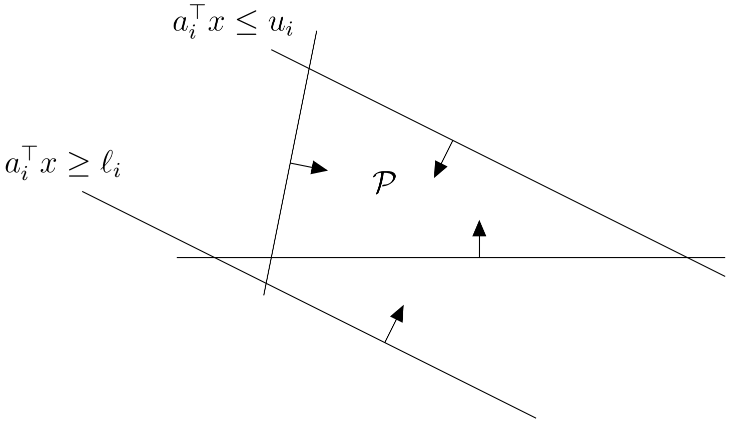

Figure 1: A lower bound ℓ_i on constraint i of (P).

In fact, we will suppose more strongly that for each $i \in \{1, ..., m\}$ we know ℓ_i and $\lambda_i \in \mathbb{R}^m$

that satisfy:

$$(LB_i): \quad \begin{cases} A\lambda_i = -a_i \\ \lambda_i \geq 0 \\ -\lambda_i^\top u \geq \ell_i \ , \end{cases}$$

and observe that when (P) is feasible, it follows from (LB_i) that any $x \in \mathcal{P}$ satisfies

$$a_i^\top x = -\lambda_i^\top A^\top x \geq -\lambda_i^\top u \geq \ell_i \ ,$$

i.e., λ_i certifies the lower bound ℓ_i on $a_i^\top x$ over all $x \in \mathcal{P}$. We will define ℓ_i to be a *certified lower bound for constraint i of (P) with certificate* $\lambda_i \in \mathbb{R}^m$ if ℓ_i and λ_i together satisfy (LB_i).

It will be convenient to collect the certified lower bounds into $\ell = (\ell_1, \ldots, \ell_m)^\top \in \mathbb{R}^m$ and their certificates columnwise into a matrix $\Lambda = [\lambda_1 | \cdots | \lambda_m] \in \mathbb{R}^{m \times m}$, and define ℓ to be a *certified lower bound for (P) with certificate matrix* $\Lambda \in \mathbb{R}^{m \times m}$ if ℓ and Λ satisfy

$$(LB): \quad \begin{cases} A\Lambda = -A \\ \Lambda \geq 0 \\ -\Lambda^\top u \geq \ell \ , \end{cases}$$

where the matrix inequalities $\Lambda \geq 0$ are considered entry-wise. Just as above, if (P) is feasible and ℓ is a certified lower bound for (P) with certificate Λ, then ℓ is a lower bound for $A^\top x$ over all $x \in \mathcal{P}$ because for any $x \in \mathcal{P}$ it holds that

$$A^\top x = -\Lambda^\top A^\top x \geq -\Lambda^\top u \geq \ell \ .$$

In general, it is not such an easy task to construct such lower bounds ℓ and certificates Λ – short of solving systems of inequalities of size at least as large as that of (P). However, in the often-occurring case when (P) contains box constraints (of the form $\underline{b} \leq x \leq \bar{b}$), such lower bounds and certificates are quite simple to write down, which we show in Section 2. (Recall that if (P) is infeasible, then any ℓ_i is a lower bound. In theory, we can find a solution to (LB_i) by obtaining a nonnegative solution to $A\hat{\lambda} = -a_i$ by Assumption 1.1 and then adding to it a suitably large multiple of a solution to (Alt).)

We can use a certified lower bound ℓ together with an arbitrary given $d \in \mathbb{R}^m$ satisfying $d > 0$ to construct a parametrized ellipsoid $E(d, \ell)$ that contains $\mathcal{P}$:

$$\mathcal{P} \subseteq E(d, \ell) := \left\{ x \in \mathbb{R}^n : (A^\top x - \ell)^\top D (A^\top x - u) \leq 0 \right\} \ , \tag{1}$$

(where $D := \operatorname{diag}(d)$ is the diagonal matrix with diagonal d), since $x \in \mathcal{P} \Rightarrow (a_i^\top x - u_i) d_i (a_i^\top x - \ell_i) \leq 0$ for all $i = 1, \ldots, m$. Using some elementary algebraic manipulation, we can re-write $E(d, \ell)$ as:

$$E(d, \ell) = \left\{ x \in \mathbb{R}^n : (x - y(d, \ell))^\top ADA^\top (x - y(d, \ell)) \leq f(d, \ell) \right\} \ , \tag{2}$$

where

$$\begin{aligned} y(d, \ell) &:= \tfrac{1}{2}(ADA^\top)^{-1} AD(u + \ell) \ , \\ f(d, \ell) &:= \tfrac{1}{4}(u + \ell)^\top DA^\top (ADA^\top)^{-1} AD(u + \ell) - \ell^\top D u \ , \end{aligned}$$

and we see from (2) that $y(d,\ell)$ is the center of the ellipsoid $E(d,\ell)$, $ADA^\top$ is the so-called shape matrix, and $\sqrt{f(d,\ell)}$ captures the scale factor of the ellipsoid.

The representation (1)-(2) was introduced by Burrell and Todd [2] to generate dual variables in the ellipsoid method. They developed a variant of the standard ellipsoid method with deep cuts that represented each ellipsoid in the form $E(d,\ell)$ (with $d \geq 0$, not necessarily positive); the difference was that sometimes it was necessary to update the lower bounds before applying the standard deep cut update.

In the Oblivious Ellipsoid Algorithm that we develop in this paper, we will also update the ellipsoid $E(d,\ell)$ by updating its parameters $(d,\ell) \rightarrow (\tilde{d},\tilde{\ell})$ (as opposed to explicitly updating the center and shape matrix as is done in the conventional ellipsoid algorithm). Hence ℓ (and its certification matrix Λ) should be thought of as parameters that are given an initial value and then are updated in the course of running the algorithm. We will also maintain d positive throughout.

Our Oblivious Ellipsoid Algorithm will update ℓ in synch with updates of Λ so that the updated ℓ is always certified by the updated Λ. For motivation why OEA updates ℓ and Λ, suppose at a given iteration we have ℓ that is certified by Λ and it holds that ℓ_j satisfies $\ell_j > u_j$ for some $j \in \{1,...,m\}$ (so that clearly (P) is infeasible). Then it is straightforward to verify (see Burrell and Todd [2], and also Corollary 5.1 here) that $\bar{\lambda}_j := \lambda_j + e_j$ is feasible for (Alt) and so is a type-L certificate of infeasibility. This will be our main method for constructing a type-L certificate of infeasibility in our algorithm, so we state this result formally as follows.

Remark 1.1. Suppose ℓ_j is a certified lower bound for inequality j with certificate λ_j, and that $\ell_j > u_j$. Then (P) is infeasible, and $\bar{\lambda}_j := \lambda_j + e_j$ is feasible for (Alt) and hence is a type-L certificate of infeasibility.

We can also use ℓ and Λ satisfying (LB) to construct certificates of infeasibility that are different from a type-L certificate. Let us show two ways that this can be done, which we will call *type-Q* and *type-E* certificates of infeasibility, respectively.

Type-Q Certificate of Infeasibility. Let $d \in \mathbb{R}^m$ satisfying $d > 0$ be given, and let ℓ be a certified lower bound for (P) with certificate matrix Λ. It follows from (1) and (2) that if $f(d,\ell) \leq 0$ and $A^\top y(d,\ell) \not\leq u$, then (P) is infeasible. Thus, $d \in \mathbb{R}^m$, $\ell \in \mathbb{R}^m$, and $\Lambda \in \mathbb{R}^{m\times m}$ that satisfy

$$\begin{aligned} d &> 0 \\ A\Lambda &= -A \\ \Lambda &\geq 0 \\ -\Lambda^\top u &\geq \ell \\ f(d,\ell) &\leq 0 \\ A^\top y(d,\ell) &\not\leq u \end{aligned}$$

comprise a certificate of infeasibility, which we will refer to as a type-Q certificate of infeasibility, where Q stands for *quadratic* because the fifth system above is a quadratic inequality in ℓ. (And later in this paper, we will show how to construct a type-L certificate of infeasibility from a type-Q certificate of infeasibility; see Proposition 5.3.)

Type-E Certificate of Infeasibility. Let $d \in \mathbb{R}^m$ satisfying $d > 0$ be given, and let ℓ be a certified lower bound for (P) with certificate matrix Λ. Suppose that $f(d, \ell) > 0$, whereby from (2) it follows that $E(d, \ell)$ has positive volume.

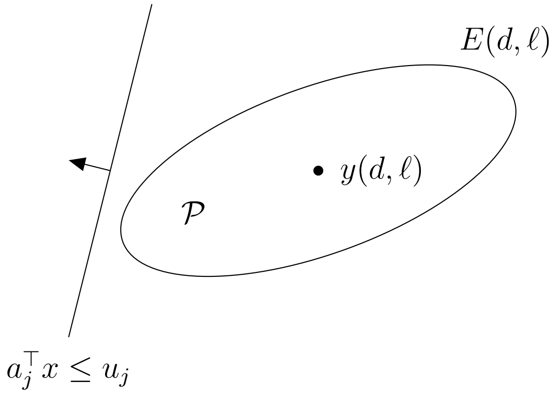

Figure 2: All points in $E(d, \ell)$ violate constraint j of (P).

It then follows from (1) and (2) that if there exists $j \in \{1, ..., m\}$ satisfying

$$u_j < \min_{x \in E(d,\ell)} a_j^\top x \ ,$$

namely every point in $E(d, \ell)$ violates constraint j of (P), then (P) is infeasible (see Figure 2). Now notice that

$$\min_{x \in E(d,\ell)} a_j^\top x = a_j^\top y(d, \ell) - \sqrt{f(d, \ell)}\sqrt{a_j^\top (ADA^\top)^{-1} a_j} \ .$$

Thus $d \in \mathbb{R}^m$, $\ell \in \mathbb{R}^m$, $\Lambda \in \mathbb{R}^{m \times m}$, and $j \in \{1, ..., m\}$ that satisfy

$$\begin{aligned} d &> 0 \\ f(d, \ell) &> 0 \\ A\Lambda &= -A \\ \Lambda &\geq 0 \\ -\Lambda^\top u &\geq \ell \\ u_j &< a_j^\top y(d, \ell) - \sqrt{f(d, \ell)}\sqrt{a_j^\top (ADA^\top)^{-1} a_j} \end{aligned}$$

comprise a certificate of infeasibility of (P), which we will refer to as a type-E certificate of infeasibility, where E stands for *ellipsoid* because the bound arises from minimization over the ellipsoid $E(d, \ell)$ as just described. Burrell and Todd [2] show how to construct a type-L certificate of infeasibility from a type-E certificate of infeasibility, which we will review in Proposition 5.1 and Corollary 5.1.

We note that there can of course be many other types of certificates of infeasibility beyond the three types just described.

1.1 Schematic of the Oblivious Ellipsoid Algorithm

Algorithm 1 below is an informal schematic of our Oblivious Ellipsoid Algorithm. (For the full algorithm description of OEA, see Algorithm 4 and the surrounding discussion.)

Algorithm 1 Schematic of Oblivious Ellipsoid Algorithm (OEA)

```
Input: data $A$ and $u$, certified lower bound $\ell$ for $(P)$ with certificate matrix $\Lambda$, and $d > 0$.
1: Compute $y(d,\ell)$. If $A^\top y(d,\ell) \leq u$, then Return $y(d,\ell)$ as a solution of $(P)$ and Stop.
2: Compute $f(d,\ell)$. If $f(d,\ell) \leq 0$, then construct and Return a certificate of infeasibility and Stop.
3: Compute the most violated constraint: $j \leftarrow \mathrm{argmax}_{i\in\{1,\ldots,m\}}\, a_i^\top y(d,\ell) - u_i$.
4: (Possibly) update certificate $\lambda_j$ if its best lower bound can be improved.
5: If $\min_{x\in E(d,\ell)} a_j^\top x_j > u_j$, then construct and Return a certificate of infeasibility and Stop.
6: Update ellipsoid $E(d,\ell)$ by updating $(d,\ell) \to (\tilde{d},\tilde{\ell})$.
7: Re-set $(d,\ell) \leftarrow (\tilde{d},\tilde{\ell})$ and Goto Step 1.
```

The iterates of Algorithm 1 are d, ℓ, and Λ. In Step 1, we perform a "standard" ellipsoid algorithm step where we check if the center $y(d,\ell)$ of the ellipsoid $E(d,\ell)$ is a feasible solution of (P), and if so we output $y(d,\ell)$ and stop. If we proceed to Step 2, then $A^\top y(d,\ell) \nleq u$. In Step 2, we check if $f(d,\ell) \leq 0$, and if this holds, then d, ℓ, and Λ comprise a type-Q certificate of infeasibility, from which we can construct a type-L certificate of infeasibility (as will be shown in Proposition 5.3). In Step 3 we perform another standard ellipsoid algorithm step wherein we compute the index of the most violated constraint. (Actually, in the standard ellipsoid method it is sufficient to compute the index of any violated constraint, but computing the most violated constraint will be crucial for establishing the convergence guarantee of the Oblivious Ellipsoid Algorithm when (P) is infeasible.) In Step 4, we possibly update the lower bound certificate λ_j if the update certifies a better lower bound than the largest lower bound currently certified. In Step 5, we check if $\min_{x\in E(d,\ell)} a_j^\top x_j > u_j$, and if this condition is satisfied, then d, ℓ, and Λ comprise a type-E certificate of infeasibility, from which we can construct and return a type-L certificate of infeasibility (as will be shown in Proposition 5.1 and Corollary 5.1). In Step 6, we update the ellipsoid $E(d,\ell)$ by computing new values $(\tilde{d},\tilde{\ell})$ of the parameters of $E(\cdot,\cdot)$ which replace the current values (d,ℓ) in Step 7.

1.2 Summary of Main Results

We briefly summarize our main results concerning the Oblivious Ellipsoid Algorithm. In the case when (P) is infeasible, OEA will compute a type-L certificate of infeasibility in

$$\left\lfloor 2m(m+1)\ln\left(\frac{m+1}{2m}\frac{\|u-\ell\|}{\tau(A,u)}\right)\right\rfloor$$

iterations, where ℓ is the initial lower bound for (P) certified by the initial Λ, and $\tau(A,u)$ is a geometric condition number that naturally captures the extent of feasibility or infeasibility of (P); see Theorem 7.1 as well as Corollary 7.1 which specializes the above bound to the case where (P) contains box constraints. Each iteration of OEA requires $O(m^2)$ arithmetic operations in a straightforward implementation, whence the total computational complexity of OEA when (P) is infeasible is $O(m^4 \ln \frac{1}{\tau})$. (We assume for simplicity here that u and $u - \ell$ are $\Theta(1)$.)

In the case when (P) is feasible, OEA will compute a solution of (P) in

$$\left\lfloor 2n(m+1) \ln \left(\frac{\|u - \ell\|}{2\rho(A)\tau(A,u)} \right) \right\rfloor$$

iterations, where ℓ is the initial certified lower bound for (P) certified by the initial Λ, $\rho(A)$ is a geometric condition measure which captures the distance to unboundedness of (P), and $\tau(A,u)$ is the geometric condition measure mentioned above (which corresponds to the radius of the largest inscribed ball in the feasible region $\mathcal{P}$ in the feasible case); see Theorem 7.2 as well as Corollary 7.2 which specializes the above bound to the case where (P) contains box constraints (in which case $\rho(A)$ plays no role). Since each iteration of OEA requires $O(m^2)$ operations, the total computational complexity of OEA when (P) is feasible is $O(m^3 n \ln \frac{1}{\rho\tau})$.

The iteration bound in the feasible case follows from standard volume-reduction arguments. However, in the infeasible case the iteration bound follows from a proof that at each iteration a novel potential function is reduced. The introduction of this potential function is another contribution of our paper; see Section 7.1. We emphasize that the algorithm we develop does not know whether the problem is feasible or infeasible (hence oblivious); however, in either case, it makes progress at each iteration towards determining that fact (both in volume reduction and in the potential function). We leave open the possibility of a less oblivious algorithm that strives to make greater progress in decreasing the volume or the potential function.

Let us compare the computational complexity bounds above to the strategy of running the standard ellipsoid algorithm in parallel simultaneously on (P) and (Alt) (which we denote by the acronym "SEAP" here and in Table 1). Our comparison assumes a straightforward implementation of the algorithms involved; however, we show how to reduce the number of operations per iteration of each algorithm using a more complicated implementation in Appendix A. The first two rows of Table 1 present the relevant bounds in the comparison. The standard ellipsoid scheme SEAP has potentially superior computational complexity over OEA – $O(mn^3 \ln \frac{1}{\rho\tau})$ rather than $O(m^3 n \ln \frac{1}{\rho\tau})$ total operations when (P) is feasible, and $O(m(m-n)^3 \ln \frac{1}{\tau})$ rather than $O(m^4 \ln \frac{1}{\tau})$ when (P) is infeasible and $\frac{m}{\rho}$ is sufficiently small. For instances when (P) is infeasible, $m = O(m-n)$, and $\frac{m}{\rho}$ is sufficiently small, the two algorithms have the same computational complexity, namely $O(m^4 \ln \frac{1}{\tau})$. And for instances when (P) is infeasible, $m = O(m-n)$, and $\frac{m}{\rho}$ is sufficiently large, OEA has potentially superior computational complexity over SEAP – $O(m^4 \ln \frac{1}{\tau})$ rather than $O(m^4 \ln \frac{m}{\rho\tau})$.

We also present two modified versions of OEA, which we call OEA-No-Alt and OEA-MM, that require a smaller number of operations per iteration than OEA – $O(mn)$ instead of $O(m^2)$. Here we briefly describe these versions and their computational complexity. Recall from earlier in this section that OEA maintains at each iteration a lower bound vector ℓ that is certified by a corresponding certificate matrix Λ, and that ℓ and Λ are updated at each iteration. The increased work per iteration of OEA – $O(m^2)$ instead of $O(mn)$ – is the result of maintaining/updating the matrix Λ. It turns out that the certificate matrix Λ is

Algorithm	Number of Iterations		Operations per Iteration (standard implementation)		Total Number of Operations	
	(P) is Feasible	(P) is Infeasible	(P) is Feasible	(P) is Infeasible	(P) is Feasible	(P) is Infeasible
SEAP	$O(n^2 \ln \frac{1}{\rho\tau})$	$O((m-n)^2 \ln \frac{m}{\rho\tau})$	$O(mn)$	$O(m(m-n))$	$O(mn^3 \ln \frac{1}{\rho\tau})$	$O(m(m-n)^3 \ln \frac{m}{\rho\tau})$
OEA	$O(mn \ln \frac{1}{\rho\tau})$	$O(m^2 \ln \frac{1}{\tau})$	$O(m^2)$	$O(m^2)$	$O(m^3 n \ln \frac{1}{\rho\tau})$	$O(m^4 \ln \frac{1}{\tau})$
OEA-No-Alt	$O(mn \ln \frac{1}{\rho\tau})$	$O(m^2 \ln \frac{1}{\tau})$	$O(mn)$	$O(mn)$	$O(m^2 n^2 \ln \frac{1}{\rho\tau})$	$O(m^3 n \ln \frac{1}{\tau})$
OEA-MM	$O(mn \ln \frac{1}{\rho\tau})$	$O(m^2 \ln \frac{1}{\tau})$	$O(mn)$	$O(mn)$	$O(m^2 n^2 \ln \frac{1}{\rho\tau})$	$O(m^3 n \ln \frac{1}{\tau})$

Table 1: Computational complexity comparison of Standard Ellipsoid Algorithm in Parallel (SEAP), Oblivious Ellipsoid Algorithm (OEA), and the modified versions OEA-No-Alt and OEA-MM.

not actually used anywhere in the computations in the algorithm; rather its sole purpose is to produce a Type-L certificate of infeasibility (a solution of (Alt)) after such infeasibility is detected. If one is only interested in solving Challenge II, i.e., correctly detecting infeasibility (but not necessarily producing a solution of (Alt)), then the updates of Λ in OEA can be removed from the steps of the algorithm, which simplifies the work per iteration and yields $O(mn)$ operations per iteration as opposed to $O(m^2)$ operations per iteration for OEA. The resulting modified method is called OEA-No-Alt because it does not produce a solution of (Alt), and its computational complexity bounds are shown in the third row of Table 1. We point out that in the case when (P) is infeasible and $n \ll m$, the last column of the table indicates that the computational complexity of OEA-No-Alt for proving infeasibility (by correctly detecting that (P) is infeasible) is $O(m^3 n \ln \frac{1}{\tau})$, which is superior to the $O(m^4 \ln \frac{m}{\rho\tau})$ computational complexity of SEAP – at the expense of not producing a solution to (Alt). We refer the reader to Section 8 for details. (However, we present a specific reformulation of (P) in Appendix A that reduces the number of operations required in each iteration of both SEAP and OEA-No-Alt, eliminating this complexity advantage at the cost of possibly increasing the values of the condition measure τ and ρ of the problem.)

