CONDITION NUMBER COMPLEXITY OF AN ELEMENTARY ALGORITHM FOR RESOLVING A CONIC LINEAR SYSTEM [1]

Marina Epelman[2]
M.I.T.

Robert M. Freund[3]
M.I.T.

February, 1997

Abstract

We develop an algorithm for resolving a conic linear system (FP_d), which is a system of the form

$$(\text{FP}_d): \quad b - Ax \in C_Y$$
$$x \in C_X,$$

where C_X and C_Y are closed convex cones, and the data for the system is $d = (A, b)$. The algorithm "resolves" the system in that it either finds an ϵ-solution of (FP_d) for a pre-specified tolerance ϵ, or demonstrates that (FP_d) has no solution by solving an alternative dual system. The algorithm is based on a generalization of von Neumann's algorithm for linear inequalities. The number of iterations of the algorithm is essentially bounded by $O\left(\mathcal{C}(d)^2 \ln(\mathcal{C}(d)) \ln\left(\frac{\|b\|}{\epsilon}\right)\right)$ when (FP_d) has a solution, and is bounded by $O\left(\mathcal{C}(d)^2\right)$ when (FP_d) has no solution, and so depends only on two numbers, namely the feasibility tolerance ϵ and the condition number $\mathcal{C}(d)$ of the data $d = (A, b)$ (in addition to the norm of the vector b), and is independent of the dimensions of the problem. The quantity $\mathcal{C}(d)$ is the condition number of (FP_d), originally defined by Renegar as $\mathcal{C}(d) \triangleq \|d\|/\rho(d)$, where $\rho(d)$ is smallest change in the data $\Delta d = (\Delta A, \Delta b)$ needed to create a data instance $d + \Delta d = (A + \Delta A, b + \Delta b)$ that is ill-posed, i.e., $\rho(d)$ is the "distance to ill-posedness". Each iteration of the algorithm performs a small number of matrix-vector and vector-vector multiplications (that take full advantage of the sparsity of the original data) plus a small number of other operations involving the cones C_X and C_Y that may be easy to compute or not, depending on the nature of the cones C_X and C_Y (and which are easy to compute when these cones are the nonnegative orthant R^n_+, the semi-definite cone $S^{k\times k}_+$, and/or the origin $\{0\}$). The algorithm is "elementary" in the sense that it performs only a few relatively simple mathematical operations at each iterations.

AMS Subject Classification: 90C, 90C05, 90C60
Keywords: Complexity of Convex Programming, Conditioning, Error Analysis

[1]This research has been partially supported through NSF Graduate Research Fellowship.
[2]M.I.T. Operations Research Center, Building E40-130, 77 Massachusetts Avenue, Cambridge, MA 02139-4307. e-mail: mepelman@mit.edu
[3]M.I.T. Operations Research Center, Building E40-149A, 77 Massachusetts Avenue, Cambridge, MA 02139-4307. email: rfreund@mit.edu

1 Introduction

The subject of this paper is the development and analysis of an algorithm for resolving a conic linear system, which is a system of the form:

$$\begin{array}{ll} (\text{FP}_d): & b - Ax \in C_Y \\ & x \in C_X, \end{array} \tag{1}$$

where $C_X \subset X$ and $C_Y \subset Y$ are each a closed convex cone in the n-dimensional normed linear vector space X (with norm $\|x\|$ for $x \in X$) and in the m-dimensional linear vector space Y (with norm $\|y\|$ for $y \in Y$), respectively, and where n and m are finite. Here $b \in Y$, and $A \in L(X,Y)$ where $L(X,Y)$ denotes the set of all linear operators $A : X \longrightarrow Y$. We denote by $d = (A, b)$ the "data" for the problem (FP_d). That is, the cones C_X and C_Y are regarded as fixed and given, and the data for the problem is the linear operator A together with the vector b. The problem (FP_d) is a very general format for studying the feasible regions of convex optimization problems, and has recently received much attention in the analysis of interior-point methods, see Nesterov and Nemirovskii[16] and Renegar [19] and [20], among others.

We develop an algorithm called "algorithm CLS" (for Conic Linear System) that "resolves" the system (FP_d) in that it either finds an ϵ-solution of (FP_d) for a pre-specified feasibility tolerance ϵ, or it demonstrates that (FP_d) has no solution by solving an alternative dual system. Algorithm CLS is based on a generalization of the algorithm of von Neumann studied by Dantzig [6] and [7].

There are three key motivating aspects of the research in this paper, namely the study of "elementary" algorithms for solving linear inequalities, the study of condition numbers for conic linear systems, and the merging of these first two aspects into an elementary algorithm for resolving a conic linear system whose iteration complexity is bounded appropriately by a function of the condition number. The first aspect has to do with the development and study of elementary algorithms for finding a point in a suitably described convex set, such as reflection algorithms for linear inequality systems (see [1], [15], [8], [13]), the "perceptron" algorithm [21] [22] [23] [24], and the recently revived algorithm of von Neumann (see [6], [7]). When applied to linear inequality systems, these algorithms share the following desirable properties, namely: the work per iteration is extremely low (typically involving only a few matrix-vector or vector-vector multiplications), the algorithms' iteration bounds do not typically rely on the dimensions m or n, and the algorithms fully exploit the sparsity of the original data. (Also, the performance of these algorithms can be quite competitive when applied to certain very large problems with very sparse data, see [5].) We refer to these algorithms as "elementary" in that the algorithms do not involve particularly sophisticated mathematics at each iteration, nor do the algorithms perform particularly sophisticated computation at each iteration, and in some sense these algorithms are all very unsophisticated as a result (especially compared to an interior-point algorithm or the ellipsoid algorithm). Part of the aim of this study is to develop a suitable generalization of one of these algorithms, namely von Neumann's algorithm, that will resolve a general conic linear system of the form (FP_d). This is accomplished, for the most part, in Section 6 wherein we develop algorithm CLS. Algorithm CLS does rely, however, on certain

assumptions regarding the ability to work conveniently with the cones C_X and C_Y. These assumptions are presented in Section 2 along with a discussion of their applicability. Incidentally, the assumptions are essentially satisfied when C_X is the nonnegative orthant R^n_+ or the semi-definite cone $S^{k\times k}_+$, and when C_Y is the nonnegative orthant R^n_+, the semi-definite cone $S^{k\times k}_+$, or the origin $\{0\}$.

The second key motivating aspect of this paper is the further development and study of algorithms for resolving (FP$_d$) related to the inherent "conditioning" of the system (FP$_d$). A particularly relevant way to measure the intuitive notion of conditioning of (FP$_d$) is in the condition number $\mathcal{C}(d)$ and the closely related "distance to ill-posedness" $\rho(d)$ of the data for (FP$_d$) as developed by Renegar in [18] in a more specific setting, but then generalized more fully in [19] and in [20]. Before discussing these concepts in detail, we first review what is known of their importance for the complexity of resolving (FP$_d$). In [20], Renegar presented an incredibly general interior-point (i.e., barrier) algorithm for resolving (FP$_d$) and showed, roughly speaking, that the iteration complexity bound of the algorithm depends linearly and only on two quantities: the barrier parameter for the underlying cones, and $\ln(\mathcal{C}(d))$, i.e., the logarithm of the condition number $\mathcal{C}(d)$. In [11], several results regarding the geometry of feasible regions of (FP$_d$) are developed that indicate that a suitably modified version of the ellipsoid algorithm applied to (FP$_d$) will resolve (FP$_d$) in $O(n \ln \mathcal{C}(d))$ iterations, see also [12]. Both the interior-point algorithm and the ellipsoid algorithm have an iteration complexity bound that is linear in $\ln(\mathcal{C}(d))$, and so are efficient algorithms in a sense defined by Renegar [19]. Both the interior-point algorithm and the ellipsoid algorithm are also very sophisticated algorithms, in contrast with the elementary reflection-type algorithms discussed above. (The interior-point algorithm makes implicit and explicit use of information from a special kind of barrier function (namely, a self-concordant barrier) at each iteration, and uses this information in the computation of a the next iterate by solving for the Newton step along the central trajectory. The work per iteration is $O(n^3)$ operations to compute the Newton step. The ellipsoid algorithm performs a particular linear transformation of the space at each iteration that is used to compute the projection of the current point onto a violated constraint, which in turn is used to create the direction for the next iterate, see [4]. The work per iteration of the ellipsoid algorithm is $O(n^2)$ operations.) Intuition strongly suggests that this sophistication is partly or wholly responsible for the excellent computational complexity of these methods. In contrast, the elementary algorithms cited earlier do not perform any sophisticated mathematics at each iteration, and one would not expect their complexity to be nearly as good as an interior-point algorithm or the ellipsoid algorithm.

The third and final aspect of this paper is the merging of the first two aspects into an "elementary" algorithm for resolving a conic linear system (FP$_d$) whose complexity is bounded appropriately by a function of the condition number $\mathcal{C}(d)$. This is accomplished for the most part in Theorem 6.3, where we show that algorithm CLS has an iteration complexity that essentially is

$$O\left(\mathcal{C}(d)^2 \ln(\mathcal{C}(d)) \ln\left(\frac{\|b\|}{\epsilon}\right)\right)$$

iterations when (FP$_d$) has a solution, and whose iteration complexity is essentially

$$O\left(\mathcal{C}(d)^2\right)$$

iterations when (FP$_d$) has no solution. These bounds are exponential in $\ln(\mathcal{C}(d))$, which is to be expected, given the nature of the algorithm. We do point out that the bounds are independent of either of the dimensions m or n, and that the work per iteration is very low in important cases such as linear inequality systems and semi-definite inequality systems. In these cases, it is possible for algorithm CLS to outperform an interior-point method or the ellipsoid algorithm when n is sufficiently large, the condition number $\mathcal{C}(d)$ is sufficiently small, and when the underlying data are very sparse. This is discussed in greater detail in Section 7.

We now present the development of the concepts of condition numbers and data perturbation for (FP$_d$) in detail. We denote by $d = (A, b)$ the "data" for the problem (FP$_d$). That is, the cones C_X and C_Y are regarded as fixed and given, and the data for the problem is the linear operator A together with the vector b. The set of solutions of (FP$_d$) is denoted by X_d in order to emphasize the dependence on the data d, i.e.,

$$X_d = \{x \in X \mid b - Ax \in C_Y, x \in C_X\}.$$

The space of all data $d = (A, b)$ for (FP$_d$) is denoted by $\mathcal{D}$. That is,

$$\mathcal{D} = \{d = (A, b) \mid A \in L(X, Y), b \in Y\}.$$

For $d = (A, b) \in \mathcal{D}$ we define the product norm on the cartesian product $L(X, Y) \times Y$ to be

$$\|d\| = \|(A, b)\| = \max\{\|A\|, \|b\|\} \tag{2}$$

where $\|b\|$ is the norm specified for Y and $\|A\|$ is the operator norm, namely

$$\|A\| = \max\{\|Ax\| \mid \|x\| \leq 1\}. \tag{3}$$

We define

$$\mathcal{F} = \{(A, b) \in \mathcal{D} \mid \text{ there exists } x \text{ satisfying } \ b - Ax \in C_Y, x \in C_X\}. \tag{4}$$

Then $\mathcal{F}$ corresponds to those data instances $d = (A, b)$ for which (FP$_d$) has a feasible solution. The complement of $\mathcal{F}$ is denoted by $\mathcal{F}^{\mathcal{C}}$, and so $\mathcal{F}^{\mathcal{C}}$ consists precisely of those data instances $d = (A, b)$ for which (FP$_d$) has no feasible solution.

The boundary of $\mathcal{F}$ and of $\mathcal{F}^{\mathcal{C}}$ is precisely the set

$$\mathcal{B} = \partial\mathcal{F} = \partial\mathcal{F}^{\mathcal{C}} = \text{cl}(\mathcal{F}) \cap \text{cl}(\mathcal{F}^{\mathcal{C}}) \tag{5}$$

where ∂S denotes the boundary of a set S and cl(S) denotes the closure of a set S. Note that if $d = (A, b) \in \mathcal{B}$, then (FP$_d$) is ill-posed in the sense that arbitrary small changes in the data $d = (A, b)$ will yield instances of (FP$_d$) that have feasible solutions, as well as instances of (FP$_d$) that do not have feasible solutions. Also, note that $\mathcal{B} \neq \emptyset$, since $d = (0, 0) \in \mathcal{B}$.

For a data instance $d = (A, b) \in \mathcal{D}$, the ball centered at d with radius δ is:

$$B(d, \delta) = \{\bar{d} \in \mathcal{D} \mid \|\bar{d} - d\| \leq \delta\}.$$

For a data instance $d = (A, b) \in \mathcal{D}$, the "distance to ill-posedness" is defined to be:

$$\rho(d) = \inf\{\|\Delta d\| \mid d + \Delta d \in \mathcal{B}\},$$

see [18], [19], [20], and so $\rho(d)$ is the distance of the data instance $d = (A, b)$ to the set $\mathcal{B}$ of ill-posed instances for the problem (FP_d). It is straightforward to show that

$$\rho(d) = \begin{cases} \sup\{\delta : B(d,\delta) \subset \mathcal{F}\} & \text{if } d \in \mathcal{F}, \\ \sup\{\delta : B(d,\delta) \subset \mathcal{F}^C\} & \text{if } d \in \mathcal{F}^C, \end{cases} \tag{6}$$

so that we could also define $\rho(d)$ by employing (6). The "condition number" $\mathcal{C}(d)$ of the data instance d is defined to be:

$$\mathcal{C}(d) = \frac{\|d\|}{\rho(d)} \tag{7}$$

when $\rho(d) > 0$, and $\mathcal{C}(d) = \infty$ when $\rho(d) = 0$. The condition number $\mathcal{C}(d)$ can be viewed as a scale-invariant reciprocal of $\rho(d)$, as it is elementary to demonstrate that $\mathcal{C}(d) = \mathcal{C}(\alpha d)$ for any positive scalar α. Observe that since $\tilde{d} = (\tilde{A}, \tilde{b}) = (0, 0) \in \mathcal{B}$ then for any $d \notin \mathcal{B}$ we have $\|d\| = \|d - \tilde{d}\| \geq \rho(d)$, whereby $\mathcal{C}(d) \geq 1$. The value of $\mathcal{C}(d)$ is a measure of the relative conditioning of the data instance d. Further analysis of the distance to ill-posedness has been studied in [11], Vera [25], [26], [27], [28], Filipowski [9], [10], and recently by Nunez and Freund [17].

The algorithm of von Neumann, which is generalized and used as a principal tool in this paper, can be viewed as a variation of the so-called relaxation or reflection schemes for solving systems of linear inequalities. In particular, let

$$\alpha_j^t x + \beta_j \geq 0, \quad i \in I$$

be a finite system of linear inequalities and let the solution set of this system be $P \subset R^n$. A general relaxation scheme is as follows. Choose the initial point x^0 arbitrarily. At the kth iteration, if $x^k \in P$, the algorithm terminates. Otherwise, let $j^k \in I$ be the index of a violated inequality and let $\hat{x}^k$ be the projection of x^k onto the hyperplane $H^k = \{x \mid \alpha_{j_k}^t x + \beta_{j_k} = 0\}$. Then

$$x^{k+1} = x^k + \lambda(\hat{x}^k - x^k),$$

where λ is chosen usually to be in the interval $\lambda \in (0,\ 2]$. Set $k \leftarrow k+1$ and repeat iteration.

If $\lambda_k = 1$ then x^{k+1} is the orthogonal projection of x^k onto H^k; if $\lambda_k = 2$ then x^{k+1} is the (complete) reflection of x^k through H^k; if $\lambda_k \in (0,1)$ $(\lambda_k \in (1,2))$ then x^{k+1} is considered an under-projection (over-projection) of x^k onto H^k.

There are a number of different rules for selecting the index j_k at each iteration. The three most studied of these rules are: the maximal distance rule (choose j_k to be the index of the furthest hyperplane H^k away from x^k), the maximal residual rule (choose j_k to maximize the value of $a_j^t x^k + b_j,\ j \in I$), and systematic projection rule (choose indices j_k in a cyclical fashion). It has been shown that under the maximal distance rule, depending on the choice of $\lambda \in (0,1]$ and the dimension of the set P, the sequence generated by the relaxation scheme either converges to a point in P or to a spherical surface having the affine hull of P as its axis. Finite termination and/or a geometric rate of convergence have been established for some of these methods (see [1], [15],[8], [13] for details). The von Neumann algorithm can be viewed (in the dual) as a relaxation algorithm with parameter $\lambda \in (0,1)$ being chosen dynamically at each iteration, with a particularly intelligent choice of the parameter λ at each iteration.

An outline of the paper is as follows. Section 2 contains notation, definitions, assumptions, and preliminary results. Section 3 revisits Dantzig's analysis [6] [7] of von Neumann's algorithm for solving a certain types of linear inequality systems, and presents further (stronger) complexity results in the case when the linear inequality system is not ill-posed. In particular, Theorem 3.2 essentially shows that von Neumann's algorithm is linearly convergent when the linear inequality system is feasible and not ill-posed, and that the algorithm produces dual variables that demonstrate infeasibility when the linear inequality system is infeasible. Section 4 presents a generalization of the von Neumann algorithm (appropriately called algorithm GVNA) for resolving a conic linear system in a special compact form (i.e, with a compactness constraint added). Theorem 4.1 essentially shows that algorithm GVNA is linearly convergent when the system has a feasible solution and is not ill-posed, and that algorithm GVNA produces dual variables that demonstrate infeasibility of the system when the system is infeasible. Section 5 presents properties of a particular parameterized conic linear system that are related to $\rho(d)$, the distance to ill-posedness of the data instance $d = (A, b)$ of problem (FP$_d$). Finally, in Section 6, we present the algorithm for resolving the conic linear system (FP$_d$). This algorithm is called algorithm CLS. Algorithm CLS repeatedly calls algorithm GVNA to resolve a parameterized sequence of problems related to (FP$_d$), and so algorithm GVNA and its complexity analysis are the main constructs behind algorithm CLS. The analysis of algorithm CLS also relies heavily on the properties derived in Section 5 related to the distance to ill-posedness $\rho(d)$ of the data instance $d = (A, b)$. Theorem 6.3 contains the main complexity result for algorithm CLS, and is the main result of this paper. Section 7 contains an extensive discussion and interpretation of several aspects of Theorem 6.3.

2 Preliminaries, Assumptions, and Further Notation

We will work in the setup of finite dimensional normed linear vector spaces. Both X and Y are normed linear spaces of finite dimension n and m, respectively, endowed with norms $\|x\|$ for $x \in X$ and $\|y\|$ for $y \in Y$. For $\bar{x} \in X$, let $B(\bar{x}, r)$ denote the ball centered at $\bar{x}$ with radius r, i.e.,

$$B(\bar{x}, r) = \{x \in X \mid \|x - \bar{x}\| \leq r\},$$

and define $B(\bar{y}, r)$ analogously for $\bar{y} \in Y$.

We associate with X and Y the dual spaces X^* and Y^* of linear functionals defined on X and Y, respectively, and whose (dual) norms are denoted by $\|u\|_*$ for $u \in X^*$ and $\|w\|_*$ for $w \in Y^*$. Let $c \in X^*$. In order to maintain consistency with standard linear algebra notation in mathematical programming, we will consider c to be a column vector in the space X^* and will denote the linear function $c(x)$ by $c^t x$. Similarly, for $A \in L(X, Y)$ and $f \in Y^*$, we denote $A(x)$ by Ax and $f(y)$ by $f^t y$. We denote the adjoint of A by A^t.

If C is a convex cone in X, C^* will denote the dual convex cone defined by

$$C^* = \{z \in X^* \mid z^t x \geq 0 \text{ for any } x \in C\}.$$

We will say that a cone C is *regular* if C is a closed convex cone, has a nonempty interior, and is pointed (i.e., contains no line).

