Polynomial interior point cutting plane methods

John E. Mitchell[1]
Department of Mathematical Sciences,
Rensselaer Polytechnic Institute,
Troy, NY 12180.
mitchj@rpi.edu
http://www.rpi.edu/~mitchj

July 16, 2001
Revised: April 1, 2003

Abstract

Polynomial cutting plane methods based on the logarithmic barrier function and on the volumetric center are surveyed. These algorithms construct a linear programming relaxation of the feasible region, find an appropriate approximate center of the region, and call a separation oracle at this approximate center to determine whether additional constraints should be added to the relaxation. Typically, these cutting plane methods can be developed so as to exhibit polynomial convergence. The volumetric cutting plane algorithm achieves the theoretical minimum number of calls to a separation oracle. Long-step versions of the algorithms for solving convex optimization problems are presented.

1 Introduction

Let $\mathcal{C}$ be a convex subset of $\mathbb{R}^m$. Cutting plane methods can be used to solve convex feasibility problems of the form:

Find $y \in \mathcal{C} \subseteq \mathbb{R}^m$.

Any convex set can be represented as the intersection of a (possibly infinite) collection of halfspaces. Therefore, the convex feasibility problem is equivalent to the (possibly semi-infinite) linear programming problem

Find y satisfying $\hat{A}^T y \leq \hat{c}$

where $\hat{A}$ is a $m \times \hat{n}$ matrix and $\hat{c}$ is an $\hat{n}$-vector, and $\hat{n}$ may be infinite.

We assume we have available a **separation oracle**. Given a point $\bar{y} \in \mathbb{R}^m$, such an oracle will either confirm that $\bar{y} \in \mathcal{C}$, or it will return an m-vector a and a scalar c_0 such that the constraint $a^T y \leq c_0$ is satisfied by all $y \in \mathcal{C}$ but $a^T \bar{y} > c_0$. A typical cutting plane algorithm can then be described as follows:

[1]Research partially supported by NSF Grant number CCR–9901822.

1. Choose $\bar{y} \in I\!R^m$.

2. Present $\bar{y}$ to the oracle.

3. If $\bar{y}$ is in $\mathcal{C}$, STOP: we have solved the convex feasibility problem.

4. Otherwise, use the constraint returned by the separation oracle to modify the choice $\bar{y}$ and return to Step 2.

Of course, it is necessary to describe Step 4 in more detail to really define an algorithm. The principal objective of this paper is to describe algorithms that use the logarithmic potential function or the volumetric barrier function to choose the iterate $\bar{y}$. The methods will set up linear programming approximations to $\mathcal{C}$ of the form

$$\mathcal{C} \subseteq \{y : A^T y \leq c\}$$

where A is an $m \times n$ matrix and c is an n-vector. This approximation will be refined at each iteration by the addition of the cutting plane returned by the oracle; earlier cuts may also be dropped. The methods will then use an interior point method to find the next trial point $\bar{y}$. Such an algorithm will be said to have polynomial complexity if the complexity is polynomial under the assumption that each call to the oracle requires unit time.

In order to discuss the number of iterations required by these algorithms, we need to refine what we mean by a solution. For example, assume $\mathcal{C}$ is contained in the hyperplane $p^T y = q$, where p is an m-vector and q is a scalar. For worst case analysis, we always assume that the oracle returns as unhelpful a cutting plane as possible. Thus, if $p^T \bar{y} = q + \alpha$ for some $\alpha > 0$, the oracle could return the constraint $p^T \bar{y} \leq q + \frac{\alpha}{2}$. It can be seen that no algorithm will then be able to guarantee finding a trial point $\bar{y}$ satisfying $p^T \bar{y} = q$ in a finite number of iterations.

Therefore, we assume that if $\mathcal{C}$ is nonempty then it contains a ball of radius ϵ for some tolerance $\epsilon > 0$. (An alternative remedy would be to redefine the oracle so that it checks whether the trial point is within ϵ of some point in $\mathcal{C}$. This is equivalent to moving all the boundaries out by ϵ. These issues are discussed in more detail in Grötschel *et al.* [15].) Further, we assume that such a ball is contained in the m-dimensional hypercube of half-width $\frac{1}{\epsilon}$ given by $\{y \in I\!R^m : |y_i| \leq \frac{1}{\epsilon}\}$. We define

$$L := \log_2 \left(\frac{1}{\epsilon} \right), \tag{1}$$

the amount of storage required to store ϵ. In terms of L, the hypercube can be written as $\{y \in I\!R^m : |y_i| \leq 2^L\}$.

Cutting plane methods based on the logarithmic potential function and on the volumetric center can be constructed so as to require a number of iterations that is bounded by a polynomial in m and L, so they exhibit polynomial convergence. Note that, implicitly, the number of calls to the separation oracle must also be polynomial in m and L. The analysis used to construct such a bound typically has the flavour of a proof-by-contradiction: if more iterations are taken then it must be the case that $\mathcal{C}$ is actually empty, in which case the algorithm can stop.

The ellipsoid algorithm can also be used to solve the convex feasibility problem in polynomial time. So far, the analytic center cutting plane method (ACCPM) of Goffin and Vial [10, 11, 12] has only been shown to be fully polynomial, that is, polynomial in m and ϵ.

The polynomial potential reduction cutting plane algorithm of Atkinson and Vaidya [7] is the subject of §2. At each iteration, their algorithm maintains a polytope that is guaranteed to contain $\mathcal{C}$, and uses the analytic center of this polytope as the trial point. If this point is not contained in the convex set, the oracle returns a hyperplane that separates the trial point and the convex set. This hyperplane is used to update the polytope. Further, hyperplanes currently in the polytope that are deemed 'unimportant' according to some criteria are dropped. Dropping constraints is crucial for the complexity analysis of the algorithm, and is the major difference between this algorithm and the ACCPM.

Vaidya [35] proposed a cutting plane algorithm that uses an approximation to the volumetric center as the trial point. Anstreicher [2] has examined this algorithm and, in [5], he refined some of the analysis in order to improve some of the constants in the complexity bound of the algorithm. Volumetric cutting plane algorithms are the subject of §3.

Consider now the **optimization** problem $\max\{b^T y : y \in \mathcal{C}\}$, where $b \in I\!R^m$. These feasibility algorithms can be used to solve this optimization problem. Much like the ellipsoid algorithm (see [15]), they can be applied to linear programming problems by cutting on a violated constraint when the trial point is infeasible, and cutting on the objective function when the trial point is feasible but not optimal. This would then resemble Renegar's algorithm [34] for linear programming. However, this is somewhat like a 'short step' method — when cutting on the objective function, the cut is placed such that the next iterate is in a small ellipsoid around the current point. This suggests that the progress in objective function value cannot be too much.

In §4, we discuss long-step variants of the algorithms described in §2 and §3, due to Mitchell and Ramaswamy [28, 33]. These methods incorporate the objective function

into a barrier function explicitly, motivated in part by the successful implementation of such algorithms for linear programming. Long step linear programming algorithms are not only polynomial in complexity, but can also be designed to exhibit superlinear [37] or quadratic [36] convergence asymptotically.

Good computational results have been obtained with interior point cutting plane methods. They have been used to solve stochastic programming problems [8], multi-commodity network flow problems [9], and integer programming problems [25], as well as other forms of convex optimization problems [14, 13]. For some classes of linear ordering problems, a cutting plane scheme that combines an interior point method and a simplex method has been shown to be up to ten times faster than one that just uses either of the methods individually [26]. For some max cut problems, an interior point cutting plane method has been shown to outperform considerably a simplex cutting plane method [25]. For more information on the use of interior point cutting plane methods for integer programming problems, see §5.

1.1 Notation

We let e denote a vector of ones of the appropriate dimension. If s denotes an n-vector, then S denotes a diagonal $n \times n$ matrix, with $S_{ii} = s_i$. Given a matrix A, we let a_j denote the jth column of A. The determinant of a square matrix M is denoted by $\det(M)$. The natural logarithm of a positive number x is denoted $\ln(x)$ and the logarithm in base 2 of x is denoted $\log(x)$.

We define an ellipsoid $E(M, z, r)$ for any positive definite $m \times m$ matrix M, any m-vector z, and any scalar r as

$$E(M, z, r) := \{y \in I\!R^m : (y - z)^T M(y - z) \leq r^2\}. \tag{2}$$

If the symmetric matrix M is positive semidefinite then this is represented as either $M \succeq 0$ or $0 \preceq M$. Given two symmetric matrices M and R, where $R - M$ is a positive semidefinite matrix, we write $R \succeq M$ or $M \preceq R$.

2 The Atkinson-Vaidya logarithmic potential function algorithm

Analytic center cutting plane algorithms construct a relaxation

$$Q := \{y : A^T y \leq c\} \supseteq \mathcal{C}, \tag{3}$$

where A is an $m \times n$ matrix and c is an n-vector, and then find an approximation to the analytic center of Q. We assume without loss of generality that the columns a_j of A all have norm equal to one. The analytic center is defined as follows. Let

$$s := c - A^T y \tag{4}$$

and define

$$\varphi_D(s) := -\sum_{i=1}^{n} \log(s_i). \tag{5}$$

The **analytic center** of Q is the point $\tilde{y}$ that minimizes the logarithmic barrier function

$$F(y) := \varphi_D(c - A^T y). \tag{6}$$

It is easily seen that the gradient of the barrier function is given by

$$\nabla F(y) = AS^{-1}e, \tag{7}$$

and the Hessian by

$$\nabla^2 F(y) = AS^{-2}A^T. \tag{8}$$

Since the analytic center $\tilde{y}$ is the unconstrained minimizer of this function F, it follows that $\nabla F(\tilde{y}) = 0$. We call y an **approximate analytic center** if

$$\Psi(y) := \nabla F(y)^T (\nabla^2 F(y))^{-1} \nabla F(y) \leq \frac{1}{16}, \tag{9}$$

that is, if the gradient of the barrier function is small in the norm defined by the Hessian of the barrier function.