The second modified version of OEA that we develop is called OEA-MM because it uses more memory. This version of OEA postpones updating the certificate matrix Λ until after OEA detects infeasibility in order to again reduce the operation complexity at each iteration from $O(m^2)$ to $O(mn)$. However, OEA-MM requires more memory storage for this post-processing step. Again we refer the reader to Section 8 for details.

Differences between the Oblivious Ellipsoid Algorithm and the Standard Ellipsoid Algorithm. We did not see a way to use a standard version of the ellipsoid algorithm to solve Challenge I or II. In particular, standard versions are designed to decrease the volume of the ellipsoid as much as possible at each iteration (by computing the minimum volume ellipsoid that contains the current half-ellipsoid), and we found this to be detrimental to establishing any type of guarantee when (P) is infeasible. Accordingly, we develop an alternative way to update ellipsoids (see Remark 6.1) that sufficiently decreases the volume (to obtain a guarantee when (P) is feasible) while also decreasing the value of a certain potential function that we introduce that is related to infeasibility measures. Like the volume of a full-dimensional polytope, the potential is bounded from below, which allows us to establish a guarantee when (P) is infeasible; see Section 7.1 for the details.

1.3 Literature Review

The ellipsoid method was introduced by Yudin and Nemirovsky [16] in their study of the complexity of convex optimization, and independently by Shor [12], and then famously used by Khachiyan [8] to show that linear programming (in the bit model) is polynomial-time bounded. Both Yudin-Nemirovsky and Khachiyan used a varying coordinate system to describe their ellipsoids, but Gács and Lovász in their exposition of the method [5] and almost all subsequent authors used the representation $\{x \in \mathbb{R}^n : (x-y)^\top G^{-1}(x-y) \leq 1\}$ in terms of the center y and the shape matrix inverse G. Many authors developed improvements involving deep and two-sided cuts; see the survey paper [1] and its references. Most research concentrated on linear programming, although there was a substantial research effort devoted to consequences in combinatorial optimization (see Grötschel, Lovász, and Schrijver [6]), and Ecker and Kupferschmid showed the effectiveness of the method on medium-sized nonlinear programming problems [4].

Here we are concerned with the linear case, and indeed just with linear inequalities. In the literature on linear programming, most variants of the ellipsoid method just describe the updates to the center y and the shape matrix G. In proving that the formulae for two-sided cut variants gave minimum-volume ellipsoids, Todd [13] showed that the new quadratic inequality was a convex combination of that defining the old ellipsoid and one requiring the solution to lie between the two hyperplanes defining the two-sided cut. This insight led later to the Burrell-Todd representation described above [2].

We also mention that there are variants of the ellipsoid method which enclose the feasible region in a sequence of convex bodies whose volumes decrease geometrically. One such method is the "simplex" method of Yamnitsky and Levin [15], which uses simplices instead of ellipsoids as the fundamental class of convex bodies in the algorithm. The guaranteed volume reduction of their method is smaller than that of the standard ellipsoid method ($O(\exp(n^{-2}))$ instead of $O(\exp(n^{-1}))$), but may be better than that of OEA when $m \gg n^2$. Similar to the Burrell-Todd variant of the ellipsoid method and also OEA, their method iteratively maintains a certificate that the current simplex contains the feasible region, but we are not aware of research on its complexity in detecting infeasibility.

1.4 Organization

In Section 2 we show how to easily construct initial lower bounds and certificates when (P) contains box constraints. In Section 3 we further review the ellipsoid parametrization (namely (1) and (2)) of [2] and we introduce ellipsoid slab radii, which will be an important geometric concept in the setting when (P) is infeasible. In Section 4 we introduce the condition number $\tau(A, u)$ that captures the extent of feasibility or infeasibility of the system (P), and we also introduce the condition measure $\rho(A)$ that measures the distance to unboundedness of (P). In Section 5 we review and further develop a method for updating certificates for lower bounds developed initially in [2]. In Section 6 we develop our mechanism for updating the ellipsoids from one iteration to the next. In Section 7 we formally state our algorithm along with convergence guarantees for the feasible and the infeasible cases. Lastly, in Section 8 we present two modified versions of OEA, which we denote as OEA-No-Alt and OEA-MM, together with their complexity analysis. Most of the proofs are in the appendices at the end of the paper.

1.5 Notation

The ℓ_p norm is denoted $\|\cdot\|_p$ for $1 \le p \le \infty$, and the operator norm of a matrix M is denoted by $\|M\|_{a,b} = \max_{\|v\|_a=1} \|Mv\|_b$. For convenience we denote the Euclidean (ℓ_2) norm simply by $\|\cdot\|$. For $d \in \mathbb{R}^m$, we use D to denote the diagonal matrix whose diagonal entries correspond to the entries of d. If not obvious from context, we use 0_k to denote the k-dimensional vector of zeros, and $I_{k\times k}$ to denote the identity matrix in $\mathbb{R}^{k\times k}$. Let e_i denote the i^{th} unit vector, whose dimension is dictated by context, and let $e = (1, \ldots, 1)^\top$, whose dimension is also dictated by context. We use $[k] := \{1, \ldots, k\}$. For a given k-dimensional vector v, the positive and negative componentwise parts of v are denoted by v^+ and v^-, respectively, and satisfy $v^+ \ge 0$, $v^- \ge 0$, $v = v^+ - v^-$, and $(v^+)^\top v^- = 0$. To save physical space, we use the notation $[u; v; w]$ to denote the concatenation of column vectors u, v, w into a single new column vector.

2 Initializing lower bounds and certificates for systems with box constraints

The variant of the ellipsoid method that we develop in this paper is premised on having an initial vector of lower bounds ℓ with associated initial certificate matrix Λ for the linear inequalities defining (P). In general, it is not clear how to construct such lower bounds and certificates – short of solving related systems of inequalities of size at least as large as that of the original system. However, in the often-occurring case when the linear inequality system defining (P) contains box constraints, such lower bounds and certificates are easy to write down. Suppose that the linear inequality system is given with box constraints, namely:

$$(P_B): \quad \begin{cases} \hat{A}^\top x \le \hat{u} \\ \quad x \le \bar{b} \\ \quad x \ge \underline{b} \ , \end{cases}$$

for given data $(\hat{A}, \hat{u}, \underline{b}, \bar{b}) \in \mathbb{R}^{n\times\hat{m}} \times \mathbb{R}^{\hat{m}} \times \mathbb{R}^n \times \mathbb{R}^n$ satisfying $\underline{b} \le \bar{b}$. We can re-write system (P_B) in the format $A^\top x \le u$ by defining

$$A := \begin{bmatrix} \hat{A} & I_{n\times n} & -I_{n\times n} \end{bmatrix} \tag{3}$$

$$u := \begin{bmatrix} \hat{u} \ ; \ \bar{b} \ ; \ -\underline{b} \end{bmatrix} \ , \tag{4}$$

and we can assume without loss of generality that:

$$\hat{u}_i \quad \le \quad \max_{\underline{b}\le x\le\bar{b}} a_i^\top x \quad = \quad (-(a_i)^-)^\top \underline{b} + ((a_i)^+)^\top \bar{b} \quad \text{for } i \in [\hat{m}] \tag{5}$$

(as otherwise constraint i would be redundant and can be removed). Let us now see how to conveniently write down certified lower bounds and certificates for the system $A^\top x \le u$ defined above. For $i \in [\hat{m}]$ define $\hat{\ell}_i \in \mathbb{R}$ to be:

$$\hat{\ell}_i := \min_{\underline{b} \le x \le \bar{b}} \hat{a}_i^\top x \ = \ (-(a_i)^-)^\top \bar{b} + ((a_i)^+)^\top \underline{b} \ , \tag{6}$$

and $\hat{\ell} := (\hat{\ell}_1, \ldots, \hat{\ell}_m)$; then it is straightforward to show that

$$\ell := \left[\hat{\ell} \ ; \ \underline{b} \ ; \ -\bar{b} \right] \tag{7}$$

is a valid lower bound vector for the system $A^\top x \le u$ defined above in (3)-(4). We construct the certificate λ_i of constraint i of $A^\top x \le u$ as follows. For $i = 1, ..., \hat{m}$, define

$$\lambda_i = \left[0_{\hat{m}} \ ; \ \hat{a}_i^- \ ; \ \hat{a}_i^+ \right] \ , \tag{8}$$

for $i = \hat{m}+1, ..., \hat{m}+n$, define

$$\lambda_i = \left[0_{\hat{m}} \ ; \ 0_n \ ; \ e_i \right] \ , \tag{9}$$

and for $i = \hat{m}+n+1, ..., \hat{m}+2n$, define

$$\lambda_i = \left[0_{\hat{m}} \ ; \ e_i \ ; \ 0_n \right] \ . \tag{10}$$

It is then straightforward to check that $\Lambda = [\lambda_1 | \ldots | \lambda_m]$ defined by (8-10) is a certificate for the lower bounds ℓ defined in (7). In summary, given a system of inequalities with box constraints (P_B), we can conveniently re-write the system in the format $A^\top x \le u$ and we can easily construct initial certified lower bounds ℓ along with an associated certificate matrix Λ. We can also assume without loss of generality that $l_i \le u_i$ for $i \in [m]$, for otherwise we can easily construct a type-L certificate of infeasibility.

Last of all, and somewhat separately, we will need the following result which is straightforward to show as a consequence of (5), (6), and Assumption 1.1:

$$\|u - \ell\| \le \left(\sqrt{\hat{m}+2} \right) \|\bar{b} - \underline{b}\| \ , \tag{11}$$

where u and ℓ are defined as in (4) and (7), respectively.

3 Containing ellipsoids and ellipsoid slab radii

We consider the ellipsoid parameterization originally developed in Burrell and Todd [2] and introduce the geometric notion of ellipsoid slab radii. For clarity and convenience, we re-present some definitions from Section 1.

For $\ell \in \mathbb{R}^m$ and $d \in \mathbb{R}^m$ with $d > 0$, recall from (1) the ellipsoid $E(d, \ell)$:

$$E(d, \ell) := \left\{ x \in \mathbb{R}^n : \left(A^\top x - \ell \right)^\top D \left(A^\top x - u \right) \le 0 \right\} \tag{12}$$

(where D is the diagonal matrix corresponding to d), and note that $E(d, \ell)$ has the following properties:

1. $E(d,\ell)$ is bounded; indeed from Assumption 1.1, the columns of A span $\mathbb{R}^n$ and hence $ADA^\top$ is positive definite.
2. If ℓ is a certified lower bound for (P) with some certificate Λ, then (P) is contained in the ellipsoid $E(d,\ell)$.
3. The ellipsoid $E(d,\ell)$ is invariant under positive scaling of d, that is, $E(d,\ell) = E(\alpha d,\ell)$ for any $\alpha > 0$.

Let us define the following quantities:

$$\begin{aligned}
r(\ell) &:= \tfrac{1}{2}(u+\ell) \ , \\
v(\ell) &:= \tfrac{1}{2}(u-\ell) \ , \\
B(d) &:= ADA^\top \ , \\
y(d,\ell) &:= B(d)^{-1}ADr(\ell) \ , \\
t(d,\ell) &:= A^\top y(d,\ell) - r(\ell) \ , \\
f(d,\ell) &:= v(\ell)^\top D v(\ell) - t(d,\ell)^\top D t(d,\ell) \ ,
\end{aligned}$$

and notice that $y(d,\ell)$ and $t(d,\ell)$ are invariant under positive scaling of d. Here $r(\ell)$ and $v(\ell)$ are the center and "radius" of the line segment between u and ℓ, respectively. Note that the parameterization in the above objects is over d and ℓ, as we consider the data A and u of the linear inequality system to be fixed. It is straightforward to verify the following alternative characterization of $E(d,\ell)$ using the above quantities:

$$E(d,\ell) = \left\{ x \in \mathbb{R}^n : (x - y(d,\ell))^\top B(d)\,(x - y(d,\ell)) \le f(d,\ell) \right\} \ . \tag{13}$$

Here we see that $y(d,\ell)$ is the center of $E(d,\ell)$. Furthermore, $E(d,\ell)$ has positive volume when $f(d,\ell) > 0$, is the point set $\{y(d,\ell)\}$ when $f(d,\ell) = 0$, and is the empty set when $f(d,\ell) < 0$.

Remark 3.1. When $E(d,\ell)$ has positive volume, i.e., $f(d,\ell) > 0$, and in light of the fact that $E(d,\ell)$ is invariant under positive scalings of d, we will often (for arithmetic convenience) rescale d to $d \leftarrow \frac{1}{f(d,\ell)} d$, in order for d to satisfy $f(d,\ell) = 1$.

Now suppose that $E(d,\ell)$ has positive volume, i.e., $f(d,\ell) > 0$. For each index $i \in [m]$ define the *ellipsoid slab radius* $\gamma_i(d,\ell)$ as:

$$\begin{aligned}
\gamma_i(d,\ell) := \min_{\gamma} \ & \gamma \\
\text{s.t.} \ & E(d,\ell) \subseteq \{x \in \mathbb{R}^n : |a_i^\top x - a_i^\top y(d,\ell)| \le \gamma\} \ ,
\end{aligned}$$

and is so named because $\gamma_i(d,\ell)$ is the radius γ of the smallest *slab* of the form $\{x \in \mathbb{R}^n : |a_i^\top x - a_i^\top y(d,\ell)| \le \gamma\}$ containing the ellipsoid $E(d,\ell)$. The ellipsoid slab radius $\gamma_i(d,\ell)$ has the following properties:

1. $\gamma_i(d,\ell)$ is invariant under positive scaling of d; this is because $E(d,\ell)$ and $y(d,\ell)$ are invariant under positive scaling of d, and

2. $\gamma_i(d,\ell)$ is alternatively characterized as:

$$\gamma_i(d,\ell) \;=\; \max_x \left\{ a_i^\top(x - y(d,\ell)) : x \in E(d,\ell) \right\} \;=\; \sqrt{f(d,\ell) a_i^\top (ADA^\top)^{-1} a_i} \ . \tag{14}$$

Notice that we have only defined $\gamma_i(d,\ell)$ when $E(d,\ell)$ has positive volume, namely $f(d,\ell) > 0$. Of course, we could extend the notions above to the case when $f(d,\ell) = 0$ (whereby $E(d,\ell)$ is the point set $\{y(d,\ell)\}$ and $\gamma_i(d,\ell) = 0$), but this will not be needed.

4 Condition Measures of Feasibility, Infeasibility, and Boundedness

We introduce the condition measure $\tau(A,u)$ which will be used to measure the extent of feasibility or infeasibility of (P). Define:

$$\tau(A,u) := |z^*| \quad \text{where} \quad z^* := \max_{x\in\mathbb{R}^n} \min_{i\in[m]} \ (u_i - a_i^\top x) \ . \tag{15}$$

The following proposition lists some of the relevant properties of $\tau(A,u)$. Here we use the parametric notation $\mathcal{P}_v := \{x \in \mathbb{R}^n : A^\top x \le v\}$ to keep our statements simple.

Proposition 4.1. *Basic properties of $\tau(A,u)$*

(a) If $\mathcal{P} \neq \emptyset$, then $\tau(A,u) = z^* \ge 0$ and $\tau(A,u)$ is the radius of the largest ℓ_2 ball contained in $\mathcal{P}$.

(b) If $\mathcal{P} = \emptyset$, then $\tau(A,u) = -z^* > 0$ is the smallest scalar θ for which the right-hand-side perturbed system $A^\top x \le u + \theta e$ has a feasible solution.

(c) If $\tau(A,u) = 0$, then the linear inequality system (P) is "ill-posed" in the following sense: for any $\varepsilon > 0$ there exist perturbations Δb_{feas} and Δb_{infeas} with $\|\Delta b_{\text{feas}}\|_\infty \le \varepsilon$, $\|\Delta b_{\text{infeas}}\|_\infty \le \varepsilon$ for which $\mathcal{P}_{u+\Delta b_{\text{feas}}}$ has a strict (interior) solution and $\mathcal{P}_{u+\Delta b_{\text{infeas}}} = \emptyset$.

□

Item (a) above uses the second part of Assumption 1.1 that the columns of A have unit ℓ_2 norm. Regarding (c), the concept of "ill-posedness" in this context was first developed by Renegar [9, 10, 11]; see Appendix B for further discussion of connections between $\tau(A,u)$ and Renegar's condition measure $\bar{\rho}(d)$ and a proof that $\tau(A,u) \ge \bar{\rho}(d)$. Also, $\tau(A,u)$ is constructed in a similar spirit to the condition number $\mathcal{C}(A)$ for homogeneous linear inequalities developed by Cheung and Cucker [3].

Notice in the case when (P) is infeasible that for any $x \in \mathbb{R}^n$ there is some constraint that is violated by at least $\tau(A,u)$, namely there exists $i \in [m]$ for which $a_i^\top x \ge u_i + \tau(A,u)$.

We also introduce the following condition measure (in the spirit of Renegar [9]) which captures the extent to which $\mathcal{P}$ is close to unbounded:

$$\rho(A) := \min_{\Delta A \in \mathbb{R}^{n\times m}} \{\|\Delta A\|_{1,2} : \text{there exists } v \neq 0 \text{ satisfying } [A + \Delta A]^\top v \le 0\} \ . \tag{16}$$

Indeed, observe that $\rho(A)$ is the $\|\cdot\|_{1,2}$ operator norm of the smallest perturbation of the matrix A for which the perturbed feasible region becomes unbounded and hence violates Assumption 1.1.

5 Updating certificates for lower bounds, and constructing certificates of infeasibility

Let $d > 0$, and let ℓ be a certified lower bound for (P) with certificate matrix Λ, and suppose that $E(d,\ell)$ has positive volume, i.e., $f(d,\ell) > 0$. Let $i \in [m]$ be given. From the definition of the slab radius $\gamma_i(d,\ell)$, the ellipsoid $E(d,\ell)$ is contained in the half-space $\{x \in \mathbb{R}^n : a_i^\top x \geq a_i^\top y(d,\ell) - \gamma_i(d,\ell)\}$, and therefore when (P) is feasible:

$$x \in \mathcal{P} \ \Rightarrow \ x \in E(d,\ell) \ \Rightarrow \ a_i^\top x \geq a_i^\top y(d,\ell) - \gamma_i(d,\ell) \ ,$$

and it follows that

$$L_i := a_i^\top y(d,\ell) - \gamma_i(d,\ell)$$

is a valid lower bound on constraint i of (P). This leads to the question of whether and how can one construct a certificate $\tilde{\lambda}_i$ for the lower bound L_i? This question was answered in the affirmative in Burrell and Todd [2], and we present their solution in the following proposition which shows how to use d, ℓ, and Λ to construct a certificate $\tilde{\lambda}_i \in \mathbb{R}^m$ for the lower bound L_i. (Note that it is possible that $L_i \leq \ell_i$.)

Proposition 5.1. (see [2]) *Let $d > 0$, and ℓ be a certified lower bound for (P) with certificate matrix Λ, suppose $E(d,\ell)$ has positive volume, and suppose that d has been rescaled so that $f(d,\ell) = 1$. Let $i \in [m]$ be given, and define $L_i := a_i^\top y(d,\ell) - \gamma_i(d,\ell)$, and*

$$\begin{aligned} \hat{\lambda}_i &:= \gamma_i(d,\ell) D t(d,\ell) - D A^\top B(d)^{-1} a_i \ , \\ \tilde{\lambda}_i &:= \Lambda \hat{\lambda}_i^- + \hat{\lambda}_i^+ \ . \end{aligned} \tag{17}$$

Then L_i is also a certified lower bound on constraint i of (P), with certificate $\tilde{\lambda}_i$. In particular, it holds that $-\tilde{\lambda}_i^\top u \geq L_i$. □

For completeness as well for consistency with the notation used in this paper, we present a proof of Proposition 5.1 in Appendix E.1.