Remark 2.1 *If we identify $(X^*)^*$ with X, then $(C^*)^* = C$ whenever C is a closed convex cone. If $C_X = X$, then $C_X^* = \{0\}$; if $C_X = \{0\}$, then $C_X^* = X$. Also, C is regular if and only if C^* is regular.*

We denote the set of real numbers by R and the set of nonnegative real numbers by R_+.

We now recall some facts about norms. Given a finite dimensional linear vector space X endowed with a norm $\|x\|$ for $x \in X$, the dual norm induced on the space X^* is denoted by $\|z\|_*$ for $z \in X^*$, and is defined as:

$$\|z\|_* = \max\{z^t x \mid \|x\| \leq 1\}, \tag{8}$$

and the Hölder inequality $z^t x \leq \|x\|\|z\|_*$ follows easily from this definition. We also point out that if $A = uv^t$, then it is easy to derive that $\|A\| = \|v\|_*\|u\|$.

We make the following assumption throughout concerning the cone C_X:

Assumption 1 *The cone C_X is a regular cone.*

One consequence of the assumption that C_X is regular is that the norm function $\|x\|$ can be approximated by a linear function over the cone C_X. We will measure the extent to which the norm $\|x\|$ can be approximated by a linear function over the convex cone C_X by the number β, see [11], which is defined as follows:

Definition 2.1

$$\begin{array}{lll} \beta = & \sup & \inf\ u^t x \\ & u \in X^* & x \in C_X \\ & \|u\|_* = 1 & \|x\| = 1 \quad . \end{array} \tag{9}$$

Examining Definition 2.1 in detail, let $\bar{u}$ denote that value of $u \in X^*$ that achieves the supremum. Then for any $x \in C_X$, $\beta\|x\| \leq \bar{u}^t x \leq \|x\|$, and so $\|x\|$ is approximated by the linear function $\bar{u}^t x$ to within the factor β over the cone C_X. Therefore, β measures the extent to which $\|x\|$ can be approximated by a linear function $\bar{u}^t x$ on the cone C_X. Furthermore, the linear function $\bar{u}^t x$ is the "best" such linear approximation of $\|x\|$ over the cone C_X. It is easy to see that $\beta \leq 1$, since, for example, $u^t x \leq \|u\|_*\|x\| = 1$ for u and x as in (9). The larger the value of β, the more closely $\|x\|$ is approximated by a linear function $u^t x$ over $x \in C_X$. For this reason, we refer to β as the "coefficient of linearity" of the norm $\|x\|$ over the cone C_X. We have the following properties of β and $\bar{u}$:

Remark 2.2 *(see [11])* $0 < \beta \leq 1$. *There exists $\bar{u} \in$ int C_X^* such that $\|\bar{u}\|_* = 1$ and $\beta = \min\{\bar{u}^t x | x \in C_X,\ \|x\| = 1\}$. For any $x \in C_X$, $\beta\|x\| \leq \bar{u}^t x \leq \|x\|$. The set $\{x \in C_X \mid \bar{u}^t x = 1\}$ is a bounded and closed convex set.*

We note that the constant β depends only on the norm $\|x\|$ and the cone C_X and is independent of the data (A, b) defining the problem (FP$_d$). We also make the following assumption:

Assumption 2 *The linear functional $\bar{u}$ of Remark 2.2 is known and given.*

Given any linear function $c^t x$ defined on $x \in X$, we define the following conic section optimization problem parameterized by c:

$$\begin{array}{rcccl} (\text{CSOP}_c) & z^*(c) & = & \min\limits_{x} & c^t x \\ & & & \text{s.t.} & x \in C_X \\ & & & & \bar{u}^t x = 1. \end{array} \tag{10}$$

Note that (CSOP$_c$) is the problem of minimizing a linear function $c^t x$ over the intersection of the cone C_X and the hyperplane defined by $\bar{u}^t x = 1$. The algorithm developed in this paper will need to solve the problem (CSOP$_c$) at each iteration. Let T_1 denote the number of operations needed to solve (CSOP$_c$). We predicate our algorithm on the premise that T_1 is not excessive, for otherwise the algorithm will not be very efficient.

We now pause to illustrate the above notions on two relevant instances of the cone C_X, namely the nonnegative orthant R^n_+ and the positive semi-definite cone $S^{n\times n}_+$. We first consider the nonnegative orthant. Let $X = R^n$ and $C_X = R^n_+ = \{x \in R^n | x \geq 0\}$. Then we can identify X^* with X and in so doing, $C^*_X = R^n_+$ as well. If $\|x\|$ is given by the L_1 norm $\|x\| = \sum_{j=1}^n |x_j|$, then note that $\|x\| = e^t x$ for all $x \in C_X$ (where e is the vector of ones), whereby the coefficient of linearity is $\beta = 1$ and $\bar{u} = e$. Furthermore, for any c, the problem (CSOP$_c$) is simply the problem of finding the smallest index of the vector c, so that the solution of (CSOP$_c$) is easily computed as $x_c = e^i$ where $i \in \text{argmin}\{c_j \mid j = 1, \ldots, n\}$, and $z^*(c) = c_i$. If instead of the L_1 norm, the norm $\|x\|$ is the L_p norm defined by:

$$\|x\|_p = \left(\sum_{j=1}^n |x_j|^p \right)^{1/p},$$

for $p \geq 1$, then for $x \in C_X$ it is straightforward to show that $\bar{u} = \left(n^{\left(\frac{1}{p}-1\right)} \right) e$ and the coefficient of linearity is $\beta = n^{\left(\frac{1}{p}-1\right)}$. The solution of (CSOP$_c$) is the same as in the case of the L_1 norm up to a scalar multiple.

Now consider the positive semi-definite cone, which has been shown to be of enormous importance in mathematical programming (see Alizadeh [2] and Nesterov and Nemirovskii [16]). Let $X = S^{n\times n}$ denote the set of real $n \times n$ symmetric matrices, and let $C_X = S^{n\times n}_+ = \{x \in S^{n\times n} \mid x \succeq 0\}$, where $x \succeq 0$ is the Löwner partial ordering, i.e., $x \succeq w$ if $x - w$ is a positive semi-definite symmetric matrix. Then C_X is a closed convex cone. We can identify X^* with X, and in so doing it is elementary to derive that $C^*_X = S^{n\times n}_+$, i.e., C_X is self-dual. For $x \in X$, let $\lambda(x)$ denote the n-vector of ordered eigenvalues of x. That is, $\lambda(x) = (\lambda_1(x), \ldots, \lambda_n(x))^t$ where $\lambda_i(x)$ is the ith largest eigenvalue of X. For any

$p \in [1, \infty)$, let the norm of x be defined by

$$\|x\| = \|x\|_p \;=\; \left(\sum_{j=1}^{n} |\lambda_j(x)|^p \right)^{\frac{1}{p}} \quad ,$$

i.e., $\|x\|_p$ is the L_p-norm of the vector of eigenvalues of x. (See Lewis [14] for a proof that $\|x\|_p$ is a norm.) When $p = 1$, $\|x\|_1$ is the sum of the absolute values of the eigenvalues of x. Therefore, when $x \in C_X$, $\|x\|_1 = tr(x) = \sum_{i=1}^{n} x_{ii}$ where x_{ij} is the ijth entry of the real matrix x, and so $\|x\|_1$ is a linear function on C_X. Therefore, when $p = 1$, we have $\bar{u} = I$ and the coefficient of linearity is $\beta = 1$. Furthermore, for any $c \in X^*$, the problem (CSOP$_c$) corresponds to the problem of finding the normalized eigenvector corresponding to the smallest eigenvalue of the matrix c, i.e., (CSOP$_c$) is a minimum eigenvalue problem and is solvable to within machine tolerance in $O(n^3)$ operations in practice (though not in theory), and the solution of (CSOP$_c$) is computed as $x_c = vv^t$ where v is a normalized eigenvector corresponding to the smallest eigenvalue of c, and $z^*(c) = \lambda_1(c)$. We note that solving (CSOP$_c$) does not involve much more than testing whether or not $c \succeq 0$, i.e., testing if $c \in S^{n \times n} = C_X^*$. When $p > 1$, it is easy to show for the norm $\|x\|_p$ over C_X that $\bar{u} = \left(n^{\left(\frac{1}{p} - 1\right)} \right) I$ has $\|\bar{u}\|_* = \|\bar{u}\|_q = 1$ (where $1/p + 1/q = 1$) and that $\beta = n^{\left(\frac{1}{p} - 1\right)}$. The solution of (CSOP$_c$) is the same as in the case when $p = 1$, up to a scalar multiple. Note that when $p = 2$, $\|x\|_2$ corresponds precisely to the Fröbenius norm of x, alternatively defined as $\|x\|_2 = \sqrt{tr(x^t x)}$.

The next assumption concerning the problem (FP$_d$) has to do with the space Y.

Assumption 3 . *The vector space Y is a subspace of a real finite dimensional space Z with Euclidean norm.*

For a vector $y \in Y$, define $\Pr_{C_Y}[y]$ to be the Euclidean projection of y onto the cone C_Y. The algorithm developed in this paper will need to compute at most three projections of vectors onto C_Y at each iteration. Let T_2 denote the number of operations needed to compute the projection $\Pr_{C_Y}[y]$. We predicate our algorithm on the premise that T_2 is not excessive, for otherwise the algorithm will not be very efficient.

We illustrate these additional ideas on three relevant instances of the cone C_Y, namely the nonnegative orthant R^m_+, the origin $\{0\}$, and the positive semi-definite cone $S^{k \times k}_+$. First consider the nonnegative orthant, i.e., $C_Y = R^m_+$ endowed with the Euclidean norm. Then $\Pr_{C_Y}[y] = [y]^+$ where $[y]^+_i = \max\{y_i, 0\}$, $i = 1, \ldots, m$, and so is very easy to compute, using only m operations. Second, consider the origin, i.e., $C_Y = \{0\}$, whereby $\Pr_{C_Y}[y] = 0$ and is trivial to compute (using no operations). Thirdly, consider the positive semi-definite cone, i.e., $C_Y = S^{k \times k}_+$, where $Y = S^{k \times k}$ (and $m = k(k+1)/2$) and Z is the set of all $k \times k$ real matrices, with the Euclidean (i.e., Fröbenius) norm specified on Z. For any $y \in C_Y$, $y = QDQ^t$, where $Q \in R^{k \times k}$ is an orthonormal matrix of eigenvectors and $D \in R^{k \times k}$ is a diagonal matrix of eigenvalues. Let $[D]^+$ be the nonnegative part of D in the following sense: $[D]^+$ is a diagonal matrix with $[D]^+_{ii} = [D_{ii}]^+$, $[D]^+_{ij} = 0$, $i \neq j$. Then $\Pr_{C_Y}[y] = Q[D]^+Q^t$. Therefore the computation of the projection $\Pr_{C_Y}[y]$ is roughly as difficult as computing

the eigenvector decomposition of a symmetric matrix, which in practice can be computed to machine tolerance in $O(k^3) = O(m^{(\frac{3}{2})})$ operations.

The last assumption is:

Assumption 4 $b \notin C_Y$.

Note that if Assumption 4 is not satisfied, then (FP$_d$) has a trivial solution $x = 0$.

We conclude this section with the following miscellaneous remarks and properties:

Remark 2.3 *If $y \in Y$, then $Pr_{C_Y}[y] - y \in C_Y^*$.*

Remark 2.4 *(see [11]). Whenever C_X is a regular cone, then it is possible to choose the norm on X in such a way that the coefficient of linearity $\beta = 1$.*

The following lemmas give a more precise mathematical characterization of the problem of computing the distance from a given point to the boundary of a given convex set. Let S be a closed convex set in R^m and let $f \in R^m$ be given. The distance from f to the boundary of S is defined as:

$$r = \min\{\|f - z\| \mid z \in \partial S\}. \tag{11}$$

Lemma 2.1 *Let r be defined by (11). Suppose $f \in S$. Then*

$$\begin{array}{lllll} r = & \min & \max & \theta & \\ & v & z & & \\ & \|v\| \leq 1 & s.t. & f - z - \theta v = 0 & \\ & & & z \in S. & \end{array}$$

Lemma 2.2 *Let r be defined by (11). Suppose $f \notin S$. Then*

$$\begin{array}{lll} r = & \min & \|f - z\| \\ & z & \\ & s.t. & z \in S. \end{array}$$

The following separating hyperplane result will also be used, whose proof is included here for completeness.

Proposition 2.1 *For every $x \in X$, there exists $z \in X^*$ with the property that $\|z\|_* = 1$ and $\|x\| = z^t x$.*

Proof: If $x = 0$, then any $z \in X^*$ with $\|z\|_* = 1$ will satisfy the statement of the proposition. Therefore, we suppose that $x \neq 0$. Consider $\|x\|$ as a function of x, i.e., $f(x) = \|x\|$. Then $f(\cdot)$ is a real-valued convex function, and so the subdifferential operator $\partial f(x)$ is non-empty for all $x \in X$, see [3]. Consider any $x \in X$, and let $z \in \partial f(x)$. Then

$$f(w) \geq f(x) + z^t(w - x) \text{ for any } w \in X. \tag{12}$$

Substituting $w = 0$ we obtain $\|x\| = f(x) \leq z^t x$. Substituting $w = 2x$ we obtain $2f(x) = f(2x) \geq f(x) + z^t(2x - x)$, and so $f(x) \geq z^t x$, whereby $f(x) = z^t x$. From the Hölder inequality it follows that $\|z\|_* \geq 1$. Now if we let $u \in X$ and set $w = x + u$, we obtain from (12) that $f(u)+f(x) \geq f(u+x) = f(w) \geq f(x)+z^t(w-x) = f(x)+z^t(u+x-x) = f(x)+z^t u$. Therefore, $z^t u \leq f(u) = \|u\|$, and so from (8) we obtain $\|z\|_* \leq 1$. Therefore, $\|z\|_* = 1$. ∎

3 Further Complexity Results for von Neumann's "Simple" Algorithm for Solving Linear Inequalities

The main algorithmic construct in this paper is an extension and generalization of a very simple algorithm due to von Neumann for solving certain types of linear inequality systems, that was presented and analyzed by Dantzig in [6] and [7]. In this section, we review the algorithm of von Neumann, and revisit and expand the complexity results of Dantzig [7]. The purpose of this section is to develop insight into the geometry of von Neumann's algorithm for linear inequalities and to see the underlying simplicity of the algorithm. This section serves as a guide and a prelude to the main algorithmic ideas developed in subsequent sections; however, this section can be skipped with no loss of continuity in the mathematical results.

In order to keep the presentation simple and concrete, we suspend (for this section only) the more abstract notions of normed vector spaces, cones, dual operators, etc., as articulated in Section 2, and instead work with real $m \times n$ matrices in $R^{m \times n}$, etc. We let e denote the n-vector of ones, i.e., $e = (1, 1, \ldots, 1)^t$, and let e^i denote the ith unit vector in R^n. If M is a matrix in $R^{m \times n}$, the ith column of M is denoted by M_i.

We consider the following linear inequality system:

$$\begin{aligned} \text{(P)} \qquad g - Mx &= 0 \\ x &\geq 0 \\ e^t x &= 1, \end{aligned} \tag{13}$$

where $g \in R^m$, $M \in R^{m \times n}$, and $x \in R^n$. We regard the data for this system to be the pair (M, g). Throughout this section we endow R^n with the L_1-norm, namely $\|x\| = \|x\|_1 = \sum_{j=1}^n |x_j|$ for $x \in R^n$, and we endow R^m with the L_2-norm, namely $\|s\| = \|s\|_2 = \sqrt{\sum_{j=1}^n |s_j|^2}$ for $s \in R^m$. Furthermore, we define the matrix norm for M to be the operator norm, namely,

$$\|M\| = \text{maximum}\{\|Mx\| \mid \|x\| \leq 1\} = \max\{\|M_i\| \mid i = 1, \ldots, n\}.$$

The "alternative" system of (P) is:

$$\text{(A)} \qquad M^t s - e(g^t s) > 0. \tag{14}$$

Farkas' Lemma yields the following theorem of the alternative:

Proposition 3.1 *Exactly one of the systems (P) in (13) or (A) in (14) has a solution.*

Notice that solving the system (P) in (13) is equivalent to solving the following optimization problem:

$$\begin{array}{lll} \text{(OP)} & \min\limits_{x} & ||g - Mx|| \\ & \text{s.t.} & x \geq 0 \\ & & e^t x = 1. \end{array} \tag{15}$$

If (P) has a feasible solution, the optimal value of (OP) is 0; otherwise, the optimal value of (OP) is strictly positive. For a given positive ϵ we call x an ϵ-solution of (P) if x is a feasible point for optimization problem (OP) with objective value at most ϵ, i.e., x is an ϵ-solution of (P) if

$$\begin{array}{c} ||g - Mx|| \leq \epsilon \\ x \geq 0 \\ e^t x = 1. \end{array} \tag{16}$$

Given a pre-specified positive tolerance $\epsilon > 0$, the algorithm of von Neumann seeks an ϵ-solution of (P) (as defined in (16)) by iteratively improving the objective value of (OP). It will produce upon termination either an ϵ-solution of (P), or will produce a solution s of the alternative system (A) (14), thus proving infeasibility of (P).

Throughout this section, we will say that a point x is "admissible" if x is feasible for the optimization problem (OP), i.e., x is admissible if $x \geq 0$ and $e^t x = ||x||_1 = 1$.

As a means towards motivating von Neumann's algorithm geometrically, it is helpful to view the problem (P) in the "column" space R^m, by viewing the vector g and the columns of the matrix M as points in the space R^m. Let $\mathcal{H} = \{Mx \mid x \geq 0,\ e^t x = 1\}$ be the convex hull of $M_1, \ldots, M_n$, i.e., $\mathcal{H}$ is the convex hull of the n columns $M_1, \ldots, M_n$ of M. (P) can then be interpreted as the problem of assigning nonnegative weights x_j to the points M_j, $j = 1, \ldots, n$, so that the weighted combination yields the point g, i.e., $Mx = g, x \geq 0$, and where the weights x_j sum to one, i.e, $e^t x = 1$.

We now describe a generic iteration of the von Neumann algorithm. At the beginning of the iteration we have an admissible point $\bar{x}$, i.e., $\bar{x} \geq 0$ and $e^t \bar{x} = 1$. Let $\bar{v}$ be the "residual" at this point, namely, $\bar{v} = g - M\bar{x}$. Then the objective value of (OP) at $\bar{x}$ is $||\bar{v}||$. If $||\bar{v}|| \leq \epsilon$, where ϵ is a pre-specified tolerance parameter, then $\bar{x}$ is an ϵ-solution of (P), and the algorithm terminates. Otherwise, the algorithm finds an index i such that $i \in \text{argmin}_{j=1,\ldots,n}\{\bar{v}^t(g - M_j)\}$. Note that this computation is equivalent to solving the problem

$$\min\{\bar{v}^t(g - Mp) \mid p \geq 0,\ e^t p = 1\} = \min\{\bar{v}^t(g - Mp) \mid p \text{ is admissible}\}$$

$$= \min\{\bar{v}^t(g - M_j) \mid j = 1, \ldots, n\}.$$

Let e^i be the unit vector corresponding to the index i found above. If $\bar{v}^t(g - Me^i) > 0$, it follows that $\bar{s} = -\bar{v}$ satisfies $M^t\bar{s} - e(g^t\bar{s}) > 0$, whereby $\bar{s}$ is a solution of the alternative system (A) of (14), and so the algorithm terminates, having established infeasibility of (P). Otherwise, the direction $e^i - x$ turns out to be a direction of potential improvement of the objective value of (OP). The algorithm takes a step in this direction, with step size found by a constrained line-search. In particular, let $\tilde{x}(\lambda) = \bar{x} + \lambda(e^i - \bar{x})$. Then the next iterate of the algorithm is computed as $\tilde{x} = \tilde{x}(\lambda^*)$ where $\lambda^* = \text{argmin}_{\lambda\in[0,1]}||g - M\tilde{x}(\lambda)||$. The value of λ^* is the solution to the one-dimensional optimization problem:

$$\begin{array}{lll} \min & ||g - M(\bar{x} + \lambda(e^i - \bar{x}))|| & = \quad \min \quad ||(1-\lambda)\bar{v} + \lambda(g - M_i)|| \\ \text{s.t.} & 0 \le \lambda \le 1 & \quad \text{s.t.} \quad 0 \le \lambda \le 1. \end{array}$$

This second portrayal of the of the one-dimensional is equivalent to a simple quadratic program in one variable, and its optimal solution is computed as:

$$\lambda^* = \frac{\bar{v}^t(\bar{v} - (g - M_i))}{||\bar{v} - (g - M_i)||^2}.$$

Note that $\lambda^* \ge 0$ since $\bar{v}^t(g - M_i) \le 0$, and therefore the numerator is positive. Furthermore, $||\bar{v} - (g - M_i)||^2 = ||\bar{v}||^2 + ||g - M_i||^2 - 2\bar{v}^t(g - M_i) \ge ||\bar{v}||^2 - \bar{v}^t(g - M_i) = \bar{v}^t(\bar{v} - (g - M_i))$. This implies that the numerator of λ^* is no larger than the denominator, therefore $\lambda^* \le 1$.