As shown by Nesterov and Nemirovskii [32], if y is an approximate analytic center then taking Newton steps will give quadratic convergence to the analytic center. Further, if $\Psi(y) \leq 0.25$ then an approximate analytic center can be recovered in one Newton step. When taking a Newton step, y is updated to

$$y \leftarrow y - (\nabla^2 F(y))^{-1} \nabla F(y). \tag{10}$$

Global convergence analysis requires examination of the value of both the logarithmic barrier function $F(y)$ and its Hessian $\nabla^2 F(y)$. If the value of the logarithmic barrier function at the analytic center becomes large, the polyhedron has become narrow, that is, one of the slack variables has become small. If it is too narrow to contain a ball of radius ϵ then we can conclude that $\mathcal{C}$ is empty. The value of $\log(\det(\nabla^2 F(y)))$ at the analytic center is related to the volume of the polyhedron, and this is used explicitly in volumetric barrier algorithms. As this value grows, the

volume of the polyhedron is shrinking. It follows that if $\log(\det(\nabla^2 F(y)))$ becomes sufficiently large, we can conclude that $\mathcal{C}$ is empty. We discuss this further in §2.5.

We also define **variational quantities**:

$$\sigma_j := \frac{a_j^T (\nabla^2 F(y))^{-1} a_j}{s_j^2} \tag{11}$$

for j from 1 to n. These variational quantities give an indication of the relative importance of the inequality $a_j^T y \leq c_j$ — the larger the value of σ_j, the more important the inequality. Note that the variational quantities are scale invariant for each constraint $a_j^T y \leq c_j$. In particular, if this constraint is replaced by the equivalent constraint $k a_j^T y \leq k c_j$ for some positive scalar k, then both a_j and s_j are rescaled by the same amount; further, the contribution of the constraint to the Hessian matrix $AS^{-2}A^T$ is unchanged by this rescaling.

The variational quantities σ are equal to the diagonal of the projection matrix

$$\bar{P}(y) := S^{-1} A^T (AS^{-2}A^T)^{-1} AS^{-1}. \tag{12}$$

Given a vector $v \in I\!R^n$, the product $\bar{P}(y)v$ is the projection of v onto the range of $S^{-1}A^T$. Since $\bar{P}(y)$ is a projection matrix, it follows immediately that $0 \leq \sigma_i \leq 1$ for each component $i = 1, \ldots, m$. Further,

$$\begin{aligned} e^T \sigma &= \text{trace}(\bar{P}(y)) \\ &= \text{trace}((AS^{-2}A^T)^{-1} AS^{-2}A^T) \quad \text{from properties of trace} \\ &= m. \end{aligned}$$

2.1 An example of the variational quantities

The principal algorithmic difference between the algorithm of Atkinson and Vaidya [7] and the ACCPM of Goffin and Vial [11, 12] is that the former drops constraints that become unimportant. The variational quantities are used as one indicator for deciding whether to drop a constraint. A constraint may be dropped if its variational quantity falls below a certain threshold.

Let $Q = \{y \in I\!R^2 : -1 \leq y_1 \leq 1, -k \leq k y_2 \leq k \text{ for } k = 1, \ldots, 100\}$. Geometrically, this is just the box with the simple bounds that each y_i must be no larger than 1 in absolute value. Algebraically, there are 100 copies of the bounds on y_2. The analytic center of Q is $\tilde{y} = 0$. The gradient of the barrier function is zero at $\tilde{y}$. The Hessian at this point is

$$(AS^{-2}A^T)^{-1} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{200} \end{bmatrix}.$$

In this example, we would expect that the constraints corresponding to the bounds on y_2 are less important than those corresponding to bounds on y_1, and this is indeed reflected in the values of the variational inequalities. In particular, the variational quantities σ_j are equal to $\frac{1}{2}$ for the constraints corresponding to bounds on y_1 and they are equal to $\frac{1}{200}$ for the constraints corresponding to bounds on y_2.

Now consider the polytope Q with two additional constraints, namely $-\delta \leq y_1 \leq \delta$ for some positive scalar δ. In this case, we obtain the variational quantities

$$\sigma_j = \begin{cases} \frac{\delta^2}{2(1+\delta^2)} & \text{for constraints } -1 \leq y_1 \leq 1 \\ \frac{1}{2(1+\delta^2)} & \text{for constraints } -\delta \leq y_1 \leq \delta \\ \frac{1}{200} & \text{for constraints on } y_2. \end{cases}$$

In this situation, we can see that the weaker bound constraint on y_1 has a smaller variational quantity than the stronger bound.

2.2 The importance of the variational quantities

The variational quantities σ are used to decide whether to drop a constraint: if the value is very small then the constraint is a candidate to be dropped. See §2.3 for details. In this subsection, we attempt to give some intuition for why this is a good criterion.

Given the current strictly feasible iterate $\bar{y}$ with slack values s, the system of equations $A^T y \leq c$ can be stated equivalently as $S^{-1}A^T y \leq S^{-1}c$. Every constraint then has slack equal to one at $\bar{y}$.

The ellipsoid $E(AS^{-2}A^T, \bar{y}, 1)$ is contained in the feasible region Q. See Lemma 1 and §3 for more discussion of this ellipsoid. Assuming the ellipsoid provides a reasonable approximation to Q, if a constraint is important, there should be a point in the inscribing ellipsoid that comes close to violating the constraint. To find the point in the ellipsoid that comes closest to violating the constraint, we can maximize $a_i^T z$ over points in the ellipsoid. Equivalently, we can rescale and translate, and then maximize $\frac{1}{s_i}a_i^T(z - \bar{y})$. This gives the following quadratic optimization problem:

$$\begin{array}{ll} \max_z & \frac{1}{s_i}a_i^T(z - \bar{y}) \\ \text{s.t.} & z \in E(AS^{-2}A^T, \bar{y}, 1). \end{array}$$

The optimal value of this problem gives the proportion by which the slack can be reduced while still remaining in the ellipsoid. Solving the KKT conditions for this problem shows that the optimal solution is $z = \bar{y} + (AS^{-2}A^T)^{-1}(s_i a_i)$, with value equal to σ_i. The maximum possible value is one, when the ellipsoid touches the ith

constraint. Thus, small values of σ_i correspond to constraints i which are far from the inscribing ellipsoid $E(AS^{-2}A^T, \bar{y}, 1)$.

Therefore, a constraint that is dropped is one that is easily satisfied by every point in the inscribing ellipsoid $E(AS^{-2}A^T, \bar{y}, 1)$.

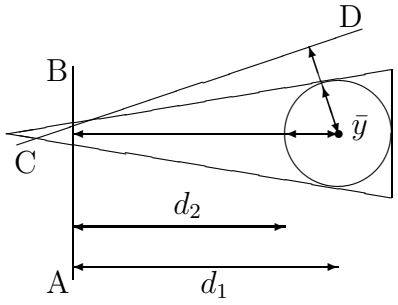

Figure 1: Dropping constraints. The variational quantity σ for the constraint given by the line AB is equal to $1 - \frac{d_2}{d_1}$.

The variational quantities σ are illustrated in Figure 1. The inscribing ellipsoid $E(AS^{-2}A^T, \bar{y}, 1)$ is shown. For this example, it is drawn as a sphere. The ellipsoid touches three constraints, so for these constraints the value of σ_i is one. Note that in general the ellipsoid might not touch any of the constraints. For the constraint illustrated by the line AB, the value of σ_i is approximately equal to 0.2. This is calculated by comparing the distance d_1 from AB to $\bar{y}$ and the distance d_2 from AB to the closest point in the ellipsoid to AB. The distance d_2 is about 80% of the distance d_1, so $\sigma_i \approx 1 - 0.8 = 0.2$. Because the ellipsoid is drawn as a sphere, the two lines giving d_1 and d_2 are collinear; in general, they need not be so. The value of σ_i for the constraint illustrated by the line CD can be found in a similar manner, leading to a value of approximately 0.5.

2.3 The algorithm

At each iteration, we have a polytope $Q = \{y : A^T y \leq c\}$. The algorithm finds an approximate analytic center $\tilde{y}$ of Q. It first determines whether it is appropriate to drop a constraint. If not, it submits the approximate analytic center to the oracle, which either confirms that $\tilde{y} \in \mathcal{C}$ or returns a hyperplane separating $\tilde{y}$ from $\mathcal{C}$. In the latter case, the hyperplane is added to the polyhedron, perhaps after weakening it slightly. A new approximate analytic center is then determined and the process is repeated. If the algorithm dropped a constraint, again a new analytic center would be determined and the process repeated. The algorithm is detailed in Figure 2.

1. **Initialize:**
   Set $n = 2m$, $A = [I,\ -I]$, and $c = \frac{1}{\epsilon}e$. Set $y = 0$ and $s = c$. Set $\kappa_i = \frac{1}{\epsilon}$ for $i = 1, \ldots, n$. Initialize the iteration counter $k = 0$. Go to Step 5.

2. **Check to drop constraints:**
   Let $\Lambda := \max\{\frac{s_i}{\kappa_i} : i = 2m+1, \ldots, n\}$. If $\Lambda \leq 2$, go to Step 5. Otherwise, if there is an index $\bar{i}$ with $\frac{s_{\bar{i}}}{\kappa_{\bar{i}}} > 2$ and $\sigma_{\bar{i}} < 0.04$, go to Step 3, else let $\bar{i}$ be any index with $\frac{s_i}{\kappa_i} > 2$ and go to Step 4.

3. **Drop constraint:**
   Drop the $\bar{i}$th row from $[A^T, c]$. Update $n \leftarrow n - 1$. Go to Step 7.

4. **Update κ:**
   Set $\kappa_{\bar{i}} = s_{\bar{i}}$. Go to Step 2.

5. **Call oracle:**
   Call the oracle with y as the trial point. If the oracle does not return a violated constraint, **STOP** with feasibility.

6. **Update relaxation:**
   Let a and c_0 denote the constraint returned by the oracle. Set $\beta = a^T y$. Add the constraint $a^T y \leq \beta$ to $[A^T, c]$ and update $n \leftarrow n + 1$. If $n \geq 2700mL$, **STOP**, with the conclusion that $\mathcal{C}$ is empty.

7. **Update y:**
   Find a new approximate analytic center y in O(1) Newton steps. Update $s = c - A^T y$. Set $\kappa_n = c_n - a^T y$ if we came here from Step 6. If $s_i \leq \frac{\epsilon}{5400mL}$ for some i, **STOP**, with the conclusion that $\mathcal{C}$ is empty. Otherwise, return to Step 2.

Figure 2: A polynomial potential reduction cutting plane algorithm

We have already discussed the termination tolerance ϵ. The algorithm initializes with Q equal to the box defined by the $2m$ simple bounds $-\frac{1}{\epsilon} \leq y_i \leq \frac{1}{\epsilon}$. These constraints are never dropped in the course of the algorithm, and they always correspond to indices $1, \ldots, 2m$. The analytic center of this box is the origin, of course.