As a corollary, we can construct a type-L certificate of infeasibility from a type-E certificate of infeasibility:

Corollary 5.1. *Under the set-up of Proposition 5.1, if $L_i := a_i^\top y(d,\ell) - \gamma_i(d,\ell) > u_i$, then $\bar{\lambda}_i := \tilde{\lambda}_i + e_i$ is a type-L certificate of infeasibility.*

Proof. From Proposition 5.1 it holds that $\tilde{\lambda}$ is a certificate for the lower bound $L_i := a_i^\top y(d,\ell) - \gamma_i(d,\ell) > u_i$. Hence $A\tilde{\lambda}_i = -a_i$, $\tilde{\lambda}_i \geq 0$, and $-u^\top \tilde{\lambda}_i \geq L_i > u_i$. It follows that $A\bar{\lambda}_i = 0$, $\bar{\lambda}_i \geq 0$, and $-u^\top \bar{\lambda}_i \geq L_i - u_i > 0$, and so $\bar{\lambda}_i$ is a type-L certificate of infeasibility. □

The following proposition ties together ellipsoids and slab radii, the condition measure $\tau(A,u)$, updates of certificates of lower bounds, and type-L certificates of infeasibility.

Proposition 5.2. *Under the set-up of Proposition 5.1, let $i := \operatorname{argmax}_{h \in [m]}(a_h^\top y(d,\ell) - u_h)$, and let L_i and $\tilde{\lambda}_i$ be as defined therein. If $\gamma_i(d,\ell) < \tau(A,u)$, then $\bar{\lambda}_i := \tilde{\lambda}_i + e_i$ is a type-L certificate of infeasibility.*

Proof. If $\mathcal{P}$ were nonempty, then a ball of radius $\tau(A,u)$ would be contained in a slab of radius $\gamma_i(d,\ell)$, so that $\tau(A,u) \le \gamma_i(d,\ell)$. Hence $\mathcal{P} = \emptyset$ and so from the definition of $\tau(A,u)$ it follows that

$$a_i^\top y(d,\ell) - u_i \ge \tau(A,u) > \gamma_i(d,\ell) = a_i^\top y(d,\ell) - L_i \ ,$$

where the first inequality is from the definition of $\tau(A,u)$ in the case when $\mathcal{P} = \emptyset$, the second inequality is by supposition, and the equality is from the definition of L_i in Proposition 5.1. It then follows that $L_i > u_i$. Hence the desired result follows from Corollary 5.1. □

Propositions 5.1 and 5.2 are premised on $E(d,\ell)$ having positive volume, i.e., $f(d,\ell) > 0$. If $f(d,\ell) \le 0$, then either $f(d,\ell) < 0$ or $f(d,\ell) = 0$. If $f(d,\ell) < 0$, then $E(d,\ell) = \emptyset$ from (13), which implies $\mathcal{P} = \emptyset$, and if $f(d,\ell) = 0$, then (13) implies that either $\mathcal{P} = \{y(d,\ell)\}$ (which is easy to verify by checking if $A^\top y(d,\ell) \le u$) or $\mathcal{P} = \emptyset$. Below we present Procedure 2, which accomplishes the task of constructing a type-L certificate of infeasibility when $f(d,\ell) \le 0$ and $A^\top y(d,\ell) \not\le u$, i.e., constructing a type-L certificate of infeasibility from a type-Q certificate of infeasibility. As will be proved, Procedure 2 is well-defined, in that for each step until termination, the specified quantities exist (for example, the scalar β in Step 2 exists). Steps 2 through 7 of Procedure 2 decrease the i-th and j-th entries of ℓ such that the updated parameterized ellipsoid $E(d,\ell)$ has positive volume and the condition $a_k^\top y(d,\ell) - \gamma_k(d,\ell) > u_k$ holds for some index $k \in [m]$. Note that in Step 7 it holds that Λ is still a certificate for the updated lower bounds ℓ because the components of the updated ℓ have either been decreased or remain the same. Steps 8 through 10 use Proposition 5.1 to construct a certificate $\tilde{\lambda}_k$ for the lower bound $L_k := a_k^\top y(d,\ell) - \gamma_k(d,\ell)$ that satisfies $L_k > u_k$. In Step 11, we return $\bar{\lambda}_k := \tilde{\lambda}_k + e_k$, which by Remark 1.1 is a type-L certificate of infeasibility.

Procedure 2 Constructing a type-L certificate of infeasibility from a type-Q certificate of infeasibility

```
Input: $d > 0$, certified lower bounds $\ell$ with certificate matrix $\Lambda$, satisfying $f(d,\ell) \le 0$ and $A^\top y(d,\ell) \not\le u$
If $\ell \not\le u$, select an index $j$ for which $\ell_j > u_j$ and Return $\bar{\lambda}_j := \lambda_j + e_j$ and Stop.
Select any index $i \in [m]$, and compute $\beta \ge 0$ such that $f(d, \ell - \beta e_i) = 0$.
$\ell \leftarrow \ell - \beta e_i$.
Compute an index $j \in [m]$ for which $a_j^\top y(d,\ell) \le u_j$.
Compute an index $k \in [m]$ for which $a_k^\top y(d,\ell) > u_k$.
Compute $\varepsilon > 0$ such that $f(d, \ell - \varepsilon e_j) > 0$ and $a_k^\top y(d, \ell - \varepsilon e_j) - \gamma_k(d, \ell - \varepsilon e_j) > u_k$.
$\ell \leftarrow \ell - \varepsilon e_j$.
$d \leftarrow \frac{1}{f(d,\ell)} d$.
$\hat{\lambda}_k \leftarrow \gamma_k(d,\ell) D t(d,\ell) - D A^\top B(d)^{-1} a_k$.
$\tilde{\lambda}_k \leftarrow \Lambda \hat{\lambda}_k^- + \hat{\lambda}_k^+$.
Return $\bar{\lambda}_k := \tilde{\lambda}_k + e_k$ and Stop.
```

Proposition 5.3 below establishes the correctness of Procedure 2. The proof of Proposition 5.3 in Appendix E.1 also indicates how to efficiently implement Steps 2 and 6 of Procedure 2 using the mechanics of the quadratic formula.

Proposition 5.3. *The output of Procedure 2 is a type-L certificate of infeasibility.* □

6 Updating the ellipsoid $E(d, \ell)$

Let $d > 0$, let ℓ be a certified lower bound for (P) with certificate Λ, and suppose that the ellipsoid $E(d, \ell)$ has positive volume. Also suppose that d has been scaled so that $f(d, \ell) = 1$. In this section, we discuss a procedure for updating the ellipsoid $E(d, \ell)$. We will assume that the center violates some constraint, i.e., the condition $a_j^\top y(d, \ell) > u_j$ holds for some $j \in [m]$. Otherwise, $y(d, \ell)$ is feasible, and so we would have no reason to construct a new ellipsoid. In the same spirit, we will also assume that the condition $a_j^\top y(d, \ell) - u_j \leq \gamma_j(d, \ell)$ holds because if it does not, then we have a type-E certificate of infeasibility and can construct a type-L certificate of infeasibility (see Corollary 5.1), and again we would have no reason to construct a new ellipsoid.

We update the ellipsoid $E(d, \ell)$ by updating its parameters d and ℓ, and it will be convenient to write the update procedure in five elementary steps. At the end of this section, we will provide some motivation for these steps.

Procedure 3 Updating ellipsoid $E(d, \ell)$ by updating its parameters d and ℓ

```
Input: $d > 0$ and certified lower bound $\ell$ with certificate matrix $\Lambda$ satisfying $f(d, \ell) = 1$,
    and $j \in [m]$ such that $0 < a_j^\top y(d, \ell) - u_j \leq \gamma_j(d, \ell)$.
1: $\hat{\ell} \leftarrow \ell - \frac{2(t_j(d,\ell) - v_j(\ell))}{d_j \gamma_j(d,\ell)^2} e_j$.
2: Compute $y(d, \hat{\ell})$. If $A^\top y(d, \hat{\ell}) \leq u$, then Return $y(d, \hat{\ell})$ as a solution to $(P)$ and Stop.
3: Compute $f(d, \hat{\ell})$. If $f(d, \hat{\ell}) \leq 0$, then call Procedure 2 and Stop.
4: $d \leftarrow \frac{1}{f(d,\hat{\ell})} d$.
5: $\tilde{\ell} \leftarrow \hat{\ell} + \frac{2(2v_j(d,\hat{\ell}) - \gamma_j(d,\hat{\ell}))}{(m-1) d_j \gamma_j(d,\hat{\ell})^2 + 2} e_j$.
6: $\tilde{d} \leftarrow d + \frac{2}{m-1} \frac{1}{\gamma_j(d,\hat{\ell})^2} e_j$.
7: $\tilde{d} \leftarrow \frac{1}{f(\tilde{d},\tilde{\ell})} \tilde{d}$.
8: Return $\tilde{d}$ and $\tilde{\ell}$, and Stop.
```

Below we establish several properties of this procedure. Proofs of the results are in Appendix E.2.

In Step 1 of Procedure 3, we update the j-th coordinate of ℓ to obtain

$$\ell^{(1)} := \ell - \frac{2(t_j(d, \ell) - v_j(\ell))}{d_j \gamma_j(d, \ell)^2} e_j \ . \tag{18}$$

Note that the numerator above is $a_j^\top y(d, \ell) - u_j > 0$, so the jth lower bound is decreased. The effect is that the center of the new ellipsoid $E(d, \ell^{(1)})$ will satisfy the jth inequality at equality, see (19) below. While this may hurt the volume reduction achieved, it is helpful in our analysis of the infeasible case. Lemma 6.1 establishes a few properties of this update.

Lemma 6.1. *Let $d > 0$ and let ℓ be a certified lower bound for (P) with certificate matrix Λ, and suppose that $f(d,\ell) = 1$. Also suppose that some $j \in [m]$ satisfies $0 < a_j^\top y(d,\ell) - u_j \leq \gamma_j(d,\ell)$. Let $\ell^{(1)}$ be defined as in (18). Then $\ell^{(1)}$ is a lower bound for (P) with certificate matrix Λ, and the following hold:*

$$a_j^\top y(d,\ell^{(1)}) = u_j \ , \ \textit{and} \tag{19}$$

$$f(d,\ell^{(1)}) = 1 - \left(\frac{a_j^\top y(d,\ell) - u_j}{\gamma_j(d,\ell)}\right)^2 \ < \ 1 \ . \tag{20}$$

□

From Lemma 6.1 and the suppositions that $0 < a_j^\top y(d,\ell) - u_j \leq \gamma_j(d,\ell)$, it holds that $f(d,\ell^{(1)}) \geq 0$. If $f(d,\ell^{(1)}) = 0$, then either $\mathcal{P} = \{y(d,\ell)\}$ in which case in Step 2 we return $y(d,\ell^{(1)})$ as a solution to (P), or $A^\top y(d,\ell) \not\leq u$ in which case in Step 3 we call Procedure 2 to construct a type-L certificate of infeasibility, as discussed and proved in Proposition 5.3 of Section 5.

In Steps 4-7, we compute updates

$$d^{(1)} = \frac{1}{f(d,\ell^{(1)})} d \ , \tag{21}$$

$$\ell^{(2)} = \ell^{(1)} + \frac{2(2v_j(\ell^{(1)}) - \gamma_j(d^{(1)},\ell^{(1)}))}{(m-1)d_j^{(1)}\gamma_j(d^{(1)},\ell^{(1)})^2 + 2} e_j \ , \tag{22}$$

$$d^{(2)} = d^{(1)} + \frac{2}{m-1}\frac{1}{\gamma_j(d^{(1)},\ell^{(1)})^2} e_j \ . \tag{23}$$

$$d^{(3)} = \frac{1}{f(d^{(2)},\ell^{(2)})} d^{(2)} \ , \tag{24}$$

Lemma 6.2 establishes a few properties of these updates:

Lemma 6.2. *Let $d > 0$, and let ℓ be a certified lower bound for (P) with certificate matrix Λ, and suppose that $f(d,\ell) = 1$. Also suppose that some $j \in [m]$ satisfies $0 < a_j^\top y(d,\ell) - u_j \leq \gamma_j(d,\ell)$, and suppose in addition that λ_j is a certificate for the lower bound $L_j := a_j^\top y(d,\ell) - \gamma_j(d,\ell)$. Let $\ell^{(1)}$ be defined as in (18), and suppose that $f(d,\ell^{(1)}) > 0$. Let $d^{(1)}$, $\ell^{(2)}$, $d^{(2)}$, and $d^{(3)}$ be defined as in (21), (22), (23), and (24), respectively. Then*

(a) $\ell_j^{(2)} \leq \max\{\ell_j, L_j\}$, and hence $\ell^{(2)}$ is a certified lower bound for (P) with certificate matrix Λ,

(b) $\gamma_j(d^{(1)},\ell^{(1)}) > 0$, and hence (22) and (23) are well-defined, and

(c) it holds that $d^{(3)} = \dfrac{m^2-1}{m^2}\left(d^{(1)} + \dfrac{2}{m-1}\dfrac{1}{\gamma_j(d^{(1)},\ell^{(1)})^2} e_j\right)$. □

Remark 6.1. It is possible to show that the ellipsoid $E(d^{(3)},\ell^{(2)})$ contains the half ellipsoid described by the intersection of the ellipsoid $E(d^{(1)},\ell^{(1)})$ and the half-space $\{x \in \mathbb{R}^n : a_j^\top x \leq u_j\} = \{x \in \mathbb{R}^n : a_j^\top x \leq a_j^\top y(d^{(1)},\ell^{(1)})\}$. The ellipsoid $E(d^{(3)},\ell^{(2)})$ is not the minimum volume ellipsoid containing the half ellipsoid, but it would be if we substituted n for m in updates (22) and (23). We use m instead of n in order to establish a convergence guarantee for our algorithm in the setting in which (P) is infeasible; we will clarify and elaborate on this idea in Section 7.1.

So far, we have separately studied the first step and the last four steps of the update procedure. Theorem 6.1 below provides a more unified perspective on the update procedure:

Theorem 6.1. *Let $d > 0$ and ℓ be a certified lower bound for (P) with certificate matrix Λ, and suppose that $f(d, \ell) = 1$. Also let $j \in [m]$ be given and suppose that $0 < a_j^\top y(d, \ell) - u_j \leq \gamma_j(d, \ell)$. Let $\ell^{(1)}$ be defined as in (18), and suppose that $f(d, \ell^{(1)}) > 0$. Let $d^{(1)}$, $\ell^{(2)}$, $d^{(2)}$, and $d^{(3)}$ be defined as in (21), (22), (23), and (24) respectively. Then*

$$d^{(3)} = \alpha(d, \ell) \left(d + \frac{2}{m-1} \frac{1}{\gamma_j(d, \ell)^2} e_j \right) ,$$

where $\alpha(d, \ell) > \frac{m^2-1}{m^2}$. □

Remark 6.2. For those familiar with the ellipsoid algorithm and the Burrell-Todd representation, we now give some motivation for the steps in our update procedure. As we will describe in our convergence analysis, our aim is to guarantee some volume reduction, but at the same time achieve progress in the case of infeasibility. For this, we need to maintain the Burrell-Todd representation, but it appears we cannot push for aggressive volume reduction. In the original ellipsoid algorithm, the quadratic inequality defining the new ellipsoid is the sum of that defining the old ellipsoid, say

$$(x - y)^\top B^{-1} (x - y) \leq 1,$$

and the multiple

$$\frac{2}{(n-1) a_j^\top B a_j}$$

of the quadratic inequality

$$(a_j^\top x - a_j^\top y)(a_j^\top x - a_j^\top y - (a_j^\top B a_j)^{1/2}) \leq 0$$

(see [13]). However, since the old inequality was not in Burrell-Todd form, nor is the new one. In deep cut and two-sided cut variants, the old ellipsoid is given by a Burrell-Todd representation. After choosing the constraint j, the lower bound ℓ_j is possibly updated, and if so, the old ellipsoid is also updated before again taking a combination of quadratic inequalities to define the new ellipsoid. In our version, this cannot be done without jeopardizing the analysis. Therefore, the first step is decreasing the lower bound ℓ_j to $\ell_j^{(1)}$ so that the new center lies on the constraint $a_j^\top x = u_j$. Then the quadratic inequality defining the new ellipsoid is the sum of that defining the intermediate ellipsoid,

$$(x - y(d^{(1)}, \ell^{(1)}))^\top (AD^{(1)}A^\top)(x - y(d^{(1)}, \ell^{(1)}) \leq 1 ,$$

and the multiple

$$\frac{2}{(m-1)\gamma_j(d^{(1)}, \ell^{(1)})^2}$$

of the quadratic inequality

$$(a_j^\top x - u_j)(a_j^\top x - L_j^{(1)}) \le 0 \ ,$$

where $L_j^{(1)} := a_j^\top y(d^{(1)}, \ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)})$. Note that, due to the common factor $(a_j^\top x - u_j)$, the terms $d_j^{(1)}(a_j^\top x - u_j)(a_j^\top x - \ell_j^{(1)})$ from the first inequality and

$$\frac{2}{(m-1)\gamma_j(d^{(1)}, \ell^{(1)})^2}(a_j^\top x - u_j)(a_j^\top x - L_j^{(1)})$$

can be combined, and this leads to the updates for $\ell^{(2)}$ and $d^{(2)}$ and the new Burrell-Todd representation.

7 Oblivious Ellipsoid Algorithm

A formal description of our oblivious ellipsoid algorithm (OEA) is presented in Algorithm 4; the description is essentially a more formal and detailed version of the schematic version of OEA presented in Algorithm 1.

Algorithm 4 Oblivious Ellipsoid Algorithm (OEA)

```
Input: data $(A, u)$, certified lower bound $\ell$ for $(P)$ with certificate matrix $\Lambda$, and $d > 0$.
 1: Compute $y(d, \ell)$. If $A^\top y(d, \ell) \le u$, then Return $y(d, \ell)$ as a solution of $(P)$ and Stop.
 2: Compute $f(d, \ell)$. If $f(d, \ell) \le 0$, then call Procedure 2 and Stop.
 3: $d \leftarrow \frac{1}{f(d,\ell)} d$.
 4: Compute most violated constraint: $j \leftarrow \mathrm{argmax}_{i \in [m]}\, a_i^\top y(d, \ell) - u_i$.
 5: If $\ell_j < L_j := a_j^\top y(d, \ell) - \gamma_j(d, \ell)$, then update the certificate for constraint $j$:
 6:     $\hat{\lambda}_j := \gamma_j(d, \ell) D t(d, \ell) - D A^\top B(d)^{-1} a_j$.
 7:     $\tilde{\lambda}_j := \Lambda \hat{\lambda}_j^- + \hat{\lambda}_j^+$.
 8:     $\lambda_j \leftarrow \tilde{\lambda}_j$.
 9: If $L_j > u_j$, then Return type-L certificate of infeasibility $\bar{\lambda}_j := \lambda_j + e_j$ and Stop.
10: Update $E(d, \ell)$ by updating ellipsoid parameters $d$ and $\ell$:
11:     Call Procedure 3 with input $d$, $\ell$, $\Lambda$ to obtain output $\tilde{d}, \tilde{\ell}$.
12: Re-set $(d, \ell) \leftarrow (\tilde{d}, \tilde{\ell})$ and Goto Step 1.
```

Let us briefly consider the steps of Algorithm 4 that are different from the steps of the schematic Algorithm 1. In Step 2 when $f(d, \ell) \le 0$, we call Procedure 2 (see Section 5) to construct and return a type-L certificate of infeasibility. In Steps 5-8 we use Proposition 5.1 to update the certificate λ_j when we can construct a new certificate that certifies a better lower bound (namely L_j). In Step 9 when $L_j > u_j$, we return $\bar{\lambda}_j$, which is a type-L certificate of infeasibility by Corollary 5.1. Lastly, in Step 11, we use Procedure 3 (of Section 6) to update the ellipsoid by updating its parameters.