Notice that the new point $\tilde{x}$ found above is a convex combination of the points $\bar{x}$ and the unit vector e^i, and therefore is also an admissible point, i.e, $\tilde{x} \ge 0$ and $e^t\tilde{x} = 1$.

A picture of the algorithm is shown in Figure 1. In the figure, the point g lies in the convex hull of the points $M_1, \ldots, M_5$, and so (P) has a feasible solution. The current admissible point is $\bar{x}$, but note in the figure that $M\bar{x}$ is not equal to g. The direction $M_5 - g$ makes the most negative inner product (most obtuse angle) with $M\bar{x} - g$, and so the index $i = 5$ is chosen for the computation of the next iterate. The next iterate, $\tilde{x}$, is chosen by computing that convex combination of the current iterate $\bar{x}$ and the unit vector e^5 for which the resulting vector $\tilde{x}(\lambda) = (1 - \lambda)\bar{x} + \lambda e^5$ minimizes the value of $||M\tilde{x}(\lambda) - g||$. The resulting value is shown as $M\tilde{x}$ in the figure.

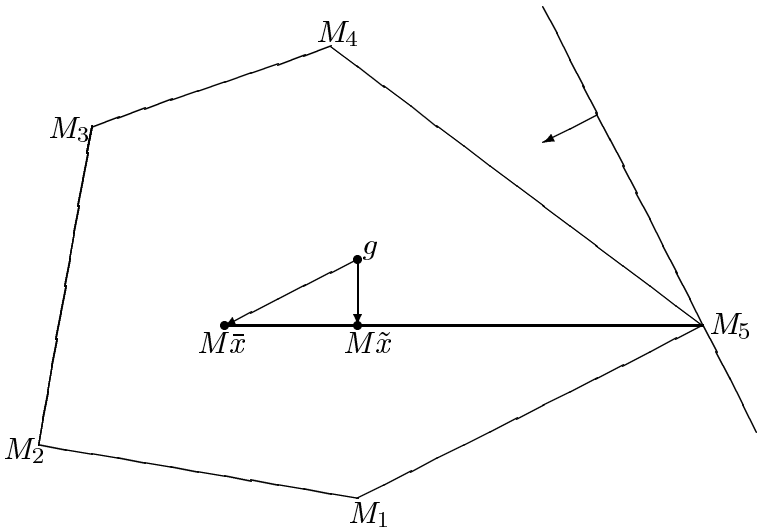

Figure 1:

The formal description of the algorithm is as follows:

von Neumann Algorithm:

- *Data:* (M, g, x^0, ϵ); x^0 admissible: $x^0 \geq 0,\ e^t x^0 = 1,\ \epsilon > 0$.
- *Initialization:* The algorithm is initialized with x^0.
- *Iteration k, $k \geq 1$:* At the start of the iteration we have an admissible point x^{k-1} : $x^{k-1} \geq 0,\ e^t x^{k-1} = 1$.

  Step 1 Compute the residual at x^{k-1}: $v^{k-1} = g - Mx^{k-1}$. If $\|v^{k-1}\| \leq \epsilon$, stop. Return as output x^{k-1} as an ϵ-solution to (P).

  Step 2 Else, find a unit vector e^i such that

  $$(v^{k-1})^t(g - Me^i) = (v^{k-1})^t(g - M_i) = \min\{(v^{k-1})^t(g - M_j) \mid j = 1, \ldots, n\}.$$

  If $(v^{k-1})^t(g - M_i) > 0$, stop, return as output $s = -v^{k-1}$ as a solution to (A).

  Step 3 Else, let

  $$\lambda^{k-1} = \frac{(v^{k-1})^t(v^{k-1} - (g - M_i))}{\|v^{k-1} - (g - M_i)\|^2},$$
  $$x^k = x^{k-1} + \lambda^{k-1}(e^i - x^{k-1}).$$

  Step 4 Let $k \leftarrow k + 1$, go to **Step 1**.

We regard this algorithm as very "elementary" in the sense that the mathematics of each iteration is not very sophisticated, and consequently the work per iteration involves very few computations. At iteration k, the algorithm must perform a small number of vector-vector and matrix-vector multiplications (where these computations can take full advantage of the sparsity of M and/or g as appropriate). In addition, the algorithm must identify the smallest component of the n-vector $(eg^t - M^t)v^{k-1}$ (which takes n operations).

We now analyze the computational complexity of the algorithm. We first present the following result of Dantzig [7] that provides an upper bound on the size of the residual throughout the algorithm. We include the proof for completeness as well as to motivate and lay the foundation for subsequent analysis.

Lemma 3.1 *(Dantzig [7]) Suppose that the von Neumann algorithm has completed k iterations, $k \geq 1$. Then $x^k \geq 0$, $e^t x^k = 1$, and*

$$\|g - Mx^k\| = \|v^k\| \leq \frac{\|M - ge^t\|}{\sqrt{k}} \leq \frac{\|M\| + \|g\|}{\sqrt{k}}.$$

Proof: From the prior discussion, all iterates of the algorithm are admissible, so that $x^k \geq 0$ and $e^t x^k = 1$ for all k. We prove the bound on the norm of the residual by induction on k.

For $k = 1$, $v^1 = g - Mx^1 = g(e^t x^1) - Mx^1 = (ge^t - M)x^1$, since x^1 is an admissible point. Hence,

$$||v^1|| = ||(ge^t - M)x^1|| \leq ||ge^t - M||\,||x^1||_1 = ||ge^t - M||\,e^t x^1 = \frac{||M - ge^t||}{\sqrt{1}}.$$

Next suppose by induction that $||v^{k-1}|| \leq \frac{||M-ge^t||}{\sqrt{k-1}}$. At the end of iteration k we have

$$||v^k|| = ||g - Mx^k|| = ||g - M\left(x^{k-1} + \lambda^{k-1}(e^i - x^{k-1})\right)||$$

$$= ||(1-\lambda^{k-1})(g - Mx^{k-1}) + \lambda^{k-1}(g - M_i)|| = ||(1-\lambda^{k-1})v^{k-1} + \lambda^{k-1}(g - M_i)||, \quad (17)$$

where e^i was found in Step 2 of the algorithm at iteration k. Also, recall that λ^{k-1} is the value of λ that minimizes $||g - M\left(x^{k-1} + \lambda(e^i - x^{k-1})\right)||$ over all $\lambda \in [0,1]$. Therefore, to obtain an upper bound on $||v^k||$ we can substitute any value of $\lambda \in [0,1]$ instead of λ^{k-1} in (17) above. We will substitute $\lambda = \frac{1}{k}$. Making this substitution and squaring (17), we obtain

$$||v^k||^2 = ||(1-\lambda^{k-1})v^{k-1} + \lambda^{k-1}(g - M_i)||^2 \leq \left|\left|\frac{k-1}{k}v^{k-1} + \frac{1}{k}(g - M_i)\right|\right|^2$$

$$= \left(\frac{1}{k}\right)^2 \left[(k-1)^2||v^{k-1}||^2 + ||g - M_i||^2 + 2(k-1)(v^{k-1})^t(g - M_i)\right]. \quad (18)$$

Since the algorithm did not terminate at Step 2 of the kth iteration, it must be true that $(v^{k-1})^t(g - M_i) \leq 0$. Furthermore,

$$||g - M_i|| = ||(ge^t - M)e^i|| \leq \max_{||x||_1=1} ||(ge^t - M)x|| = ||ge^t - M||.$$

Substituting these two inequalities into (18) we obtain

$$||v^k||^2 \leq \left(\frac{1}{k}\right)^2 \left[(k-1)^2||v^{k-1}||^2 + ||ge^t - M||^2\right].$$

From the inductive hypothesis, we have $||v^{k-1}|| \leq \frac{||M-ge^t||}{\sqrt{k-1}}$, and substituting this in the above yields:

$$||v^k||^2 \leq \left(\frac{1}{k}\right)^2 \left[(k-1)^2\frac{||M - ge^t||^2}{k-1} + ||ge^t - M||^2\right] = \frac{||M - ge^t||^2}{k}.$$

Therefore, $||v^k|| \leq \frac{||M-ge^t||}{\sqrt{k}} \leq \frac{||M||+||ge^t||}{\sqrt{k}} = \frac{||M||+||g||}{\sqrt{k}}$. ∎

Lemma 3.1 can then be used directly to establish the following complexity bound:

Theorem 3.1 *(Dantzig [7]) Suppose that the von Neumann algorithm is initiated with data* (M, g, x^0, ϵ). *If (P) has a feasible solution, the algorithm will find an* ϵ*-solution of (P) in at most*

$$\left\lceil \frac{4\max\{||M||^2, ||g||^2\}}{\epsilon^2} \right\rceil$$

iterations. ∎

Note in Theorem 3.1 that the dimension n of the problem plays no direct role in the complexity bound, which is interesting and potentially advantageous. Note also that Theorem 3.1 only covers the case when problem (P) has a feasible solution, and so says nothing about the infeasible case. Furthermore, note that the complexity bound is exponential in $\ln(1/\epsilon)$, and so it would seem that the von Neumann algorithm is an exponential algorithm. However, as shown below, a different analysis of the algorithm shows that the algorithm is actually linear in $\ln(1/\epsilon)$ when problem (P) is "well-posed". We now develop this result.

In order to motivate this next complexity result, we first introduce another geometric construct. Recall the definition of $\mathcal{H}$: $\mathcal{H} = \{Mx \mid x \geq 0,\ e^t x = 1\}$, i.e., $\mathcal{H}$ is the convex hull of the columns of the matrix M. Now let r denote the distance from the vector g to the boundary of $\mathcal{H}$. More formally, we define:

$$r = \inf\{||g - h|| \mid h \in \partial\mathcal{H}\} \quad \text{where} \quad \mathcal{H} = \{Mx \mid x \geq 0,\ e^t x = 1\}. \tag{19}$$

Recalling the geometric interpretation of $\mathcal{H}$, $r = 0$ in (19) precisely when the vector g is on the boundary of the convex hull of the columns of M. Thus, when $r = 0$, the problem (P) has a feasible solution, but arbitrarily small changes in the data (M, g) can yield instances of (P) that have no feasible solution. Therefore when $r = 0$ we can rightfully call the problem (P) unstable, or in the language of data perturbation and condition numbers, the problem (P) is "ill-posed." One can obviously interpret r as the smallest change in the data right-hand-side vector g needed to cause the resulting perturbed instance of (P) to be ill-posed. (However, it should be mentioned in passing that this interpretation can actually be strengthened, and that r is in fact the smallest change in the data (M, g) needed to cause the resulting perturbed instance of (P) to be ill-posed.)

The following proposition gives a useful characterization of the value of r.

Proposition 3.2 *Let* r *be defined by (19).*

If (P) has a feasible solution, then

$$\begin{array}{rlllcllll} r = & \min & \max & \theta & & = & \min & \max & \theta \\ & v & h & & & & v & x & \\ & ||v|| \leq 1 & s.t. & g - h - \theta v = 0 & & & ||v|| \leq 1 & s.t. & g - Mx - \theta v = 0 \\ & & & h \in \mathcal{H} & & & & & x \geq 0 \\ & & & & & & & & e^t x = 1. \end{array}$$

If (P) does not have a feasible solution, then

$$\begin{array}{rlcll} r = & \min & ||g-h|| & = & \min \quad ||g-Mx|| \\ & h & & & x \\ & s.t. & h \in \mathcal{H} & s.t. & x \geq 0 \\ & & & & e^t x = 1. \end{array}$$

Proof: The proof is a straightforward application of Lemmas 2.1 and 2.2. If (P) has a feasible solution, then $g \in \mathcal{H}$, and the characterization of r follows from Lemma 2.1. If (P) does not have a feasible solution, then $g \notin \mathcal{H}$, and the characterization of r follows from Lemma 2.2. ∎

Essentially, Proposition 3.2 states that when (P) has a feasible solution, then r is the radius of the largest ball centered at g and contained in the convex hull of the columns of M; when (P) does not have a feasible solution, then r is the distance from g to the convex hull of the columns of M.

We now present an analysis of the performance of the von Neumann algorithm in terms of the quantity r.

We first prove the following technical result.

Proposition 3.3 *Suppose that (P) has a feasible solution. Let $v^k \neq 0$ be the residual at point x^k, and let e^i be the unit vector computed in Step 2 at the $(k+1)$st iteration. Then* $(v^k)^t(g - Me^i) + r||v^k|| \leq 0$.

Proof: Since (P) has a feasible solution, r is the radius of the largest ball centered at g that is contained in $\mathcal{H}$. From Proposition 3.2, there exists a point $h \in \mathcal{H}$ such that $g + r\frac{v^k}{||v^k||} = h$. By the definition of $\mathcal{H}$, $h = Mx$ for some $x \geq 0$ and $e^t x = 1$, and so

$$g - Mx = -r\frac{v^k}{||v^k||}.$$

Recall that $e^i = \text{argmin}\{(v^k)^t(g - Mp) \mid p \text{ is admissible}\}$. Therefore,

$$(v^k)^t(g - Me^i) \leq (v^k)^t(g - Mx) = -(v^k)^t r\frac{v^k}{||v^k||} = -r||v^k||,$$

which implies that

$$(v^k)^t(g - Me^i) + r||v^k|| \leq 0. \quad ∎$$

Now let us estimate the rate of decrease of the norm of the residual at each iteration of the von Neumann algorithm. As we observed in the proof of Lemma 3.1,

$$||v^k|| = ||(1 - \lambda^{k-1})v^{k-1} + \lambda^{k-1}(g - M_i)||.$$

Substituting the expression for λ^{k-1} given in Step 3 of the algorithm, algebraic manipulation yields:

$$||v^k||^2 = \frac{||v^{k-1} - (g - M_i)||^2||v^{k-1}||^2 - ((v^{k-1})^t(v^{k-1} - (g - M_i))^2}{||v^{k-1} - (g - M_i)||^2}$$

$$= \frac{||v^{k-1}||^2||g - M_i||^2 - ((v^{k-1})^t(g - M_i))^2}{||v^{k-1} - (g - M_i)||^2} = \frac{||v^{k-1}||^2||g - M_i||^2 - ((v^{k-1})^t(g - M_i))^2}{||v^{k-1}||^2 + ||g - M_i||^2 - 2(v^{k-1})^t(g - M_i)}.$$

Recall from Proposition 3.3 that $(v^{k-1})^t(g - Me^i) \leq -r||v^{k-1}|| \leq 0$. Therefore, the numerator in the above expression is bounded from above by $||v^{k-1}||^2||g-M_i||^2-r^2||v^{k-1}||^2$, and the denominator is bounded from below by $||g - M_i||^2$, which yields:

$$||v^k||^2 \leq \frac{||v^{k-1}||^2||g - M_i||^2 - r^2||v^{k-1}||^2}{||g - M_i||^2} = ||v^{k-1}||^2 \left(1 - \frac{r^2}{||g - M_i||^2}\right)$$

$$\leq ||v^{k-1}||^2 \left(1 - \frac{r^2}{||ge^t - M||^2}\right) \leq ||v^{k-1}||^2 e^{-\left(\frac{r^2}{||ge^t - M||^2}\right)},$$

since $||g - M_i|| = ||(ge^t - M)e^i|| \leq ||ge^t - M||$ and $1 - t \leq e^{-t}$ for all t.

We can therefore bound the size of the residual $||v^k||$ inductively:

$$||v^k|| \leq ||v^0||e^{-\left(\frac{kr^2}{2||ge^t - M||^2}\right)}.$$

Therefore, to guarantee that $||v^k|| \leq \epsilon$, it suffices that

$$||v^0||e^{-\left(\frac{kr^2}{2||ge^t - M||^2}\right)} \leq \epsilon.$$

From the above inequality, it is easy to derive that

$$k = \left\lceil \frac{2||ge^t - M||^2}{r^2} \ln\left(\frac{||v^0||}{\epsilon}\right) \right\rceil \leq \left\lceil \frac{8\max\{||g||^2, ||M||^2\}}{r^2} \ln\left(\frac{||v^0||}{\epsilon}\right) \right\rceil$$

iterations are sufficient, when (P) has a feasible solution.

When (P) does not have a feasible solution, the quantity r of (19) is characterized as $r = \inf\{||g - Mx|| \mid x \text{ is admissible}\}$, from Proposition 3.2. Therefore, $||v^k|| = ||g - Mx^k|| \geq r$ for all k. On the other hand, Lemma 3.1 states that if the kth iteration is completed (i.e., the algorithm completes all four steps of the iteration, without establishing infeasibility or an ϵ-solution), then $||v^k|| \leq \frac{||ge^t - M||}{\sqrt{k}}$. Therefore, if k iterations are completed, it follows that

$$r \leq ||v^k|| \leq \frac{||ge^t - M||}{\sqrt{k}}.$$

Hence,

$$k \leq \frac{||ge^t - M||^2}{r^2} \leq \frac{4\max\{||g||^2, ||M||^2\}}{r^2}.$$

Therefore, at most

$$\left\lfloor \frac{4\max\{||g||^2, ||M||^2\}}{r^2} \right\rfloor$$

(complete) iterations of the algorithm will be needed to establish the infeasibility of (P).

Combining these results obtained for the two cases we obtain the following:

Theorem 3.2 *Let r be defined by (19), and suppose that $r > 0$. Then the von Neumann algorithm initialized with data (M, g, x^0, ϵ) will either find an ϵ-solution of (P) or prove infeasibility of (P) by exhibiting a solution of (A). The algorithm will terminate in at most*

$$\left\lceil \frac{8\max\{||g||^2, ||M||^2\}}{r^2} \ln\left(\frac{||g - Mx^0||}{\epsilon}\right) \right\rceil$$

iterations when (P) has a feasible solution. The algorithm will terminate in at most

$$\left\lfloor \frac{4\max\{||g||^2, ||M||^2\}}{r^2} \right\rfloor$$

iterations when (P) does not have a feasible solution. ∎

Now let us compare and contrast the complexity results in Theorem 3.1 and Theorem 3.2. When (P) has a feasible solution, r is the radius of the largest ball centered at g and contained in the convex hull of the columns of M. The bound in Theorem 3.2 will dominate when ϵ is small and when r is not too small, as the bound grows only linearly in $\ln(1/\epsilon)$ and linearly in $1/r$. When $r = 0$, the bound in Theorem 3.2 is meaningless, but in this case problem (P) is ill-posed in the sense that arbitrarily small changes in the data (M, g) can yield instances of (P) that have no feasible solution. When (P) has no feasible solution, it is straightforward to show that $r > 0$, and so the complexity bound in Theorem 3.2 is always meaningful in this case. Recalling the geometric interpretation of r as the smallest change in the data (M, g) needed to create an ill-posed instance of (P), we see that the bounds in Theorem 3.2 are exponential in $\ln(1/r)$. This is inferior to the interior point algorithm of Renegar [20] (as one would expect), whose complexity results for solving (P), for instance, are linear in $\ln(1/r)$. However, the work per iteration in the von Neumann algorithm is very low as compared to an interior point algorithm.