The decision to drop a constraint i is based on two values: κ_i and σ_i. If σ_i is small then the constraint is far from an inscribed ellipsoid centered at the current iterate. The κ values are initialized to be the slacks of the corresponding constraints. If the slack doubles then the constraint becomes a candidate to be dropped. A κ value can be reset to the current slack if it is decided to not drop a constraint based on examination of the values of σ; this slack then has to double again before the constraint can once more become a candidate to be dropped.

The κ values of the box constraints are set equal to the initial slack values and never reset. Note that the slack variables for these constraints cannot increase by more than a factor of two over the course of the algorithm, so the ratio $\frac{s_i}{\kappa_i}$ is always smaller than 2.

When a constraint is added, it is weakened so that the current iterate y satisfies it exactly. In their original presentation, Atkinson and Vaidya [7] weakened the added constraint further, setting $\beta = a^T y + 4\sqrt{a^T (AS^{-2}A^T)^{-1} a}$; note that this sets the variational quantity for this added constraint equal to 1/16. With this weakening, a new approximate analytic center can be recovered in one step. It follows from the work of Goffin , Luo, and Ye [10] and Mitchell and Todd [29] that the constraint can be added right through the current iterate and a new approximate analytic center found in O(1) Newton steps.

We show in §2.4 that the new analytic center can be recovered in one Newton step when a constraint is dropped. The termination criteria used in steps 6 and 7 are justified in §2.5, where we show that these criteria result in a polynomial time algorithm, under the assumption that the oracle requires unit time. We sketch the principal ideas of the proofs of local and global convergence; details can be found in [7].

2.4 Local convergence analysis

In this section, we show that a new approximate analytic center can be obtained quickly when a constraint is added or dropped. These are the inner iterations of the algorithm. A bound on the number of outer iterations will then lead to a proof of convergence of the algorithm.

2.4.1 Adding a constraint

Recovering a new analytic approximate center after the addition of a cut through the current approximate center in O(1) steps is discussed in the recent survey paper by Goffin and Vial [12]. Goffin and Vial also show that it is possible to add p constraints simultaneously, and recover a new approximate analytic center in $O(p\log(p))$ Newton steps.

2.4.2 Dropping a constraint

A constraint $\bar{i}$ is dropped only if its slack has doubled since $\kappa_{\bar{i}}$ was last reset and its variational quantity is small. The iterate y was an approximate analytic center before the constraint was dropped. Because the constraint was not very important at y, it also wasn't very important at the analytic center. Therefore, the new analytic center is close to the old one, and it can be shown that, after dropping the constraint, we have $\Psi(y) \leq 0.25$, so one Newton step gives a new approximate analytic center. For details, see [7].

2.5 Global convergence analysis

The logarithmic barrier function $F(y)$ is used to prove global convergence of the algorithm. This proof uses both volume and width arguments. In order to simplify the presentation, we assume that the algorithm works with exact analytic centers. The results presented can be extended to show convergence when approximate analytic centers are calculated instead.

2.5.1 Volume arguments

The volume arguments require relating the value of the Hessian of the potential function to the size of certain ellipsoids. The function $F(y)$ is strictly convex, so its Hessian matrix $\nabla^2 F(y)$ is positive semidefinite. Therefore, we can use it to define ellipsoids — such ellipsoids are called *Hessian ellipsoids*. Now, at the analytic center $\tilde{y}$, the gradient of $F(\tilde{y})$ is zero. It will follow that if y is in a small Hessian ellipsoid centered at the analytic center, then $F(y)$ will be close to its minimum value $F(\tilde{y})$. The following can be shown:

Lemma 1 *The polytope Q satisfies*

$$E(\nabla^2 F(\tilde{y}), \tilde{y}, 1) \subseteq Q \subseteq E(\nabla^2 F(\tilde{y}), \tilde{y}, n),$$

where n is the number of constraints in the description of Q.

Thus, we can obtain inscribing and circumscribing ellipsoids for Q.

Now, the volume of the ellipsoid $E(M, z, r)$ is

$$\text{vol}(E(M, z, r)) = \frac{r^m \text{vol}(E(I, 0, 1))}{\sqrt{\det(M)}}, \tag{13}$$

where $E(I, 0, 1)$ denotes the unit ball centered at the origin. It follows that as the determinant of the logarithmic barrier increases, the volume of Q decreases, if we keep the number of constraints approximately constant. Atkinson and Vaidya [7] prove the following crucial theorem, showing that that the determinant of $\nabla^2 F(\tilde{y})$ grows exponentially in the number of constraints, n. The proof of the theorem requires that unimportant constraints, as measured by the slack values s and the variational quantities σ, be dropped.

Theorem 1 *If $\frac{s_i}{\kappa_i} \leq 2$ for all the constraints, then*

$$\det \nabla^2 F(\tilde{y}) > 2^{-m} \epsilon^{2m} (1.01)^{n-2m}.$$

It follows as a straightforward corollary to this theorem that the number of constraints n defining Q must be bounded, or otherwise the volume of $\mathcal{C}$ would be smaller than a ball of radius ϵ. The proof requires combining the result of the theorem with (13).

Corollary 1 *If the algorithm adds sufficiently many constraints that $n \geq 2700mL$ then $\mathcal{C}$ is empty.*

This corollary justifies the termination criterion in Step 6 of the algorithm.

2.5.2 Width arguments

The width arguments require showing that eventually the polytope Q becomes too narrow to contain a ball of radius ϵ, even if the number of constraints remains bounded by $2700mL$. This is shown by demonstrating that if the value of the logarithmic potential function at the analytic center becomes large then a slack variable has become small, indicating that the width of Q has also become small. If a slack variable becomes sufficiently small then the feasible region for the current relaxation is no longer large enough to contain a ball of radius ϵ, so $\mathcal{C}$ must be empty.

Other things being equal, as the slack variables become smaller, the logarithmic potential function becomes larger. As we will see, the cutting plane algorithm in Figure 2 causes the value of the logarithmic barrier function to increase each time Step 7 is completed.

Thus, the width argument requires three components:

1. The value of the logarithmic barrier function at the approximate analytic center increases as the algorithm proceeds.

2. An increase in the value of the barrier function correlates with a decrease in the slack of some variable.

3. The number of constraints is bounded; this follows from the volume arguments in §2.5.1.

Define width(a_i) to be the variation in $a_i^T y$ over the current polytope. We have the following lemma.

Lemma 2 *The width of Q in the direction a_i is bounded by width$(a_i) \leq n(c_i - a_i^T \tilde{y})$ for $i = 1, \ldots, n$.*

Proof: The gradient of the logarithmic barrier function is zero at the analytic center $\tilde{y}$, so we have

$$0 = \nabla F(\tilde{y}) = -\sum_{i=1}^{n} \frac{a_i}{c_i - a_i^T \tilde{y}}.$$

Taking the inner product of this with $\tilde{y} - x$ and rearranging gives

$$\begin{aligned} n &= \sum_{i=1}^{n} \frac{a_i^T(\tilde{y} - x) + c_i - a_i^T \tilde{y}}{c_i - a_i^T \tilde{y}} \\ &= \sum_{i=1}^{n} \frac{c_i - a_i^T x}{c_i - a_i^T \tilde{y}} \end{aligned}$$

which holds for all x. Each term in the summand is nonnegative for any feasible point x. Choose x to be the feasible point that minimizes $a_i^T x$, so width$(a_i) = c_i - a_i^T x$. We obtain

$$\frac{\text{width}(a_i)}{c_i - a_i^T \tilde{y}} \leq n.$$

Rearranging gives the required result. □

Note that if $F(\tilde{y}) \geq n(L + \log(n))$ then we must have $c_i - a_i^T \tilde{y} \leq \frac{\epsilon}{n}$ for some constraint i. It follows from this lemma that we have width$(a_i) \leq \epsilon$ if the logarithmic

barrier function is sufficiently large. Therefore, $\mathcal{C}$ cannot contain a ball of radius ϵ, so it must be empty.

It remains to be shown that $F(\tilde{y})$ grows as the algorithm proceeds. This requires consideration of a normalized version of $F(y)$. Thus, given a point y and the corresponding slacks $s = c - A^T y$, we define

$$G(y) := -\sum_{i=1}^{n} \log(\frac{s_i}{\kappa_i}) = -\sum_{i=1}^{n} \log(s_i) + \sum_{i=1}^{n} \log(\kappa_i) = F(y) + \sum_{i=1}^{n} \log(\kappa_i). \tag{14}$$

Now, κ_i can be no larger than the largest possible value of the ith slack term. Because $||a_i||_2 = 1$, the slack can be no larger than the diagonal distance across the cube, so

$$\kappa_i \leq \frac{2\sqrt{m}}{\epsilon}. \tag{15}$$

Thus,

$$F(y) = G(y) - \sum_{i=1}^{n} \log(\kappa_i) \geq G(y) - n(1 + 0.5\log(m) - \log(\epsilon)). \tag{16}$$

The volume arguments show that n is bounded, so it suffices to show that $G(y)$ increases sufficiently. There are three cases to consider:

- Step 3 of the algorithm, a constraint is dropped:
  Dropping a constraint causes the analytic center to change. Because the ratio $\frac{s_i}{\kappa_i} > 2$, the value of G at the old analytic center increases by at least $\log(2)$. It can be shown that the new analytic center $\tilde{y}^{\text{new}}$ is sufficiently close to the old analytic center $\tilde{y}^{\text{old}}$ that $G(\tilde{y}^{\text{new}}) \geq G(\tilde{y}^{\text{old}}) + 0.65$, because σ_i is sufficiently small.

- Step 4 of the algorithm, one of the κ_i is increased:
  In this case, κ_i is set equal to the value of s_i, so κ_i is at least doubled. The analytic center is unchanged, so $G(\tilde{y})$ increases by at least $\log(2)$.

- Step 6 of the algorithm, a constraint is added:
  The analytic center changes. The new analytic center lies outside a sufficiently large ellipsoid centered at the old analytic center so that it can be shown that there is a guaranteed increase in the the old $G(y)$ (that is, $G(y)$ calculated without the new constraint). Note that we chose κ_i for the new constraint so that it contributes zero to the new normalized function $G(\tilde{y})$. Thus, even after adding in the term for the new constraint, we still have $G(\tilde{y}^{new}) \geq G(\tilde{y}^{old}) + 0.01$.