Remark 7.1. The operations complexity of an iteration of OEA (with appropriate rank-1 updates) is $O(m^2)$ because the most expensive computation that can occur in an iteration is computing $\Lambda \hat{\lambda}_j^-$ in Step 7.

Remark 7.2. It turns out that the condition $f(d,\ell) \leq 0$ in Step 2 can only be satisfied at Step 2 during the first iteration of the algorithm. This is because at later iterations Procedure 3 in Step 11 detects if this condition holds, calls Procedure 2, and then terminates. (And it is straightforward to check that if Procedure 3 completes a full iteration with output $\tilde{d}$ and $\tilde{\ell}$, then $f(\tilde{d},\tilde{\ell}) > 0$.)

In a similar spirit, Step 3 only needs to be implemented during the first iteration of the algorithm because Procedure 3, called in Step 11, returns parameters $\tilde{d}$ and $\tilde{\ell}$ that satisfy $f(\tilde{d},\tilde{\ell}) = 1$.

7.1 Computational Guarantees when (P) is Infeasible

In this subsection we examine the computational complexity of Algorithm 4 in the case when (P) is infeasible. We start with the following elementary proposition that bounds the slab radii in terms of the (normalized) components of d. (Proofs of the results of Section 7.1 appear in Appendix E.3.)

Proposition 7.1. *Let $d \in \mathbb{R}^m_{++}$ and $\ell \in \mathbb{R}^m$ such that $f(d,\ell) > 0$. For all $i \in [m]$ it holds that*

$$\gamma_i(d,\ell) \leq \left(\frac{d_i}{f(d,\ell)}\right)^{-\frac{1}{2}} .$$

□

It then follows from Proposition 5.2 and Proposition 7.1 that we can construct a type-L certificate of infeasibility if the entries of the normalized iterate $\frac{1}{f(d,\ell)}d$ eventually become large enough so that they satisfy

$$\left(\frac{1}{f(d,\ell)}d_i\right)^{-\frac{1}{2}} < \tau(A,u) \text{ for all } i \in [m] .$$

In order to prove that this condition will eventually hold, we first introduce the following potential function $\phi(d,\ell)$:

$$\phi(d,\ell) := \prod_{i=1}^m \max\left\{\left(\frac{1}{f(d,\ell)}d_i\right)^{-\frac{1}{2}}, \frac{m}{m+1}\tau(A,u)\right\} ,$$

and we will show in this subsection that this potential function sufficiently decreases over the iterations in the case when (P) is infeasible. For notational convenience, define

$$\mu_i(d,\ell) := \max\left\{\left(\frac{1}{f(d,\ell)}d_i\right)^{-\frac{1}{2}}, \frac{m}{m+1}\tau(A,u)\right\} ,$$

and therefore $\phi(d,\ell) = \prod_{i=1}^m \mu_i(d,\ell)$. Note that $\phi(d,\ell)$ is bounded from below, namely $\phi(d,\ell) \geq \left(\frac{m}{m+1}\tau(A,u)\right)^m$. Lemma 7.1 below states that after updating d and ℓ in Procedure 3, $\phi(d,\ell)$ decreases by at least the multiplicative factor $e^{-\frac{1}{2(m+1)}}$.

Lemma 7.1 (Potential function decrease)**.** *Let $d > 0$ and $\ell \in \mathbb{R}^m$ satisfy $f(d, \ell) > 0$, and similarly let $\tilde{d} > 0$ and $\tilde{\ell} \in \mathbb{R}^m$ satisfy $f(\tilde{d}, \tilde{\ell}) > 0$. Let $j \in [m]$ be given, and suppose that d, ℓ, $\tilde{d}$, $\tilde{\ell}$ satisfy:*

$$\frac{1}{f(\tilde{d}, \tilde{\ell})}\tilde{d} = \alpha\left(\frac{1}{f(d, \ell)}d + \frac{2}{m-1}\frac{1}{\gamma_j(d, \ell)^2}e_j\right) \ ,$$

for a scalar $\alpha \geq \frac{m^2-1}{m^2}$. If $\left(\frac{d_j}{f(d,\ell)}\right)^{-\frac{1}{2}} \geq \tau(A, u)$, then

$$\phi(\tilde{d}, \tilde{\ell}) \leq e^{-\frac{1}{2(m+1)}}\phi(d, \ell) \ .$$

□

With Lemma 7.1 in hand, we now state and prove our main computational guarantee for Algorithm 4 in the case when (P) is infeasible.

Theorem 7.1. *Let $\ell \in \mathbb{R}^m$ be certified lower bounds for (P) with certificate matrix Λ. Let $d := e \in \mathbb{R}^m$. If (P) is infeasible, Algorithm 4 with input A, u, ℓ, Λ, and d will stop and return a type-L certificate of infeasibility in at most*

$$\left\lfloor 2m(m+1)\ln\left(\frac{m+1}{2m}\frac{\|u - \ell\|}{\tau(A, u)}\right)\right\rfloor$$

iterations. □

Proof. In the notation of the theorem d and ℓ are the initial values used as input to Algorithm 4. First note that if $f(d, \ell) \leq 0$, then it follows from Proposition 5.3 that Step 2 of Algorithm 4 will return a certificate of infeasibility of (P) at the very first iteration. Also, if $\sqrt{f(e, \ell)} < \tau(A, u)$, then it follows from Proposition 7.1 that $\gamma_i(e, \ell) < \tau(A, u)$ for all $i \in [m]$, whereby for the index j in Step 4 it holds that ${a_j}^\top y(d, \ell) - u_j \geq \tau(A, u) > \gamma_j(e, \ell)$ which implies that $L_j > u_j$ in Step 9 of Algorithm 4, and so it follows from Corollary 5.1 that Algorithm 4 will return a certificate of infeasibility of (P) at Step 9 of the very first iteration. We therefore suppose for the rest of the proof that $f(e, \ell) > 0$ and $\sqrt{f(e, \ell)} \geq \tau(A, u)$.

From the definition of the potential function, it therefore holds for the initial values of $d = e$ and ℓ that $\phi(e, \ell) = \Pi_{i=1}^m \sqrt{f(e, \ell)}$. Notice that $f(e, \ell) = v(\ell)^\top Iv(\ell) - t(e, \ell)^\top It(d, \ell) \leq v(\ell)^\top Iv(\ell) = (\frac{1}{2}\|u - \ell\|)^2$, whereby $\phi(e, \ell) \leq (\frac{1}{2}\|u - \ell\|)^m$.

Suppose that Algorithm 4 has completed k iterations, and let $\hat{d}$ and $\hat{\ell}$ denote the values of d and ℓ upon completion of iteration k. It then follows from Lemma 7.1 that

$$\left(\frac{m}{m+1}\tau(A, u)\right)^m \leq \phi(\hat{d}, \hat{\ell}) \leq e^{-\frac{k}{2(m+1)}} \ \phi(d, \ell) \leq e^{-\frac{k}{2(m+1)}}(\tfrac{1}{2}\|u - \ell\|)^m \ ,$$

where the first inequality uses the absolute lower bound on $\phi(\cdot, \cdot)$ from its definition, and the second inequality uses Lemma 7.1. Taking logarithms of both sides and rearranging terms yields the inequality $k \leq 2m(m+1)\ln\left(\frac{m+1}{2m}\frac{\|u-\ell\|}{\tau(A,u)}\right)$ which proves the result. □

Corollary 7.1 specializes Theorem 7.1 to instances of linear inequality systems with box constraints (P_B) from Section 2. The corollary follows immediately from Theorem 7.1 and inequality (11).

Corollary 7.1. *Consider the linear inequality system with box constraints* (P_B)*, and let* A*,* u*,* ℓ*, and* Λ *be defined as in* (7)-(10)*. Let* $d := e \in \mathbb{R}^m$*. If* (P_B) *is infeasible, Algorithm 4 with input* A*,* u*,* ℓ*,* Λ*, and* d *will stop and return a type-L certificate of infeasibility in at most*

$$\left\lfloor 2m(m+1)\ln\left(\frac{(m+1)(\sqrt{\hat{m}}+2)}{2m}\frac{\|\bar{b}-\underline{b}\|}{\tau(A,u)}\right)\right\rfloor$$

iterations.

□

7.2 Computational Guarantees when (P) is Feasible

In this subsection we examine the computational complexity of Algorithm 4 in the case when (P) is feasible. Our analysis is in some sense standard, in that we show that upon updating the values of d and ℓ in Procedure 3, the volume of the newly updated ellipsoid $E(d,\ell)$ decreases by a sufficient amount. For $d \in \mathbb{R}^m_{++}$ and $\ell \in \mathbb{R}^m$ satisfying $f(d,\ell) > 0$, the (relative) volume of $E(d,\ell)$ is:

$$\operatorname{vol} E(d,\ell) := \frac{(f(d,\ell))^{\frac{n}{2}}}{\sqrt{\det ADA^\top}}$$

(relative in that it ignores the dimensional constant $c_n = \frac{\pi^{(n/2)}}{\Gamma(n/2+1)}$). Lemma 7.2 below states that after updating d and ℓ in Procedure 3, the volume of the ellipsoid $E(d,\ell)$ decreases by at least the multiplicative factor $e^{-\frac{1}{2(m+1)}}$.

Lemma 7.2 (Volume decrease)**.** *Let* $d > 0$ *and* $\ell \in \mathbb{R}^m$ *satisfy* $f(d,\ell) > 0$*, and similarly let* $\tilde{d} > 0$ *and* $\tilde{\ell} \in \mathbb{R}^m$ *satisfy* $f(\tilde{d},\tilde{\ell}) > 0$*. Let* $j \in [m]$ *be given, and suppose that* d*,* ℓ*,* $\tilde{d}$*,* $\tilde{\ell}$ *satisfy:*

$$\frac{1}{f(\tilde{d},\tilde{\ell})}\tilde{d} = \alpha\left(\frac{1}{f(d,\ell)}d + \frac{2}{m-1}\frac{1}{\gamma_j(d,\ell)^2}e_j\right) \ ,$$

for a scalar $\alpha \geq \frac{m^2-1}{m^2}$*. Then*

$$\operatorname{vol} E(\tilde{d},\tilde{\ell}) \leq e^{-\frac{1}{2(m+1)}} \operatorname{vol} E(d,\ell) \ .$$

□

With Lemma 7.2 in hand, we now state and prove our main computational guarantee for Algorithm 4 in the case when (P) is feasible. Note that the theorem uses the condition number $\rho(A)$, which was introduced in (16) and measures the distance to unboundedness as discussed earlier.

Theorem 7.2. *Let* $\ell \in \mathbb{R}^m$ *be certified lower bounds for* (P) *with certificate matrix* Λ*. Let* $d := e \in \mathbb{R}^m$*. If* (P) *is feasible, Algorithm 4 with input* A*,* u*,* ℓ*,* Λ*, and* d *will stop and return a feasible solution of* (P) *in at most*

$$\left\lfloor 2n(m+1)\ln\left(\frac{\|u-\ell\|}{2\rho(A)\tau(A,u)}\right)\right\rfloor$$

iterations. □

Proof. In the notation of the theorem $d = e$ and ℓ are the initial values used as input to Algorithm 4. Let us first bound the volume of the initial ellipsoid $E(d, \ell)$. From the bound on $f(e, \ell)$ in the proof of Theorem 7.1 we have:

$$\operatorname{vol} E(d,\ell) = \frac{(f(e,\ell))^{\frac{n}{2}}}{\sqrt{\det AA^\top}} \le \frac{(\frac{1}{2}\|u-\ell\|)^n}{\rho(A)^n} = \left(\frac{\|u-\ell\|)}{2\rho(A)}\right)^n \ ,$$

where the bound in the denominator above uses Proposition E.1. Suppose that Algorithm 4 has completed k iterations, and let $\hat{d}$ and $\hat{\ell}$ denote the values of d and ℓ upon completion of iteration k. Next notice that $E(\hat{d}, \hat{\ell}) \supset \mathcal{P} \supset B(c, \tau(A, u))$ for some $c \in \mathcal{P}$ where the second inclusion follows from Proposition 4.1. Therefore a lower bound on $\operatorname{vol} E(\hat{d}, \hat{\ell})$ is $\tau(A, u)^n$. It then follows from Lemma 7.2 that

$$\tau(A,u)^n \le \operatorname{vol} E(\hat{d},\hat{\ell}) \le e^{-\frac{k}{2(m+1)}} \ \operatorname{vol} E(d,\ell) \le e^{-\frac{k}{2(m+1)}} \left(\frac{\|u-\ell\|)}{2\rho(A)}\right)^n \ ,$$

where the second inequality uses Lemma 7.1 and the third inequality uses the upper bound on $\operatorname{vol} E(d, \ell)$. Taking logarithms of both sides and rearranging terms yields the inequality $k \le 2n(m+1)\ln\left(\frac{\|u-\ell\|}{2\rho(A)\tau(A,u)}\right)$ which proves the result. □

Corollary 7.2 specializes Theorem 7.2 to instances of linear inequality systems with box constraints (P_B) from Section 2. The corollary follows from Theorem 7.2, inequality (11), and the fact that if $A^\top x \le u$ contains box constraints, then $\det AA^\top > 1$ (and so $\rho(A)$ vanishes in the guarantee).

Corollary 7.2. *Consider the linear inequality system with box constraints (P_B), and let A, u, ℓ, and Λ be defined as in* (7)-(10)*. Let $d := e \in \mathbb{R}^m$. If (P_B) is feasible, Algorithm 4 with input A, u, ℓ, Λ, and d will stop and return a feasible solution of (P_B) in at most*

$$\left\lfloor 2n(m+1)\ln\left(\frac{\sqrt{\hat{m}+2}\|\bar{b}-\underline{b}\|}{2\tau(A,u)}\right)\right\rfloor$$

iterations.

□

We conclude this section by pointing to the computational complexity of OEA. Remark 7.1 states that the operations complexity of an iteration of OEA is $O(m^2)$ operations. Combining this with the iteration complexity of Theorem 7.1 and Theorem 7.2 yields the computational complexity bounds for OEA in the second row of Table 1.

8 Modified Versions of Algorithm OEA

In this section we present two modified versions of OEA, which we call OEA-No-Alt and OEA-MM, for Challenges II and I, respectively.

8.1 OEA-No-Alt

OEA-No-Alt is a simpler version of OEA that does not iteratively update the information needed to produce a type-L certificate of infeasibility. The algorithm still proves infeasibility by correctly detecting infeasibility when (P) is infeasible, but it does not produce a solution of (Alt), hence the notation "OEA-No-Alt." The modified algorithm is based on two rather elementary observations about OEA, as follows.

The first observation concerns the role of the updates of Λ in OEA. Observe that the certificate matrix Λ is never used anywhere in the computational rules in OEA nor in the updates of any objects other than Λ itself; these updates of Λ are pure "record-keeping" and their sole purpose is to eventually produce a Type-L certificate of infeasibility (a solution of (Alt)) after such infeasibility is detected and the algorithm needs no further iterations. Hence, if one is not interested in actually computing a solution of (Alt), any and all updates of Λ can be omitted. By omitting the updates of Λ the algorithm will no longer produce a solution of (Alt) in the case when (P) is infeasible, and hence we denote this simplified version of OEA as OEA-No-Alt. Nevertheless the updated values of Λ exist (but are just not computed).

A somewhat formal description of OEA-No-Alt is as follows. Instead of calling Procedure 2 in Step 2 when $f(d, \ell) \leq 0$ (implying that (P) is infeasible), OEA-No-Alt simply declares infeasibility and stops. In Step 5 there is no update of the certificate for constraint j, and Steps 6-8 are thus omitted. And instead of returning a type-L certificate of infeasibility in Step 9 when $L_j > u_j$ (implying (P) is infeasible), OEA-No-Alt simply declares infeasibility and stops. Finally, in Step 11, instead of calling Procedure 2 inside of Procedure 3 (implying (P) is infeasible), OEA-No-Alt simply declares infeasibility and stops.

Notice from the above formal description of OEA-No-Alt that the stopping criteria in the case when (P) is infeasible are identical to that in the original OEA. Hence, in the case when (P) is infeasible, OEA-No-Alt will stop when and only when it detects infeasibility exactly as in the original OEA.

The second observation concerns the operations complexity of an iteration of OEA. The computational complexity of an iteration of OEA is $O(mn)$ operations except for the updates of the certificate matrix Λ, which are $O(m^2)$ operations. Therefore, if we eliminate the updates of the matrix Λ, the operations complexity of an iteration of the resulting algorithm is $O(mn)$ operations.

The above analysis yields the following computational complexity result for OEA-No-Alt.

Corollary 8.1. *Let $\ell \in \mathbb{R}^m$ be certified lower bounds for (P) with certificate matrix Λ. Let $d := e \in \mathbb{R}^m$. If (P) is infeasible, Algorithm OEA-No-Alt with input A, u, ℓ, Λ, and d will correctly detect infeasibility, proving that (P) is infeasible, with the same iteration bound as given in Theorem 7.1. Therefore the total computational complexity of Algorithm OEA-No-Alt is $O(m^3 n \ln \frac{1}{\tau})$ operations.* □

The computational complexity bounds for OEA-No-Alt in the third row of Table 1 follow directly from the above observations.

8.2 OEA-MM

OEA-MM is very similar to OEA-No-Alt, except that it computes and stores certain information at each iteration that can be used after-the-fact to later construct the final certificate matrix Λ that would have been produced by the complete OEA algorithm. In this way, if (P) is infeasible, the final Λ can be constructed after-the-fact and used to produce the solution of (Alt) exactly as in the complete OEA. And if (P) is feasible, no certificate matrix is needed and so computing Λ is unnecessary. OEA-MM is based on the following notions.

1. Just like OEA-No-Alt, OEA-MM does not iteratively update the certificate matrix Λ, and in this aspect it is identical to OEA-No-Alt. By not updating the certificate matrix Λ at each iteration, the per-iteration complexity is reduced to $O(mn)$ operations per iteration just like in the algorithm OEA-No-Alt.

2. However, in the interest of having the capability of computing the solution of (Alt) after-the-fact that would have been computed by the complete algorithm OEA if (P) is infeasible, OEA-MM computes and stores the information needed to re-construct the certificate of infeasibility that would be computed by the complete algorithm OEA. For this reason the algorithm has the notation "-MM" for more memory.

3. If (P) is infeasible, the information stored at each iteration is then used to construct the type-L certificate of infeasibility that the complete algorithm OEA would have computed.

Before going into the details of algorithm OEA-MM, we first step back and examine certain properties of the complete algorithm OEA under the assumption that (P) is infeasible. OEA constructs a type-L certificate of infeasibility either in Step 9 of Algorithm 4 or in Step 11 of Procedure 2 (after being called by Algorithm 4). Let k be the number of iterations of algorithm OEA in which the certificate matrix is updated in Step 7 of Algorithm 4 or Step 10 of Procedure 2 (an upper bound on k is given in Theorem 7.1). Let us denote the i-th certificate matrix that OEA constructs by $\Lambda^{(i)}$ for $i \in [k]$. For consistency, we denote the initial given certificate matrix as $\Lambda^{(0)}$.

OEA updates the previous certificate matrix $\Lambda^{(i-1)}$ to the new certificate matrix $\Lambda^{(i)}$ in Step 7 of Algorithm 4 or Step 10 of Procedure 2 by first computing the relevant index $j_i := j$ in Step 4 of Algorithm 4 or $j_i := k$ in Step 5 of Procedure 2, along with the vector $\hat{\lambda}_{(i)} := \hat{\lambda}_j$ in Step 6 of Algorithm 4 or $\hat{\lambda}_{(i)} := \hat{\lambda}_k$ in Step 9 of Procedure 2. Finally, according to Step 7 of Algorithm 4 or Step 10 of Procedure 2, we obtain $\Lambda^{(i)}$ by updating $\Lambda^{(i-1)}$ which works out in full matrix form to be:

$$\Lambda^{(i)} = \Lambda^{(i-1)}[I - e_{j_i}e_{j_i}^\top + \hat{\lambda}^-_{(j_i)}e_{j_i}^\top] + [\hat{\lambda}^+_{(j_i)}e_{j_i}^\top] = \Lambda^{(i-1)}M_{(i)} + B_{(i)} \ , \tag{25}$$

where

$$M_{(i)} := I - e_{j_i}e_{j_i}^\top + \hat{\lambda}^-_{(j_i)}e_{j_i}^\top \ \text{ and } \ B_{(i)} := \hat{\lambda}^+_{(j_i)}e_{j_i}^\top \ .$$

First notice that given $\Lambda^{(0)}$ and if we have computed and stored the matrix pairs $(M_{(1)}, B_{(1)}), \ldots, (M_{(k)}, B_{(k)})$, we can construct the final certificate matrix $\Lambda^{(k)}$ by inductively using (25), and then construct the type-L certificate of infeasibility by the computation

$\bar{\lambda}_{j_k} := \Lambda^{(k)} e_{j_k} + e_{j_k}$. Next notice that it is sufficient to compute and store the vector-index pairs $(\hat{\lambda}_{(1)}, j_1), ..., (\hat{\lambda}_{(k)}, j_k)$ rather than the full matrices $(M_{(1)}, B_{(1)}), \ldots, (M_{(k)}, B_{(k)})$ because for each iteration i the information contained in the pair $(\hat{\lambda}_{(i)}, j_i)$ is sufficient to construct the matrices $(M_{(i)}, B_{(i)})$. We will refer to the sequence $\{(\hat{\lambda}_{(i)}, j_i)\}_{i \in [k]}$ as the *certificate-index sequence* of OEA.

