CONICAL PROJECTION ALGORITHMS FOR LINEAR PROGRAMMING

by

Clovis Gonzaga

Memorandum No. UCB/ERL M87/11

5 March 1987

COVER PAGE

CONICAL PROJECTION ALGORITHMS FOR LINEAR PROGRAMMING

by

Clovis Gonzaga

Memorandum No. UCB/ERL M87/11

5 March 1987

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

TITLE PAGE

CONICAL PROJECTION ALGORITHMS FOR LINEAR PROGRAMMING

by

Clovis Gonzaga

Memorandum No. UCB/ERL M87/11

5 March 1987

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Conical Projection Algorithms For Linear Programming

Clovis Gonzaga

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, California 94720

ABSTRACT

The Linear Programming Problem is manipulated to be stated as a Non-Linear Programming Problem in which Karmarkar's logarithmic potential function is minimized in the positive cone generated by the original feasible set. The resulting problem is then solved by a master algorithm that iteratively rescales the problem and calls an internal unconstrained non-linear programming algorithm. Several different procedures for the internal algorithm are proposed, giving priority either to the reduction of the potential function or of the actual cost. We show that Karmarkar's algorithm is equivalent to this method in the special case in which the internal algorithm is reduced to a single steepest descent iteration. All variants of the new algorithm have the same complexity as Karmarkar's method, but the amount of computation is reduced by the fact that only one projection matrix must be calculated for each call of the internal algorithm.

Keywords: conical projection methods, Karmarkar's LP algorithm, linear programming, polynomial-time algorithms

November 20, 1986

On leave from the Dept. of Systems Engineering and Computer Science, COPPE, Federal University of Rio de Janeiro, Brasil.

Conical Projection Algorithms For Linear Programming

Clovis Gonzaga

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, California 94720

1. Introduction

The Linear Programming Algorithm created by Karmarkar [9] is based on the use of typical non-linear programming techniques, evolving by a sequence of line searches along internal feasible descent directions for his logarithmic potential function. It has been noticed [13],[3] that the method resembles barrier function methods and methods of centers, but the precise role of non-linear programming procedures in his approach has not yet been clearly stated. The nature of the projective transformation used in Karmarkar's algorithm is vaguely understood, as well as the role played by the unit simplex on which the action takes place.

In this paper we intend to provide answers to these questions, and show that the unit simplex is not needed at all. By isolating the utilization of non-linear programming procedures the path to improve the algorithm performance is opened, resulting in greater reductions of the objective function between computations of the projection matrix.

We shall present an algorithm with two levels of hierarchy, composed of a "master" algorithm that manipulates the Linear Programming Problem and calls an "internal" non-linear programming algorithm that provides a strategy for improving the current solution. The master algorithm re-scales the Linear Programming Problem by means of a linear transformation that places the current solution at the point $e = [1, 1, \ldots, 1]'$. An "internal problem" is then defined, consisting of a non-linear potential function to be minimized in a linear space with positivity constraints, and any non-linear programming algorithm can be used to reduce the value of its objective function. An approach similar to this was independently developed by Ye [14], allowing for multiple feasible descent iterations in a more structured internal problem.

In section 2 we state the linear programming problem and show how to manipulate it into a format that allows the application of projective techniques. The manipulation results in a problem that is equivalent to the original one in the strong sense of having the same feasible set, same dimension and cost. Section 3 describes the complete algorithm, section 4 shows how to generate lower bounds to the value of an optimal solution, and section 5 describes different options to the choice of descent directions for the internal algorithm. This choice can have two preferred goals: reducing the potential function or reducing the cost. We study both possibilities of priorizing one goal over the other, by generating steepest descent directions for the potential function subject to non positive variations of the cost, and steepest descent directions for the cost subject to a minimum descent for the potential. The first possibility was already studied by Anstreicher [2], with similar results.

Section 6 will discuss the convergence of the algorithms based on the first line-search performed by the internal algorithm in each iteration, with the following results: polynomial convergence is achieved if the feasible set is compact, reproducing Karmarkar's [9] result. The same polynomial bound is achieved by the other descent directions under a weaker condition, requiring compactness only for the optimal set.

2. The Linear Programming Problem

The algorithms described in this paper will deal with linear programming problems in the following special format:

$$\begin{aligned} &\textit{minimize} \quad c'x \\ &\textit{subject to} \quad A\,x = 0 \\ &\qquad\qquad a'x = 1 \\ &\qquad\qquad x \geq 0 \end{aligned} \tag{P}$$

where $n > m > 0$, $c, x \in R^n$, $a \in R^m$, A is an $m \times n$ matrix.

The following hypotheses must be satisfied:
An initial non optimal feasible solution $x^0 > 0$ is known as well as a lower bound v^0 for the value v of an optimal solution.

We shall define the following sets:

$$\begin{aligned} S &= \{x \in R^n \mid A\,x = 0\,, a'x = 1\,, x \geq 0\} \\ C &= \{x \in R^n \mid A\,x = 0\,, a'x > 0\,, x \geq 0\} \\ D &= \{x \in R^n \mid A\,x = 0\,, x > 0\} \end{aligned} \tag{1}$$

These three sets are respectively the feasible set for (P), the positive cone generated by it, and the subspace generated by S and the origin intercepted with the positive orthant. Conical projection methods will work in C. Since D is simpler than C, the following lemma will be useful in the future:

2.1 *Lemma:* If $a \geq 0$ or S is compact then $D \subset C$.
Proof: If $a \geq 0$, then for any $x > 0$, $a'x > 0$, since $a \neq 0$. This proves the first assertion.
By contradiction, assume that $x > 0$ is such that $A\,x = 0$ and $a'x = 0$. Since D is convex and $x^0 \in D$, then for any integer $i > 0$ the point $x^i = \frac{1}{i}x^0 + \frac{i-1}{i}x \in D$, and $a'x^i = \frac{a'x^0}{i} = \frac{1}{i}$.
Consequently, $i\,x^i \in S$, since $a'i\,x^i = 1\,, A\,x^i = 0$.
This contradicts the compactness assumption, since the sequence $(\|i\,x^i\|)$ is unbounded.