Last of all, the von Neumann algorithm can be viewed as an instance of the Frank-Wolfe algorithm applied to the problem

$$\begin{array}{ll} \min\limits_{x} & ||g - Mx||^2 \\ \text{s.t.} & x \geq 0 \\ & e^t x = 1. \end{array}$$

This observation was communicated verbally to the second author by Yuri Nesterov.

4 A Generalized von Neumann Algorithm for a Conic Linear System in Compact Form

In this section we consider a generalization of the linear inequality system (P) of (13) to a more general conic linear system in compact form, and we extend the von Neumann algorithm of the previous section to handle the more general conic linear system.

We will work with a conic linear system of the form:

$$\begin{array}{rl} \text{(P)} & g - Mx \in C_Y \\ & x \in C_X \\ & \bar{u}^t x = 1, \end{array} \tag{20}$$

where $C_X \subset X$ and $C_Y \subset Y$ are each a closed convex cone in the (finite) n-dimensional normed linear vector space X (with norm $\|x\|$ for $x \in X$) and in the (finite) m-dimensional linear vector space Y (with norm $\|y\|$ for $y \in Y$), respectively. Here $b \in Y$, and $A \in L(X,Y)$ where $L(X,Y)$ denotes the set of all linear operators $A : X \longrightarrow Y$. where $C_X \in X$ and $C_Y \in Y$ are closed convex cones. Assumption 1, Assumption 2, and Assumption 3 are presumed valid for this system, and in particular the linear function $\bar{u}^t x$ in (20) is as described in Assumption 2. Notice that the system (20) is a generalization of (13). Indeed, setting $\bar{u} = e$ and letting $C_Y = \{0\}$ and $C_X = \{x \in R^n \mid x \geq 0\}$, we obtain (13).

The "alternative" system to (P) of (20) is:

$$\begin{array}{rl} \text{(A)} & M^t s - \bar{u}(g^t s) \in \text{int} C_X^* \\ & s \in C_Y^*, \end{array} \tag{21}$$

and a generalization of Farkas' Lemma yields the following duality result:

Proposition 4.1 *Exactly one of the systems (P) in (20) and (A) in (21) has a solution.*

Notice that solving the feasibility problem (P) in (20) is equivalent to solving the following optimization problem:

$$\begin{array}{lll} \min & \|g - Mx - y\| & \\ x, y & & \\ \text{s.t.} & x \in C_X & \\ & \bar{u}^t x = 1 & \\ & y \in C_Y. & \end{array} \tag{22}$$

By the definition of the projection operator $\text{Pr}_{C_Y}[\cdot]$, $y = \text{Pr}_{C_Y}[g - Mx]$ achieves the best objective value of the optimization program in (22) for a given x satisfying $x \in C_X$ and $\bar{u}^t x = 1$, over all $y \in C_Y$. Therefore, the optimization problem (22) can be reformulated as

$$\begin{array}{lll} \text{(OP)} & \min & \|g - Mx - \text{Pr}_{C_Y}[g - Mx]\| \\ & x & \\ & \text{s.t.} & x \in C_X \\ & & \bar{u}^t x = 1. \end{array} \tag{23}$$

If (P) has a feasible solution, the optimal value of (OP) is 0; otherwise, the optimal value of (OP) is strictly positive. For a given positive ϵ we call x an ϵ-solution of (P) if x is a feasible point for optimization problem (OP) with objective value at most ϵ, i.e., x is an ϵ-solution of (P) if

$$\begin{array}{c} ||g - Mx - \Pr_{C_Y}[g - Mx]|| \leq \epsilon \\ x \in C_X \\ \bar{u}^t x = 1. \end{array} \tag{24}$$

Given a pre-specified tolerance $\epsilon > 0$, the algorithm we describe in this section seeks an ϵ-solution of (P) (as defined by (24)) by iteratively improving the objective value of (OP). It will produce upon termination either a point x that is an ϵ-solution of (P), or will produce a solution s to the alternative system (A) (21), thus demonstrating infeasibility of (P).

Throughout this section, we will say that a point x is "admissible" if x is feasible for the optimization problem (OP), namely $x \in C_X$ and $\bar{u}^t x = 1$.

We now describe a generic iteration of the algorithm. At the beginning of the iteration we have an admissible point $\bar{x}$, i.e., $\bar{x} \in C_X$ and $\bar{u}^t \bar{x} = 1$. Let $\bar{z} = g - M\bar{x}$ and $\bar{y} = \Pr_{C_y}[\bar{z}]$. Let $\bar{v}$ be the "residual" at the point $\bar{x}$, namely, $\bar{v} = \bar{z} - \bar{y}$. From Remark 2.3, we know that $\bar{v} \in -C_Y^*$. Notice that $\|\bar{v}\| = ||\bar{z} - \bar{y}|| = ||g - M\bar{x} - \Pr_{C_Y}[g - M\bar{x}]||$, so, $\|\bar{v}\|$ is the objective value of (OP) at point $\bar{x}$. If $||\bar{v}|| \leq \epsilon$, where ϵ is a pre-specified tolerance parameter, then $\bar{x}$ is an ϵ-solution of (P), and the algorithm terminates. Otherwise, the algorithm calls an oracle to solve the following instance of the conic section optimization problem (CSOP) of (10):

$$\begin{array}{lll} \min & \bar{v}^t(g - Mp) = & \min \quad \bar{v}^t(g\bar{u}^t - M)p \\ p & & p \\ \text{s.t.} & p \in C_X & \text{s.t.} \quad p \in C_X \\ & \bar{u}^t p = 1 & \quad\;\; \bar{u}^t p = 1, \end{array} \tag{25}$$

where (25) is an instance of the (CSOP$_c$) with $c = (-M^t + \bar{u}g^t)\bar{v}$. Let $\bar{p}$ be a solution to the problem (25). If $\bar{v}^t(g\bar{u}^t - M)\bar{p} > 0$, then it follows from the optimality of $\bar{p}$ that $(\bar{v}^t(g\bar{u}^t - M))^t \in \text{int}C_X^*$. Therefore, $\bar{s} = -\bar{v} \in C_Y^*$ and satisfies $M^t\bar{s} - \bar{u}(g^t\bar{s}) \in \text{int}C_X^*$. Therefore, $\bar{s}$ is a solution to (A) (21), and the algorithm terminates, having established the infeasibility of (P).

On the other hand, if $\bar{v}^t(g\bar{u}^t - M)\bar{p} \leq 0$, $\bar{p} - \bar{x}$ turns out to be a direction of potential improvement of the objective function of (OP). The algorithm takes a step in the direction $\bar{p} - \bar{x}$ with stepsize found by constrained line-search. In particular, let

$$\tilde{x}(\lambda) = \bar{x} + \lambda(\bar{p} - \bar{x}) \quad \text{and} \quad \tilde{y}(\lambda) = \Pr_{C_Y}[g - M\tilde{x}(\lambda)].$$

Then the next iterate $\tilde{x}$ is computed as $\tilde{x} = \tilde{x}(\lambda^*)$, where

$$\lambda^* = \text{argmin}_{\lambda \in [0,1]} ||g - M\tilde{x}(\lambda) - \tilde{y}(\lambda)||.$$

The value of λ^* and the corresponding value of $\bar{y}(\lambda^*)$ is the solution to the following optimization problem:

$$\begin{array}{lll} \min & ||g - M(\bar{x} + \lambda(\bar{p} - \bar{x})) - y|| & = \quad \min \quad ||(1-\lambda)\bar{z} + \lambda(g - M\bar{p}) - y|| \\ \text{s.t.} & 0 \leq \lambda \leq 1 & \text{s.t.} \quad 0 \leq \lambda \leq 1 \\ & y \in C_Y & \quad\;\; y \in C_Y. \end{array} \tag{26}$$

Notice that $\tilde{x}$ is a convex combination of the two admissible points $\bar{x}$ and $\bar{p}$ and therefore $\tilde{x}$ is also admissible. Also, λ^* above is computed as the solution of the nonlinear optimization problem (26). Finding such λ^* can be a difficult task computationally, even when the projection operator $\Pr_{C_y}[\cdot]$ is easy to compute. Fortunately, the analysis below will show that certain approximations of λ^*, which can be computed easily in closed form, are sufficient for our purposes.

The formal description of the algorithm is as follows:

Algorithm GVNA

- *Data:* (M, g, x^0, ϵ); x^0 is admissible: $x^0 \in C_X,\ \bar{u}^t x^0 = 1,\ \epsilon > 0$.
- *Initialization:* The algorithm is initialized with x^0.
- *Iteration k, $k \geq 1$:* At the start of the iteration we have an admissible point x^{k-1} : $x^{k-1} \in C_X,\ \bar{u}^t x^{k-1} = 1$.

  Step 1 Compute $z^{k-1} = g - Mx^{k-1}$ and $y^{k-1} = \Pr_{C_Y}[z^{k-1}]$. Compute the residual at the point x^{k-1}: $v^{k-1} = z^{k-1} - y^{k-1}$. If $||v^{k-1}|| < \epsilon$, stop. Return as output x^{k-1} as an ϵ-solution to (P).

  Step 2 Else, solve the following conic section optimization problem:

$$\begin{array}{lll} \min\limits_{p} & (v^{k-1})^t(g - Mp) = & \min\limits_{p} \quad (v^{k-1})^t(g\bar{u}^t - M)p \\ \text{s.t.} & p \in C_X & \text{s.t.} \quad p \in C_X \\ & \bar{u}^t p = 1 & \quad\ \bar{u}^t p = 1. \end{array} \tag{27}$$

  Let p^{k-1} be an optimal solution of the optimization problem (27). If $(v^{k-1})^t(g\bar{u}^t - M)p^{k-1} > 0$, stop. Return as output $s = -v^{k-1}$ as a solution to (A).

  Step 3 Else, $(v^{k-1})^t(g\bar{u}^t - M)p^{k-1} \leq 0$, and let

$$(\lambda^{k-1}, y^k) = \text{argmin}\{||g - M(x^{k-1} + \lambda^{k-1}(p^{k-1} - x^{k-1})) - y|| \mid 0 \leq \lambda \leq 1,\ y \in C_Y\},$$

  and

$$x^k = x^{k-1} + \lambda^{k-1}(p^{k-1} - x^{k-1}).$$

  Step 4 Let $k \leftarrow k + 1$, go to Step 1.

Notice that y^k is defined twice for each value of $k \geq 1$: first, it is computed as a solution to the minimization problem in Step 3 of iteration k, then it is calculated in Step 1 of iteration $k + 1$. It can be easily shown that the values of y^k are the same in both cases.

Analogous to the von Neumann algorithm, we regard algorithm GVNA as "elementary" in that the algorithm does not rely on particularly sophisticated mathematics at each iteration (each iteration must perform a few matrix-vector and vector-vector multiplications, in addition to several Euclidean projections onto C_Y and must solve an instance of the conic section optimization problem (CSOP$_c$)). Furthermore the work per iteration will be low so long as T_1 (the number of operations needed to solve the conic section optimization problem (CSOP$_c$)) and T_2 (the number of operations needed to perform the Euclidean projection onto C_Y) are small. This is seen as follows. At iteration k, the algorithm must

perform a small number of vector-vector and matrix-vector multiplications (where these computations can take full advantage of the sparsity of M and/or g as appropriate). At Step 1 of iteration k, algorithm GVNA must compute a projection onto the cone C_Y. At Step 2 of iteration k, the algorithm must solve an instance of the conic section optimization problem (CSOP$_c$) (10) (with $c = (-M^t + \bar{u}g^t)v^{k-1}$). At Step 3 of iteration k, the algorithm calls for exact minimization over $\lambda \in [0,1]$ and $y \in C_Y$ of the distance from the point $g - M(x^{k-1} + \lambda(p^{k-1} - x^{k-1}))$ to the cone C_Y when defining λ^{k-1} and y^k. However, in the analysis of the algorithm which follows, we will show that it is only necessary to evaluate this minimum distance problem at two particular values of $\lambda \in [0,1]$ and $y \in C_Y$, the computation of which involves only a few matrix-vector and vector-vector multiplications. A thorough evaluation of the work per iteration of algorithm GVNA is presented in Remark 4.1 at the end of this section.

We now analyze the computational complexity of the algorithm. We first prove a result, analogous to Lemma 3.1, that provides an upper bound on the size of the residual throughout the algorithm. In the following Lemma, recall the definition of β as given in (9).

Lemma 4.1 *Suppose that algorithm GVNA has completed k iterations, $k \geq 1$. Then $x^k \in C_X$, $\bar{u}^t x^k = 1$, $y^k \in C_Y$, and*

$$||g - Mx^k - y^k|| = ||v^k|| \leq \frac{||M - g\bar{u}^t||}{\beta\sqrt{k}} \leq \frac{||M|| + ||g||}{\beta\sqrt{k}}.$$

Proof: First note that if x is any admissible point (i.e., $x \in C_X$ and $\bar{u}^t x = 1$), then $||x|| \leq \frac{\bar{u}^t x}{\beta} = \frac{1}{\beta}$, and so

$$||g - Mx|| = ||(g\bar{u}^t - M)x|| \leq ||M - g\bar{u}^t||||x|| \leq \frac{||M - g\bar{u}^t||}{\beta}. \tag{28}$$

From the discussion preceding the formal statement of the algorithm, all iterates of the algorithm are admissible, so that $x^k \in C_X$ and $\bar{u}^t x^k = 1$ for all k. We prove the bound on the norm of the residual by induction on k.

For $k = 1$, $||v^1|| = ||z^1 - \mathrm{Pr}_{C_Y}[z^1]|| \leq ||z^1 - 0||$, since $0 \in C_Y$. Substituting the expression for z^1 and using the fact that x^1 is admissible, we obtain:

$$||v^1|| \leq ||g - Mx^1|| = ||(g\bar{u}^t - M)x^1|| \leq ||M - g\bar{u}^t||||x^1|| \leq \frac{||M - g\bar{u}^t||}{\beta} = \frac{||M - g\bar{u}^t||}{\beta\sqrt{1}},$$

where the last inequality above derives from (28).

Next suppose by induction that $||v^{k-1}|| \leq \frac{||M - g\bar{u}^t||}{\beta\sqrt{k-1}}$. At the end of iteration k we have

$$\begin{aligned} ||v^k|| &= ||g - Mx^k - y^k|| = ||(1 - \lambda^{k-1})(g - Mx^{k-1}) + \lambda^{k-1}(g - Mp^{k-1}) - y^k|| \\ &= ||(1 - \lambda^{k-1})z^{k-1} + \lambda^{k-1}(g - Mp^{k-1}) - y^k||, \end{aligned} \tag{29}$$

where p^{k-1} found in Step 2 of the algorithm satisfies $(v^{k-1})^t(g\bar{u}^t - M)p^{k-1} \leq 0$ (from Step 3). Also, recall that λ^{k-1} and y^k were defined in Step 3 as the minimizers of $||(1-\lambda)z^{k-1} +$

$\lambda(g - Mp^{k-1}) - y||$ over all $\lambda \in [0,1]$ and $y \in C_Y$. Therefore, in order to obtain an upper bound on $||v^k||$, we can substitute any $\lambda \in [0,1]$ and $y \in C_Y$ in (29). We will substitute

$$\lambda = \tilde{\lambda} \triangleq \frac{1}{k} \quad \text{and} \quad y = \tilde{y} \triangleq \frac{k-1}{k} y^{k-1}. \tag{30}$$

Making this substitution, we obtain:

$$||v^k|| \leq \left|\left| \frac{k-1}{k} z^{k-1} + \frac{1}{k}(g - Mp^{k-1}) - \frac{k-1}{k} y^{k-1} \right|\right| = \frac{1}{k} ||(k-1)v^{k-1} + (g - Mp^{k-1})||. \tag{31}$$

Squaring (31) yields:

$$||v^k||^2 \leq \frac{1}{k^2} \left((k-1)^2 ||v^{k-1}||^2 + ||g - Mp^{k-1}||^2 + 2(k-1)(v^{k-1})^t (g - Mp^{k-1}) \right). \tag{32}$$

Since the algorithm did not terminate at Step 2 of the kth iteration, it must be true that $(v^{k-1})^t(g - Mp^{k-1}) \leq 0$. Also, since p^{k-1} is admissible, $||g - Mp^{k-1}|| \leq \frac{||M - g\bar{u}^t||}{\beta}$ (from (28)). Combining these results with the inductive bound on $||v^{k-1}||$ and substituting into (32) above yields

$$||v^k||^2 \leq \frac{1}{k^2} \left((k-1)^2 \frac{||M - g\bar{u}^t||^2}{\beta^2 (k-1)} + \frac{||M - g\bar{u}^t||^2}{\beta^2} \right) = \frac{||M - g\bar{u}^t||^2}{\beta^2 k}. \tag{33}$$

Also, $||M - g\bar{u}^t|| \leq ||M|| + ||g\bar{u}^t|| = ||M|| + ||g|| \cdot ||\bar{u}||_* = ||M|| + ||g||$. Combining these results together, we obtain

$$||v^k|| \leq \frac{||M - g\bar{u}^t||}{\beta\sqrt{k}} \leq \frac{||M|| + ||g||}{\beta\sqrt{k}}. \quad \blacksquare$$

Lemma 4.1 implies the following complexity bound:

Lemma 4.2 *Suppose that algorithm GVNA is initiated with data (M, g, x^0, ϵ). If (P) has a feasible solution, the algorithm will find an ϵ-solution of (P) in at most*

$$\left\lceil \frac{4 \max\{||M||^2, ||g||^2\}}{\beta^2 \epsilon^2} \right\rceil$$

iterations. $\blacksquare$

Note as in the von Neumann algorithm that the dimension n of the problem plays no role in this iteration complexity bound, and that this complexity bound is exponential in $\ln(1/\epsilon)$.

We now develop another line of analysis of the algorithm, which will be used to show that the iteration complexity is linear in $\ln(1/\epsilon)$ when the problem (P) is "well-posed". Let

$$\mathcal{H} = \mathcal{H}_M = \{Mx + y \mid x \in C_X,\ \bar{u}^t x = 1,\ y \in C_Y\}. \tag{34}$$

Define

$$r = r(M, g) = \inf\{||g - h|| \mid h \in \partial\mathcal{H}\} \tag{35}$$

where $\mathcal{H}$ is defined above in (34). As in Section 3, it is possible to interpret r as the smallest change in the data (M, g) needed to create an ill-posed instance of (P). This interpretation of r will be proved in Section 5, and then will be utilized in Section 6.

Notice that both $\mathcal{H} = \mathcal{H}_M$ and $r = r(M, g)$ are specific to a given data instance (M, g) of (P), i.e., their definitions depend on the problem data M and g. We will, however, often omit problem data M and g from the notation for $\mathcal{H} = \mathcal{H}_M$ and $r = r(M, g)$. It should be clear from the context which data instance we are referring to.

Proposition 4.2 *Let $r = r(M, g)$ be defined as in (34) and (35). If (P) has a feasible solution, then*

$$\begin{array}{rccccc|ccccc} r = & \min & \max & \theta & & & = & \min & \max & \theta & \\ & v & h & & & & & v & x, y & & \\ & \|v\| \le 1 & s.t. & g - h - \theta v = 0 & & & & \|v\| \le 1 & s.t. & g - Mx - \theta v - y = 0 & \\ & & & h \in \mathcal{H} & & & & & & x \in C_X & \\ & & & & & & & & & \bar{u}^t x = 1 & \\ & & & & & & & & & y \in C_Y. & \end{array} \tag{36}$$

If (P) does not have a feasible solution, then

$$\begin{array}{rccc|ccc} r = & \min & \|g - h\| & = & \min & \|g - Mx - y\| \\ & h & & & x, y & & \\ & s.t. & h \in \mathcal{H} & & s.t. & x \in C_X \\ & & & & & \bar{u}^t x = 1 \\ & & & & & y \in C_Y. \end{array} \tag{37}$$

Proof: The proof is a straightforward consequence of Lemmas 2.1 and 2.2. If (P) has feasible solution, then $g \in \mathcal{H}$, and the characterization of r follows from Lemma 2.1. If (P) does not have a feasible solution, then $g \notin \mathcal{H}$, and the characterization of r follows from Lemma 2.2. ∎

We now present an analysis of the performance of algorithm GVNA in terms of the quantity $r = r(M, g)$.