Based on the above discussion, we obtain OEA-MM from algorithm OEA with the following modifications:

1. OEA-MM foregoes Steps 7 and 8 of Algorithm 4 and Step 10 Procedure 2. Accordingly, OEA-MM does not update the certificate matrix Λ.

2. After implementing Step 6 of Algorithm 4 and Step 9 of Procedure 2, OEA-MM stores $\hat{\lambda}_j$ and j as a pair $(\hat{\lambda}_j, j)$ in memory. These pairs comprise the certificate-index sequence $(\lambda_{(1)}, j_1), ..., (\lambda_{(k)}, j_k)$.

It follows from Theorems 7.1 and 7.2 that the number k of certificate-index pairs that need to be stored by OEA-MM satisfies $k = O(\max\{m^2 \ln(\|u - \ell\|/\tau(A, u)), mn \ln(\|u - \ell\|/(\rho(A)\tau(A, u))\})$.

Finally, notice that Step 9 of Algorithm 4 or Step 11 of Procedure 2 are where the type-L certificate of infeasibility is computed in the complete OEA. A naive (and inefficient) way to accomplish the computation in these steps in OEA-MM would be to construct the full final certificate matrix $\Lambda^{(k)}$ by inductively using (25), and then to construct the type-L certificate of infeasibility by the computation $\bar{\lambda}_{j_k} := \Lambda^{(k)} e_{j_k} + e_{j_k}$. However, we can take advantage of the inductive recursion in the construction of $\Lambda^{(k)}$ in (25) to instead compute just the type-L certificate of infeasibility $\bar{\lambda}_{j_k}$ via a sequence of k back-solves. The detailed computation is presented in Procedure 5 below.

Procedure 5 Construction of Type-L Certificate from (Stored) Certificate-Index Sequence

Input: initial certificate matrix $\Lambda^{(0)}$ and certificate-index sequence $\{(\hat{\lambda}_{(i)}, j_i)\}_{i \in [k]}$.

```
Initialize $w^k \leftarrow e_{j_k}$ and $z^k \leftarrow e_{j_k}$ .
for $i = k : 1$ do
    $w^{i-1} \leftarrow w^i + (\hat{\lambda}^-_{(i)} - e_{j_i})(e_{j_i}^\top w^i)$ .
    $z^{i-1} \leftarrow \hat{\lambda}^+_{(i)} e_{j_i}^\top w^i + z^i$ .
end for
Return $\bar{\lambda} := \Lambda^{(0)} w^0 + z^0$, and Stop.
```

The following proposition establishes the correctness of Procedure 5. The proof of Proposition 8.1 is given in Appendix E.4.

Proposition 8.1. *The output of Procedure 5 satisfies $\bar{\lambda} = \Lambda^{(k)} e_{j_k} + e_{j_k}$ and hence is a type-L certificate of infeasibility.* □

Based on this procedure, the third and final modification of OEA is as follows:

3. Once Step 9 of Algorithm 4 or Step 11 of Procedure 2 must be executed, OEA-MM instead runs Procedure 5 to construct a Type-L certificate from the initial certificate matrix $\Lambda^{(0)}$ and the certificate-index sequence $\{(\lambda_{(i)}, j_i)\}_{i\in[k]}$ stored in memory.

The computational complexity of implementing Steps 3 and 4 in Procedure 5 is $O(m)$, and Step 6 requires $O(m^2)$ operations. From the earlier discussion, the computational complexity of the number of iterations k of Procedure 5 is $O(m^2)$. Thus the computational complexity of Procedure 5 is $O(m^3)$. Finally, because OEA-MM foregoes Step 7 of Algorithm 4, the total computational complexity of OEA-MM is $O(m^3 n \ln \frac{1}{\tau})$.

The above analysis yields the following computational complexity result for OEA-MM.

Corollary 8.2. *Let $\ell \in \mathbb{R}^m$ be certified lower bounds for (P) with certificate matrix Λ. Let $d := e \in \mathbb{R}^m$. If (P) is infeasible, Algorithm OEA-MM with input A, u, ℓ, Λ, and d will stop and return a type-L certificate of infeasibility, with the same iteration bound as given in Theorem 7.1. Therefore the total computational complexity of Algorithm OEA-MM is $O(m^3 n \ln \frac{1}{\tau})$ operations.*

Finally, the computational complexity bounds for OEA-MM in the fourth row of Table 1 follow directly from the above observations.

Appendices

A Reducing the Operation Counts for an Iteration of the Ellipsoid Algorithm

The first two subsections of this appendix present two different transformations for implementing the standard ellipsoid algorithm to solve the problem (Alt), whose feasible region lies in the nullspace of A (which we denote by $Null(A)$). The first transformation preserves Euclidean distances and hence aspects of the original conditioning of (Alt), and is presented in Section A.1. The second transformation is guaranteed to reduce the operation counts per iteration of the ellipsoid algorithm from $O(mn)$ to $O(np)$ operations per iteration where $p := m - n$, but the condition measures $\tau(\cdot)$ and $\rho(\cdot)$ are changed by the transformation. This is presented in Section A.2. Section A.3 of this appendix shows how the transformation in Section A.2 can be applied when solving (P) using either the standard ellipsoid algorithm, OEA, OEA-No-Alt, or OEA-MM. This transformation is guaranteed to reduce the operation counts per iteration of these versions by the factor m/p – but the condition measures $\tau(\cdot)$ and $\rho(\cdot)$ are changed by the transformation.

A.1 QR factorization to implement the iterations of the standard ellipsoid algorithm for solving (Alt)

This subsection considers a QR factorization approach to implement the standard ellipsoid method for solving (Alt), that preserves Euclidean distances (and hence key features of problem geometry of (Alt)). Specifically, we show how to parameterize the nullspace of A (which we denote by $Null(A)$) in a way that preserves Euclidean distances in (Alt). We initially compute the QR factorization of $A^\top$: we write

$$A^\top = QR = [Y, Z]\begin{bmatrix} R_Y \\ 0 \end{bmatrix} = YR_Y \ ,$$

where Q is an $m \times m$ orthogonal matrix, partitioned into its first n and last p columns, and R is an $m \times n$ upper triangular matrix partitioned into its first n and last p rows. We then have $AZ = R_Y^\top Y^\top Z = 0$, and the columns of Z form an orthonormal basis for the nullspace of A (recall that Assumption 1.1 implies A has rank n, so that its nullspace has dimension p). Then $\{\lambda \in \mathbb{R}^m : A\lambda = 0\} = \{Z\mu : \mu \in \mathbb{R}^p\}$ and, since Z has orthonormal columns, the distance between two points λ in $Null(A)$ coincides with the distance between their corresponding μ's. This property preserves the Euclidean geometry of the problem.

To compute the factorization we calculate a sequence of elementary reflectors of the form $Q_i := I - 2w_i w_i^\top$, where each w_i is a unit vector, to reduce $A^\top$ to upper triangular form column by column, so that

$$Q_n \cdots Q_2 Q_1 A^\top = R \ , \qquad Q = Q_1 Q_2 \cdots Q_n \ .$$

This requires $O(mn^2)$ arithmetic operations. We can next if desired compute Z column by column in $O(mnp)$ operations. We can then apply the ellipsoid method to seek a point satisfying the transformed version of (Alt) which is

$$(Alt') : \quad \left\{ \begin{array}{rcl} Z\mu & \geq & 0 \\ u^\top Z\mu & < & 0 \\ \|\mu\| & \leq & 1 \ , \end{array} \right.$$

starting with the initial ellipsoid equal to the unit ball, and where we have added the unit ball constraint due to the positive homogeneity (of degree 1) of the rest of the system. At every iteration we need to evaluate the constraints at the current center. If we have computed and stored Z explicitly, this requires $O(mp)$ operations; alternatively we can augment μ with n leading zeros and apply the Q_i's sequentially to process the first two constraints of (Alt') in $O(mn)$ operations. We compute the normal vector of a violated constraint by inspecting Z, or alternatively by computing $Z^\top e_j$ or $Z^\top u$ in $O(mn)$ operations. We then proceed to update the inverse shape matrix (or a factorization of the shape matrix) in $O(p^2)$ operations. Choosing the better of the above alternatives, we see that each iteration of the standard ellipsoid algorithm applied to solve (Alt') uses $O(mp)$ operations. (Although Q differs from the identity by a matrix of rank n, we do not see how to reduce the operation count for updating the shape matrix when $n \ll m$.)

Let us also see how to implement the above computational steps of the ellipsoid method directly in the space of λ's. At a particular iteration, we suppose that we have an ellipsoid in μ-space defined by its center $\bar{\mu}$ and its inverse shape matrix M, and for simplicity we presume that the quadratic inequality right-hand side is 1). Then Z transforms this ellipsoid into the space of λ's as

$$\{\lambda \in \mathbb{R}^m : A\lambda = 0, (\lambda - \bar{\lambda})^\top (ZMZ^\top)(\lambda - \bar{\lambda}) \leq 1\} \ ,$$

where $\bar{\lambda} := Z\bar{\mu}$. We can work with this center $\bar{\lambda}$ and the "inverse shape matrix" $\hat{M} := ZMZ^\top$. (Of course, this matrix has rank p and is not invertible.) Evaluating the constraints of (Alt') at $\mu = \bar{\mu}$ corresponds exactly to evaluating the constraints $\lambda \geq 0,\ u^\top \lambda < 0$ at $\lambda = \bar{\lambda}$. A constraint normal v in μ-space corresponds to the constraint normal $\hat{v}$ in λ-space via the correspondence $v = Z^\top \hat{v}$. Furthermore, the update of the inverse shape matrix

$$M_+ = \delta \left(M - \sigma \frac{Mvv^\top M}{v^\top Mv} \right)$$

in μ-space corresponds exactly to the update of the "inverse shape matrix" in λ-space given by

$$\hat{M}_+ = \delta \left(\hat{M} - \sigma \frac{\hat{M}\hat{v}\hat{v}^\top \hat{M}}{\hat{v}^\top \hat{M}\hat{v}} \right) ,$$

where the scalar parameters δ and σ are chosen appropriate to p-dimensional space rather than m-dimensional space. The initial "inverse shape matrix" is $ZZ^\top$ and can be computed in $O(m^2 \min\{n, p\})$ operations. However, updating $\hat{M}$ at each iteration requires $O(m^2)$ operations. We summarize the above in the following.

Proposition A.1. *Using the QR factorization approach outlined above, the standard ellipsoid algorithm can be implemented to solve* (Alt) *via the transformed problem* (Alt') *starting with the unit ball in either* $Null(A)$ *or* μ*-space. Euclidean distances are preserved by the transformations involved, and hence any Euclidean ball of radius* $\bar{r}$ *in* $Null(A)$ *corresponds to a Euclidean ball of radius* $\bar{r}$ *in* μ*-space. Each iteration of the ellipsoid algorithm uses* $O(mp)$ *operations in* μ*-space and* $O(m^2)$ *operations in* $Null(A)$. □

Remark A.1. Because the QR factorization approach preserves Euclidean distances, it follows from Proposition A.1 and Lemma C.1 that the total number of operations required to compute a solution of (Alt) using the standard ellipsoid algorithm in μ-space is $O(mp^3 \ln(\frac{m}{\rho(A)\tau(A,u)}))$. This bound is shown in the last column of the first row of Table 1.

A.2 A matrix partition factorization to reduce the operations count of an iteration of the standard ellipsoid algorithm for solving (Alt)

This subsection considers a matrix partitioning approach to implement the standard ellipsoid algorithm for solving (Alt) that reduces the operation counts of an iteration of the ellipsoid algorithm, albeit at the possible expense of worsening the conditioning of the transformed problem. We will parameterize $Null(A)$ in a way that will decrease the per-iteration operations of the standard ellipsoid algorithm for the unbalanced cases in which $n \ll m$ or $p \ll m$. We can partition A (assuming, for simplicity, that the leading $n \times n$ submatrix is nonsingular) as $[A_B, A_N]$. Then the nullspace of A can be represented as

$$\left\{ \lambda = \begin{pmatrix} \lambda_B \\ \lambda_N \end{pmatrix} = \begin{bmatrix} H \\ -I \end{bmatrix} \mu : \mu \in \mathbb{R}^p \right\} ,$$

where

$$H := A_B^{-1} A_N \ . \tag{26}$$

Computing H takes $O(mn^2)$ operations, but needs to be done only once. Here $\mu \in \mathbb{R}^p$ again parametrizes the subspace, but now Euclidean distance is not preserved between corresponding pairs of points in $Null(A)$ and μ-space (unless $A_N = 0$). Consider the following transformed version of (Alt):

$$\begin{aligned} H\mu &\geq 0 \\ -\mu &\geq 0 \\ (u_B^\top A_B^{-1} A_N - u_N^\top)\mu &< 0 \\ \|\mu\| &\leq 1 \ , \end{aligned}$$

where we have similarly partitioned u into $(u_B; u_N)$, and have added the unit ball constraint due to the positive homogeneity (of degree 1) of the rest of the system just as we did earlier. We can apply the ellipsoid method to seek a point satisfying the above system starting with the initial ellipsoid equal to the unit ball centered at the origin. For convenience, let us write the above transformed system as

$$(Alt'') \left\{ \begin{array}{rl} -H\mu & \leq 0 \\ \bar{g}^\top \mu & < 0 \\ \mu & \leq 0 \\ \|\mu\| & \leq 1 \ , \end{array} \right.$$

where $\bar{g} = (A_N^\top A_B^{-\top} u_B - u_N)$, and notice that by defining the $p \times (n+1)$ matrix $G := [-H^\top, \bar{g}]$, then $G^\top \mu$ is comprised of the left-hand side of the first $(p+1)$ *general* inequalities of (Alt'') above. Since (Alt'') contains $m+1$ linear inequalities in p variables in addition to the unit ball constraint, each iteration would normally require $O(mp)$ operations to evaluate the constraints at the current center, and $O(p^2)$ operations to apply the inverse of the shape matrix to a constraint vector v of the above system and to update the inverse or a Cholesky factorization of the $p \times p$ shape matrix. However, because there are only $n+1$ general inequalities (which comprise the matrix G), both of these counts can be reduced to $O(np)$. This is immediately apparent for evaluating the constraints because evaluating $\mu \leq 0$ requires $O(p)$ operations. Below we show how to perform the other tasks in $O(np)$ operations when $n < p$; otherwise, we have $p^2 = O(np)$ already.

Recall from above that our initial ellipsoid is a unit ball; accordingly, the initial shape matrix is the identity matrix. At each iteration of the ellipsoid algorithm, we add a multiple of a rank-one matrix of the form $vv^\top$ to the current shape matrix and then positively rescale the shape matrix, where v is a constraint vector from (Alt''). Notice that we can presume that v corresponds to one of the rows of the linear inequalities in (Alt''), since if the only violated constraint at the current center is the unit ball constraint $\|\mu\| \leq 1$, then the center is indeed a solution of (Alt) and we are done. Hence, at every iteration the shape matrix is of the form

$$B = E + GDG^\top, \tag{27}$$

where E and D are positive semidefinite diagonal matrices of order p and $n+1$, respectively. Note that E is nonsingular, and we will without loss of generality assume that D is as well because we can restrict our attention to the columns of G that correspond to the positive diagonal entries of D. Clearly we can apply B to any vector at a cost of only $O(np)$ operations with this form. By the Inverse Matrix Modification Formula (commonly called the Sherman-Morrison-Woodbury formula, but due to earlier work by Guttman and Duncan — see Hager [7]) ,

$$B^{-1} = E^{-1} - E^{-1}G(D^{-1} + G^\top E^{-1}G)^{-1}G^\top E^{-1} .$$

Hence, if we have the inverse or a factorization of the inner matrix

$$J := D^{-1} + G^\top E^{-1}G ,$$

we can also apply B^{-1} to any vector in only $O(np)$ operations. It remains to verify that we can update the shape matrix in $O(np)$ operations. At each iteration, we either increase a single entry of D or E, and then scale the resulting matrix. If we keep the scalings separate, we have just a change of a single entry of D^{-1} or E^{-1} in J, which leads to a rank-one update. We can therefore update the inverse or a factorization of J in just $O(n^2)$ operations. (The case when a column is added to G, corresponding to a diagonal entry of D increasing from zero, can also be handled in $O(n^2)$ operations, but we omit the details.)

Thus, all the steps of an iteration of the ellipsoid method applied to (Alt'') require just $O(np)$ operations. We summarize the above discussion in the following proposition.

Proposition A.2. *Using the matrix partitioning approach outlined above, the standard ellipsoid algorithm can be implemented to solve (Alt) via the transformed problem (Alt'') starting with the unit ball in μ-space. Each iteration of the ellipsoid algorithm in μ-space can be implemented using $O(np)$ operations. Euclidean distances are not preserved by the transformations involved, and hence a Euclidean ball of radius $\bar{r}$ in Null(A) does not necessarily correspond to a Euclidean ball of radius $\bar{r}$ in μ-space.* □

Remark A.2. Because the matrix partitioning approach requires $O(np)$ operations per iteration, the total number of operations required to compute a solution of (Alt) using the standard ellipsoid algorithm applied to (Alt'') is $O(np^3 \ln(\frac{m}{\rho(\tilde{A})\tau(\tilde{A},u)}))$, where $\hat{A} := [I, A_B^{-1}A_N]$ is the transformed data matrix and $\tilde{A}$ is a rescaling of $\hat{A}$ so that its columns have unit norm. Thus while this approach reduces the per-iteration operations by the factor m/n, the values of the problem condition measures $\rho(\cdot)$ and $\tau(\cdot)$ are changed by the transformation.

A.3 A matrix partition factorization to reduce the operations count of an iteration of the standard ellipsoid algorithm for solving (P)

Let us now consider the matrix partition factorization of Section A.2 applied to using the ellipsoid algorithm to solve (P). Because (P) contains m inequalities in n variables, a straightforward implementation of the ellipsoid algorithm uses $O(mn)$ operations to evaluate the constraints at the current center, and $O(n^2)$ operations to apply the inverse of the shape matrix to the vector v of the current violated constraint ($v = a_j$ for some $j \in \{1, \ldots, m\}$), and to update the inverse or a factorization of the shape matrix. Applying the matrix partition approach to (P) and using identical notation as in Section A.2, we can write (P) as

$$\begin{aligned} A_B^T x &\le u_B \\ A_N^T x &\le u_N \ , \end{aligned}$$

and hence we can transform (P) into the following system (P') that is defined in terms of the linearly transformed variables $z := A_B^T x$:

$$(P') : \begin{cases} \quad z \le u_B \\ H^T z \le u_N \ , \end{cases}$$

where H is given by (26) above. Notice that the invertible linear transformation $z := A_B^T x$ is not guaranteed to preserve Euclidean distances. The system (P') is comprised of the n inequalities $z \le u_B$ and p general inequalities in n variables. It follows immediately that we can evaluate the constraints at the current center in $O(np)$ operations. When $p < n$, we can apply the inverse of the shape matrix to a constraint vector and update the inverse or a Cholesky factorization of the shape matrix in $O(np)$ operations by using the techniques described above for B in (27). And when $n < p$, we have $n^2 = O(np)$ already.