Initially, the only constraints are the box constraints, and the origin is the analytic center. The κ_i are chosen so that $G(\tilde{y}) = 0$ for this initial step. Since $G(\tilde{y})$ increases

by at least γ for some fixed constant $\gamma > 0$ at each iteration, we have that $G(\tilde{y}) \geq \gamma p$ after p iterations.

2.6 Conclusions and future research

It follows from (16) and from the boundedness of n that after p iterations we have $F(\tilde{y}) \geq \gamma p - \psi$ for a constant ψ. Eventually, $F(\tilde{y})$ will be sufficiently large that some s_i will be small enough to indicate that the polytope is too flat to contain a ball of radius ϵ. The following theorem makes this more precise:

Theorem 2 *The algorithm will stop in $O(mL^2)$ iterations, either because a feasible point has been found, or because the polyhedron cannot contain a ball of radius ϵ. It can be determined that the polyhedron does not contain such a ball either because its volume has shrunk (indicated by the presence of a large number of constraints) or because it has become very flat (indicated by a very small slack value and a large potential function value).*

The constant term hidden in the "big-O" analysis is quite large. It is of interest to reduce this constant, in order to develop a more practical algorithm. It is also of interest to try to reduce the exponent on L, obtaining perhaps a bound on the number of iterations of the form $O(mL)$, similar to the bound shown for many interior point algorithms for linear programming. One extension of this algorithm is to solve the optimization problem using a long-step approach; this is the subject of §4.1.

3 A volumetric cutting plane algorithm

Vaidya [35] introduced the volumetric barrier function and proved several interesting properties of a volumetric barrier cutting plane algorithm. His ideas were developed further by Anstreicher [2, 5], and Ramaswamy and Mitchell [33]. In particular, Ramaswamy and Mitchell showed that cutting planes could be added right through the current point and that the optimization problem could be solved using these techniques, and Anstreicher introduced a number of refinements in the proof techniques that considerably improved the constants involved in the complexity analysis, making the algorithm more practical.

As in §2, we approximate the convex set $\mathcal{C}$ using a finite set of inequalities, so $\mathcal{C} \subseteq \{y : A^T y \leq c\} =: Q$, where A is an $m \times n$ matrix and c is an n-vector. Let y be a strictly feasible point in Q and let $s = c - A^T y > 0$. The *volumetric barrier function*

for Q at the point y is

$$V(y) := \frac{1}{2} \ln \det(AS^{-2}A^T). \tag{17}$$

The *volumetric center* ω of Q is the point that minimizes $V(y)$. For any strictly feasible point y with corresponding slacks s, the ellipsoid $E(AS^{-2}A^T, y, 1)$ is contained within Q. It follows from (13) that minimizing $V(y)$ is equivalent to choosing y to maximize the volume of the inscribing ellipsoid $E(AS^{-2}A^T, y, 1)$ centered at y.

Recall the variational quantities σ_j defined in (11)

$$\sigma_j := \frac{a_j^T (AS^{-2}A^T)^{-1} a_j}{s_j^2}$$

for j from 1 to n. These quantities are the diagonal terms of the matrix $\bar{P}(y)$ defined in (12). It can be shown that the gradient of the volumetric barrier function is

$$\nabla V(y) = -AS^{-1}\sigma \tag{18}$$

and the Hessian is

$$\nabla^2 V(y) = AS^{-1}(3\Sigma - 2\bar{P}(y)^{(2)})S^{-1}A^T, \tag{19}$$

where Σ is the diagonal matrix with entries equal to the components of σ and $\bar{P}(y)^{(2)}$ is the Hadamard product of $\bar{P}(y)$ with itself. It is expensive to calculate the entries of this Hessian matrix. The principal cost is due to the dense nature of $\bar{P}(y)^{(2)}$: first, it is expensive to calculate all the entries in $\bar{P}(y)$, and then it is expensive to include the resulting dense matrix in a sequence of matrix-matrix multiplications. Vaidya exploited the properties of σ given above to motivate using an approximation to the diagonal of $3\Sigma - 2\bar{P}(y)^{(2)}$. In particular, Vaidya [35] defined the matrix

$$P(y) := A^T S^{-2} \Sigma A. \tag{20}$$

He was then able to show that

$$P(y) \preceq \nabla^2 V(y) \preceq 3P(y). \tag{21}$$

A polynomial time approximate Newton's method can be used to find the volumetric center ω of a polyhedron. The direction d used at each step is found by solving the system of equations $P(y)d = -\nabla V(y)$.

An appropriate multiple of $V(y)$ is an $O(m\sqrt{n})$-self-concordant barrier for the polyhedron, so, in principal, properties of self concordant barrier functions [32] can be used to construct algorithms. However, most of the proofs of the results for volumetric center algorithms are derived directly, exploiting the dependence of $V(y)$ on σ.

3.1 Comparing the volumetric center and the analytic center

The volumetric barrier function is related to the Hessian of the logarithmic barrier function. In particular, if $\bar{y}$ strictly satisfies the constraints $A^T y \leq c$, then the Hessian of the logarithmic barrier function at $\bar{y}$ is $\nabla^2 F(\bar{y}) = AS^{-2}A^T$ with S defined appropriately, and

$$V(y) = \frac{1}{2} \ln \det(\nabla^2 F(\bar{y})). \tag{22}$$

The complexity bound proved for the volumetric barrier cutting plane method is smaller than that for the analytic center cutting plane method. This is in part because the volumetric center is closer to a geometrical center (for example, the center of mass) than the analytic center, at least for some problems.

For example, consider the following polyhedron. Let $Q := \{y \in I\!R : y \geq -1, py \leq p \text{ for } p = 1, \ldots, 100\} = [-1, 1]$. Geometrically, the center of this region is the origin.

The logarithmic barrier function is

$$F(y) = -\log(1+y) - 100\log(1-y) - \sum_{p=1}^{100} \log(p)$$

with gradient

$$\frac{dF}{dy} = -\frac{1}{1+y} + \frac{100}{1-y}.$$

Setting the gradient equal to zero shows that the analytic center is

$$y = -\frac{99}{101}.$$

The volumetric barrier function is

$$V(y) = \frac{1}{2} \ln \left(\frac{1}{(1+y)^2} + \frac{100}{(1-y)^2} \right).$$

The diagonal entries of the projection matrix $\bar{P}(y)$ are

$$\sigma_i = \begin{cases} \frac{1}{(1+y)^2} \frac{1}{\frac{1}{(1+y)^2} + \frac{100}{(1-y)^2}} & \text{for constraint } y \geq -1 \\ \frac{1}{(1-y)^2} \frac{1}{\frac{1}{(1+y)^2} + \frac{100}{(1-y)^2}} & \text{for constraint } py \leq p \end{cases}$$

so the gradient of the volumetric barrier function is

$$\frac{dV}{dy} = \frac{-\frac{1}{(1+y)^3} + \frac{100}{(1-y)^3}}{\frac{1}{(1+y)^2} + \frac{100}{(1-y)^2}}.$$

It follows that the volumetric center satisfies

$$100(1+y)^3 - (1-y)^3 = 0,$$

which is zero at $\omega \approx -0.6455$. Thus, the volumetric center is closer than the analytic center to the geometric center of this feasible region.

Cutting planes generated at the volumetric center will often be deeper cuts than those generated at the analytic center, as they would be in this example.

3.2 An algorithm

A volumetric barrier cutting plane algorithm is given in Figure 3. The initial relaxation consists of box constraints, $-\frac{1}{\epsilon}e \leq y \leq \frac{1}{\epsilon}e$. The box constraints are never dropped. An approximate volumetric center for the current relaxation is found, and a constraint is then added or dropped as appropriate, and the process is repeated. Approximate volumetric centers are defined in §3.3. If any σ_i drops below a threshold $\sigma_{\min}$ then a constraint is dropped; this process takes place in Steps 2 and 3 of the algorithm and is discussed further in §3.3. When the relaxation is updated, it is necessary to find a new approximate volumetric center in Step 6; this is also described in §3.3.

The termination criterion of the algorithm in Step 7 requires comparing the value of the volumetric barrier at the current iterate, $V(y)$, with a threshold. The rationale for this criterion is presented in §3.4.

3.3 Local convergence

In this subsection, we show how the number of Newton steps required in one call to Step 6 of the algorithm given in Figure 3 can be bounded. In the next subsection, we show how the number of calls to Step 6 can be bounded. The product of these two bounds will bound the overall number of Newton steps required by the volumetric center cutting plane algorithm.

Given a set of constraints $A^T y \leq c$, the volumetric center ω minimizes the volumetric barrier function $V(y) = \frac{1}{2}\ln\det(A^T S^{-2} A)$. The volumetric center is approached in the limit by an algorithmic approach, so it is necessary to work with approximate volumetric centers. The condition for a point to be an approximate volumetric center can be expressed as a condition on the norm of the gradient of the volumetric barrier function in the norm given by the approximation $P(y)$ to the Hessian of the volumetric barrier function. (This is similar to the definition of an approximate *analytic* center.) In order to define an approximate volumetric center formally, we need to define some other quantities.

1. **Initialize:**
   Set $n = 2m$, $A = [I, \ -I]$, and $c = \frac{1}{\epsilon}e$. Set $y = 0$ and $s = c$. Initialize the iteration counter $k = 0$. Initialize $\sigma_{\min}$ to some value between 0 and ϵ. Go to Step 4.

2. **Check to drop constraints:**
   Calculate σ_i for each constraint. Let $\bar{i}$ be the index with the smallest value of σ_i, for $i = 2m + 1, \ldots, n$. If $\sigma_{\bar{i}} < \sigma_{\min}$, go to Step 3, else go to Step 4.

3. **Drop constraint:**
   Drop the $\bar{i}$th row from $[A^T, c]$. Update $n \leftarrow n - 1$. Go to Step 6.

4. **Call oracle:**
   Call the oracle with y as the trial point. If the oracle does not return a violated constraint, **STOP** with feasibility.

5. **Update relaxation:**
   Let a and c_0 denote the constraint returned by the oracle. Set $\beta = a^T y$. Add the constraint $a^T y \leq \beta$ to $[A^T, c]$ and update $n \leftarrow n + 1$.

6. **Update y:**
   Find a new approximate volumetric center y. Update $s = c - A^T y$.

7. **Infeasible termination:**
   If $V(y)$ is sufficiently large, **STOP**, with the conclusion that $\mathcal{C}$ is empty. Otherwise, return to Step 2.