We first prove the following technical result.

Proposition 4.3 *Suppose that (P) has a feasible solution. Let $v^k \neq 0$ be the residual at point x^k, and let p^k be the direction found in Step 2 of the algorithm at the $(k+1)$st iteration. Then $(v^k)^t(g - Mp^k) + r\|v^k\| \le 0$, where $r = r(M, g)$ is defined in (34) and (35).*

Proof: Since (P) has a feasible solution, r is defined by (36). Therefore, there exists a point $h \in \mathcal{H}$ such that $g - h + r\frac{v^k}{||v^k||} = 0$. By the definition of $\mathcal{H}$, $h = Mx + y$ for some admissible point x and $y \in C_Y$. It follows that

$$g - Mx = y - r\frac{v^k}{||v^k||}$$

for some $y \in C_Y$. Recall that $p^k = \text{argmin}\{(v^k)^t(g - Mp) \mid p \in C_X,\ \bar{u}^t p = 1\}$. Therefore,

$$(v^k)^t(g - Mp^k) \leq (v^k)^t(g - Mx) =$$

$$(v^k)^t y - (v^k)^t r\frac{v^k}{||v^k||} = (v^k)^t y - r||v^k|| \leq -r||v^k||,$$

since $v^k \in -C_Y^*$ from Remark 2.3. Therefore

$$(v^k)^t(g - Mp^k) + r||v^k|| \leq 0. \ \blacksquare$$

Now let us estimate the rate of decrease of the norm of the residual at each iteration of algorithm GVNA. As we observed in the proof of Lemma 4.1,

$$||v^k|| = ||(1 - \lambda^{k-1})z^{k-1} + \lambda^{k-1}(g - Mp^{k-1}) - y^k||, \tag{38}$$

where λ^{k-1} and y^k were defined in Step 3, and are the minimizers of $||(1 - \lambda)z^{k-1} + \lambda(g - Mp^{k-1}) - y||$ over all $\lambda \in [0, 1]$ and $y \in C_Y$. Therefore, in order to obtain an upper bound on $||v^k||$, we can substitute any $\lambda \in [0, 1]$ and $y \in C_Y$ in (38). We will use

$$\lambda = \check{\lambda} \triangleq \frac{(v^{k-1})^t(v^{k-1} - (g - Mp^{k-1}))}{||v^{k-1} - (g - Mp^{k-1})||^2} \text{ and } y = \check{y} \triangleq (1 - \check{\lambda})y^{k-1}. \tag{39}$$

First, notice that $0 \leq \check{\lambda} \leq 1$. To see why this is true, recall that at Step 3 of the algorithm, that $(v^{k-1})^t(g - Mp^{k-1}) \leq 0$, and therefore the numerator of $\check{\lambda}$ is nonnegative. Also, $||v^{k-1} - (g - Mp^{k-1})||^2 = ||v^{k-1}||^2 + ||g - Mp^{k-1}||^2 - 2(v^{k-1})^t(g - Mp^{k-1}) \geq ||v^{k-1}||^2 - (v^{k-1})^t(g - Mp^{k-1}) = (v^{k-1})^t(v^{k-1} - (g - Mp^{k-1}))$. This implies that the numerator of $\check{\lambda}$ is no larger than the denominator. Also, from Proposition 4.3, the denominator is positive, whereby $0 \leq \check{\lambda} \leq 1$.

Substituting the above values of $\check{\lambda}$ and $\check{y}$ into (38), we obtain the desired overestimate:

$$||v^k|| \leq ||(1 - \check{\lambda})(z^{k-1} - y^{k-1}) + \check{\lambda}(g - Mp^{k-1})|| = ||(1 - \check{\lambda})v^{k-1} + \check{\lambda}(g - Mp^{k-1})||. \tag{40}$$

Squaring (40) and substituting the expressions for $\check{\lambda}$ and $\check{y}$ given in (39), algebraic manipulation yields:

$$||v^k||^2 \leq (1 - \check{\lambda})^2||v^{k-1}||^2 + \check{\lambda}^2||g - Mp^{k-1}||^2 + 2\check{\lambda}(1 - \check{\lambda})(v^{k-1})^t(g - Mp^{k-1})$$

$$= \frac{||g - Mp^{k-1}||^2||v^{k-1}||^2 - ((v^{k-1})^t(g - Mp^{k-1}))^2}{||v^{k-1} - (g - Mp^{k-1})||^2}. \tag{41}$$

Recall from Proposition 4.3, that $(v^{k-1})^t(g - Mp^{k-1}) \leq -r||v^{k-1}||$. Thus, $||v^{k-1}||^2(||g - Mp^{k-1}||^2 - r^2)$ is an upper bound on the numerator of (41). Also, $||v^{k-1} - (g - Mp^{k-1})||^2 = ||v^{k-1}||^2 + ||g - Mp^{k-1}||^2 - 2(v^{k-1})^t(g - Mp^{k-1}) \geq ||g - Mp^{k-1}||^2$. Substituting this into (41) yields

$$||v^k||^2 \leq \frac{||v^{k-1}||^2(||g - Mp^{k-1}||^2 - r^2)}{||g - Mp^{k-1}||^2}$$

$$= \left(1 - \frac{r^2}{||g - Mp^{k-1}||^2}\right) ||v^{k-1}||^2 \leq \left(1 - \frac{r^2}{||g\bar{u}^t - M||^2||p^{k-1}||^2}\right) ||v^{k-1}||^2$$

$$\leq \left(1 - \left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2\right) ||v^{k-1}||^2 \tag{42}$$

where the last inequality derives from (28). Applying the inequality $1 - t \leq e^{-t}$ for $t = \left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2$, we obtain:

$$||v^k||^2 \leq ||v^{k-1}||^2 e^{-\left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2},$$

or

$$||v^k|| \leq ||v^{k-1}|| e^{-\frac{1}{2}\left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2}. \tag{43}$$

We can therefore bound the size of the residual $||v^k||$ inductively:

$$||v^k|| \leq ||v^0|| e^{-\frac{k}{2}\left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2}$$

Therefore, to guarantee that $||v^k|| \leq \epsilon$, it suffices that

$$||v^0|| e^{-\frac{k}{2}\left(\frac{\beta r}{||g\bar{u}^t - M||}\right)^2} \leq \epsilon.$$

From the above inequality, it is easy to derive that

$$k = \left\lceil \frac{2||g\bar{u}^t - M||^2}{\beta^2 r^2} \ln\left(\frac{||v^0||}{\epsilon}\right) \right\rceil \leq \left\lceil \frac{8\max\{||g||^2, ||M||^2\}}{\beta^2 r^2} \ln\left(\frac{||v^0||}{\epsilon}\right) \right\rceil$$

iterations are sufficient, when (P) has a feasible solution .

When (P) does not have a feasible solution, $r = r(M, g)$ is characterized by (37). Therefore, for the kth iterate of algorithm GVNA, $||v^k|| = ||g - Mx^k - y^k|| \geq r$. On the other hand, Lemma 4.1 states that if the kth iteration is completed (i.e., the algorithm completes all four steps of the iteration, without either finding an ϵ-solution or demonstrating infeasibility), then $||v^k|| \leq \frac{||g\bar{u}^t - M||}{\beta\sqrt{k}}$. Therefore, if k iterations are completed, it follows that

$$r \leq ||v^k|| \leq \frac{||g\bar{u}^t - M||}{\beta\sqrt{k}}.$$

Hence,

$$k \leq \frac{||g\bar{u}^t - M||^2}{\beta^2 r^2}.$$

Therefore, at most

$$\left\lfloor \frac{||g\bar{u}^t - M||^2}{\beta^2 r^2} \right\rfloor \leq \left\lfloor \frac{(||g|| + ||M||)^2}{\beta^2 r^2} \right\rfloor \leq \left\lfloor \frac{4\max\{||g||^2, ||M||^2\}}{\beta^2 r^2} \right\rfloor$$

(complete) iterations of the algorithm will be needed to establish the infeasibility of (P).

Combining these results obtained for the two cases we obtain the following:

Theorem 4.1 *Let r be defined by (34) and (35), and suppose that $r > 0$. Then algorithm GVNA initialized with data (M, g, x^0, ϵ) will either find an ϵ-solution of (P) or prove infeasibility of (P) by exhibiting a solution of (A). The algorithm will terminate in at most*

$$\left\lceil \frac{8\max\{||g||^2, ||M||^2\}}{\beta^2 r^2} \ln\left(\frac{||g - Mx^0 - y^0||}{\epsilon}\right) \right\rceil$$

iterations when (P) has a feasible solution. The algorithm will terminate in at most

$$\left\lfloor \frac{(||g|| + ||M||)^2}{\beta^2 r^2} \right\rfloor \leq \left\lfloor \frac{4\max\{||g||^2, ||M||^2\}}{\beta^2 r^2} \right\rfloor$$

iterations when (P) does not have a feasible solution. ∎

Recall now that Step 3 of the algorithm calls for exact minimization over $\lambda \in [0,1]$ of the distance from the point $g - M(x^{k-1} + \lambda(p^{k-1} - x^{k-1}))$ to the cone C_Y when defining λ^{k-1}. However, in our analysis of the algorithm we only used upper bounds on the size of the residual at the kth iteration given by particular values of $\lambda \in [0,1]$ and $y \in C_Y$. In particular, in the proof of Lemma 4.1 we used the values $\tilde{\lambda} = \frac{1}{k}$ and $\tilde{y} = \frac{k-1}{k} y^{k-1}$ (see (30)), and in the proof of Proposition 4.3 we used the values $\check{\lambda} = \frac{(v^{k-1})^t (v^{k-1} - (g - Mp^{k-1}))}{||v^{k-1} - (g - Mp^{k-1})||^2}$ and $\check{y} = (1 - \check{\lambda}) y^{k-1}$ (see (39)). Therefore, if we use $\tilde{\lambda}$ or $\check{\lambda}$ as λ^{k-1} (choosing the one that provides us with a residual of a smaller size), we can avoid solving an exact line-search problem while maintaining all the complexity results stated in the Lemmas and the Theorem of this section.

To complete the analysis of algorithm GVNA, we now discuss the computational work performed per iteration. At Step 2 of the algorithm, the algorithm must solve an instance of the conic section optimization problem (CSOP$_c$), and recall that T_1 denotes the number of operations needed to solve an instance of (CSOP$_c$). The algorithm also computes projections of three vectors onto the cone C_Y at each iteration (to compute y^{k-1} as well as to choose which of the values $\tilde{\lambda}$ and $\check{\lambda}$ to use as a stepsize, where $\tilde{\lambda}$ and $\check{\lambda}$ are defined in (30) and (39)), and recall that T_2 denotes the number of operations needed to compute a projection onto the cone C_Y. (It is presumed that neither T_1 nor T_2 are excessive.) We have the following Remark:

Remark 4.1 *Each iteration of algorithm GVNA requires at most*

$$T_1 + 3T_2 + 5m + n + 3mn$$

operations, where T_1 is the number of operations needed to solve an instance of ($CSOP_c$) and T_2 is the number of operations needed to compute a projection onto the cone C_Y. The term $5m+n+3mn$ derives from counting the matrix-vector and vector-vector multiplications. The number of operations required to perform these multiplications can be significantly reduced if the matrices and vectors involved are sparse.

5 Properties of a Parameterized Conic Linear System in Compact Form

In this section we return to the feasibility problem (FP_d) for a general conic linear system, defined in (1):

$$\begin{array}{ll} (\text{FP}_d) & b - Ax \in C_Y \\ & x \in C_X. \end{array} \tag{44}$$

We motivate an approach to solving (FP_d) using algorithm GVNA as a subroutine, and we prove several properties related to this approach that will be necessary for an analysis of the algorithm for solving (FP_d) that will be presented in Section 6.

The "alternative" system of (FP_d) is:

$$\begin{array}{ll} (\text{FA}_d) & A^t s \in C_X^* \\ & s \in C_Y^* \\ & s^t b < 0, \end{array} \tag{45}$$

and a separating hyperplane argument yields the following partial theorem of the alternative:

Proposition 5.1 *If (FA_d) has a solution, then (FP_d) has no solution. If (FP_d) has no solution, then the following system (46) has a solution:*

$$\begin{array}{l} A^t s \in C_X^* \\ s \in C_Y^* \\ s^t b \leq 0 \\ s \neq 0. \end{array} \tag{46}$$

∎

We will call a point x an ϵ-solution to the problem (FP$_d$) if there exists a point y such that

$$\begin{array}{l} ||b - Ax - y|| \leq \epsilon \\ \quad x \in C_X \\ \quad y \in C_Y. \end{array} \tag{47}$$

Notice that x is an ϵ-solution if and only if x is a feasible point for the following optimization problem corresponding to (FP$_d$):

$$\begin{array}{ll} \min & ||b - Ax - y|| \\ x, y & \\ \text{s.t.} & x \in C_X \\ & y \in C_Y \end{array}$$

with objective value at most ϵ.

Proposition 5.2 *For any given $\epsilon > 0$, either (FP$_d$) has an ϵ-solution or (FA$_d$) has a solution, or both.*

Proof: Suppose that $\epsilon > 0$ is given, and that (FP$_d$) does not have an ϵ-solution. We will show that in this case (FA$_d$) has a solution. Consider the following two sets: $S_1 = \{b + v\frac{\epsilon}{2} \mid v \in Y,\ ||v|| \leq 1\}$, and $S_2 = \{Ax + y \mid x \in C_X, y \in C_Y\}$. Since (FP$_d$) does not have an ϵ-solution, then $S_1 \cap S_2 = \emptyset$. Moreover, S_1 is a convex set, and S_2 is a convex cone. Therefore, these two sets can be separated by a hyperplane containing 0. In particular, there exists $s \in Y^*$ such that $s \neq 0$ and

$$s^t(Ax + y) \geq 0 \ \text{ for any } x \in C_X, \ \text{ for any } y \in C_Y, \text{ and } s^t(b + v\frac{\epsilon}{2}) \leq 0 \ \text{ for any } ||v|| \leq 1.$$

Therefore, in particular, $s \in C_Y^*$ and $A^t s \in C_X^*$. Let $v \in Y$ be chosen such that $||v|| = 1$ and $s^t v = ||s||_*$ (see Proposition 2.1). Then $0 \geq s^t(b + v\frac{\epsilon}{2}) = s^t b + \frac{\epsilon ||s||_*}{2}$. Therefore, $s^t b < 0$. Hence, s is a solution of (FA$_d$). ∎

The set of solutions of the system (FP$_d$) is not necessarily a bounded set. In order to apply algorithm GVNA as a means to solve (FP$_d$), it seems reasonable to homogenize the right-hand-side of (FP$_d$) and to add a compactifying constraint, such as:

$$\begin{array}{ll} (\mathrm{P}_0) & b\mu - Aw \in C_Y \\ & \mu + \bar{u}^t w = 1 \\ & \mu \geq 0,\ w \in C_X. \end{array} \tag{48}$$

Notice that (P$_0$) is a feasibility problem of the form (P) (see (20)), where

$$g' = 0,\ M' = [-b, A],$$

$$x' = (\mu, w),$$

$$C_X' = \{\alpha \mid \alpha \geq 0\} \times C_X,$$

$$\bar{u}' = (1, \bar{u}).$$

It can be shown that in the space $X' = X \times R^1$ with the norm defined as $||(\mu, w)|| = |\mu| + ||w||$ the linear approximation of the norm function over the cone C'_X (see Definition 2.1) is indeed given by the vector $\bar{u}' = (1, \bar{u})$. Moreover, if β' is the coefficient of linearity of the norm $||(\mu, w)||$ over the cone C'_X as given in Definition 2.1, it is easy to see that $\beta' = \beta$.

Since (P_0) is a feasibility problem of the form (P), we could apply algorithm GVNA to the system (P_0). If we have a point (μ, w) that is a feasible solution of the system (P_0) with $\mu > 0$, then $x = \frac{w}{\mu}$ is a solution to the original system (FP_d). However, if $\mu = 0$, we cannot apply the above transformation to obtain a solution for the original system. Even if $\mu > 0$, another problem arises if we seek an ϵ-solution of (FP_d): if the above transformation is to yield an ϵ-solution of (FP_d), we must ensure that (w, μ) is an $(\epsilon\mu)$-solution of (P_0), which is troubling in that algorithm GVNA cannot guarantee that μ will stay sufficiently positive.

In order to overcome this difficulty, we propose a different transformation of the system (FP_d) which has the structure needed for applying algorithm GVNA and at the same time has certain tractable properties, as will be shown later in this section. For a given scalar δ, consider the parameterized system

$$\begin{array}{ll} (P_\delta) & b\delta + b\mu - Aw \in C_Y \\ & \mu \geq 0,\ w \in C_X \\ & \mu + \bar{u}^t w = 1. \end{array} \tag{49}$$

Notice that (P_δ) is system of the form (P) (20) where

$$g = b\delta,\ M = [-b, A],$$
$$x' = (\mu, w),$$
$$C'_X = \{\alpha \mid \alpha \geq 0\} \times C_X,$$
$$\bar{u}' = (1, \bar{u}),$$

and solving (P_δ) is equivalent to solving the optimization problem

$$\begin{array}{lll} (OP_\delta) & \min\limits_{\mu, w} & ||b\delta + b\mu - Aw - \mathrm{Pr}_{C_Y}[b\delta + b\mu - Aw]|| \\ & \text{s.t.} & \mu \geq 0,\ w \in C_X \\ & & \mu + \bar{u}^t w = 1 \end{array}$$

which is of the form (OP) (23). As in Section 4, we call a vector (μ, w) "admissible" if (μ, w) is feasible for (OP_δ), and we call (μ, w) a γ-solution of (P_δ) for a specified tolerance $\gamma \geq 0$ if (μ, w) is admissible and $||b\delta + b\mu - Aw - \mathrm{Pr}_{C_Y}[b\delta + b\mu - Aw]|| \leq \gamma$.

It is easy to obtain approximate solutions to the system (FP_d) from approximate solutions to the system (P_δ):

Proposition 5.3 *If $\delta > 0$ and $(\bar{\mu}, \bar{w})$ is an $\epsilon \cdot \delta$-solution of the system (P_δ), then $x = \frac{\bar{w}}{\bar{\mu} + \delta}$ is an ϵ-solution of the system (FP_d).* ∎

Finally, observe that the system (P_0) can be viewed as the instance of the system (P_δ) with the parameter $\delta = 0$.