It follows from the above discussion that the standard ellipsoid algorithm, OEA-No-Alt, and OEA-MM all require $O(np)$ operations per iteration. To claim a similar result for OEA, we need to show how to update the certificate matrix Λ in $O(mp)$ operations per iteration. It is not hard to see that because of the special form of the reformulated constraint matrix, it suffices to store and update the last p rows of Λ. If infeasibility is detected, the first n rows of Λe_j can be obtained by using the defining equation $A\Lambda = -A$. As a result, we can perform each iteration of OEA in $O(mp)$ operations. We summarize this discussion in the following.

Proposition A.3. *Using the matrix partitioning approach above, the standard ellipsoid algorithm, OEA, OEA-No-Alt, and OEA-MM can be implemented to solve (P) via the transformed problem (P'). Each iteration of the standard ellipsoid algorithm, OEA-No-Alt, and OEA-MM can be implemented using $O(np)$ operations, while each iteration of OEA can be implemented using $O(mp)$ operations. Euclidean distances are not preserved by the transformations involved, and hence a Euclidean ball of radius $\bar{r}$ in z-space does not necessarily correspond to a Euclidean ball of radius $\bar{r}$ in x-space.*

Remark A.3. Similar to Remark A.2, the total number of operations required to compute a solution of (P) using the standard ellipsoid algorithm, OEA-No-Alt, or OEA-MM applied to (P') is $O(n^3 p \ln(\frac{m}{\rho(\tilde{A})\tau(\tilde{A},u)}))$, and for OEA it is $O(mn^2 p \ln(\frac{m}{\rho(\tilde{A})\tau(\tilde{A},u)}))$. Here $\hat{A} := [I, H^\top] = [I, A_B^{-1} A_N]$ is the transformed data matrix and $\tilde{A}$ is a rescaling of $\hat{A}$ so that its columns have unit norm. This approach reduces the per-iteration operations by the factor m/p, but the values of the problem condition measures $\rho(\cdot)$ and $\tau(\cdot)$ are changed by the transformation.

B Connection between $\tau(A, u)$ and Renegar's distance to ill-posedness $\bar{\rho}(d)$

The paper [9] by Renegar develops a rather complete data-perturbation-theoretic condition measure theory for conic optimization using a data-dependent measure $\bar{\rho}$ that is naturally tied to a variety of geometric, analytic, numerical, and algorithmic properties of conic optimization problems. We will show below that $\tau(A, u) \geq \bar{\rho}(A, u)$, but first we need to establish the setting and then give a formal definition of $\bar{\rho}$.

The condition measure $\bar{\rho}$ is concerned with data-instance-specific conic systems and their state changes as the data is perturbed. Restricting our discussion to the case of linear inequality systems of the form (P), let us define the data $d = (A, u) \in \mathbb{R}^{n\times m} \times \mathbb{R}^m$ and $\mathcal{P}_d := \mathcal{P}_{A,u} := \{x \in \mathbb{R}^n : A^\top x \leq u\}$. (We slightly abuse notation in calling the data d in order to be consistent with the notation used in the condition measure theory.) The feasible and infeasible data instances are then defined as $\mathcal{F} := \{d \in \mathbb{R}^{n\times m} \times \mathbb{R}^m : \mathcal{P}_d \neq \emptyset\}$ and $\mathcal{I} := \{d \in \mathbb{R}^{n\times m} \times \mathbb{R}^m : \mathcal{P}_d = \emptyset\}$. We will define the following norm on the data: $\|d\| := \|(A, u)\| := \max\{\|A\|_{1,2}, \|u\|_\infty\}$ where recall that the operator norm of a matrix M is $\|M\|_{1,2} = \max_{\|v\|_1=1} \|Mv\|_2$. The condition measure $\bar{\rho}(d)$ is then defined as:

$$\bar{\rho}(d) := \begin{cases} \inf_{d+\Delta d\in\mathcal{I}} \|\Delta d\| & \text{if } d \in \mathcal{F} \\ \\ \inf_{d+\Delta d\in\mathcal{F}} \|\Delta d\| & \text{if } d \in \mathcal{I} \end{cases},$$

which is essentially the size of the smallest data perturbation $\Delta d = (\Delta A, \Delta u)$ for which $\mathcal{P}_{d+\Delta d}$ changes from nonempty to empty, or *vice versa*. $\bar{\rho}(d)$ is called the "distance to ill-posedness" because the optimal or nearly-optimal perturbed data $d + \Delta d$ lies on the set of ill-posed instances $\partial\mathcal{F} = \partial\mathcal{I}$. It is simple to show that $\bar{\rho}(d)$ and $\tau(A, u)$ are related as follows:

$$\tau(A, u) \geq \bar{\rho}(A, u) \ . \tag{28}$$

To see this, first consider the case when $d = (A, u) \in \mathcal{F}$, and define $\Delta A = 0$ and $\Delta u = (-\tau(A, u) - \varepsilon)e$, and notice from the definition of $\tau(A, u)$ that $\mathcal{P}_{d+\Delta d} = \emptyset$ for all $\varepsilon > 0$, whereby $\bar{\rho}(d) \leq \|\Delta d\| = \tau(A, u) + \varepsilon$, and it then follows that $\bar{\rho}(d) \leq \tau(A, u)$. Next consider the case when $d = (A, u) \in \mathcal{I}$, and define $\Delta A = 0$ and $\Delta u = \tau(A, u)e$, and notice from the definition of $\tau(A, u)$ that $\mathcal{P}_{d+\Delta d} \neq \emptyset$, whereby $\bar{\rho}(d) \leq \|\Delta d\| = \tau(A, u)$.

The inequality (28) is the "good" direction for complexity of the ellipsoid method, since the computational complexity shown herein is $O(\ln(1/\tau(A, u))) \leq O(\ln(1/\bar{\rho}(A, u)))$, which automatically bounds the computational complexity of the ellipsoid method in terms of $\bar{\rho}(A, u)$.

C Iteration Complexity of a Standard Ellipsoid Algorithm for Computing a Solution of (Alt)

In the case when (P) is infeasible, we derive a bound on the iteration complexity of computing a solution of (Alt) using the standard ellipsoid algorithm, that depends only on m, $\tau(A, u)$, and $\rho(A)$. Let $Null(A)$ denote the nullspace of A and let P_A denote the ℓ_2 projection matrix onto $Null(A)$, namely $P_A = I - A^T(AA^T)^{-1}A$. Because (Alt) is positively homogeneous, we can augment (Alt) by adding a unit ℓ_2 ball constraint:

$$(Alt''') : \begin{cases} A\lambda & = & 0 \\ \lambda & \geq & 0 \\ u^\top\lambda & < & 0 \\ \|\lambda\| & \leq & 1 \; , \end{cases}$$

and then solve (Alt''') using the standard ellipsoid algorithm in $Null(A)$ (which is a $p := (m-n)$-dimensional subspace of $\mathbb{R}^m$) starting with the unit ball as the starting ellipsoid. Appendix A describes how the algorithm can be implemented. Let $\mathcal{D}$ denote the set of solutions of (Alt'''). Recall from (15) and (16) the definitions of the condition measure $\tau(A,u)$ which measures just how infeasible the system (P) is, and $\rho(A)$ which measures how "bounded" are the inequalities in (P) in the sense of how much their normals must be perturbed in order to have a non-trivial recession cone.

Let $B(c,r)$ denote the ℓ_2 ball in $\mathbb{R}^m$ centered at c with radius r. Critical to bounding the number of iterations of the ellipsoid algorithm when solving (Alt''') is the existence of a ball $B(\lambda^c, r)$ that satisfies

$$\lambda^c \in Null(A) \quad \text{and} \quad B(\lambda^c, r) \cap Null(A) \subset \mathcal{D} \; .$$

If this is the case, then the standard ellipsoid algorithm will need at most $\lceil 2p(p+1)\ln(1/r)\rceil$ iterations to compute a solution of $\mathcal{D}$. It turns out that we can bound the radius r of the largest such ball from above and below in terms of $\tau(A,u)$ and $\rho(A)$ as the following lemma indicates.

Lemma C.1. *In the case when (P) is infeasible, let r^* be the supremum of r for which there exists λ^c satisfying*

$$\lambda^c \in Null(A) \quad \textit{and} \quad B(\lambda^c, r) \cap Null(A) \subset \mathcal{D} \; . \tag{29}$$

Then the following holds:

$$\frac{\|P_A u\|}{\tau(A,u)\sqrt{m}} + 1 \;\; \leq \;\; \frac{1}{r^*} \;\; \leq \;\; \left(\frac{\|P_A u\|}{\tau(A,u)} + 1\right)\left(\frac{m}{\rho(A)} + \sqrt{m} + 1\right) \; . \tag{30}$$

□

For clarity, these bounds can be weakened to:

$$\frac{\|P_A u\|}{\tau(A,u)\sqrt{m}} \;\leq\; \frac{1}{r^*} - 1 \;\leq\; \frac{(4m+1)\|P_A u\|}{\tau(A,u)\rho(A)} \tag{31}$$

(this follows since $\rho(A) \leq 1$ and $\tau(A,u) \leq \|P_A u\|$, see below), whereby we see that $\tau(A,u)$ approximates r^* to within a factor of $m/\rho(A)$. By combining Lemma C.1 with Proposition A.1, we obtain the following bound for the standard ellipsoid algorithm for computing a solution of $\mathcal{D}$, where $p := m - n$ is the dimension of $Null(A)$.

Theorem C.1. *In the case when (P) is infeasible, the number of iterations of the standard ellipsoid algorithm applied to solve $\mathcal{D}$ starting with the unit ball in Null(A) is at most*

$$\left\lceil 2p(p+1)\ln\left(\frac{(4m+2)\|P_A u\|}{\tau(A,u)\rho(A)}\right)\right\rceil \ .$$

Furthermore, each iteration of the ellipsoid method can be implemented using $O(mp)$ operations.

□

In the case when (P) is infeasible, the linear optimization problem describing $\tau(A,u)$, together with its dual problem, can be written as:

$$\begin{array}{rcllcrrcl} \tau(A,u) & = & \min\limits_{x,\tau} & \tau & & = & \max\limits_{\lambda} & -u^T\lambda & \\ & & \text{s.t.} & A^T x \ \le \ u + e\tau & & & \text{s.t.} & A\lambda & = \ 0 \\ & & & & & & & e^T\lambda & = \ 1 \\ & & & & & & & \lambda & \ge \ 0 \ , \end{array} \tag{32}$$

where we call the left-side and right-side problems above the primal and dual, respectively. Notice that $(x,\tau) = ((AA^T)^{-1}Au, \|P_A u\|)$ is feasible for the primal problem, and hence $\tau(A,u) \le \|P_A u\|$. Thus a natural condition measure for infeasibility is $\|P_A u\|/\tau(A,u)$ which is (positively) scale invariant and satisfies $\|P_A u\|/\tau(A,u) \in [1,\infty)$. Also, because the columns of A have unit norm, it follows that $\|A\|_{1,2} = 1$ and $\rho(A) \le 1$ and hence $1/\rho(A)$ is a natural condition measure that satisfies $1/\rho(A) \in [1,\infty)$.

If we normalize u for discussion's sake so that $\|P_A u\| = 1$ and we ignore constants, then the left inequality in (31) states that $\tau(A,u)\sqrt{m}$ is an upper bound on the largest ball radius of a ball in $\mathcal{D}$. And the right inequality in (31) states that the largest such ball radius r^* is only guaranteed to be as large as $\tau(A,u)\rho(A)/m$. Hence when m is of the same order as $p = m-n$ and $m/\rho(A) \gg 1/\tau(A,u)$, then OEA will have a better complexity bound than the bound for the standard ellipsoid algorithm in Theorem C.1. Indeed because the complexity bound for OEA in the infeasible case relies only on $\tau(A,u)$ and m, and has no dependence on $\rho(A)$, it can outperform the standard ellipsoid algorithm in terms of its iteration complexity bound.

Proof of Lemma C.1: To prove the left side of (30), let λ, r^* satisfy the limiting supremum in (29), which therefore satisfy $A\lambda = 0$, $\|\lambda\| + r^* = 1$, and

$$Ad = 0 \ , \ \|d\| \le 1 \ \Longrightarrow \ u^T(\lambda + r^* d) \le 0 \text{ and } \lambda + r^* d \ge 0 \ .$$

It therefore follows using $d = P_A u/\|P_A u\|$ that $-u^T\lambda \ge r^* u^T d = r^* u^T P_A u/\|P_A u\| = r^*\|P_A u\|$. Furthermore, setting $\lambda' := \lambda/e^T\lambda$, it follows that λ' is feasible for the dual problem in (32) and hence

$$\tau(A,u) \ge -u^T\lambda' \ge \frac{r^*\|P_A u\|}{e^T\lambda} \ge \frac{r^*\|P_A u\|}{\sqrt{m}\|\lambda\|} = \frac{r^*\|P_A u\|}{\sqrt{m}(1-r^*)} \ ,$$

and rearranging the terms above yields the left side of (30).

To prove the right side of (30), we proceed as follows. Define $\bar{e} := \frac{1}{m}e$, and for $\rho < \rho(A)$ define the perturbation matrix $\Delta A := \frac{\rho}{\|A\bar{e}\|}A\bar{e}e^T$. It then follows that $\|\Delta A\|_{1,2} = \rho < \rho(A)$, whereby from the definition of $\rho(A)$ in (16) there does not exist $v \neq 0$ satisfying $[A+\Delta A]^T v \leq 0$. It then follows from a theorem of the alternative that there exists $\lambda \in \mathbb{R}^m$ satisfying:

$$[A + \Delta A]\lambda = 0 \ , \ \lambda \geq 0 \ , \ \|\lambda\| = 1 \ . \tag{33}$$

Define $\tilde{\lambda} := \lambda + \frac{\rho e^T\lambda}{\|A\bar{e}\|}\bar{e}$, and it follows from (33) that

$$A\tilde{\lambda} = 0 \ , \ \tilde{\lambda} \geq 0 \ , \ \|\tilde{\lambda}\| \leq 1 + \frac{\rho e^T\lambda}{\sqrt{m}\|A\bar{e}\|} \ , \ \tilde{\lambda}_j \geq \frac{\rho e^T\lambda}{m\|A\bar{e}\|} \ \text{ for } j \in [m] \ . \tag{34}$$

Next let $\check{\lambda}$ solve the dual problem in (32), whereby $\check{\lambda}$ satisfies

$$A\check{\lambda} = 0 \ , \ \check{\lambda} \geq 0 \ , \ e^T\check{\lambda} = 1 \ , \ -u^T\check{\lambda} = \tau(A, u) \ . \tag{35}$$

Now the idea is to take a nonnegative combination of $\check{\lambda}$ and $\tilde{\lambda}$ in a way that guarantees that resulting combination satisfies the inequalities of $\mathcal{D}$ as much as possible. We accomplish this by defining:

$$\bar{\lambda} := \check{\lambda} + \alpha\tilde{\lambda} \quad \text{and} \quad \bar{r} := \frac{\alpha\rho e^T\lambda}{m\|A\bar{e}\|} \ , \quad \text{where} \ \ \alpha := \frac{\tau(A,u)m\|A\bar{e}\|}{\|P_A u\|(m\|\tilde{\lambda}\|\|A\bar{e}\| + \rho e^T\lambda)} \ . \tag{36}$$

Notice that $A\bar{\lambda} = 0$. We will now show that $\bar{\lambda}$ and $\bar{r}$ also satisfy:

$$Ad = 0 \ , \ \|d\| \leq 1 \implies \bar{\lambda} + \bar{r}d \geq 0 \text{ and } u^T(\bar{\lambda} + \bar{r}d) \leq 0 \ . \tag{37}$$

This then implies that upon rescaling $\bar{\lambda}$ and $\bar{r}$ by $(\|\bar{\lambda}\| + \bar{r})$ to

$$\bar{\lambda}' := \frac{\bar{\lambda}}{\|\bar{\lambda}\| + \bar{r}} \quad \text{and} \quad \bar{r}' := \frac{\bar{r}}{\|\bar{\lambda}\| + \bar{r}} \ , \tag{38}$$

for all sufficiently small positive ε it holds that

$$\bar{\lambda}' \in Null(A) \quad \text{and} \quad B(\bar{\lambda}', \bar{r}' - \varepsilon) \cap Null(A) \subset \mathcal{D} \ , \tag{39}$$

and the proof will be completed by then showing a relevant lower bound on the value of $\bar{r}'$. Let us therefore now show (37). It follows from (34), (35), and (36) that $\bar{\lambda}_j \geq \alpha\tilde{\lambda}_j \geq \frac{\alpha\rho e^T\lambda}{m\|A\bar{e}\|} = \bar{r}$, and hence for d satisfying $Ad = 0$ and $\|d\| \leq 1$ it holds that $\bar{\lambda}_j + \bar{r}d_j \geq \bar{r}(1 - \|d\|) \geq 0$ and hence $\bar{\lambda} + \bar{r}d \geq 0$. Again considering d satisfying $Ad = 0$ and $\|d\| \leq 1$, it also follows from (34), (35), and (36) that

$$\begin{aligned}
u^T(\bar{\lambda} + \bar{r}d) &= u^T(\check{\lambda} + \alpha\tilde{\lambda} + \bar{r}d) \\
&= -\tau(A,u) + \alpha u^T P_A\tilde{\lambda} + \bar{r}u^T P_A d \\
&\leq -\tau(A,u) + \alpha\|P_A u\|\|\tilde{\lambda}\| + \bar{r}\|P_A u\| \\
&= -\tau(A,u) + \alpha\|P_A u\|\|\tilde{\lambda}\| + \tfrac{\alpha\rho e^T\lambda\|P_A u\|}{m\|A\bar{e}\|} \\
&= -\tau(A,u) + \alpha\left(\tfrac{\|P_A u\|(m\|\tilde{\lambda}\|\|A\bar{e}\| + \rho e^T\lambda)}{m\|A\bar{e}\|}\right) \\
&= -\tau(A,u) + \tau(A,u) = 0 \ ,
\end{aligned}$$

thus demonstrating (37). Applying the rescaling in (38), it indeed follows that (39) holds. Therefore $r^* \geq \bar{r}' - \varepsilon$ for all sufficiently small positive ε, and hence $r^* \geq \bar{r}'$ and

$$
\begin{aligned}
\frac{1}{r^*} \leq \frac{1}{\bar{r}'} &= 1 + \frac{\|\bar{\lambda}\|}{\bar{r}} \\
&\leq 1 + \frac{\|\check{\lambda}\| + \alpha\|\tilde{\lambda}\|}{\bar{r}} \\
&\leq 1 + \frac{1}{\bar{r}} + \frac{\alpha}{\bar{r}}\left(1 + \frac{\rho e^T\lambda}{\sqrt{m}\|A\bar{e}\|}\right) \\
&= 1 + \frac{1}{\bar{r}} + \frac{\alpha}{\bar{r}} + \frac{\alpha\rho e^T\lambda}{\bar{r}\sqrt{m}\|A\bar{e}\|} \\
&= 1 + \frac{m\|A\bar{e}\|}{\alpha\rho e^T\lambda} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \sqrt{m} \\
&= 1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \frac{m\|A\bar{e}\|\|P_A u\|}{\tau(A,u)\rho e^T\lambda}\left(\frac{m\|\tilde{\lambda}\|\|A\bar{e}\| + \rho e^T\lambda}{m\|A\bar{e}\|}\right) \\
&= 1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \frac{m\|A\bar{e}\|\|P_A u\|}{\tau(A,u)\rho e^T\lambda}\left(\|\tilde{\lambda}\| + \frac{\rho e^T\lambda}{m\|A\bar{e}\|}\right) \\
&= 1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \frac{\|P_A u\|}{\tau(A,u)}\left(\frac{m\|A\bar{e}\|}{\rho e^T\lambda}\|\tilde{\lambda}\| + 1\right) \\
&\leq 1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \frac{\|P_A u\|}{\tau(A,u)} + \frac{\|P_A u\| m\|A\bar{e}\|}{\tau(A,u)\rho e^T\lambda}\left(1 + \frac{\rho e^T\lambda}{\sqrt{m}\|A\bar{e}\|}\right) \\
&= 1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda} + \frac{\|P_A u\|}{\tau(A,u)}\left(1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda}\right) \\
&= \left(1 + \sqrt{m} + \frac{m\|A\bar{e}\|}{\rho e^T\lambda}\right)\left(1 + \frac{\|P_A u\|}{\tau(A,u)}\right) \\
&\leq \left(1 + \sqrt{m} + \frac{m}{\rho}\right)\left(1 + \frac{\|P_A u\|}{\tau(A,u)}\right) .
\end{aligned}
$$

As this inequality holds for all $\rho < \rho(A)$, it also holds for $\rho = \rho(A)$, proving the right-side inequality in (30). □

D Two Update Formulas

Propositions D.1 and D.2 below are straightforward results that follow from the Sherman-Morrison formula and algebraic manipulation.