Figure 3: A volumetric barrier cutting plane algorithm

Given a point y satisfying $A^T y < c$, let $\sigma_{\bar{i}}$ be the smallest value of σ_i. Set

$$\nu := \min \left\{ (2\sqrt{\sigma_{\bar{i}}} - \sigma_{\bar{i}})^{-1/2}, \sqrt{\frac{1+\sqrt{m}}{2}} \right\}. \tag{23}$$

Anstreicher proves the following result showing that if the gradient of the volumetric barrier is sufficiently small at y, measured in an appropriate norm, then y is close to the volumetric center ω and $V(y)$ is close to the optimal value $V(w)$:

Theorem 3 ([5], Theorem 2.6, Corollary 2.7.) *Let y satisfy $A^T y < c$. Let g denote the gradient of the volumetric barrier function at y. Assume $\nu||g||_{P(y)^{-1}} \leq \gamma \leq \frac{1}{6}$. Then $\nu||\omega - y||_{P(y)} \leq 1$. If $\gamma = \frac{4}{27}$ then $V(y) - V(\omega) \leq \frac{0.0232}{\nu^2}$ and if $\gamma = \frac{1}{8}$ then $V(y) - V(\omega) \leq \frac{0.0113}{\nu^2}$.*

Thus, we regard y as an approximate volumetric center if

$$\nu||g||_{P(y)^{-1}} \leq \gamma \tag{24}$$

for some appropriate γ. If the current iterate y at Step 6 is an approximate volumetric center, then we move to Step 2. Otherwise, we need to improve y. This is done by using an approximate Newton step in the direction d satisfying $P(y)d = -\nabla V(y)$. For an appropriate step length, this will give a decrease of $O(\frac{1}{\sqrt{n}})$ in the value of the volumetric barrier function (Theorem 3.1 in [35]). The technical lemmas that are needed to prove this result look at points $\bar{y} = y + d$ for some feasible step d satisfying $||S^{-1}A^T d||_\infty = \delta < 1$ and obtain bounds on the difference between $V(\bar{y})$ and $V(y)$ and between $P(\bar{y})$ and $P(y)$, as a function of δ.

When a constraint is dropped, the criterion used in Step 3 ensures that the value of the volumetric barrier function at the volumetric center only decreases slightly, and, further, the old approximate volumetric center is sufficiently close to the volumetric center of the modified polytope that a new approximate volumetric center can be recovered in one iteration.

When adding a constraint, Vaidya [35] proposed weakening the added constraint so that a new approximate analytic center can be recovered in $O(1)$ Newton steps. These weakened cuts were also used in [2]. Ramaswamy and Mitchell [33] proposed adding central cuts, as presented in Step 5. They showed that an approximate analytic center can now be recovered in $O(\sqrt{m})$ Newton steps, and these central cuts were also used in [5]. When the cuts are placed in central position, the direction proposed in [29] is used to move off the added constraint, and then Newton steps are used to recover an approximate volumetric center.

3.4 Global convergence

Global convergence of volumetric center algorithms is usually shown by demonstrating that eventually the volumetric barrier function becomes too large for the feasible region to contain a ball of radius ϵ. This enables us to place an upper bound on the number of iterations required: either the algorithm will find a feasible point in this time, or no feasible point exists. For example, Anstreicher proves a result similar to the following theorem (Lemma 3.1, [2]):

Theorem 4 *Let the current polyhedral approximation Q to $\mathcal{C}$ have n constraints. If the value of the volumetric barrier at the volumetric center ω of Q is greater than $V_{\max} := mL + m\ln(n)$ then the volume of $\mathcal{C}$ is smaller than that of an m-dimensional sphere of radius ϵ.*

Proof: Let y^* be the analytic center of Q. Then P is contained within an ellipsoid centered at y^*:

$$P \subseteq \{y \in I\!R^n : (y - y^*)^T \nabla^2 F(y^*)(y - y^*) \leq n^2\}.$$

It follows that the volume of Q is no larger than the volume of this ellipsoid, so, from (13), the volume of $\mathcal{C}$ is bounded by

$$\mathrm{Vol}(\mathcal{C}) \leq \Upsilon_m n^m (\det(\nabla^2 F(y^*)))^{-1/2},$$

where Υ_m denotes the volume of the unit ball $E(I, 0, 1)$ in $I\!R^m$. Equation (22) then gives

$$\begin{aligned}
\mathrm{Vol}(\mathcal{C}) &\leq \Upsilon_m n^m e^{-V(y^*)} && \text{from (22)} \\
&\leq \Upsilon_m n^m e^{-V(\omega)} && \text{since } \omega \text{ minimizes } V(.) \\
&\leq \Upsilon_m n^m \epsilon^n n^{-m} && \text{from our hypothesis and (1)} \\
&= \Upsilon_m \epsilon^m, &&
\end{aligned}$$

as required. □

Notice that this maximum possible value of the volumetric barrier at the volumetric center increases logarithmically in the number of constraints, n. If we can show that there is a guaranteed constant increase in $V(\omega)$ when a constraint is added, then eventually this value will be larger than V_{max}, if no constraints are ever dropped — see Figure 4. This is shown by Vaidya [35], under the assumption that the variational

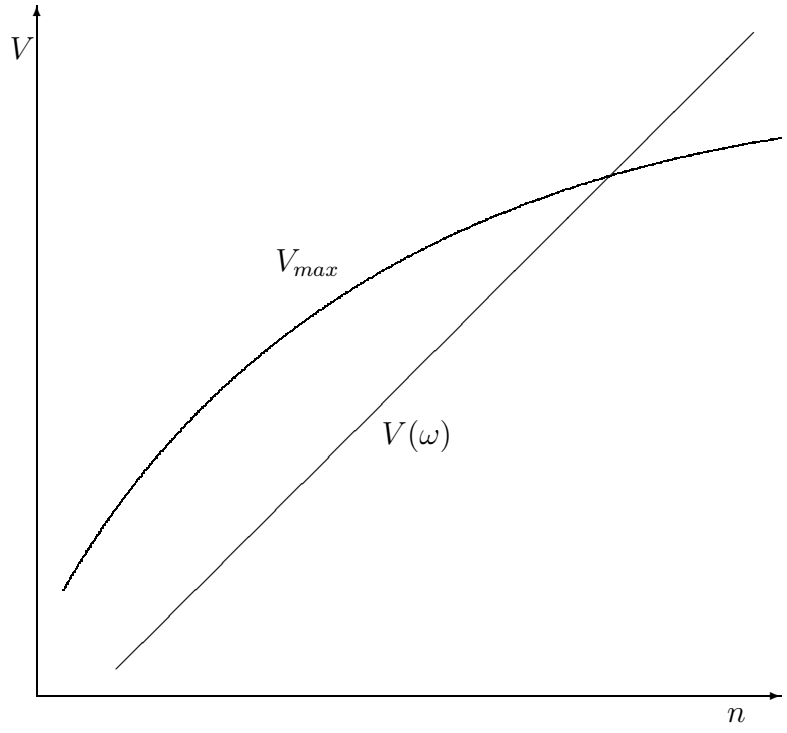

Figure 4: Comparing V_{max} and $V(\omega)$. The maximum number of constraints that can be generated is the value of n where the curve V_{max} and the line $V(\omega)$ cross.

quantities are bounded away from zero. It is possible to modify this result to obtain a complexity result when constraints are not dropped. However, the complexity result of $O(m^2L^2)$ calls to the oracle provided by this approach is weaker than that provided by an alternative, and the analysis is more complicated.

Therefore, other complexity results (including another result in [35]) also consider dropping constraints. Now the structure of the complexity results requires showing the increase in $V(\omega)$ when a constraint is added is at least ΔV^+, and also the decrease when a constraint is dropped is no more than ΔV^-, for constants ΔV^+ and ΔV^- satisfying $\Delta V^+ - \Delta V^- \geq k$ for some positive $O(1)$ constant k. We keep the original box constraints in the formulation, so the number of constraints describing Q at any stage is at least $2m$. Thus, if we drop a constraint, then the value of the volumetric center may decrease, but the decrease is less than the increase seen when the constraint was originally added. It follows that the increase in the value of the volumetric center is at least

$V(\omega) -$ the original value

$$
\begin{aligned}
&\geq \quad \text{(number of constraints added and still in relaxation)} \ \times \Delta V^{+} \\
&\qquad + \text{(number of constraints added and subsequently dropped)} \ \times k \\
&\geq \quad k \times \ \text{(total number of constraints added)} \\
&\geq \quad \frac{kp}{2}
\end{aligned}
$$

where p is the number of outer iterations performed by the algorithm, that is, the number of visits to Step 6. Thus, the value of the volumetric barrier at the volumetric center ω is bounded below by a function that increases linearly in the number of outer iterations performed by the algorithm.

The results of this section have been stated in terms of the exact volumetric center ω. They can be modified to use approximate volumetric centers as defined in §3.3. The overall complexity of the algorithm can be summarized as follows:

Theorem 5 *The volumetric center cutting plane algorithm either finds a feasible solution or proves that $\mathcal{C}$ is empty in $O(mL)$ calls to the oracle and either $O(mL)$ or $O(m^{1.5}L)$ approximate Newton steps, depending on the choice of parameters.*

Note that the number of calls to the oracle required by the volumetric algorithm is smaller than the corresponding number for either the ellipsoid algorithm or the logarithmic barrier algorithm presented in §2. This complexity of $O(mL)$ calls to the separation oracle is optimal — see Nemirovskii and Yudin [31].

3.5 Extensions and future research

This algorithm is still far from practical, because of the large constants involved. It is of interest to reduce the constant terms in the complexity analysis in order to make the algorithm more practical, and Anstreicher [5] has made a start on this.

The algorithm has been extended to add multiple cuts — see [5, 33]. Neither of these approaches uses the restart technique developed by Goffin and Vial [11] for analytic center cutting plane algorithms. A similar result for the volumetric center would be of interest.

Vaidya [35] discussed solving optimization problems using his algorithm. He suggested cutting on the objective function if the current approximate volumetric center is feasible, and otherwise cutting on a violated constraint. Such an approach has also been proposed to extend other algorithms for the convex feasibility problem, including the ellipsoid method and analytic center cutting plane algorithms. Ramaswamy and Mitchell [33] proposed an alternative, long-step, approach, and we discuss this further in §4.2.