Recall from Theorem 4.1 of Section 4 that one of the important parameters in the complexity bound for algorithm GVNA is the distance $r = r(M, g)$ of the right-hand-side vector g of the system (P) to the boundary of the set $\mathcal{H} = \mathcal{H}_M = \{Mx+y \mid x \in C_X,\ \bar{u}^t x = 1,\ y \in C_Y\}$, defined in (34) and (35). For the system (P_δ), we have $M = [-b, A]$, $g = b\delta$, and the appropriate substitution for $\bar{u}$ is $\bar{u}' = (1, \bar{u})$. Then in order to describe $r = r(M, g)$ for this system, we first describe the set $\mathcal{H}$ corresponding to the system (P_δ):

$$\begin{aligned}\mathcal{H} = \mathcal{H}_M &= \{Mx' + y \mid x' \in C'_X, y \in C_Y, (\bar{u}')^t x' = 1\} \\ &= \{-b\mu + Aw + y \mid \mu \geq 0, w \in C_X, y \in C_Y, \bar{\mu} + u^t x = 1\}. \end{aligned} \tag{50}$$

For each value of the parameter δ, let $R(\delta)$ denote the corresponding value of the distance r to the boundary of $\mathcal{H}$, i.e, $R(\delta) = r = r(M, g) = r([-b, A], b\delta)$ for the choice of M and g above. Using Proposition 4.2, we can characterize $R(\delta)$ as follows:

Proposition 5.4 *If (P_δ) has a feasible solution, then*

$$\begin{aligned} R(\delta) = \quad & \min_{v} \quad \max_{h,\theta} \quad \theta \\ & ||v|| \leq 1 \quad s.t. \quad b\delta - h - \theta v = 0 \\ & \qquad\qquad\quad h \in \mathcal{H} \\ = \quad & \min_{v} \quad \max_{\mu,w,y,\theta} \quad \theta \\ & ||v|| \leq 1 \quad s.t. \quad b\delta + b\mu - Aw - \theta v - y = 0 \\ & \qquad\qquad\quad \mu \geq 0,\ w \in C_X \\ & \qquad\qquad\quad \mu + \bar{u}^t w = 1 \\ & \qquad\qquad\quad y \in C_Y. \end{aligned} \tag{51}$$

If (P_δ) does not have a feasible solution,

$$\begin{aligned} R(\delta) = \min_{h} \quad & \|b\delta - h\| \quad = \quad \min_{\mu,w,y} \quad \|b\delta + b\mu - Aw - y\| \\ h \in \mathcal{H} \quad & \qquad\qquad\quad s.t. \quad \mu \geq 0,\ w \in C_X \\ & \qquad\qquad\qquad\quad \mu + \bar{u}^t w = 1 \\ & \qquad\qquad\qquad\quad y \in C_Y. \end{aligned} \tag{52}$$

∎

Notice that if $b\delta \in \partial\mathcal{H}$, both definitions (51) and (52) will yield $R(\delta) = 0$.

We now develop properties of (P_δ) and $R(\delta)$ relevant to our analysis. Suppose that (FP_d) has a feasible solution. Then $0 \in \mathcal{H}$. Furthermore, by the definition of $\mathcal{H}$ (see (50)), the system (P_δ) is feasible if and only if the vector $b\delta \in \mathcal{H}$. Therefore, we are interested in the values of δ for which $b\delta \in \mathcal{H}$. Define

$$\bar{\delta} = \max\{\delta \mid (P_\delta) \text{ has a solution}\} = \max\{\delta \mid b\delta \in \mathcal{H}\}. \tag{53}$$

Since $\mathcal{H}$ is a closed convex set, $\bar{\delta}$ is well-defined. Note that with this definition, it must be true that $b\bar{\delta} \in \partial\mathcal{H}$, which is illustrated in Figure 2.

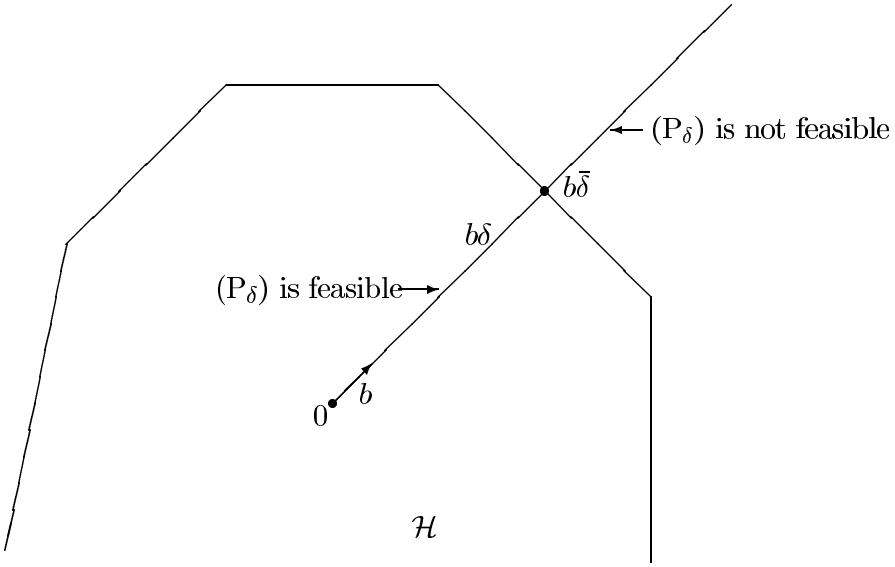

Figure 2

Also, notice that $\bar{\delta} < +\infty$, which is an immediate consequence of Assumption 4. We have the following result, which further characterizes the behavior of $R(\delta)$:

Lemma 5.1 *Suppose (FP_d) has a solution and $R(0) > 0$. Then $\bar{\delta}$ given by (53) satisfies $0 < \bar{\delta} < +\infty$. $R(\delta)$ is a concave function over the range $\delta \in [0, \bar{\delta}]$, and $R(\delta)$ is a nondecreasing convex function over the range $\delta \in [\bar{\delta}, +\infty)$. Furthermore, $R(\delta) \geq \left|\frac{\delta - \bar{\delta}}{\bar{\delta}}\right| R(0)$ for all $\delta \geq 0$.*

Proof: If (FP_d) has a solution, then (P_0) also has a solution. From (51) of Proposition 5.4 and using $v = -\frac{b}{\|b\|}$, we see that there exists $\theta \geq R(0) > 0$ for which $\frac{b\theta}{\|b\|} \in \mathcal{H}$. Therefore, (P_δ) has a solution for $\delta = \frac{\theta}{\|b\|}$, and hence $\bar{\delta} \geq \delta > 0$, since $\theta > 0$.

Next, we show that $R(\delta)$ is a concave function over the range $\delta \in [0, \bar{\delta}]$. Suppose $\delta_1, \delta_2 \in [0, \bar{\delta}]$ and $\lambda \in [0, 1]$. Let $\delta = \lambda\delta_1 + (1 - \lambda)\delta_2$. We want to show that $R(\delta) \geq$

$\lambda R(\delta_1) + (1-\lambda)R(\delta_2)$. This is equivalent to showing that for any vector v with unit norm, $b\delta + (\lambda R(\delta_1) + (1-\lambda)R(\delta_2))v \in \mathcal{H}$.

We know that $b\delta_1 + R(\delta_1)v \in \mathcal{H}$ and $b\delta_2 + R(\delta_2)v \in \mathcal{H}$. Therefore,

$$b\delta + (\lambda R(\delta_1) + (1-\lambda)R(\delta_2))v = \lambda(b\delta_1 + R(\delta_1)v) + (1-\lambda)(b\delta_2 + R(\delta_2)v) \in \mathcal{H},$$

since $\mathcal{H}$ is a convex set, thus demonstrating the concavity of $R(\delta)$ over the range $\delta \in [0, \bar{\delta}]$.

As argued earlier, $\bar{\delta} < \infty$ and $R(\bar{\delta}) = 0$. Hence for $\delta \in [0, \bar{\delta}]$

$$R(\delta) = R\left(\frac{\delta}{\bar{\delta}}\bar{\delta} + \left(1 - \frac{\delta}{\bar{\delta}}\right)0\right) \geq \frac{\delta}{\bar{\delta}}R(\bar{\delta}) + \left(1 - \frac{\delta}{\bar{\delta}}\right)R(0) = \frac{\bar{\delta} - \delta}{\bar{\delta}}R(0) = \left|\frac{\bar{\delta} - \delta}{\bar{\delta}}\right|R(0).$$

Next we examine the behavior of $R(\delta)$ for $\delta \geq \bar{\delta}$. First we show that $R(\delta)$ is a nondecreasing convex function for $\delta \geq \bar{\delta}$. Suppose $\delta_1, \delta_2 \geq \bar{\delta}$. From (52) of Proposition 5.4, we have $R(\delta) = \min\{\|b\delta - h\| \mid h \in \mathcal{H}\}$ for all $\delta > \bar{\delta}$. Suppose $h_1, h_2 \in \mathcal{H}$ are such that $R(\delta_1) = \|b\delta_1 - h_1\|$ and $R(\delta_2) = \|b\delta_2 - h_2\|$. Let $\delta = \lambda\delta_1 + (1-\lambda)\delta_2,\ \lambda \in [0,1]$. Then

$$R(\delta) = \min\{\|b\delta - h\| \mid h \in \mathcal{H}\} \leq \|b\delta - (\lambda h_1 + (1-\lambda)h_2)\|$$

$$= \|\lambda(b\delta_1 - h_1) + (1-\lambda)(b\delta_2 - h_2)\| \leq \lambda\|b\delta_1 - h_1\| + (1-\lambda)\|b\delta_2 - h_2\| = \lambda R(\delta_1) + (1-\lambda)R(\delta_2),$$

and therefore $R(\delta)$ is a convex function over the range $\delta \geq \bar{\delta}$.

To show monotonicity, observe that $R(\bar{\delta}) = 0$ and $R(\delta)$ is nonnegative for all $\delta \geq \bar{\delta}$. Therefore the convexity of $R(\delta)$ over all $\delta \geq \bar{\delta}$ implies that $R(\delta)$ is nondecreasing for all values of $\delta \geq \bar{\delta}$.

Since $b\bar{\delta} \in \partial\mathcal{H}$ and $\mathcal{H}$ is a closed convex set,there exists a supporting hyperplane of $\mathcal{H}$ at $\bar{\delta}b$, i.e., there exists a vector $\bar{h}$ with $\|\bar{h}\|_2 = 1$ such that $\bar{h}^t h \leq \bar{\delta}\bar{h}^t b$ for all $h \in \mathcal{H}$. Also, since $R(0)\bar{h} \in \mathcal{H}$,

$$R(0) = R(0)\bar{h}^t\bar{h} \leq \bar{\delta}\bar{h}^t b.$$

Now, for any $h \in \mathcal{H}$ and $\delta \geq \bar{\delta}$,

$$\|\delta b - h\|_2 = \|\delta b - h\|_2\|\bar{h}\|_2 \geq \bar{h}^t(\delta b - h) = \bar{h}^t(\bar{\delta}b - h + (\delta - \bar{\delta})b)$$

$$= (\delta - \bar{\delta})\bar{h}^t b + \bar{\delta}\bar{h}^t b - \bar{h}^t h \geq (\delta - \bar{\delta})\bar{h}^t b \geq \frac{\delta - \bar{\delta}}{\bar{\delta}}R(0) = \left|\frac{\bar{\delta} - \delta}{\bar{\delta}}\right|R(0).$$

∎

As a last step in our analysis of the properties of the family of problems (P_δ), we establish a relationship between the distance to ill-posedness $\rho(d)$ (see (6)) of the system (FP_d) and $R(0)$ for (FP_d).

First suppose that (FP_d) has a solution. Then (P_0) also has a solution, and by substituting $\delta = 0$ into (51) and applying Theorem 3.2 of [11], it follows that $\beta R(0) \leq \rho(d) \leq R(0)$.

Next suppose that (FP$_d$) does not have a solution. By substituting $\delta = 0$ into (52) and applying Theorem 3.9 of [11], it also follows that $\beta R(0) \leq \rho(d) \leq R(0)$.

Therefore

$$\beta R(0) \leq \rho(d) \leq R(0) \tag{54}$$

and, as a result, we can restate Lemma 5.1 as follows:

Lemma 5.2 *Suppose (FP$_d$) has a solution and $\rho(d) > 0$. Then $\bar{\delta}$ given by (53) satisfies $0 < \bar{\delta} < +\infty$. $R(\delta)$ is a concave function over the range $\delta \in [0, \bar{\delta}]$, and $R(\delta)$ is a nondecreasing convex function over the range $\delta \in [\bar{\delta}, +\infty)$. Furthermore, $R(\delta) \geq \left|\frac{\delta - \bar{\delta}}{\bar{\delta}}\right| \rho(d)$ for all $\delta \geq 0$.*

We conclude this section with a lower bound on $\bar{\delta}$ in terms of the condition number $\mathcal{C}(d)$ (see (7)) of the system (FP$_d$):

Proposition 5.5 *Suppose (FP$_d$) has a solution. Then*

$$\frac{1}{\mathcal{C}(d)} \leq \frac{\rho(d)}{||b||} \leq \bar{\delta}. \tag{55}$$

Proof: We have $R(0) \geq \rho(d)$ from (54) and $R(0) = \inf\{||0 - h|| \mid h \in \partial\mathcal{H}\} \leq ||b\bar{\delta}||$, since $b\bar{\delta} \in \partial\mathcal{H}$ by the definition of $\bar{\delta}$. Therefore, $||b||\bar{\delta} = ||b\bar{\delta}|| \geq R(0) \geq \rho(d)$. Since $b \neq 0$ from Assumption 4, we have

$$\bar{\delta} \geq \frac{\rho(d)}{||b||}.$$

Recall that $\mathcal{C}(d) = \frac{||d||}{\rho(d)} = \frac{\max\{||A||, ||b||\}}{\rho(d)} \geq \frac{||b||}{\rho(d)}$. Therefore,

$$\frac{\rho(d)}{||b||} \geq \frac{1}{\mathcal{C}(d)},$$

and the statement of the Proposition follows. ∎

6 Algorithm CLS for Solving a Conic Linear System

In this section, we present an algorithm called "CLS" (for Conic Linear System) for resolving the conic linear system (FP$_d$), and we establish a complexity result for algorithm CLS in Theorem 6.3. Algorithm CLS either finds an ϵ-solution of (FP$_d$) or demonstrates the infeasibility of the system (FP$_d$) by providing a solution to the alternative system (FA$_d$) (see (45)). Algorithm CLS is a combination of two other algorithms, namely algorithm FCLS (Feasible Conic Linear System) which is used to find an ϵ-solution of (FP$_d$), and algorithm ICLS (Infeasible Conic Linear System), which is used to find a solution to the alternative system (FA$_d$). We first proceed by presenting algorithms FCLS and ICLS, and studying their complexity. We then combine algorithms FCLS and ICLS to form algorithm CLS and study its complexity.

6.1 Algorithm FCLS

Algorithm FCLS is designed to find an ϵ-solution of (FP$_d$) when (FP$_d$) is feasible, by constructing an $(\epsilon\delta)$-solution (μ, w) of the system (P$_\delta$)(where $\delta > 0$ is chosen in such a way that (P$_\delta$) is feasible if (FP$_d$) is feasible), and by transforming (μ, w) into an ϵ-solution x of (FP$_d$) via the transformation $x = w/(\delta + \mu)$ (see Proposition 5.3). It follows from (53) that it is necessary that the parameter δ satisfies $\delta \in (0, \bar{\delta}]$ to ensure that (P$_\delta$) is feasible whenever (FP$_d$) is feasible. From Proposition 5.5, then, it is sufficient to choose $\delta \in (0, \frac{\rho(d)}{\|b\|}]$ to ensure that (P$_\delta$) is feasible whenever (FP$_d$) is feasible. However, the value of $\rho(d)$ is not known in advance, and its determination is no easier than solving (FP$_d$) itself (see [11]). To overcome this difficulty, consider the strategy of applying algorithm GVNA to try to find an $(\epsilon\delta)$-solution of (P$_\delta$) for a sequence of values of δ of the form $\delta = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots$. If algorithm GVNA finds an $(\epsilon\delta)$-solution (μ, w) of (P$_\delta$) then $x = w/(\delta+\mu)$ is an ϵ-solution of (FP$_d$) and the procedures stops. If algorithm GVNA establishes infeasibility of (P$_\delta$) (which corresponds to the case when δ is "too large", i.e., $\delta > \bar{\delta}$), the value of δ is halved and one then applies algorithm GVNA to the new system (P$_\delta$). The algorithm repeatedly halves δ until δ becomes smaller than $\bar{\delta}$, and the system (P$_\delta$) becomes feasible.

Each iteration of the above strategy involves applying algorithm GVNA to the system (P$_\delta$). We know from Theorem 4.1 that the bound on the running time of algorithm GVNA applied to the system (P$_\delta$) is quadratic in $\frac{1}{R(\delta)}$. Furthermore, the analysis of the function $R(\delta)$ in Lemma 5.2 shows that as δ approaches $\bar{\delta}$, $R(\delta)$ approaches 0, which implies an excessive running time for algorithm GVNA if δ is very close to $\bar{\delta}$. Since the value of $\bar{\delta}$ is unknown, the above strategy cannot guarantee that the choice of δ at each iteration will stay sufficiently far away from $\bar{\delta}$. We therefore utilize the following strategy to overcome this difficulty: at each iteration, we apply algorithm GVNA simultaneously to the system (P$_\delta$) and to the system (P$_{\frac{1}{2}\delta}$) where δ is chosen as described above and run the two algorithms in parallel until one of the two algorithms terminates. We will show below that for any value of $\delta > 0$ at least one of the two systems (P$_\delta$) and (P$_{\frac{1}{2}\delta}$) will have a tractable running time, which will lead to good bounds on the running time of the entire method. Let $\hat{\delta}$ denote the value of either δ or $\frac{1}{2}\delta$, corresponding to the run of algorithm GVNA on system (P$_\delta$) or (P$_{\frac{1}{2}\delta}$) that terminates first. If algorithm GVNA finds an $(\epsilon\hat{\delta})$-solution $(\hat{\mu}, \hat{w})$ of (P$_{\hat{\delta}}$), then $x = \hat{w}/(\hat{\delta} + \hat{\mu})$ is an ϵ-solution of (FP$_d$) and the procedures terminates. If algorithm GVNA establishes infeasibility of (P$_{\hat{\delta}}$) (which corresponds to the case when $\hat{\delta}$ is "too large", i.e., $\hat{\delta} > \bar{\delta}$), the value of δ is halved again, and one then applies algorithm GVNA in parallel to the new systems (P$_\delta$) and (P$_{\frac{1}{2}\delta}$). The formal statement of the algorithm is as follows:

Algorithm FCLS

- *Data:* (A, b, ϵ)
- *Iteration* k, $k > 0$
  Step 1 $\delta^k = \delta' = 2^{-(k-1)}$, $\delta'' = \frac{1}{2}\delta'$
  Step 2 Simultaneously run algorithm GVNA with the data sets $(M', g', (x^0)', \epsilon')$ and

$(M'', g'', (x^0)'', \epsilon'')$ defined as follows:

$$(M', g', (x^0)', \epsilon') = ([-b, A], \delta' b, (0, 1), \delta' \epsilon)$$

$$(M'', g'', (x^0)'', \epsilon'') = ([-b, A], \delta'' b, (0, 1), \delta'' \epsilon)$$

and with $\bar{u}' = (1, \bar{u})$ substituting for $\bar{u}$, until one of the two runs terminated.

Step 3 Let $(\hat{M}, \hat{g}, \hat{x}, \hat{\epsilon})$ denote the data set of the run that has terminated first, and let $\hat{\delta}$ denote the corresponding value of δ (either δ' or δ'').

If algorithm GVNA has returned an $\hat{\epsilon}$ solution $(\hat{\mu}, \hat{w})$ of $\mathrm{P}_{\hat{\delta}}$, stop. Return $x = \frac{\hat{w}}{\hat{\mu}+\hat{\delta}}$ as an ϵ-solution of (FP_d).

Else, set $k \leftarrow k + 1$ and repeat **Step 1.**

(Note that although Step 2 of each iteration of algorithm FCLS calls for algorithm GVNA to be run in parallel on two data sets, terminating when one of the two runs terminates, there is no necessity for parallel computation *per se*. Let d' and d'' denote the two data sets of Step 2 of the algorithm. A sequential implementation of Step 2 is to run one iteration of algorithm GVNA on data d', followed by one iteration of algorithm GVNA on d'', followed by the next iteration of algorithm GVNA on data d', followed by the next iteration of algorithm GVNA on d'', etc., until one of the iterations yields the stopping criterion.)

We now analyze the complexity of algorithm FCLS. We assume that (FP_d) has a feasible solution, and that $\rho(d) > 0$. The analysis of the complexity of the algorithm FCLS consists of three steps. First, we analyze the running time of the kth iteration. Next, we will provide an upper bound on the total number of iterations. Finally, we will combine the two results to derive a worst-case running time of FCLS.