Proposition D.1. *Suppose $d \in \mathbb{R}^m_{++}$ and $\ell \in \mathbb{R}^m$ satisfy $f(d,\ell) = 1$. Let $j \in [m]$, $\delta \in \mathbb{R}_+$, and $\tilde{d} = d + \delta e_j$. Then,*

$$B(\tilde{d})^{-1} = B^{-1} - \left(\frac{\delta}{1 + \delta\gamma_j(d,\ell)^2}\right) B^{-1}a_j a_j^T B^{-1}, \tag{40}$$

$$t(\tilde{d},\ell) = t - \left(\frac{\delta}{1+\delta\gamma_j(d,\ell)^2}\right) t_j A^T B^{-1} a_j, \tag{41}$$

and

$$f(\tilde{d},\ell) = 1 + \delta v_j(\ell)^2 - \left(\frac{\delta}{1+\delta\gamma_j(d,\ell)^2}\right) t_j(d,\ell)^2. \tag{42}$$

Proof. For notational convenience, we suppress the dependence on d and l in the quantities $y(\cdot,\cdot)$, $t(\cdot,\cdot)$, and $f(\cdot,\cdot)$, and we write y for $y(d,\ell)$ and $\tilde{y}$ for $y(\tilde{d},\ell) = y(d+\delta e_j,\ell)$, and similarly for t and f; we also write B for $B(d)$ and $\tilde{B}$ for $B(\tilde{d})$. Let $\theta := \frac{\delta}{1+\delta\gamma_j(d,\ell)^2}$. From the Sherman-Morrison formula, (40) holds, and hence

$$\begin{aligned}
\tilde{y} &= \tilde{B}^{-1} A\tilde{D} r \\
&= \left(B^{-1} - \theta B^{-1} a_j a_j^\top B^{-1}\right)(ADr + \delta r_j a_j) \\
&= y + (\delta r_j - \theta a_j^\top y - \theta\delta r_j \gamma_j^2) B^{-1} a_j \\
&= y + \theta(r_j - a_j^\top y) B^{-1} a_j \\
&= y - \theta t_j B^{-1} a_j.
\end{aligned} \tag{43}$$

It follows from (43) that $\tilde{t} = A^\top \tilde{y} - r = t - \theta t_j A^\top B^{-1} a_j$, and thus we have (41). Hence,

$$\begin{aligned}
\tilde{t}^\top \tilde{D} \tilde{t} &= (t - \theta t_j A^\top B^{-1} a_j)^\top (D + \delta e_j e_j^\top)(t - \theta t_j A^\top B^{-1} a_j) \\
&= t^\top D t + \theta^2 t_j^2 \gamma_j^2 + \delta t_j^2 - 2\delta\theta t_j^2 \gamma_j^2 + \delta\theta^2 t_j^2 \gamma_j^4 \\
&= t^\top D t + \theta t_j^2 \ ,
\end{aligned} \tag{44}$$

where the second equality follows from $ADt = 0$. Also,

$$v^\top \tilde{D} v = v^\top (D + \delta e_j e_j^\top) v = v^\top D v + \delta v_j^2 \ . \tag{45}$$

From (44) and (45),

$$\begin{aligned}
f(\tilde{d},\ell) &= v^\top D v - t^\top D t + \delta v_j^2 - \theta t_j^2 \\
&= f(d,\ell) + \delta v_j^2 - \theta t_j^2 \\
&= 1 + \delta v_j^2 - \theta t_j^2 \ ,
\end{aligned}$$

and thus (42) holds. □

Proposition D.2. *Suppose $d \in \mathbb{R}^m_{++}$ and $\ell \in \mathbb{R}^m$. Let $j \in [m]$, $\beta \in \mathbb{R}$, and $\tilde{\ell} = \ell + \beta e_j$. Then*

$$y(d,\tilde{\ell}) = y(d,\ell) + \tfrac{1}{2}\beta d_j (B(d))^{-1} a_j, \tag{46}$$

$$t(d,\tilde{\ell}) = t(d,\ell) + \tfrac{1}{2}\beta d_j A^\top (B(d))^{-1} a_j - \tfrac{1}{2}\beta e_j, \tag{47}$$

$$f(d,\tilde{\ell}) = f(d,\ell) + \beta(t_j(d,\ell) - v_j(\ell)) d_j + \tfrac{1}{4}\beta^2 d_j^2 a_j^\top B(d)^{-1} a_j \ . \tag{48}$$

Proof. For notational convenience, we suppress the dependence on d and l in the quantities $y(\cdot,\cdot)$, $t(\cdot,\cdot)$, $f(\cdot,\cdot)$, $r(\cdot,\cdot)$, and $v(\cdot,\cdot)$, and we write y for $y(d,\ell)$ and $\tilde{y}$ for $y(d,\tilde{\ell}) = y(d,\ell+\beta e_j)$, and similarly for t, f, r, and v; we also write B for $B(d)$. Because $\tilde{r} = r + (\beta/2) e_j$,

$$\tilde{y} = B^{-1} AD\tilde{r} = y + \frac{\beta}{2} d_j B^{-1} a_j.$$

Thus, equation (46) holds. From (46),

$$\tilde{t} = A^\top \tilde{y} - \tilde{r} = t + \frac{\beta}{2} d_j A^\top B^{-1} a_j - \frac{\beta}{2} e_j,$$

and so equation (47) holds. It follows from (47) that

$$\begin{aligned} &\tilde{t}^\top D \tilde{t} \\ &= (t + \frac{\beta}{2} d_j A^\top B^{-1} a_j - \frac{\beta}{2} e_j)^\top D (t + \frac{\beta}{2} d_j A^\top B^{-1} a_j - \frac{\beta}{2} e_j) \\ &= t^\top D t - \beta t_j d_j + \frac{1}{4}\beta^2 (d_j A^\top B^{-1} a_j - e_j)^\top D (d_j A^\top B^{-1} a_j - e_j) \\ &= t^\top D t - \beta t_j d_j + \frac{1}{4}\beta^2 (d_j - d_j^2 a_j^\top B^{-1} a_j), \end{aligned} \tag{49}$$

where the second equality is from $ADt = 0$. Because $\tilde{v} = v - (\beta/2) e_j$,

$$\tilde{v}^\top D \tilde{v} = v^\top D v - \beta v_j d_j + \frac{1}{4}\beta^2 d_j. \tag{50}$$

From (49) and (50),

$$f(d, \tilde{\ell}) = f(d, \ell) + \beta d_j (t_j - v_j) + \frac{1}{4}\beta^2 d_j^2 a_j^\top B^{-1} a_j \ ,$$

and thus (48) holds. □

E Proofs of Results

E.1 Proofs for Section 5

Proof of Proposition 5.1. We need to show that $\tilde{\lambda}_i$ and L_i satisfy (LB_i). First note that $\tilde{\lambda}_i = \Lambda \hat{\lambda}_i^- + \hat{\lambda}_i^+ \geq 0$ because $\Lambda \geq 0$, $\hat{\lambda}_i^- \geq 0$, and $\hat{\lambda}_i^+ \geq 0$. Next observe that

$$A\tilde{\lambda}_i = A\Lambda\hat{\lambda}_i^- + A\hat{\lambda}_i^+ = A(\hat{\lambda}_i^+ - \hat{\lambda}_i^-) = A\hat{\lambda}_i \ , \tag{51}$$

where the second equality follows from $A\Lambda = -A$. Also note that

$$A\hat{\lambda}_i = \gamma_i(d,\ell) ADt(d,\ell) - ADA^\top B(d)^{-1} a_i = -a_i \ , \tag{52}$$

where the second equality follows from $ADt(d,\ell) = 0$ and the definition of $B(d)$. From (51) and (52), it holds that $A\tilde{\lambda}_i = -a_i$, and so it remains to verify that $-\tilde{\lambda}_i^\top u \geq L_i$.

For notational convenience, define

$$z = y(d,\ell) - \frac{1}{\gamma_i(d,\ell)} B(d)^{-1} a_i \ ,$$

and note that $\hat{\lambda}_i = \gamma_i(d,\ell) D(A^\top z - r(\ell))$. Also, for each $j \in [m]$, define

$$\mu_j = \tfrac{1}{2}\gamma_i(d,\ell) d_j \left[2a_j^\top z (a_j^\top z - r_j(\ell)) - (a_j^\top z - \ell_j)(a_j^\top z - u_j) \right] \ ,$$

and observe that if $(\hat{\lambda}_i)_j \neq 0$ (so $d_j \neq 0$ and $a_j^\top z \neq r_j(\ell)$), then

$$\frac{\mu_j}{(\hat{\lambda}_i)_j} = a_j^\top z - \frac{(a_j^\top z - \ell_j)(a_j^\top z - u_j)}{2(a_j^\top z - r_j(\ell))} = u_j + \frac{(a_j^\top z - u_j)^2}{2(a_j^\top z - r_j(\ell))} \tag{53}$$

$$= \ell_j + \frac{(a_j^\top z - \ell_j)^2}{2(a_j^\top z - r_j(\ell))} \ . \tag{54}$$

Now if $(\hat{\lambda}_i)_j > 0$, then $\text{sgn}((\hat{\lambda}_i)_j) = \text{sgn}(a_j^\top z - r_j(\ell))$, and it follows from multiplying (53) by $(\hat{\lambda}_i)_j$ that

$$\mu_j = (\hat{\lambda}_i)_j \left[u_j + \frac{(a_j^\top z - u_j)^2}{2(a_j^\top z - r_j(\ell))} \right] \geq (\hat{\lambda}_i)_j u_j \ .$$

Similarly if $(\hat{\lambda}_i)_j < 0$, then $\text{sgn}((\hat{\lambda}_i)_j) = \text{sgn}(a_j^\top z - r_j(\ell))$, and it follows from multiplying (54) by $(\hat{\lambda}_i)_j$ that

$$\mu_j = (\hat{\lambda}_i)_j \left[\ell_j + \frac{(a_j^\top z - \ell_j)^2}{2(a_j^\top z - r_j(\ell))} \right] \geq (\hat{\lambda}_i)_j \ell_j \ .$$

Finally, if $(\hat{\lambda}_i)_j = 0$, then either $d_j = 0$ in which case $\mu_j = 0$, or $a_j^\top z = r_j(\ell)$ in which case

$$\mu_j = -\tfrac{1}{2}\gamma_i(d,\ell) d_j (a_j^\top z - \ell_j)(a_j^\top z - u_j) \geq 0 \ .$$

Thus from the above we obtain:

$$\begin{aligned}
(\hat{\lambda}_i^+)^\top u - (\hat{\lambda}_i^-)^\top \ell \leq \sum_{j=1}^m \mu_j &= \hat{\lambda}_i^\top A^\top z - \tfrac{1}{2}\gamma_i(d,\ell) \sum_{j=1}^m d_j (a_j^\top z - \ell_j)(a_j^\top z - u_j) \\
&= -a_i^\top z - \tfrac{1}{2}\gamma_i(d,\ell) \sum_{j=1}^m d_j (a_j^\top z - \ell_j)(a_j^\top z - u_j) \\
&= -a_i^\top z \\
&= \gamma_i(d,\ell) - a_i^\top y(d,\ell) \ ,
\end{aligned} \tag{55}$$

where the second equality follows from (52) and the third equality follows from the fact that z satisfies the inequality defining $E(d,\ell)$ with equality. Rearranging (55) yields

$$L_i = a_i^\top y(d,\ell) - \gamma_i(d,\ell) \leq (\hat{\lambda}_i^-)^\top \ell - (\hat{\lambda}_i^+)^\top u \leq -(\hat{\lambda}_i^-)^\top \Lambda^\top u - (\hat{\lambda}_i^+)^\top u = -\tilde{\lambda}_i^\top u \ ,$$

where the second inequality follows from $-\Lambda^\top u \geq \ell$ and $\hat{\lambda}_i^- \geq 0$. □

Proof of Proposition 5.3. Examining Step 1 of Procedure 2, we see from Remark 1.1 that $\bar{\lambda}_j$ is a type-L certificate of infeasibility if $\ell_j > u_j$. If the procedure does not exit at Step 1, it holds that $\frac{1}{2}(u - \ell) = v(\ell) \geq 0$. And from the discussion of Procedure 2 directly preceding Proposition 5.3, $\bar{\lambda}_k := \lambda_k + e_k$ is a type-L certificate of infeasibility as long as Steps 2 – 11 of Procedure 2 can be executed as stipulated. The only steps that require proof of such viability are Steps 2, 4, 5, and 6.

We first examine Step 2. Let $i \in [m]$ be selected, and define $\bar{\ell} := \ell - \beta e_i$; then from Proposition D.2 it follows that

$$f(d, \bar{\ell}) := f(d, \ell - \beta e_i) = f(d, \ell) - d_i(a_i^\top y(d, \ell) - u_i)\beta + \tfrac{1}{4} d_i^2 a_i^\top B(d)^{-1} a_i \beta^2 \ ,$$

using (48) and the fact that $t_i(d, \ell) - v_i(\ell) = a_i^\top y(d, \ell) - u_i$. As the above expression is a strictly convex quadratic in β and $f(d, \ell) \leq 0$, there is a positive value of β for which $f(d, \ell - \beta e_i) = 0$, and in fact using the quadratic formula this value of β works out to be:

$$\beta = \frac{2(a_i^\top y(d, \ell) - u_i) + 2\sqrt{(a_i^\top y(d, \ell) - u_i)^2 - f(d, \ell) a_i^\top B(d)^{-1} a_i}}{d_i a_i^\top B(d)^{-1} a_i} \ .$$

Thus Step 2 is executable. Let us next consider Step 4. After Steps 2 and 3 are computed, it holds that $f(d, \ell) = 0$. We must show that there exists an index $j \in [m]$ such that $a_j^\top y(d, \ell) \leq u_j$. Suppose there is no such index; then for all $s \in [m]$ it holds that

$$t_s(d, \ell) = a_s^\top y(d, \ell) - r_s(\ell) > u_s - r_s(\ell) = v_s(\ell) \geq 0 \ ,$$

where the last inequality follows since the original input lower bounds satisfied $v(\ell) \geq 0$ and the updated value of ℓ in Step 3 is less than or equal to the original value, whereby it still holds that $v(\ell) \geq 0$. It then follows that

$$f(d, \ell) = v(\ell)^\top D v(\ell) - t(d, \ell)^\top D t(d, \ell) < 0 \ ,$$

which yields a contradiction. Thus there exists an index $j \in [m]$ such that $a_j^\top y(d, \ell) \leq u_j$, whereby Step 4 is executable.

To see why Step 5 is executable, note that the input to Procedure 2 satisfied $f(d, \ell) \leq 0$ and $A^\top y(d, \ell) \not\leq u$ (for the original input value ℓ) and hence implied that $\mathcal{P} = \emptyset$. Thus for any y there is a violated inequality of the system (P).

Last of all we show that Step 6 is implementable. At the start of Step 6 we have $f(d, \ell) = 0$, $a_j^\top y(d, \ell) \leq u_j$, and $a_k^\top y(d, \ell) > u_k$. From Proposition D.2 and the fact that $f(d, \ell) = 0$, it follows that

$$\begin{aligned} f(d, \ell - \varepsilon e_j) &= f(d, \ell) - \varepsilon(a_j^\top y(d, \ell) - u_j) d_j + \varepsilon^2 \tfrac{1}{4} d_j^2 a_j^\top B(d)^{-1} a_j \\ &= \varepsilon(u_j - a_j^\top y(d, \ell)) d_j + \varepsilon^2 \tfrac{1}{4} d_j^2 a_j^\top B(d)^{-1} a_j \ , \end{aligned}$$

and thus $f(d, \ell - \varepsilon e_j) > 0$ for all $\varepsilon > 0$. Accordingly, it is sufficient to show that we can take $\varepsilon > 0$ and sufficiently small such that $a_k^\top y(d, \ell - \varepsilon e_j) - \gamma_k(d, \ell - \varepsilon e_j) > u_k$. Let us denote $\bar{\ell} := \ell - \varepsilon e_j$ for notational convenience, and $h(\varepsilon) := a_k^\top y(d, \ell - \varepsilon e_j) - \gamma_k(d, \ell - \varepsilon e_j) - u_k$. From Proposition D.2 and the characterization of slab radii in (14), it holds for all $\varepsilon > 0$ that

$$\begin{aligned} h(\varepsilon) \ &= \ a_k^\top y(d, \ell) - u_k - \varepsilon \tfrac{1}{2} d_j a_k^\top B(d)^{-1} a_j \\ &\quad - \Big(\varepsilon(u_j - a_j^\top y(d, \ell)) d_j + \varepsilon^2 \tfrac{1}{4} d_j^2 a_j^\top B(d)^{-1} a_j\Big)^{1/2} (a_k^\top B^{-1}(d) a_k)^{1/2} \\ \\ &= \ \delta - \varepsilon \tfrac{1}{2} d_j a_k^\top B(d)^{-1} a_j - \Big(\varepsilon(u_j - a_j^\top y(d, \ell)) d_j + \varepsilon^2 \tfrac{1}{4} d_j^2 a_j^\top B(d)^{-1} a_j\Big)^{1/2} (a_k^\top B^{-1}(d) a_k)^{1/2} \ , \end{aligned}$$

where $\delta := a_k^\top y(d, \ell) - u_k > 0$. Now notice that $h(0) = \delta > 0$ and by continuity it holds that $h(\varepsilon) > 0$ for all $\varepsilon > 0$ and sufficiently small. Thus Step 6 is implementable. Furthermore, the equation $h(\varepsilon) = \delta/2$ can be rearranged so that squaring both sides yields a quadratic in ε, and so Step 6 can be implemented using the mechanics of the quadratic formula. □

E.2 Proofs for Section 6

For notational convenience we define:

$$\begin{aligned}\beta^{(1)} &:= \frac{-2(t_j(d,\ell) - v_j(\ell))}{d_j \gamma_j(d,\ell)^2} \ , \\ \beta^{(2)} &:= \frac{2(2v_j(\ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)}))}{(m-1)d_j^{(1)}\gamma_j(d^{(1)},\ell^{(1)})^2 + 2} \ .\end{aligned}$$

Note that $\ell^{(1)} = \ell + \beta^{(1)} e_j$ and $\ell^{(2)} = \ell^{(1)} + \beta^{(2)} e_j$.