Anstreicher has investigated the use of the volumetric algorithm to solve semidefinite programming problems [6] and for problems with convex quadratic constraints [4].

Vaidya [35] proposed combining the volumetric and logarithmic barrier functions. This has been pursued further by Anstreicher [6, 3]. The algorithms considered in these references are not cutting plane algorithms, but algorithms for problems where the constraints are given explicitly. Anstreicher [3] was able to obtain a complexity of $O(n^{1/4}m^{1/4}L)$ for linear programming with a short step algorithm combining the two barrier functions; this is better than the standard $O(\sqrt{\max\{m,n\}}L)$ complexity for short step potential reduction algorithms, especially if m and n differ markedly. Nesterov and Nemirovskii (§5.5 of [32]) derive the self-concordancy parameter for both the volumetric barrier and the combined barrier; simplified proofs appear in [3]. The combined barrier is an $O(\sqrt{mn})$-logarithmically homogeneous self-concordant barrier.

4 Long-step polynomial approaches to solving the convex optimization problem

The algorithms presented in §2 and §3 are designed to solve the convex feasibility problem. These feasibility algorithms can be extended to solve the optimization problem

$$\begin{array}{lll} \max & b^T y & \\ \text{subject to} & y \in \mathcal{C}, & (OPT) \end{array}$$

by cutting on a violated constraint when the trial point is infeasible, and cutting on the objective function when the trial point is feasible but not optimal. However, this is somewhat like a 'short step' method — when cutting on the objective function, the cut is placed such that the next iterate is in a small ellipsoid around the current point. This suggests that the progress in objective function value cannot be too much.

In this section, we look at long step cutting plane algorithms for the optimization problem. These methods use potential functions that include the objective function explicitly, allowing good progress in improving the objective when the current approximate center is feasible. In a long step algorithm for linear programming, the barrier parameter μ is reduced by a constant factor at each iteration, and this factor does not depend on the problem size. The iterates trace the central trajectory of the polytope. Long step methods perform far better in practice than short step methods, although their worst case theoretical complexity results are slightly worse. Another advantage of the explicit inclusion of the objective function in the potential function is

that it obviates the need to add multiple copies of cuts corresponding to the objective function. These algorithms also maintain primal and dual feasible iterates, allowing for early termination when the sub-optimality is determined to be within acceptable limits.

Thus, the algorithms are essentially long step path-following algorithms that use a polyhedral approximation to $\mathcal{C}$, and whenever infeasibility is encountered a scheme similar to that in either §2 or §3 is used until the current iterate again becomes feasible. The extension of the method of §2 is described in §4.1 and the extension of the volumetric cutting plane method is presented in §4.2.

The linear programming relaxation of the optimization problem (OPT) can be written

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & A^T y & \leq & c & \qquad (P) \end{array}$$

with dual

$$\begin{array}{lrcll} \min & c^T x & & & \\ \text{s.t.} & Ax & = & b & \qquad (D) \\ & x & \geq & 0. & \end{array}$$

It should be noted that any feasible solution to (D) will provide an upper bound on the optimal value of the optimization problem, since (P) is a relaxation of the problem of interest.

In addition to the assumptions made earlier regarding ϵ, we assume that it suffices to find a solution to (OPT) to within an accuracy ϵ. Further, we assume that $\mathcal{C}$ is nonempty.

4.1 A long step logarithmic barrier cutting plane algorithm

The algorithm we present in this section extends the algorithm presented in §2 by including a term for the objective function in the barrier function. We use the barrier function

$$f(y, \mu) := -\frac{b^T y}{\mu} + F(y) \tag{25}$$

where μ is a positive scalar and $F(y)$ was defined in (6). For a fixed value of $\mu > 0$, it is desirable to minimize $f(y, \mu)$, leading to a balance between the objective function and centrality. The barrier parameter μ is driven to zero, leading to increasing emphasis on the objective function and convergence to an optimal solution. The algorithm has a complexity of $O(nL^2)$ iterations, the same as that for the algorithm presented in §2.

Luo *et al.* [22] have extended the fully polynomial algorithm recently surveyed by Goffin and Vial [12] to a long-step method with exponential complexity, although it

does not require that constraints be dropped. Den Hertog *et al.* [17, 18, 16] and Kaliski *et al.* [19] discuss an algorithmic framework very similar to the one described in this section, combining long steps with the addition and deletion of cuts to yield a long step cutting plane algorithm. These papers prove complexity bounds that are polynomial in the total number of cuts that become active, and not necessarily polynomial in the dimension of the space. In their conclusions, the authors of [16] mention that proving possible polynomial complexity of their algorithm remains open. In [19], the authors point out that results from [10] suggest that their algorithm is fully polynomial.

At any given iteration, we operate with a relaxation of (OPT). The algorithm is initialized with the relaxation

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & y & \leq & \frac{1}{\epsilon} e & \qquad (P_0) \\ & -y & \leq & \frac{1}{\epsilon} e & \\ & b^T y & \leq & \frac{1}{\epsilon}\sqrt{m} & \end{array}$$

where, by our assumptions, the feasible set of this relaxation contains $\mathcal{Y}$, the set of optimal solutions to (OPT). The first $2m$ hyperplanes included in this initial relaxation are the box hyperplanes, familiar from the algorithm in Figure 2. The final constraint does not affect the feasible region of the initial relaxation. As the algorithm obtains better upper bounds on the optimal value of the full problem, the right hand side of this upper bound constraint is updated.

At iteration k, we would have something like :

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & y & \leq & \frac{1}{\epsilon} e_n & \qquad (P_k) \\ & -y & \leq & \frac{1}{\epsilon} e_n & \\ & b^T y & \leq & u_k & \\ & \bar{A}_k^T y & \leq & \bar{c}_k & \end{array}$$

where $\bar{A}_k \in I\!R^{m \times \bar{n}_k}$, and u_k is some upper bound on the optimal objective function value. This can be written more simply as

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & A_k^T y & \leq & b_k & \qquad (P_k) \end{array}$$

with $A_k \in I\!R^{m \times n_k}$, so $n_k = 2m + 1 + \bar{n}_k$. We refer to the feasible region of (P_k) as Q_k. Throughout most of the discussion, we may omit the subscripts k since that causes no ambiguity.

The gradient of the barrier function $f(y, \mu)$ is given by

$$\nabla f(y, \mu) = -\frac{b}{\mu} + \sum_{i=1}^{n} \frac{a_i}{s_i} = -\frac{b}{\mu} + A^T S^{-1} e \tag{26}$$

and the Hessian by

$$\nabla^2 f(y, \mu) = A^T S^{-2} A. \tag{27}$$

For a given value of μ, $y(\mu)$ denotes the unique minimizer of this barrier function. We refer to this unique point when we use the term *(exact) μ-center*. Because the polytope we have at any iteration is always bounded, it has a unique exact μ-center for any $\mu > 0$. In a similar manner to (9), we define an *approximate μ-center* to be a point y satisfying

$$\Psi_f(y, \mu) := \nabla f(y, \mu)^T (\nabla^2 f(y, \mu))^{-1} \nabla f(y, \mu) \leq \frac{1}{16}. \tag{28}$$

It should be noted that $\Psi_f(y, \mu)$ is the square of the measure of centrality $\delta(y, \mu)$ given by

$$\begin{aligned} \delta(y, \mu) \quad := \quad & \min_x \left|\left| \frac{Sx}{\mu} - e \right|\right| \\ & \text{s.t. } Ax = b. \end{aligned} \tag{29}$$

For a given y and μ, it is easy to find the solution x to this quadratic program. It follows from (29) that each component of Sx must be close to μ, so in particular the duality gap $s^T x$ can be bounded in terms of $\delta(y, \mu)$ and μ:

$$\begin{aligned} s^T x &= \sum_{i=1}^{n} s_i x_i \\ &\leq \sum_{i=1}^{n} \mu(1 + \delta(y, \mu)) \\ &= n\mu(1 + \delta(y, \mu)). \end{aligned}$$

This is used in the analysis of the algorithm to provide an upper bound on the optimal value of (OPT). If $\delta(y, \mu) \leq 1$ then the corresponding x is feasible in the dual to (P_k). This is a constrained version of the dual problem to (OPT), so the value of x gives an upper bound on the optimal value of (OPT). In particular, we obtain an upper bound of

$$u = b^T y + n\mu(1 + \delta(y, \mu)) \tag{30}$$

for any feasible y in the current relaxation with $\delta(y, \mu) \leq 1$. If, in addition, $y \in \mathcal{C}$, and y is an approximate μ-center, then it is within $1.25n\mu$ of optimality. This upper

bound is used in the algorithm to give a constraint. The constraint is redundant when it is added, but it may later become useful if constraints are dropped. However, the principal use of the constraint is in the analysis of the algorithm, because it allows an explicit bounding of various quantities, including the size of a polytope Q^*.

This polytope Q^* is guaranteed to contain the optimal solution to (OPT). Before defining Q^*, it is necessary to define some more notation. The algorithm moves from one approximate μ-center to another. If the approximate μ-center is in $\mathcal{C}$ either a constraint is dropped (using the criteria of the algorithm of §2) or the barrier parameter is reduced. If the barrier parameter is reduced, let y_{prev} denote the current feasible approximate μ-center, let μ_{prev} denote the barrier parameter before it is reduced, and let n_{prev} denote the number of constraints before the barrier parameter is reduced. Standard interior point iterations are then taken until a new approximate μ-center $\tilde{y}$ is found. This point $\tilde{y}$ may not be in $\mathcal{C}$. The polytope Q^* is defined as follows:

$$Q^* := Q \cap \{y \in \Re^m : b^T y \geq \zeta\} \tag{31}$$

where ζ is equal to $\min(b^T\tilde{y}, b^T y_{prev}) - 1.25 n_{prev}\mu_{prev}$. Thus, this polytope is certain to contain the optimal set to the problem of interest and the current point $\tilde{y}$. This property is important, and enables the construction of termination criteria based on the volume and width of Q^*. This property would hold without the $1.25 n_{prev}\mu_{prev}$ term in the definition of ζ; this extra term is used to relate the width of Q^* to the width of Q. It is important to note that the polytope Q^* is used only in the theoretical analysis — lower-bound constraints that restrict subsequent iterates to be in the polytope Q^* are not added explicitly, although they could be.