We start by analyzing the running time of the kth iteration of FCLS. Each iteration of FCLS consists of running algorithm GVNA in parallel on the data instances for the two problems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$, until one of the runs terminates. Note that computing the running time of either of the two instances provides an upper bound on the running time of the kth iteration. We will therefore estimate the running time of algorithm GVNA on the more "convenient" instance, i.e., the instance that will give a better complexity bound. As should be intuitively clear, the running time of the kth iteration of FCLS will be strongly related to the values of $\delta' = \delta^k$ and $\delta'' = \frac{1}{2}\delta'$ and their position relative to $\bar{\delta}$.

First consider the case when $\delta'' > \bar{\delta}$. In this case, both $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ have no feasible solution. We will analyze the running time of algorithm GVNA running on $(\mathrm{P}_{\delta'})$. It follows from Theorem 4.1 that algorithm GVNA will take at most

$$\left\lfloor \frac{(||b||\delta' + |||[-b, A]|||)^2}{\beta^2 R(\delta')^2} \right\rfloor \tag{56}$$

(completed) iterations of algorithm GVNA to establish infeasibility of $(\mathrm{P}_{\delta'})$. In order to estimate $R(\delta')$, notice that $\delta' = 2\delta'' \geq 2\bar{\delta}$. From Lemma 5.2, we conclude that

$$R(\delta') \geq R(2\bar{\delta}) \geq \left| \frac{2\bar{\delta} - \bar{\delta}}{\bar{\delta}} \right| \rho(d) = \rho(d). \tag{57}$$

Substituting for $R(\delta')$ in (56) and recognizing that $|||[-b, A]||| = \|d\|$ and $\delta' \leq 1$, we conclude that it will take at most

$$\left\lfloor \frac{(\|b\| + \|d\|)^2}{\beta^2 \rho(d)^2} \right\rfloor \tag{58}$$

iterations of algorithm GVNA to terminate the kth iteration of FCLS.

Next, consider the case when $\delta'' \leq \bar{\delta} < \delta'$. In this case, $(\mathrm{P}_{\delta''})$ has a solution and $(\mathrm{P}_{\delta'})$ does not have a solution. Note the following trivial fact: either $\delta' > \frac{4}{3}\bar{\delta}$, or $\delta'' \leq \frac{2}{3}\bar{\delta}$. We will analyze these two sub-cases separately below.

In the first sub-case, $\delta' > \frac{4}{3}\bar{\delta}$, and we analyze the running time of algorithm GVNA on the system $(\mathrm{P}_{\delta'})$ for the sub-case . It follows from Theorem 4.1 that it will take at most

$$\left\lfloor \frac{(\|b\|\delta' + |||[-b, A]|||)^2}{\beta^2 R(\delta')^2} \right\rfloor \tag{59}$$

(completed) iterations of algorithm GVNA to demonstrate the infeasibility of $(\mathrm{P}_{\delta'})$. In order to estimate $R(\delta')$, we exploit the inequality $\delta' > \frac{4}{3}\bar{\delta}$. From Lemma 5.2, we conclude that

$$R(\delta') \geq R\left(\frac{4}{3}\bar{\delta}\right) \geq \left| \frac{\frac{4}{3}\bar{\delta} - \bar{\delta}}{\bar{\delta}} \right| \rho(d) = \frac{1}{3}\rho(d). \tag{60}$$

Substituting for $R(\delta')$ in (59) and recognizing that $|||[-b, A]||| = \|d\|$ and $\delta' \leq 1$, we conclude that it will take at most

$$\left\lfloor \frac{9(\|b\| + \|d\|)^2}{\beta^2 \rho(d)^2} \right\rfloor \tag{61}$$

iterations of algorithm GVNA to terminate this iteration of FCLS.

In the second sub-case, $\delta'' \leq \frac{2}{3}\bar{\delta}$, and we analyze the running time of algorithm GVNA on the system $(\mathrm{P}_{\delta''})$ for this sub-case. It follows from Theorem 4.1 that it will take at most

$$\left\lceil \frac{8 \max\{\|b\delta''\|^2, |||[-b, A]|||^2\}}{\beta^2 R(\delta'')^2} \ln\left(\frac{\|b\delta'' + b \cdot 1 - A \cdot 0 - y^0\|}{\epsilon\delta''} \right) \right\rceil \tag{62}$$

iterations of algorithm GVNA initialized at the point $(\mu, w) = (1, 0)$ to find an $(\epsilon\delta'')$-solution of $(\mathrm{P}_{\delta''})$. To estimate $R(\delta'')$, let $\delta'' = \alpha\bar{\delta}$. From the inequality $\delta'' \leq \frac{2}{3}\bar{\delta}$, it follows that $0 < \alpha \leq \frac{2}{3}$. From Lemma 5.2, we conclude that

$$R\left(\delta''\right) = R(\alpha\bar{\delta}) \geq \left| \frac{\alpha\bar{\delta} - \bar{\delta}}{\bar{\delta}} \right| \rho(d) = (1 - \alpha)\rho(d) \geq \frac{1}{3}\rho(d). \tag{63}$$

After substituting for $R(\delta'')$ in (62) and recognizing that $\delta'' \leq 1$, algebraic manipulation of (62) yields that at most

$$\left\lceil \frac{72 \max\{\|b\|^2, \|d\|^2\}}{\beta^2 \rho(d)^2} \ln\left(\frac{\|b\delta'' + b - y^0\|}{\epsilon\delta''} \right) \right\rceil = \left\lceil \frac{72\|d\|^2}{\beta^2 \rho(d)^2} \ln\left(\frac{\|b\delta'' + b - y^0\|}{\epsilon\delta''} \right) \right\rceil \tag{64}$$

iterations of algorithm GVNA are needed to find an $(\epsilon\delta'')$-solution of $(\mathrm{P}_{\delta''})$.

Combining (61) and (64), we conclude that in the case when $\delta'' \leq \bar{\delta} < \delta'$, at most

$$\max\left\{ \left\lfloor \frac{9(\|b\| + \|d\|)^2}{\beta^2 \rho(d)^2} \right\rfloor, \left\lceil \frac{72\|d\|^2}{\beta^2 \rho(d)^2} \ln\left(\frac{\|b\delta'' + b - y^0\|}{\epsilon\delta''} \right) \right\rceil \right\} \tag{65}$$

iterations of algorithm GVNA are needed to complete the kth iteration of FCLS.

Lastly, consider the case when $\delta' \leq \bar{\delta}$. In this case, both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ have solutions. We will analyze the running time of algorithm GVNA running on $(\mathrm{P}_{\delta''})$. It follows from Theorem 4.1 that it will take at most

$$\left\lceil \frac{8\max\{||b\delta''||^2, ||[-b, A]||^2\}}{\beta^2 R(\delta'')^2} \ln\left(\frac{||b\delta'' + b - y^0||}{\epsilon\delta''}\right)\right\rceil \tag{66}$$

iterations of algorithm GVNA initialized at the point $(\mu, w) = (1, 0)$ to find an $(\epsilon\delta'')$-solution of $(\mathrm{P}_{\delta''})$. To estimate $R(\delta'')$, let $\delta'' = \alpha\bar{\delta}$. Since $2\delta'' = \delta' \leq \bar{\delta}$, it follows that $0 < \alpha \leq \frac{1}{2}$. From Lemma 5.2, we conclude that

$$R\left(\delta''\right) = R(\alpha\bar{\delta}) \geq \left|\frac{\alpha\bar{\delta} - \bar{\delta}}{\bar{\delta}}\right| \rho(d) = (1-\alpha)\rho(d) \geq \frac{1}{2}\rho(d). \tag{67}$$

After substituting for $R(\delta'')$ in (66) and recognizing that $\delta'' \leq 1$, algebraic manipulation of (66) yields that at most

$$\left\lceil \frac{32||d||^2}{\beta^2\rho(d)^2} \ln\left(\frac{||b\delta'' + b - y^0||}{\epsilon\delta''}\right)\right\rceil \tag{68}$$

iterations of algorithm GVNA are needed to find an $(\epsilon\delta'')$-solution of $(\mathrm{P}_{\delta''})$.

Now let us analyze the number of iterations that algorithm FCLS will perform. Observe that both $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ have no feasible solution when $\delta'' = \frac{1}{2}\delta^k > \bar{\delta}$. Since $\delta^k = (\frac{1}{2})^{k-1}$, it is easy to see that algorithm FCLS will perform at most

$$\max\left(\left\lfloor \log_2\left(\frac{1}{\bar{\delta}}\right)\right\rfloor, 0\right) = \left\lfloor \log_2\left(\frac{1}{\bar{\delta}}\right)\right\rfloor^+ \tag{69}$$

iterations in which both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ are infeasible.

When δ^k satisfies $\frac{1}{2}\delta^k = \delta'' \leq \bar{\delta} < \delta^k = \delta'$, algorithm FCLS performs an iteration in which exactly one of the two systems (P_{δ^k}) and $(\mathrm{P}_{\frac{1}{2}\delta^k})$ has a feasible solution. If algorithm FCLS does not find an approximate solution of that system, algorithm FCLS proceeds with the next iteration, in which $\delta^{k+1} = \frac{1}{2}\delta^k$, and it follows that both systems $(\mathrm{P}_{\delta^{k+1}})$ and $(\mathrm{P}_{\frac{1}{2}\delta^{k+1}})$ have solutions. Therefore, an approximate solution of one of the two systems will be found during this iteration, which will terminate algorithm FCLS.

Summarizing, algorithm FCLS will perform at most $\left\lfloor \log_2\left(\frac{1}{\bar{\delta}}\right)\right\rfloor^+$ iterations in which both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ are infeasible, one iteration in which exactly one system of $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ is feasible, and at most one iteration in which both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ are feasible.

Substituting the lower bound on $\bar{\delta}$ from Proposition 5.5 into (69) and (58), algebraic manipulation of these expressions yield that algorithm FCLS will perform at most

$$\left\lfloor \log_2\left(\frac{1}{\bar{\delta}}\right)\right\rfloor^+ \leq \left\lfloor \log_2\left(\mathcal{C}(d)\right)\right\rfloor \tag{70}$$

iterations in which both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ are infeasible. Each of these iterations will perform at most

$$2\left\lfloor \frac{(||b|| + ||d||)^2}{\beta^2\rho(d)^2}\right\rfloor \leq 2\left\lfloor \frac{4||d||^2}{\beta^2\rho(d)^2}\right\rfloor = 2\left\lfloor \frac{4\mathcal{C}(d)^2}{\beta^2}\right\rfloor \tag{71}$$

iterations of algorithm GVNA (from (58) and since algorithm GVNA is run in parallel on two systems).

Then algorithm FCLS will perform one iteration in which exactly one of the systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ is feasible. This iteration will perform at most

$$2\max\left\{\left\lfloor\frac{36||d||^2}{\beta^2\rho(d)^2}\right\rfloor,\left\lceil\frac{72||d||^2}{\beta^2\rho(d)^2}\ln\left(\frac{||b\delta''+b-y^0||}{\epsilon\delta''}\right)\right\rceil\right\} \tag{72}$$

iterations of algorithm GVNA (from (65) and, again, since algorithm GVNA is run in parallel on two systems). Since y^0 was chosen so as to minimize $||b\delta''+b-y||$ over all points $y\in C_Y$, we can replace y^0 by 0 to obtain an upper bound. Also, during this iteration $\bar{\delta}<\delta'=2\delta''\leq\delta^1=1$, so we can bound the expression in (72) above by

$$\begin{aligned}
&2\max\left\{\left\lfloor\frac{36||d||^2}{\beta^2\rho(d)^2}\right\rfloor,\left\lceil\frac{72||d||^2}{\beta^2\rho(d)^2}\ln\left(\frac{\frac{1}{2}||b||+||b||}{\epsilon(\frac{1}{2}\bar{\delta})}\right)\right\rceil\right\}\\
&=2\max\left\{\left\lfloor\frac{36\mathcal{C}(d)^2}{\beta^2}\right\rfloor,\left\lceil\frac{72\mathcal{C}(d)^2}{\beta^2}\ln\left(\frac{3||b||}{\epsilon\bar{\delta}}\right)\right\rceil\right\}\\
&\leq 2\max\left\{\left\lfloor\frac{36\mathcal{C}(d)^2}{\beta^2}\right\rfloor,\left\lceil\frac{72\mathcal{C}(d)^2}{\beta^2}\ln\left(\frac{3\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil\right\}.
\end{aligned} \tag{73}$$

Lastly, algorithm FCLS will perform at most one iteration in which both systems $(\mathrm{P}_{\delta'})$ and $(\mathrm{P}_{\delta''})$ are feasible. This iteration will perform at most

$$2\left\lceil\frac{32||d||^2}{\beta^2\rho(d)^2}\ln\left(\frac{||b\delta''+b-y^0||}{\epsilon\delta''}\right)\right\rceil \tag{74}$$

iterations of algorithm GVNA (from (68) and, again, since algorithm GVNA is run in parallel on two systems) . Again, we can replace y^0 by 0. Suppose the current iteration is not the first iteration of algorithm FCLS. Then at the previous iteration of algorithm FCLS one of the two systems was infeasible. Hence, $\frac{1}{2}\bar{\delta}\leq\delta'\leq\bar{\delta}$, whereby $\delta''\geq\frac{1}{4}\bar{\delta}$. Also $\delta''\leq\frac{1}{2}$. Therefore, we can bound the expression in (74) above by

$$\begin{aligned}
&2\left\lceil\frac{32\mathcal{C}(d)^2}{\beta^2}\ln\left(\frac{||\frac{1}{2}b+b||}{\epsilon(\frac{1}{4}\bar{\delta})}\right)\right\rceil\\
&=2\left\lceil\frac{32\mathcal{C}(d)^2}{\beta^2}\ln\left(\frac{6||b||}{\epsilon\bar{\delta}}\right)\right\rceil\\
&\leq 2\left\lceil\frac{32\mathcal{C}(d)^2}{\beta^2}\ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil,
\end{aligned} \tag{75}$$

using Proposition 5.5. If, however, the current iteration is in fact the first iteration of algorithm FCLS, then $\delta''=\frac{1}{2}$ and $\delta'=1\leq\bar{\delta}$, whereby an upper bound on (74) is

$$2\left\lceil\frac{32||d||^2}{\beta^2\rho(d)^2}\ln\left(\frac{\frac{3}{2}||b||}{\frac{1}{2}\epsilon}\right)\right\rceil$$

$$= 2\left\lceil \frac{32\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{3||b||}{\epsilon}\right)\right\rceil \leq 2\left\lceil \frac{32\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil. \quad (76)$$

Combining (70), (71), (73), (75), and (76), we obtain:

Theorem 6.1 *Suppose (FP_d) has a solution and that $\rho(d) > 0$. If we apply algorithm FCLS with data (A, b, ϵ) where $\epsilon > 0$, then the algorithm will return an ϵ-solution of (FP_d) in at most*

$$2\left(\left\lfloor \log_2\left(\mathcal{C}(d)\right)\right\rfloor \cdot \left\lfloor \frac{4\mathcal{C}(d)^2}{\beta^2}\right\rfloor + \max\left\{\left\lfloor \frac{36\mathcal{C}(d)^2}{\beta^2}\right\rfloor, \left\lceil \frac{72\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{3\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil\right\}$$

$$+ \left\lceil \frac{32\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil\right)$$

iterations of algorithm GVNA. The work per iteration of algorithm GVNA is as described in Remark 4.1. ∎

6.2 Algorithm ICLS

Algorithm ICLS is designed to demonstrate infeasibility of (FP_d), by providing a solution of the alternative system (FA_d) (45), in the case when (FP_d) is infeasible. Recall the system P_0 of (48), which is an instance of (P_δ) for $\delta = 0$. Using (21), the alternative system of (P_0) is

$$(A_0) \quad \begin{array}{l} A^t s \in \text{int} C_X^* \\ s \in C_Y^* \\ s^t b < 0, \end{array}$$

and note that if s is a solution of (A_0), then s is also a solution of (FA_d). Therefore it is appealing to use algorithm GVNA on the system (P_0), to seek a solution of (A_0), which is precisely what we do. This leads to the following algorithm for demonstrating the infeasibility of (FP_d), called algorithm ICLS:

Algorithm ICLS

- *Data:* (A, b, ϵ)
- **Step 1** Run algorithm GVNA with data set $(\bar{M}, \bar{g}, \bar{x}, \bar{\epsilon}) = ([-b, A], 0, (0, 1), 0)$ and with $\bar{u}' = (1, \bar{u})$ substituting for $\bar{u}$. If algorithm GVNA terminates and returns a solution s of (A_0) then s solves (FA_d), demonstrating infeasibility of (FP_d).

Note that in algorithm ICLS, that algorithm GVNA is run on the system (P_0) with a feasibility tolerance of $\bar{\epsilon} = 0$, at Step 1. Suppose that (FP_d) has no solution, and suppose that $\rho(d) > 0$. Then algorithm GVNA cannot return an $\bar{\epsilon} = 0$-solution of the system (P_0).

To see why this is true, suppose the contrary. Then there exists (μ, w) feasible for (48), and so from Theorem 3.9 of [11], $\rho(d) = 0$, a contradiction.

Suppose that algorithm GVNA demonstrates the infeasibility of the system (P_0). It follows from Theorem 4.1 that it will do so in at most

$$\left\lfloor \frac{|||[-b, A]||^2}{\beta^2 R(0)^2} \right\rfloor \tag{77}$$

iterations. Furthermore, recall from (54) that $R(0) \geq \rho(d)$. Also, $|||[-b, A]||| = ||d||$. Therefore, an upper bound on (77) is:

$$\left\lfloor \frac{||d||^2}{\beta^2 \rho(d)^2} \right\rfloor \leq \left\lfloor \frac{\mathcal{C}(d)^2}{\beta^2} \right\rfloor \tag{78}$$

iterations of algorithm GVNA. This leads to the following complexity bound:

Theorem 6.2 *Suppose that (FP_d) has no solution, and that $\rho(d) > 0$. If we apply algorithm ICLS with data (A, b, ϵ), then the algorithm will terminate in at most*

$$\left\lfloor \frac{\mathcal{C}(d)^2}{\beta^2} \right\rfloor$$

iterations of algorithm GVNA. Upon termination, algorithm ICLS will return a solution s of (FA_d), thereby demonstrating infeasibility of (FP_d). The work per iteration of algorithm GVNA is as described in Remark 4.1. ∎

6.3 Algorithm CLS

Algorithm CLS, described below, is a combination of algorithm FCLS and algorithm ICLS developed above. Algorithm CLS is designed to resolve the system (FP_d) by either finding an ϵ-solution of the system (FP_d) or demonstrating the infeasibility of (FP_d) by providing a solution of (FA_d). Since it is not known in advance whether (FP_d) has a solution or not, the algorithm CLS will run both algorithms FCLS and ICLS in parallel, and will terminate when either one of the two algorithms FCLS or ICLS terminates. The formal description of algorithm CLS is as follows:

CLS

- *Data:* (A, b, ϵ)
- **Step 1** Run algorithms FCLS and ICLS simultaneously on the data set (A, b, ϵ), until one of the two algorithms terminates. If algorithm FCLS terminates first, stop and return its output as an ϵ-solution of (FP_d). If algorithm ICLS terminates first, proceed to Step 2.

- **Step 2** If algorithm ICLS return a solution s of (FA_d), stop: (FP_d) is infeasible. Else, continue running algorithm FCLS until it terminates.

(Again, there is no necessity for parallel computation *per Ye* in Step 1. A sequential implementation of Step 1 is to run one iteration of algorithm FCLS, followed by one iteration of algorithm ICLS, followed by the next iteration of algorithm FCLS, followed by the next iteration of algorithm ICLS, etc., until one of the iterations terminates.)

Note that Step 2 of algorithm CLS is necessary for the following reason. If algorithm ICLS terminates first, it will return exactly one of two possible outputs: (i) either it will return a solution to (FA_d), thus proving infeasibility of (FP_d), or (ii) it will find an exact solution of (P_0), if (FP_d) is feasible. In the latter case, the solution returned by GVNA in algorithm ICLS is an exact solution of (P_0), but it cannot necessarily be transformed into an ϵ-solution of (FP_d) (see the discussion in Section 5). For this reason, we discard the solution of (P_0) and continue running algorithm FCLS until it provides an ϵ-solution of (FP_d).