Proof of Lemma 6.1. Observe that

$$\ell_j - \ell_j^{(1)} = \frac{2(t_j(d,\ell) - v_j(\ell))}{d_j \gamma_j(d,\ell)^2} = \frac{2(a_j^\top y(d,\ell) - u_j)}{d_j \gamma_j(d,\ell)^2} > 0 \ , \tag{56}$$

and so $\ell_j^{(1)} < \ell_j$ and hence $\ell^{(1)} \leq \ell$, whereby $\ell^{(1)}$ is a lower bound for (P) with certificate matrix Λ since Λ is a certificate matrix for ℓ and $\ell^{(1)} \leq \ell$. From (46) with $\beta = \beta^{(1)}$ we have:

$$\begin{aligned} a_j^\top y(d, \ell^{(1)}) &= a_j^\top y(d,\ell) + \tfrac{1}{2}\beta^{(1)} d_j \gamma_j(d,\ell)^2 \\ &= a_j^\top y(d,\ell) - (t_j(d,\ell) - v_j(\ell)) \\ &= a_j^\top y(d,\ell) - (a_j^\top y(d,\ell) - r_j(\ell) - v_j(\ell)) \\ &= u_j \ , \end{aligned}$$

which shows (19). And from (48) with $\beta = \beta^{(1)}$ we have:

$$\begin{aligned} f(d, \ell^{(1)}) &= 1 + \beta^{(1)}(t_j(d,\ell) - v_j(\ell))d_j + \tfrac{1}{4}(\beta^{(1)})^2 d_j^2 \gamma_j(d,\ell)^2 \\ &= 1 - \left(\frac{t_j(d,\ell) - v_j(\ell)}{\gamma_j(d,\ell)}\right)^2 \\ &= 1 - \left(\frac{a_j^\top y(d,\ell) - u_j}{\gamma_j(d,\ell)}\right)^2 \ , \end{aligned} \tag{57}$$

which demonstrates the equality in (20). The inequality in (20) follows since j is the index of a violated constraint, hence $a_j^\top y(d,\ell) > u_j$. □

Proof of Lemma 6.2. We first prove item (a). First suppose that $\beta^{(2)} \leq 0$. Then

$$\ell_j^{(2)} = \ell_j^{(1)} + \beta^{(2)} \leq \ell_j^{(1)} < \ell_j \leq \max\{\ell_j, L_j\} \, ,$$

where the strict inequality uses (56). Next suppose that $\beta^{(2)} > 0$. Then $\beta^{(2)} \leq 2v_j(\ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)})$, and therefore

$$\begin{aligned} \ell_j^{(2)} &\leq \ell_j^{(1)} + 2v_j(\ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)}) \\ &= u_j - 2v_j(\ell^{(1)}) + 2v_j(\ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)}) \\ &= a_j^\top y(d, \ell^{(1)}) - \gamma_j(d^{(1)}, \ell^{(1)}) \ , \end{aligned} \tag{58}$$

where the last equality follows from (19). Also note that

$$\gamma_j(d,\ell^{(1)}) = f(d,\ell^{(1)})^{\frac{1}{2}}(a_j^\top B(d)^{-1}a_j)^{\frac{1}{2}} = f(d,\ell^{(1)})^{\frac{1}{2}}\gamma_j(d,\ell). \tag{59}$$

From the invariance of $\gamma_j(\cdot,\ell^{(1)})$ under positive scaling of the first argument,

$$\begin{aligned}
&\left(a_j^\top y(d,\ell) - \gamma_j(d,\ell)\right) - \left(a_j^\top y(d,\ell^{(1)}) - \gamma_j(d^{(1)},\ell^{(1)})\right) \\
&= \gamma_j(d,\ell^{(1)}) - \gamma_j(d,\ell) + a_j^\top y(d,\ell) - a_j^\top y(d,\ell^{(1)}) \\
&= \left(f(d,\ell^{(1)})^{\frac{1}{2}} - 1\right)\gamma_j(d,\ell) + a_j^\top y(d,\ell) - u_j \\
&= \left(\sqrt{1 - \left(\frac{a_j^\top y(d,\ell) - u_j}{\gamma_j(d,\ell)}\right)^2} - 1\right)\gamma_j(d,\ell) + a_j^\top y(d,\ell) - u_j \\
&\geq 0,
\end{aligned}$$

where the second equality follows from (19) and (59), the third equality uses (57), and the inequality from the fact that $\sqrt{1-x^2} \geq 1-x$ for any scalar $0 \leq x \leq 1$. Therefore

$$a_j^\top y(d,\ell^{(1)}) - \gamma_j(d^{(1)},\ell^{(1)}) \leq a_j^\top y(d,\ell) - \gamma_j(d,\ell)\ , \tag{60}$$

and (58) and (60) combine to yield

$$\ell_j^{(2)} \leq a_j^\top y(d,\ell) - \gamma_j(d,\ell) \leq \max\{\ell_j, L_j\}\ ,$$

which completes the proof of (a).

Next, (b) is immediate since $f(d^{(1)},\ell^{(1)}) = 1$ from (21).

Finally, we prove (c). We need to prove that

$$f(d^{(2)},\ell^{(2)}) = \frac{m^2}{m^2-1} = 1 + \frac{1}{m^2-1}\ . \tag{61}$$

Note that

$$\begin{aligned}
f(d^{(2)},\ell^{(2)}) &= f(d^{(1)},\ell^{(1)}) + [f(d^{(2)},\ell^{(1)}) - f(d^{(1)},\ell^{(1)})] + [f(d^{(2)},\ell^{(2)}) - f(d^{(2)},\ell^{(1)})] \\
&= 1 + [f(d^{(2)},\ell^{(1)}) - 1] + [f(d^{(2)},\ell^{(2)}) - f(d^{(2)},\ell^{(1)})]\ . \qquad (62)
\end{aligned}$$

We now proceed to evaluate the two terms in brackets.

For notational convenience let

$$\begin{aligned}
y^{(1)} &= y(d^{(1)},\ell^{(1)})\ , \\
t^{(1)} &= t(d^{(1)},\ell^{(1)})\ , \\
\gamma &= \gamma_j(d^{(1)},\ell^{(1)})\ , \\
v_j^{(1)} &= v_j(\ell^{(1)})\ , \\
r_j^{(1)} &= r_j(\ell^{(1)})\ , \\
B &= B(d^{(1)})\ .
\end{aligned}$$

Then from Proposition D.1 with $\delta = 2/[(m-1)\gamma^2]$ and $\theta = \delta/(1+\delta\gamma^2)$, we have $B(d^{(2)})^{-1} = B^{-1} - \theta B^{-1}a_j a_j^T B^{-1}$, and so

$$a_j^\top B(d^{(2)})^{-1}a_j = \gamma^2 - \theta\gamma^4 = \frac{1}{1+\delta\gamma^2}\gamma^2 = \frac{m-1}{m+1}\gamma^2 \ .$$

Also, $t(d^{(2)}, \ell^{(1)}) = t^{(1)} - \theta t_j^{(1)} A^T B^{-1} a_j$, and so

$$t_j(d^{(2)}, \ell^{(1)}) = t_j^{(1)}(1-\theta\gamma^2) = \frac{m-1}{m+1}t_j^{(1)} \ .$$

Lastly,

$$f(d^{(2)}, \ell^{(1)}) - 1 = \frac{2}{(m-1)\gamma^2}(v_j^{(1)})^2 - \frac{2}{(m+1)\gamma^2}(t_j^{(1)})^2 \ .$$

From the invariance of $y(\cdot, \ell^{(1)})$ under positive scaling and (19), it holds that

$$t_j^{(1)} = a_j^\top y^{(1)} - r_j^{(1)} = a_j^\top y(d, \ell^{(1)}) - r_j^{(1)} = u_j - r_j^{(1)} = v_j^{(1)} \ , \tag{63}$$

and so

$$f(d^{(2)}, \ell^{(1)}) - 1 = \frac{4}{(m^2-1)\gamma^2}(v_j^{(1)})^2 \ . \tag{64}$$

Next, from Proposition D.2, we have

$$f(d^{(2)}, \ell^{(2)}) - f(d^{(2)}, \ell^{(1)}) = \beta^{(2)}(t_j(d^{(2)}, \ell^{(1)}) - v_j^{(1)})d_j^{(2)} + \frac{1}{4}(\beta^{(2)})^2(d_j^{(2)})^2 a_j^T(B(d^{(2)}))^{-1}a_j \ . \tag{65}$$

Note that from the definition of $d_j^{(2)}$ and $\beta^{(2)}$, it follows that

$$\beta^{(2)}d_j^{(2)} = \frac{2(2v_j^{(1)} - \gamma)}{(m-1)\gamma^2} \ .$$

We can now evaluate the terms in (65).

Since $t_j(d^{(2)}, \ell^{(1)}) = \frac{m-1}{m+1}t_j^{(1)} = \frac{m-1}{m+1}v_j^{(1)}$, the first term on the right-hand side is

$$\beta^{(2)}d_j^{(2)}\left(\frac{m-1}{m+1}v_j^{(1)} - v_j^{(1)}\right) = \frac{2(2v_j^{(1)} - \gamma)}{(m-1)\gamma^2} \cdot \frac{-2v_j^{(1)}}{m+1} = \frac{-4(2(v_j^{(1)})^2 - \gamma v_j^{(1)})}{(m^2-1)\gamma^2}.$$

The second term on the right-hand side is equal to

$$\frac{1}{4}(\beta^{(2)}d_j^{(2)})^2\frac{m-1}{m+1}\gamma^2 = \frac{(2v_j^{(1)} - \gamma)^2}{(m^2-1)\gamma^2}.$$

Combining these terms and substituting them in (65) gives

$$f(d^{(2)}, \ell^{(2)}) - f(d^{(2)}, \ell^{(1)}) = -\frac{4}{(m^2-1)\gamma^2}(v_j^{(1)})^2 + \frac{1}{m^2-1},$$

which with (64) and (62) yields (61). □

Proof of Theorem 6.1. Define $\alpha(d, \ell) := \frac{m^2-1}{m^2}\frac{1}{f(d,\ell^{(1)})}$. Note that $\alpha(d, \ell) > \frac{m^2-1}{m^2}$ because $0 < f(d, \ell^{(1)}) < 1$ using the hypothesis of the theorem and Lemma 6.1. From the invariance of $\gamma_j(\cdot, \ell^{(1)})$ under positive scaling of the first argument, it holds that:

$$\gamma_j(d^{(1)}, \ell^{(1)})^2 = \gamma_j(d, \ell^{(1)})^2 = f(d, \ell^{(1)})\gamma_j(d, \ell)^2 \ . \tag{66}$$

The result now follows from (21) and Lemma 6.2 (c). □

E.3 Proofs for Section 7

Proof of Proposition 7.1. For any $x \in E(d,\ell)$ it holds that

$$d_i(x-y(d,\ell))a_ia_i^\top(x-y(d,\ell)) \le (x-y(d,\ell))^\top ADA^\top(x-y(d,\ell)) \le f(d,\ell) \ ,$$

and therefore $|a_i^\top x - a_i^\top y(d,\ell)| \le \left(\frac{f(d,\ell)}{d_i}\right)^{\frac{1}{2}}$ for all $x \in E(d,\ell)$. The result then follows from the definition of $\gamma_i(d,\ell)$. □

Proof of Lemma 7.1. We first show that

$$\mu_j(\tilde{d}) \le \frac{m}{m+1}\mu_j(d) \ . \tag{67}$$

Note that $\mu_j(d) = \sqrt{\frac{f(d,\ell)}{d_j}}$ because $\sqrt{\frac{f(d,\ell)}{d_j}} \ge \tau(A,u) \ge \frac{m}{m+1}\tau(A,u)$. From Proposition 7.1 and $\alpha \ge \frac{m^2-1}{m^2}$ it follows that:

$$\frac{\tilde{d}_j}{f(\tilde{d},\ell)} = \alpha\left(\frac{d_j}{f(d,\ell)} + \frac{2}{m-1}\frac{1}{\gamma_j(d,\ell)^2}\right) \ge \frac{m^2-1}{m^2}\left(1+\frac{2}{m-1}\right)\frac{d_j}{f(d,\ell)} = \left(\frac{m+1}{m}\right)^2\frac{d_j}{f(d,\ell)} \ ,$$

and therefore

$$\begin{aligned}\mu_j(\tilde{d}) &= \max\left\{\sqrt{\frac{f(\tilde{d},\ell)}{\tilde{d}_j}}, \frac{m}{m+1}\tau(A,u)\right\} \\ &\le \max\left\{\frac{m}{m+1}\sqrt{\frac{f(d,\ell)}{d_j}}, \frac{m}{m+1}\tau(A,u)\right\} = \frac{m}{m+1}\mu_j(d) \ ,\end{aligned}$$

where the last equality follows from $\sqrt{\frac{f(d,\ell)}{d_j}} \ge \tau(A,u)$ and $\mu_j(d) = \sqrt{\frac{f(d,\ell)}{d_j}}$.

Next we show that for all $i \in [m]$, $i \ne j$, it holds that:

$$\mu_i(\tilde{d}) \le \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}}\mu_i(d) \ . \tag{68}$$

Note that $\tilde{d}_i/f(\tilde{d},\ell) = \alpha d_i/f(d,l)$, from which

$$\sqrt{\frac{f(\tilde{d},\ell)}{\tilde{d}_i}} = \sqrt{\frac{1}{\alpha}}\sqrt{\frac{f(d,l)}{d_i}} \le \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}}\sqrt{\frac{f(d,l)}{d_i}}.$$

Thus if $\mu_i(\tilde{d}) = \frac{m}{m+1}\tau(A,u)$, we have

$$\mu_i(\tilde{d}) = \frac{m}{m+1}\tau(A,u) \le \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}}\frac{m}{m+1}\tau(A,u) \le \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}}\mu_i(d),$$

while if $\mu_i(\tilde{d}) = \sqrt{\frac{f(\tilde{d},\ell)}{\tilde{d}_i}}$, we have

$$\mu_i(\tilde{d}) = \sqrt{\frac{f(\tilde{d},\ell)}{\tilde{d}_i}} \leq \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}} \sqrt{\frac{f(d,l)}{d_i}} \leq \left(\frac{m^2}{m^2-1}\right)^{\frac{1}{2}} \mu_i(d).$$

Together these establish (68).

Thus from (67) and (68) we obtain:

$$\begin{aligned}
\phi(\tilde{d},\ell) &\leq \phi(d,\ell)\left(\frac{m}{m+1}\right)\left(\frac{m^2}{m^2-1}\right)^{(m-1)/2} \\
&= \phi(d,\ell)\left(1-\frac{1}{m+1}\right)\left(1+\frac{1}{m^2-1}\right)^{(m-1)/2} \\
&\leq \phi(d,\ell)e^{-\frac{1}{(m+1)}}\left(e^{-\frac{1}{(m^2-1)}}\right)^{(m-1)/2} \\
&= \phi(d,\ell)e^{-\frac{1}{2(m+1)}} ,
\end{aligned}$$

where the second inequality follows from the fact that $1+x \leq e^x$ for any scalar x. □

Proof of Lemma 7.2. For notational convenience let us assume that d and $\tilde{d}$ have been rescaled so that $f(d,\ell) = 1$ and $f(\tilde{d},\tilde{\ell}) = 1$. Then the volume ratio $E(\tilde{d},\tilde{\ell})$ and $E(d,\ell)$ is:

$$\begin{aligned}
&\frac{\operatorname{vol} E(\tilde{d},\tilde{\ell})}{\operatorname{vol} E(d,\ell)} = \left(\frac{\det(ADA^\top)}{\alpha^n \det\left(ADA^\top + \frac{2}{m-1}\frac{1}{\gamma_j(d,\ell)^2}a_j a_j^\top\right)}\right)^{\frac{1}{2}} = \left(\frac{m-1}{m+1}\right)^{\frac{1}{2}}\left(\frac{1}{\alpha}\right)^{\frac{n}{2}} \\
&\leq \left(\frac{m-1}{m+1}\right)^{\frac{1}{2}}\left(\frac{m^2}{m^2-1}\right)^{\frac{n}{2}} \leq \left(\frac{m-1}{m+1}\right)^{\frac{1}{2}}\left(\frac{m^2}{m^2-1}\right)^{\frac{m}{2}} \\
&= \left(\frac{m}{m+1}\right)\left(\frac{m^2}{m^2-1}\right)^{(m-1)/2} \leq e^{-\frac{1}{2(m+1)}},
\end{aligned}$$

where the second equality uses the matrix determinant lemma, and the last inequality follows from the fact that $1+x \leq e^x$ for any scalar x. □

Proposition E.1. $\det(AA^\top) \geq \rho(A)^{2n}$.

Proof. If $\rho(A) = 0$ the result is clearly true, so suppose for the rest of the proof that $\rho(A) > 0$. We claim that

$$\text{for each } v \text{ satisfying } \|v\| = 1 \text{ there exists } i \in [m] \text{ satisfying } a_i^\top v \geq \rho(A) \ . \qquad (69)$$

If the claim is true, then for any v with $\|v\| = 1$ it holds that $v^\top AA^\top v = \sum_{j=1}^m (a_j^\top v)^2 \geq (a_i^\top v)^2 \geq \rho(A)^2$, and so the smallest eigenvalue of $AA^\top$ is at least $\rho(A)^2$, whereby $\det(AA^\top) \geq \rho(A)^{2n}$, which proves the result.

We now prove (69) by contradiction. Suppose that the claim is false. Then there exists $\bar{v}$ with $\|\bar{v}\| = 1$ and $0 < \alpha < 1$ satisfying $A^\top \bar{v} \leq \alpha\rho(A)e$. Define $\Delta A := -\alpha\rho(A)\bar{v}e^\top$ and note that

$$(A + \Delta A)^\top(-\bar{v}) = -A^\top\bar{v} + \alpha\rho(A)\bar{v}^\top\bar{v}e \geq -\alpha\rho(A)e + \alpha\rho(A)e = 0 \ ,$$

and so from the definition of $\rho(A)$ in (16) it must hold that $\|\Delta A\|_{1,2} \geq \rho(A)$. However, it is simple to verify that $\|\Delta A\|_{1,2} = \| - \alpha\rho(A)\bar{v}e^\top\|_{1,2} = \alpha\rho(A) < \rho(A)$ which provides the desired contradiction, establishing (69). □

E.4 Proofs for Section 8.2

Proof of Proposition 8.1. We will establish a more general result that will imply the correctness of Procedure 5 as a particular instance. Recall from (25) that the certificate matrices satisfy the recursion $\Lambda^{(i)} = \Lambda^{(i-1)}M_{(i)} + B_{(i)}$ for $i = 1, \ldots, k$. Suppose we are given vectors w^k and z^k, and we wish to compute $\bar{v} := \Lambda^{(k)}w^k + z^k$. We claim that the following procedure accomplishes this task:

```
for $i = k : 1$ do
    $w^{i-1} \leftarrow M_{(i)}w^i$
    $z^{i-1} \leftarrow B_{(i)}w^i + z^i$
end for
Return $\bar{v} := \Lambda^{(0)}w^0 + z^0$
```

It is straightforward to prove using induction that the returned value $\bar{v}$ satisfies $\bar{v} = \Lambda^{(k)}w^k + z^k$. Now notice that Procedure 5 is simply an instantiation of the above procedure with $w^k := e_{j_k}$, $z^k := e_{j_k}$, and the formulas for $M_{(i)}$ and $B_{(i)}$ in Section 8.2. □

References

[1] R. Bland, D. Goldfarb, and M. J. Todd, *The ellipsoid method: a survey*, Operations Research **29** (1981), no. 6, 1039–1091.

[2] B. P. Burrell and M. J. Todd, *The ellipsoid method generates dual variables*, Mathematics of Operations Research **10** (1985), no. 4, 527–715.

[3] D. Cheung and F. Cucker, *A new condition number for linear programming*, Mathematical Programming **91** (2001), 163–174.

[4] J.G. Ecker and M. Kupferschmid, *A computational comparison of the ellipsoid algorithm with several nonlinear programming algorithms*, SIAM Journal on Control and Optimization **23** (1985), no. 5, 657–674.

[5] P. Gács and L. Lovász, *Khachiyan's algorithm for linear programming*, Mathematical Programming at Oberwolfach, Mathematical Programming Studies Vol. 14 (H. König, B. Korte, and K. Ritter, eds.), Springer, 1981.

[6] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric algorithms and combinatorial optimization*, second ed., Springer-Verlag, Berlin, 1994.

[7] W.W. Hager, *Updating the inverse of a matrix*, SIAM Review **31** (1989), 221–239.

[8] L. G. Khachiyan, *A polynomial algorithm in linear programming*, Soviet Math. Dokl. **20** (1979), no. 1, 191–194.

[9] J. Renegar, *Some perturbation theory for linear programming*, Mathematical Programming **65** (1994), no. 1, 73–91.

[10] ———, *Incorporating condition measures into the complexity theory of linear programming*, SIAM Journal on Optimization **5** (1995), no. 3, 506–524.

[11] ———, *Linear programming, complexity theory, and elementary functional analysis*, Mathematical Programming **70** (1995), no. 3, 279–351.

[12] N.Z. Shor, *Cut-off method with space extension in convex programming problems*, Cybernetics **13** (1977), 94–96.

[13] M. J. Todd, *Minimum volume ellipsoids containing part of a given ellipsoid*, Mathematics of Operations Research **7** (1980), 253–261.

[14] ———, *Ellipsoid method redux*, International Symposium on Mathematical Programming, MOS, 2018.

[15] B. Yamnitsky and L.A. Levin, *An old linear programming algorithm runs in polynomial time*, 23rd Annual Symposium on the Foundations of Computer Science (1982), 327–328.

[16] D.B. Yudin and A.S. Nemirovsky, *Informational complexity and effective methods of solution for convex extremal problems*, Matekon: Translations of Russian and East European Mathematical Economics **13** (1977), 25–45.