The algorithm is contained in Figure 5. The algorithm will terminate with optimality if the current iterate y is an approximate μ-center for a sufficiently small μ, provided $y \in \mathcal{C}$. The algorithm also has termination criteria based upon the number of constraints in the current relaxation and on the width of Q^*. These will be invoked if, for example, the algorithm finds the optimal solution while the value of μ is still large: further iterations may well produce infeasible iterates, resulting in the addition of constraints, or the new iterates may be feasible but inferior to the known feasible point, resulting in reductions in μ. Therefore, it is possible that the optimal point returned by the algorithm will not be final approximate analytic center found in Step 7 or Step 8. The algorithm stops in Step 2 if the current polytope Q^* is too narrow to contain a ball of the appropriate diameter; it stops in Step 7 if the number of constraints implies that the volume of Q^* is too small.

The principal difference between this algorithm and the one contained in Figure 2 lies in the handling of μ. If the oracle fails to find a violated constraint in Step 5

1. **Initialize:**
   Set $\mu = \frac{1}{\epsilon}$, $n = 2m + 1$, $A = [I,\ -I,\ b]$, and $c^T = [\frac{1}{\epsilon}e^T,\ \frac{1}{\epsilon}e^T,\ \frac{\sqrt{m}}{\epsilon}]$. Set $y = 0$ and $s = c$. Set $\kappa_i = c_i$ for $i = 1, \ldots, n$. Initialize the iteration counter $k = 0$. Set $u_{prev} = c_n$. Initialize x. Iterate, if necessary, to get approximate μ-center. Set constants $n_{max} := 4093mL$, and $s_{min} := 10^{-5}\epsilon^3/(2m^{1.5}L)$. Go to Step 5.

2. **Check to drop constraints:**
   If $s_i \leq s_{min}$ for some i, **STOP**: the best feasible point found so far is optimal. Let $\Lambda := \max\{\frac{s_i}{\kappa_i} : i = 2m + 2, \ldots, n\}$. If $\Lambda \leq 2$, go to Step 5. Otherwise, if there is an index $\bar{i}$ with $\frac{s_{\bar{i}}}{\kappa_{\bar{i}}} > 2$ and $\sigma_{\bar{i}} < 0.04$, go to Step 3, else let $\bar{i}$ be any index with $\frac{s_i}{\kappa_i} > 2$ and go to Step 4.

3. **Drop constraint:**
   Drop the $\bar{i}$th row from $[A^T, c]$. Update $n \leftarrow n - 1$. Go to Step 7.

4. **Update κ:**
   Set $\kappa_{\bar{i}} = s_i$. Go to Step 2.

5. **Call oracle:**
   Call the oracle with y as the trial point. If the oracle does not return a violated constraint, go to Step 8; otherwise go to Step 6.

6. **Update relaxation:**
   Let a and c_0 denote the constraint returned by the oracle. Set $\beta = a^T y$. Add the constraint $a^T y \leq \beta$ to $[A^T, c]$ and update $n \leftarrow n + 1$. If $n \geq n_{max}$, **STOP**: the best feasible point found so far is optimal.

7. **Update y:**
   Find a new approximate analytic center y in O(1) Newton steps. Update $s = c - A^T y$. Set $\kappa_n = c_n - a^T y$ if we came here from Step 6. Return to Step 2.

8. **Reduce μ:**
   Test y for optimality (is $1.25n\mu < \epsilon$?). If optimal, get dual solution x and STOP. Else, set $u = b^T y + 1.25n\mu$. Set $u_{prev} = \min\{u, u_{prev}\}$ and update the constraint $b^T y \leq u_{prev}$. Set $\mu = \mu\rho$ where $\rho \in (0.5, 1)$ is a constant independent of the problem. Take Newton steps with linesearch to find new approximate μ-center. Return to Step 2.

Figure 5: **A long step variant of the algorithm in Figure 2**

then the algorithm moves to Step 8, reduces μ, and finds a new approximate μ-center. This new point is found using a Newton method with long steps, so it may require $O(mL^2)$ steps (note that $n = O(mL)$).

It might appear that a short step approach to reducing μ would result in a smaller overall complexity of $O(\sqrt{mL}L)$ Newton steps, as it does for algorithms for linear programming. However, this is not the dominant term in the overall complexity analysis, because the number of cutting plane additions is bounded by $O(mL^2)$. Thus, for this cutting plane algorithm, the overall complexity is the same whether a short step or a long step approach to reducing μ is used.

As mentioned above, an upper bound constraint on the objective function value is added to the formulation, in Step 1. This constraint is implied by the box constraints $-\frac{1}{\epsilon}e \leq y \leq \frac{1}{\epsilon}e$. Whenever a feasible iterate is found in Step 5, the right hand side of the upper bound constraint is be updated using the formula given in (30), in Step 8, provided this leads to an improved upper bound.

One technique used in the analysis of the algorithm is to use the modified barrier function

$$f_u(y, \mu) := \frac{u - b^T y}{\mu} + G(y), \tag{32}$$

where u is the current best known upper bound on the optimal value of (OPT) and $G(y)$ was defined in (14). It follows that a reduction in the upper bound u will lead to a decrease in the modified barrier function $f_u(y, \mu)$. Local convergence analysis (that is, the number of Newton steps required in Step 7 or Step 8) is not changed by the explicit inclusion of the upper bound in the modified barrier function. The global convergence analysis is simplified because the size of $\frac{u-b^T y}{\mu}$ is limited at approximate analytic centers to be $O(n)$. Therefore, once f_u has increased sufficiently, we can conclude that $G(y)$ has also increased to a large enough value to allow termination due to either volume or width considerations.

To be more specific, it can be shown that when the barrier parameter is reduced, the decrease in $f_u(y, \mu)$ from one approximate μ-center to the next is no greater than $O(mL)$ (see Lemma 18 in [28]). The number of reductions in μ is $O(L)$, so the total decrease in f_u due to barrier parameter reduction steps is no more than $O(mL^2)$. Recall that the analysis of the algorithm in §2 required showing that the barrier function increases from one approximate analytic center to the next, and that there is a limit to how large this function can become. The decrease in the potential function due to the inclusion of the barrier parameter reduction steps is of the same order as the maximum possible increase in f_u before we can terminate due to volume or width considerations. Therefore, the overall complexity bound for the algorithm

is also $O(mL^2)$ Newton steps and $O(mL)$ calls to the oracle.

Note that the algorithm can be terminated before Q^* becomes too small, in the same way that the ellipsoid method can be terminated. In particular, if Q^* can be contained in an ellipsoid of volume no more than that of a ball of radius ϵ, then we could round to an optimal solution using simultaneous diophantine approximation, as described in Grötschel *et al.* [15]. An alternative method for rounding to an optimal solution is described in Megiddo [23].

4.2 A long-step volumetric center cutting plane algorithm

Ramaswamy and Mitchell [33] have proposed a long step variant of the volumetric cutting plane algorithm. They used the barrier function

$$\Phi_u(y,\mu) := \frac{u - b^T y}{\mu} + V(y) \tag{33}$$

where $V(y)$ is the volumetric barrier function defined in (17), u is an upper bound on the optimal value of (OPT), and μ is the barrier parameter. As in §4.1, the presence of u in the barrier function $\Phi_u(y,\mu)$ serves to bound the contribution of the objective function. It then follows that once the barrier function $\Phi_u(y,\mu)$ is sufficiently large, the volumetric barrier function $V(y)$ is also large enough to allow termination on the basis that a region similar to Q^* is too small to contain a ball of radius ϵ.

The algorithm generates an approximate μ-volumetric center (see (34) below), and tries to drop or add constraints as in the algorithm presented in §3. If no constraints should be dropped and if no violated constraints are returned by the oracle, then the barrier parameter μ is reduced and quasi-Newton steps are taken to recover a new approximate μ-volumetric center.

The algorithm is initialized with the same relaxation (P_0) given in §4.1, and the relaxation at any stage would have the form (P_k). A strictly feasible point y in the current relaxation is an *approximate μ-volumetric center* if it satisfies

$$\nu||\nabla\Phi_u(y,\mu)||_{P(y)} \leq \gamma \tag{34}$$

for some appropriate constant γ, where ν was defined in (23).

The algorithm is given in Figure 6. Local convergence in Step 5 can be shown to have the same complexity, namely $O(\sqrt{m})$ quasi-Newton steps, as for the volumetric algorithm presented in §3. Local convergence in Step 7 requires $O(m^{1.5})$ quasi-Newton steps.

It can be shown that the barrier function $\Phi_u(y,\mu)$ increases by at least a constant $\Delta\Phi^+$ when a constraint is added, and decreases by no more than a constant $\Delta\Phi^-$

1. **Initialize:**
   Set $\mu = \frac{1}{\epsilon}$, $n = 2m+1$, $A = [I,\ -I,\ b]$, and $c^T = [\frac{1}{\epsilon}e^T,\ \frac{1}{\epsilon}e^T,\ \frac{\sqrt{m}}{\epsilon}]$. Set $y = 0$ and $s = c$. Initialize the iteration counter $k = 0$. Set $u_{prev} = c_n$. Iterate, if necessary, to get approximate μ-volumetric center. Initialize $\sigma_{\min}$ to some value between 0 and ϵ. Go to Step 4.

2. **Check to drop constraints:**
   If $V(y)$ is sufficiently large, **STOP**: the best point seen so far is optimal. Calculate σ_i for each constraint. Let $\bar{i}$ be the index with the smallest value of σ_i, for $i = 2m+2, \ldots, n$. If $\sigma_{\bar{i}} < \sigma_{\min}$, go to Step 3, else go to Step 4.

3. **Drop constraint:**
   Drop the $\bar{i}$th row from $[A^T, c]$. Update $n \leftarrow n-1$. Go to Step 6.

4. **Call oracle:**
   Call the oracle with y as the trial point. If the oracle does not return a violated constraint, go to Step 7.

5. **Update relaxation:**
   Let a and c_0 denote the constraint returned by the oracle. Set $\beta = a^T y$. Add the constraint $a^T y \leq \beta$ to $[A^T, c]$ and update $n \leftarrow n+1$.

6. **Update y:**
   Find a new approximate volumetric center y. Update $s = c - A^T y$. Return to Step 2.

7. **Reduce μ:**
   Test y for optimality. If optimal, get dual solution x and STOP. Else, set $u = b^T y + 2n\mu$. Set $u_{prev} = \min\{u, u_{prev}\}$ and update the constraint $b^T y \leq u_{prev}$. Set $\mu = \mu\rho$ where $\rho \in (0.5, 1)$ is a constant independent of the problem. Take quasi-Newton steps with linesearch to find new approximate μ-center. Return to Step 2.