Combining the complexity results for algorithms FCLS and ICLS from Theorems 6.1 and 6.2 we obtain the following:

Theorem 6.3 *Suppose that $\rho(d) > 0$. The algorithm CLS will either find an ϵ-solution of the system (FP_d) or prove infeasibility of (FP_d) by exhibiting a solution of (FA_d). It will terminate in at most*

$$\left\lceil \left(\frac{21\mathcal{C}(d)}{\beta} \right)^2 \left(log_2(\mathcal{C}(d)) + \max\left\{ 1, \ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon} \right) \right\} \right) \right\rceil + 8 \tag{79}$$

iterations of algorithm GVNA when (FP_d) has a solution. It will terminate in at most

$$2\left\lfloor \frac{\mathcal{C}(d)^2}{\beta^2} \right\rfloor \tag{80}$$

iterations of algorithm GVNA when (FP_d) does not have a solution. Each iteration of algorithm GVNA uses at most

$$T_1 + 3T_2 + 5m + n + 3mn$$

operations, where T_1 is the number of operations needed to solve an instance of (CSOP_c) and T_2 is the number of operations needed to compute a projection onto the cone C_Y.

Proof: The proof is an immediate consequence of the discussion above, and the results of Theorems 6.1 and 6.2. For (79), note that two times the bound in Theorem 6.1 is:

$$4\left(\left\lfloor \log_2\left(\mathcal{C}(d)\right) \right\rfloor \cdot \left\lfloor \frac{4\mathcal{C}(d)^2}{\beta^2} \right\rfloor + \max\left\{ \left\lfloor \frac{36\mathcal{C}(d)^2}{\beta^2} \right\rfloor, \left\lceil \frac{72\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{3\mathcal{C}(d)||b||}{\epsilon} \right) \right\rceil \right\} \right) \tag{81}$$

$$+\left\lceil \frac{32\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon}\right)\right\rceil\Bigg)$$

$$\leq 4\left(\log_2(\mathcal{C}(d)) \cdot \frac{4\mathcal{C}(d)^2}{\beta^2} + \max\left\{\frac{36\mathcal{C}(d)^2}{\beta^2}, \frac{72\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{3\mathcal{C}(d)||b||}{\epsilon}\right)\right\}\right.$$

$$\left. + \frac{32\mathcal{C}(d)^2}{\beta^2} \ln\left(\frac{6\mathcal{C}(d)||b||}{\epsilon}\right)\right) + 8$$

which in turn is bounded above by the simpler expression (79), where the bound of Theorem 6.1 is doubled due to the running of algorithms FCLS and ICLS in parallel. Likewise, (80) follows from doubling the bound in Theorem 6.2. ∎

The next section contains a discussion of several aspects of this theorem, including strengths and weaknesses, etc.

7 Discussion

Observe that algorithm CLS (as well as algorithms FCLS and ICLS) consists simply of repetitively calling algorithm GVNA on a sequence of data instances (M', g'), all with the same matrix operator $M' = [-b, A]$, but where the right-hand-side is of the form $g' = \delta b$ for a geometric sequence of values of the parameter δ. Viewed in this light, algorithm CLS is essentially no more than algorithm GVNA applied to a sequence of data instances all of very similar form. We regard the "inner" iterations of algorithm CLS as the iterations of algorithm GVNA, and the "outer" iterations of algorithm CLS then consist of the repeated application of algorithm GVNA to the sequence of data instances. The total workload of algorithm CLS, as presented in Theorem 6.3, is the total number of iterations of algorithm GVNA that are called in algorithm CLS. In this perspective, algorithm CLS is "elementary" in that the the mathematics of each inner iteration is not particularly sophisticated, only involving some matrix-vector multiplications, three Euclidean projections, and the solution of one conic section optimization problem (CSOP_c), see Remark 4.1.

Discussion of Complexity Bound. The iteration bounds on the number of iterations of algorithm GVNA in Theorem 6.3 depend only on four quantities, namely: the "coefficient of linearity" β for the cone C_X (which is independent of the problem data instance $d = (A, b)$), the condition number $\mathcal{C}(d)$ of the data instance $d = (A, b)$, the feasibility tolerance ϵ, and the norm $||b||$ of the right-hand-side vector b. As discussed below, the coefficient of linearity β can actually be ignored, as one can always choose the norm $||x||$ in such a way that $\beta = 1$ without affecting the mechanics of the algorithm. Also, the dependence of the bound (79) on the norm $||b||$ arises naturally so that a re-scaling of the problem data has the desired effect on the number of iterations needed to achieve a given feasibility tolerance. Therefore, the only critical components of the bounds of Theorem 6.3 are the condition number $\mathcal{C}(d)$ and the feasibility tolerance ϵ.

The iteration bounds in Theorem 6.3 are linear in $\ln(1/\epsilon)$ (see 79), which is a desirable feature. However, the bounds are polynomial in the condition number $\mathcal{C}(d)$ of the data instance $d = (A, b)$ (see (79) and 80)), and so are exponential in $\ln(\mathcal{C}(d))$. This exponential

factor is undesirable both aesthetically as well as practically, and in the language of computational efficiency (see Renegar [19]), algorithm CLS is inefficient. Indeed, to the extent that the bounds in Theorem 6.3 are attained, the algorithm's exponential sensitivity to $\ln(\mathcal{C}(d))$ would be troublesome. Of course, this is offset somewhat when the work per inner iteration is low, i.e, when T_1 and T_2 are not excessive. In this case, algorithm CLS trades off low work per inner iteration for a greater number of inner iterations, where the number of inner iterations could be exponential in $\ln(\mathcal{C}(d))$. We note that the iteration bounds in Theorem 6.3 do not depend at all on either n or m (just as in von Neumann's algorithm of Section 3).

It is useful to compare the bounds in Theorem 6.3 to those of an interior-point algorithm applied to problem (FP), see Renegar [20]. In an interior-point algorithm, the work per iteration is the work in solving for the Newton step, which is typically $O(n^3)$ operations. The bound on the number of iterations is, generally speaking, linear in $\ln(\mathcal{C}(d))$, which is efficient, and also is linear in $\sqrt{\vartheta_1 + \vartheta_2}$, where ϑ_1 and ϑ_2 are the barrier parameters for the cones C_X and C_Y. (For most applications, it is convenient to think of ϑ_1 as $O(n)$ and ϑ_2 as $O(m)$ in the worst case.) While an interior-point algorithm will generally have better worst-case complexity than that of algorithm CLS, the complexity bound of algorithm CLS will dominate an interior-point algorithm to the extent that the dimensions n and/or m are large, the condition number $\mathcal{C}(d)$ is small, and T_1 and T_2 are small.

For example, consider an instance of (FP) corresponding to the linear programming primal feasibility problem: $Ax = b, x \geq 0$. Here we have $T_1 = n$ and $T_2 = 1$, and the work per inner iteration of algorithm CLS is $O(n^2)$ as opposed to $O(n^3)$ operations for an interior point algorithm. Also, algorithm CLS can exploit sparsity in the data $d = (A, b)$ in a way that an interior-point algorithm cannot. The iteration bound of an interior-point algorithm is linear in $\sqrt{n}$. If n is sufficiently large and $\mathcal{C}(d)$ is sufficiently small, then algorithm CLS would have a better worst-case complexity bound than an interior-point algorithm.

As another example, consider an instance of (FP) corresponding to the semi-definite programming primal feasibility problem: $Ax = b, x \succeq 0$, where $n = k(k+1)/2$ and $x \in S^{k \times k}$. Here we have $T_1 = O(k^3)$ in practice (but not in theory) and $T_2 = 1$, and the work per inner iteration of algorithm CLS is $O(k^4)$ as opposed to $O(k^6)$ operations for an interior point algorithm. As above, algorithm CLS can exploit sparsity in the data $d = (A, b)$ in a way that an interior-point algorithm cannot. The iteration bound of an interior-point algorithm is linear in $(k)^{\frac{1}{4}}$. Here again, if k is sufficiently large and $\mathcal{C}(d)$ is sufficiently small, then algorithm CLS would have a better worst-case complexity bound than an interior-point algorithm.

Together, these two examples point out two possible ways that algorithm CLS might have some practical significance, in spite of its inferior exponential bound in terms of $\ln(\mathcal{C}(d))$.

The value of β is not consequential. As stated in (79) and (80), the complexity bound of Theorem 6.3 depends on the coefficient of linearity β defined in Definition 2.1. However, as is shown in [11], it is always possible to choose the norm $\|x\|$ on X in such a way that $\beta = 1$. In fact, if u' is any given interior point of the dual cone C_X^*, then it is possible to construct the norm $\|x\|$ in such a way that $\beta = 1$ and that the "best" linear operator $\bar{u}$ is $\bar{u} = u'$. As a consequence, Theorem 6.3 could actually be restated with no dependence on

β. (Of course, since the operator norm $\|A\|$ of the matrix A depends on the norm on X, the condition number $\mathcal{C}(d)$ would change with the changed norm.) We chose not to assume $\beta = 1$ to show the generality of the results, and to facilitate using results from [11] in the analysis.

Data Instances where C_X is not regular. One of the critical assumptions needed for the development of algorithm CLS is that the cone C_X is a regular cone. This assumption does not hold for important cases of (FP) such as $Ax \leq b$ (linear inequalities and $C_X = R^n$) and $Ax \preceq b$ (semi-definite inequalities and $C_X = R^n$). However, by instead working with the dual system (i.e., the "alternative system") for these cases ($(A^t y = 0, b^t y = -1, y \geq 0)$ or $(A^t y = 0, b^t y = -1, y \succeq 0)$), one can still utilize algorithm CLS to solve these systems by interchanging the roles of the primal and the dual conic linear systems, and interchanging the role of the cone C_X with the cone C_Y^*. This interchange works whenever the cone C_Y is a regular cone.

Improved Starting Values of Algorithm FCLS. Algorithm FCLS uses the explicit starting point of $(x^0)' = (x^0, \mu^0) = (0, 1)$ and the explicit starting value of δ of $\delta = 1$ at each call of algorithm GVNA. There may be better choices of these two starting values that would lead to lower iterations when a good starting point is known or guessed in advance and/or when a good estimate of the condition number $\mathcal{C}(d)$ is known or guessed in advance. For example, notice that algorithm FCLS runs algorithm GVNA on the same system (P_δ) during two consequent outer iterations. One might wish, therefore, to initialize the second run of GVNA on this system using the last iterate of the first run as a starting point. This may be the subject of future investigation.

Converting an Approximate Solution to an Exact Solution. In [6], Dantzig employs a clever trick to extend the von Neumann algorithm into an exact algorithm that computes an exact solution to (13). His method involves solving $n + 1$ suitably chosen instances of (13) and then taking an appropriate convex combination of the resulting solutions, see [6] for details. This approach presumably could be extended to algorithm CLS, with minor modifications, and would probably increase the complexity of algorithm CLS by a factor of $O(n \ln(n))$. We chose not to present or analyze this extension, as it would involve the use of another outer loop of iterations into algorithm CLS, which already consumes too many pages of analysis for most readers' tastes.

Converting Algorithm CLS into an Optimization Algorithm. Converting algorithm CLS into an optimization algorithm is a logical extension of the work presented in this paper. Suppose that we are interested in minimizing a linear function $c^t x$ over the feasible region of (FP). Then algorithm CLS could be modified with the addition of another outer loop that will add an objective function cut of the form $c^t x \leq c^t \bar{x}$ whenever an ϵ-solution $\bar{x}$ is produced at the previous iteration. This may be a topic of future research.

Complexity of Algorithm CLS when (FP) is ill-posed. The complexity bound of Theorem 6.3 relies on the hypothesis that (FP) is not ill-posed, i.e., that $\rho(d) > 0$. Algorithm CLS is not predicted to perform well in cases when $\rho(d) = 0$. We illustrate the possibility for extremely poor computational performance when $\rho(d) = 0$ with two simple examples. For the first example, let $X = R^2, Y = R^2, \|x\| = \|x\|_1$,

$$A = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \; C_X = R^2_+, \text{ and } C_Y = -R^2_+, \quad \textit{and} \quad \bar{u} = (1, 1)^t.$$

The system (FP) then is

$$\begin{gathered} 0 - 0x_1 - 0x_2 \leq 0 \\ 1 - 0x_1 - 0x_2 \leq 0 \\ x_1 \geq 0, x_2 \geq 0, \end{gathered}$$

and clearly (FP) does not have a solution. Note that by perturbing the coefficients of A arbitrarily small, it is possible to create an instance of (FP) that has a solution, and so $\rho(d) = 0$, i.e., (FP) is ill-posed. The system (P_0) is:

$$\begin{gathered} 0\mu - 0w_1 - 0w_2 \leq 0 \\ 1\mu - 0w_1 - 0w_2 \leq 0 \\ \mu \geq 0, w1 \geq 0, w_2 \geq 0 \\ \mu + w_1 + w_2 = 1, \end{gathered}$$

and (P_0) has a solution $(\mu, w_1, w_2) = (0, 0, 1)$. Since ($\mathrm{P}_0$) has a solution, algorithm ICLS will not be able to prove infeasibility of (FP). Therefore, the only possible output of algorithm CLS is an $(\epsilon\delta)$-solution of (P_δ) for some $\delta > 0$, if one exists. The system (P_δ) is

$$\begin{gathered} (\mathrm{P}_\delta) \quad 0\delta + 0\mu - 0w_1 - 0w_2 \leq 0 \\ 1\delta + 1\mu - 0w_1 - 0w_2 \leq 0 \\ \mu \geq 0, w1 \geq 0, w_2 \geq 0 \\ \mu + w_1 + w_2 = 1, \end{gathered}$$

and so if μ is part of an $(\epsilon\delta)$-solution of (P_δ) then $\mu \in [0, 1]$ and $\delta + \mu \leq \epsilon\delta$. Therefore, if $\epsilon < 1$, then no $(\epsilon\delta)$-solution of (P_δ) can exist for any value of $\delta > 0$. Therefore, algorithm CLS will not terminate if applied to this ill-posed data instance.

Next, consider the following example: let $X = R^2, Y = R^2, \|x\| = \|x\|_1$,

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}, \; b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \; C_X = R^2_+, C_Y = \{(0,0)^t\}, \quad \text{and} \quad \bar{u} = (1,1)^t.$$

Then system (FP) is the feasible system:

$$\begin{gathered} 1 - 2x_1 - 0x_2 = 0 \\ 0 - 0x_1 - 1x_2 = 0 \\ x_1 \geq 0, \; x_2 \geq 0. \end{gathered}$$

Note that for this data instance that $\rho(d) = 0$. (To see this, observe that by making the second component of the vector b negative but arbitrarily small, that the perturbed system (FP) has no solution, and this implies that $\rho(d) = 0$.) Problem (P_δ) for (FP) is:

$$\begin{gathered} 1\delta + 1\mu - 2x_1 - 0x_2 = 0 \\ 0\delta + 0\mu - 0x_1 - 1x_2 = 0 \\ x_1 \geq 0, \; x_2 \geq 0 \\ \mu + x_1 + x_2 = 1. \end{gathered}$$

It is easy to see that the largest value of δ for which (P_δ) has a feasible solution is $\delta = 2$, and therefore $\bar{\delta} = 2$. Let us now examine the performance of algorithm CLS applied to (FP). Algorithm ICLS will not terminate with a demonstration of infeasibility, since (FP) has a solution. Algorithm FCLS will start with $\delta' = 1$ and $\delta'' = \frac{1}{2}$, which both yield feasible

instances of (P_δ). From Lemma 4.2, the bound on the number of iterations to achieve an ϵ-solution of (P_δ) (for $\delta = 1$ or $\delta = \frac{1}{2}$) will be

$$O\left(\frac{\|d\|}{\epsilon^2}\right)$$

iterations, which is exponential in $\ln(1/\epsilon)$. Furthermore, the geometry of the problem indicates that this bound would probably, and unfortunately, be achieved.

These two examples illustrate the possibilities for poor computational complexity when the problem (FP) is ill-posed, i.e., when $\rho(d) = 0$.

References

[1] S. Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6:382–392, 1954.

[2] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.

[3] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming, Theory and Algorithms*. John Wiley & Sons, Inc, New York, second edition, 1993.

[4] R. Bland, D. Goldfarb, and M.J. Todd. The ellipsoid method: a survey. *Operations Research*, 29:1039–1091, 1981.

[5] C. Chen and O.L. Mangasarian. Smoothing methods for convex inequalities and linear complementarity problems. *Mathematical Programming*, 71(1):51–70, 1995.

[6] G.B. Dantzig. Converting a converging algorithm into a polynomially bounded algorithm. Technical Report SOL 91-5, Stanford University, 1991.

[7] G.B. Dantzig. An ϵ-precise feasible solution to a linear program with a convexity constraint in $1/\epsilon^2$ iterations independent of problem size. Technical Report, Stanford University, 1992.

[8] B.C. Eaves. Piecewise linear retractions by reflection. *Linear Algebra and its Applications*, 7:93–98, 1973.

[9] Sharon Filipowski. On the complexity of solving linear programs specified with approximate data and known to be feasible. Technical Report, Dept. of Industrial and Manufacturing Systems Engineering, Iowa State University, May 1994.

[10] Sharon Filipowski. On the complexity of solving sparse symmetric linear programs specified with approximate data. Technical Report, Dept. of Industrial and Manufacturing Systems Engineering, Iowa State University, December 1994.

[11] Robert M. Freund and Jorge R. Vera. Some characterizations and properties of the "distance to ill-posedness" in conic linear systems. Technical Report W.P. #3862-95-MSA, Sloan School of Management, Massachusetts Institute of Technology, 1995.

[12] Robert M. Freund and Jorge R. Vera. Condition complexity of the ellipsoid algorithm for solving a conic linear system. Technical report, 1997. In preparation.

[13] J.L. Goffin. The relaxation method for solving systems of linear inequalities. *Mathematics of Operations Research*, 5(3):388–414, 1980.

[14] A.S. Lewis. Group invariance and convex matrix analysis. Technical Report, Dept. of Combinatorics and Optimization, University of Waterloo, Waterloo, Canada, February 1995.

[15] T.S. Motzkin and I.J. Schoenberg. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6:393–404, 1954.

[16] Yurii Nesterov and Arkadii Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1994.

[17] Manuel A. Nunez and Robert M. Freund. Condition measures and properties of the central trajectory of a linear program. Technical Report W.P. #3889-96-MSA, Sloan School of Management, Massachusetts Institute of Technology, 1996.

[18] James Renegar. Some perturbation theory for linear programming. *Mathematical Programming*, 65(1):73–91, 1994.

[19] James Renegar. Incorporating condition measures into the complexity theory of linear programming. *SIAM Journal on Optimization*, 5(3):506–524, 1995.

[20] James Renegar. Linear programming, complexity theory, and elementary functional analysis. *Mathematical Programming*, 70(3):279–351, 1995.

[21] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[22] F. Rosenblatt. On the convergence of reinforcement procedures in simple perceptrons. Report, VG-1196-G-4, Cornell Aeronautical Laboratory, Buffalo, NY, 1960.

[23] F. Rosenblatt. Perceptron simulation experiments. *Proceedings of the Institute of Radio Engineers*, 48:301–309, 1960.

[24] F. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, Washington, DC, 1962.

[25] Jorge R. Vera. Ill-posedness and the computation of solutions to linear programs with approximate data. Technical Report, Cornell University, May 1992.

[26] Jorge R. Vera. *Ill-Posedness in Mathematical Programming and Problem Solving with Approximate Data*. PhD thesis, Cornell University, 1992.

[27] Jorge R. Vera. Ill-posedness and the complexity of deciding existence of solutions to linear programs. Technical Report, Cornell University, June 1994. To appear in *SIAM Journal on Optimization*.

[28] Jorge R. Vera. On the complexity of linear programming under finite precision arithmetic. Technical Report, Dept. of Industrial Engineering, University of Chile, August 1994.