Figure 6: A long step volumetric barrier cutting plane algorithm

when a constraint is dropped, with $\Delta\Phi^+ - \Delta\Phi^-$ bounded away from zero, as in §3. As in §4.1, global convergence requires showing that the reduction in the function $\Phi_u(y, \mu)$ is limited when μ is reduced. Since the number of reductions in the barrier parameter is no more than $O(L)$, the following result was proved in [33]:

Theorem 6 *The algorithm will terminate in at most $O(m^{1.5}L)$ quasi-Newton steps, with at most $O(mL)$ constraints added.*

For the feasibility algorithm, adding constraints through the current iterate increases the number of quasi-Newton steps in Step 6 from $O(1)$ to $O(\sqrt{m})$, and this increases the overall complexity of the algorithm by a factor of $\sqrt{m}$. For the algorithm in this section, backing off the constraints does not lead to a reduction in the overall complexity, because Step 7 requires $O(m^{1.5})$ quasi-Newton steps.

5 Solving integer programming problems

Cutting plane algorithms can be used to solve integer programming problems. In this case, the set $\mathcal{C}$ in (OPT) is the convex hull of the set of feasible integer solutions. Note that this formulation can be used for nonlinear as well as linear integer programs, as long as the inequality constraints are convex. Problems with concave objective functions $f(y)$ can also be cast in the form of (OPT): introduce an extra variable Θ to give the objective function value, maximize this additional variable, and include the constraint $\Theta - f(y) \leq 0$. An optimal solution to (OPT) occurs at an extreme point of $\mathcal{C}$, so solving (OPT) will give the optimal solution to the integer program if it is unique.

Classical Gomory cutting planes for general integer linear programming problems were extended to interior point cutting plane methods in [24]. Much recent success with cutting plane methods has come with the use of problem-specific cutting planes. For experience with interior point cutting plane methods with problem-specific cutting planes, see [25] or [26]. These methods are surveyed in [27].

The algorithms described in this paper construct polyhedral approximations to the convex set of interest, $\mathcal{C}$. Mixed integer nonlinear programming problems can be solved using such an approach. For an example of a polyhedral cutting plane approach for solving maxcut problems via a semidefinite formulation, see Krishnan and Mitchell [20, 21]. Akrotirianakis *et al.* [1] solve mixed integer nonlinear programming problems using a cutting plane approach where nonlinear programming relaxations are constructed and the nonlinear programs are solved using an interior

point method. The theoretical work of Mokhtarian and Goffin [30] addresses the performance of cutting plane methods where nonlinear cuts are added, so nonlinear programming approximations to $\mathcal{C}$ are solved.

Grötschel *et al.* [15] showed the polynomial equivalence of separation and optimization for integer programming problems. In particular, they showed that if the separation oracle requires polynomial time then the optimization problem (OPT) can be solved in polynomial time, if the ellipsoid algorithm is used to solve the relaxations. The polynomial methods presented in this paper provide an alternative proof of this fundamental result.

6 Conclusions and open questions

The volumetric cutting plane algorithm for the feasibility problem gives a complexity for the number of calls to the oracle that is optimal. This complexity can be achieved whether or not the algorithm is designed to drop unimportant constraints. The complexity of the cutting plane algorithm based on the logarithmic barrier function is slightly worse, and the analysis requires that unimportant constraints be dropped. This algorithm is polynomial. It would be desirable to reduce the complexity of such an algorithm to match the complexity of the volumetric center algorithm.

Algorithms that are based on the analytic center and do not drop constraints have only been shown to be fully polynomial. It would be of interest to develop a polynomial cutting plane algorithm using the logarithmic barrier function that did not require that unimportant constraints be dropped.

The algorithms have been extended to handle optimization problems, through the introduction of a linear term for the objective function into the barrier functions. These algorithms allow long-step reductions in the barrier parameter μ. The long-step variant of the logarithmic cutting plane algorithm has the same complexity as the original, but the long-step volumetric cutting plane algorithm requires a larger number of quasi-Newton steps (although the same number of calls to the oracle) as the feasibility version.

References

[1] I. Akrotirianakis, I. Maros, and B. Rustem. An outer approximation based branch and cut algorithm for convex 0-1 MINLP problems. *Optimization Methods and Software*, 16:21–47, 2001.

[2] K. M. Anstreicher. On Vaidya's volumetric cutting plane method for convex programming. *Mathematics of Operations Research*, 22:63–89, 1997.

[3] K. M. Anstreicher. Volumetric path following algorithms for linear programming. *Mathematical Programming*, 76:245–263, 1997.

[4] K. M. Anstreicher. The volumetric barrier for convex quadratic constraints. Technical report, Department of Management Sciences, University of Iowa, Iowa City, Iowa 52242, October 1998.

[5] K. M. Anstreicher. Towards a practical volumetric cutting plane method for convex programming. *SIAM Journal on Optimization*, 9:190–206, 1999.

[6] K. M. Anstreicher. The volumetric barrier for semidefinite programming. *Mathematics of Operations Research*, 25(3):365–380, 2000.

[7] D. S. Atkinson and P. M. Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69:1–43, 1995.

[8] O. Bahn, O. Du Merle, J. L. Goffin, and J. P. Vial. A cutting plane method from analytic centers for stochastic programming. *Mathematical Programming*, 69:45–73, 1995.

[9] J.-L. Goffin, J. Gondzio, R. Sarkissian, and J.-P. Vial. Solving nonlinear multicommodity network flow problems by the analytic center cutting plane method. *Mathematical Programming*, 76:131–154, 1997.

[10] J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM Journal on Optimization*, 6:638–652, 1996.

[11] J.-L. Goffin and J.-P. Vial. Multiple cuts in the analytic center cutting plane method. *SIAM Journal on Optimization*, 11(1):266–288, 2001.

[12] J.-L. Goffin and J.-P. Vial. Convex nondifferentiable optimization: a survey focussed on the analytic center cutting plane method. *Optimization Methods and Software*, 17(5):805–867, 2002.

[13] J. Gondzio. Warm start of the primal-dual method applied in the cutting plane scheme. *Mathematical Programming*, 83:125–143, 1998.

[14] J. Gondzio, O. du Merle, R. Sarkissian, and J.-P. Vial. ACCPM — A library for convex optimization based on an analytic center cutting plane method. *European Journal of Operational Research*, 94:206–211, 1996.

[15] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, Germany, 1988.

[16] D. den Hertog, J. Kaliski, C. Roos, and T. Terlaky. A logarithmic barrier cutting plane method for convex programming problems. *Annals of Operations Research*, 58:69–98, 1995.

[17] D. den Hertog, C. Roos, and T. Terlaky. A build-up variant of the path-following method for LP. *Operations Research Letters*, 12:181–186, 1992.

[18] D. den Hertog, C. Roos, and T. Terlaky. Adding and deleting constraints in the logarithmic barrier method for LP. In D.-Z. Du and J. Sun, editors, *Advances in Optimization and Approximation*, pages 166–185. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[19] J. Kaliski, D. Haglin, C. Roos, and T. Terlaky. Logarithmic barrier decomposition methods for semi-infinite programming. *International Transactions in Operations Research*, 4:285–303, 1997.

[20] K. Krishnan and J. E. Mitchell. Cutting plane methods for semidefinite programming. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, November 2002.

[21] K. Krishnan and J. E. Mitchell. Semidefinite cutting plane approaches for the maxcut problem. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, April 2003.

[22] Z.-Q. Luo, C. Roos, and T. Terlaky. Complexity analysis of a logarithmic barrier decomposition method for semi-infinite linear programming. *Applied Numerical Mathematics*, 29(3):379–395, 1999. Special issue dedicated to the HPOPT-I workshop held in Delft, September 19–20, 1996.

[23] N. Megiddo. On finding primal- and dual-optimal bases. *ORSA Journal on Computing*, 3:63–65, 1991.

[24] J. E. Mitchell. Fixing variables and generating classical cutting planes when using an interior point branch and cut method to solve integer programming problems. *European Journal of Operational Research*, 97:139–148, 1997.

[25] J. E. Mitchell. Computational experience with an interior point cutting plane algorithm. *SIAM Journal on Optimization*, 10(4):1212–1227, 2000.

[26] J. E. Mitchell and B. Borchers. Solving linear ordering problems with a combined interior point/simplex cutting plane algorithm. In H. L. Frenk *et al.*, editor, *High Performance Optimization*, chapter 14, pages 349–366. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[27] J. E. Mitchell, P. M. Pardalos, and M. G. C. Resende. Interior point methods for combinatorial optimization. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*, volume 1, pages 189–297. Kluwer Academic Publishers, 1998.

[28] J. E. Mitchell and S. Ramaswamy. A long-step, cutting plane algorithm for linear and convex programming. *Annals of Operations Research*, 99:95–122, 2000.

[29] J. E. Mitchell and M. J. Todd. Solving combinatorial optimization problems using Karmarkar's algorithm. *Mathematical Programming*, 56:245–284, 1992.

[30] F. S. Mokhtarian and J.-L. Goffin. A nonlinear analytic center cutting plane method for a class of convex programming problems. *SIAM Journal on Optimization*, 8:1108–1131, 1998.

[31] A. S. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley, 1983.

[32] Y. E. Nesterov and A. S. Nemirovsky. *Interior Point Polynomial Methods in Convex Programming : Theory and Algorithms*. SIAM Publications. SIAM, Philadelphia, USA, 1993.

[33] S. Ramaswamy and J. E. Mitchell. A long step cutting plane algorithm that uses the volumetric barrier. Technical report, DSES, Rensselaer Polytechnic Institute, Troy, NY 12180, June 1995.

[34] J. Renegar. A polynomial-time algorithm based on Newton's method for linear programming. *Mathematical Programming*, 40:59–93, 1988.

[35] P. M. Vaidya. A new algorithm for minimizing convex functions over convex sets. *Mathematical Programming*, 73:291–341, 1996.

[36] Y. Ye, O. Güler, R. A. Tapia, and Y. Zhang. A quadratically convergent $O(\sqrt{n}L)$–iteration algorithm for linear programming. *Mathematical Programming*, 59:151–162, 1993.

[37] Y. Zhang, R. A. Tapia, and J. E. Dennis. On the superlinear and quadratic convergence of primal–dual interior point linear programming algorithms. *SIAM Journal on Optimization*, 2(2):304–324, 1992.