Problem manipulations

We now show that the formulation (P) is quite general, and completely equivalent to the usual format for the linear programming problem:

$$\begin{aligned} &\textit{minimize} \quad c'x \\ &\textit{subject to} \quad \bar{A}\,x = b \\ &\qquad\qquad x \geq 0 \end{aligned} \tag{P0}$$

Several methods have been proposed for the reduction of (P0) to (P), usually based on the addition of a compactifying constraint $e'x \leq M$, where $e = [\,1 \;\; 1 \cdots 1\,]'$, and M is a large number. We now show how the reduction can be made without changing neither the objective function nor the feasible set.

2.2 First manipulation: constraint reduction:

Let a_i', $i = 0, 1 \cdots m$ be the rows of $\bar{A}$, and assume (exchanging indices if necessary) that $b_0 \neq 0$. Define $a := \frac{a_0}{b_0}$. C can now be described by

$$\begin{aligned} a_i'x &= b_i \qquad i = 1,2,\cdots,m \\ a'x &= 1 \\ x &\geq 0 \end{aligned}$$

The constraints can be rewritten as $a_i'x = a'x\ b_i$, or finally,

$$\begin{aligned} (a_i - b_i a)'x &= 0 \qquad i = 1,2,\cdots,m \\ a'x &= 1 \\ x &\geq 0 \end{aligned}$$

The manipulation is completed by defining A with rows $(a_i - b_i a)'$. The following facts are evident:

i) The Null space of A is the subspace generated by C and the origin, and does not depend on the choice of a_0 among the rows with non-zero right-hand side.

ii) The dimension of $Null(A)$ is one unit higher then the dimension of $Null(\bar{A})$.

iii) The cone S generated by C defined in (1) does not depend on the choice of a_0.

iv) Although the vector a obviously depends on the choice of a_0, its projection a_p onto $Null(A)$ is the unique (up to a constant) direction in $Null(A)$ that is orthogonal to C. Consequently a_p does not depend on the choice of a_0

We conclude from these observations that there exists a set of equivalent manipulations (one for each choice of a_0) that produce a unique totally equivalent problem with the desired format. We stress the fact that the projected vector a_p is determined by the geometry of the problem and therefore unique.

2.3 Second manipulation: space augmentation:

A second manipulation into the format (P0) was proposed in [6], and also in [11], [2], obtained simply by introducing a new variable x_n into a problem originally defined in R^{n-1}. The new variable is constrained by $x_n = 1$, and it is enough to set $A := [\bar{A}\ \ -b\ ]$.

This reduction does change the original problem, but produces a new problem that is equivalent in the usual sense that optimal solutions to each of the problems are related by an isomorphism. We do not see any advantage in this reduction process, besides minor simplification in some proofs brought by the fact that now $a \geq 0$, and the property in lemma 2.1 is always trivially satisfied.

The Potential Function

We shall denote the value of an optimal solution by v and we shall make use of Karmarkar's logarithmic Potential Function $f(.)$ defined for all $x>0$ by

$$f(x) = n \log(c'x - v) - \sum_{j=1}^{n} \log x_j \tag{2}$$

The logarithmic form is used for convenience, since it leads to simpler expressions. Its multiplicative formulation

$$h(x) = \frac{(c'x - v)^n}{\prod_{j=1}^{n} x_j} \tag{3}$$

gives identical search directions in all algorithms, and can be used instead of $f(.)$. The function cannot be evaluated, since v is unknown. The algorithm uses instead lower bounds for v, calculated at each iteration. Given the lower bound u, the problem can be restated with the objective function $c'x - u$, and with a potential function defined for all $x \geq 0$:

$$f_u(x) = n \log(c'x - u) - \sum_{j=1}^{n} \log x_j$$

The objective function $c'x - u$ can be put in the standard format by using the following lemma:

2.4 *Lemma:* For all $x \varepsilon C$, $c'x - u = (c - ua)'x$.

Proof: Immediate, by noticing that for all $x \varepsilon C$, $ua'x = u$,since $a'x = 1$.

3. The algorithm

In this section we present the complete algorithm, structured in two levels: the master algorithm and the internal algorithm.

The master algorithm

The algorithm will generate a sequence of points (x^k) and an increasing sequence of lower bounds (v^k) for v , in such a way as to guarantee a substantial decrease in $f(.)$ in each iteration.

Each iteration of the master algorithm will perform the following operations:

- Scaling: a linear transformation brings the current solution x^k to the point $e = (1,1,..1)$.
- Projection: calculate the projection matrix onto the new linear space. This time-consuming operation is done only once for each iteration of the master algorithm.
- Calculation of a new lower bound to v .
- Definition of the internal problem, computation of a solution with a low value and then back to the original problem.

The internal algorithm makes use of the number α , to be defined in (6.1.4), and in this section we shall assume that the potential function defined in the internal problem can be reduced by α in each iteration. The proof of this fact will be the object of sections 5 and 6 .

3.1 *Algorithm*: Given a precision $\varepsilon > 0$, a lower bound $v^0 \leq v$ and $x^0 \varepsilon C$, $x^0 > 0$.

```
Set $k := 0$.
While $c'x^k - v^k > \varepsilon$ do
    Define $D := diag(x_1^k, x_2^k, \ldots, x_n^k)$.
    (scaling) Define $A^k := AD$, $a^k := Da$, $c^k := Dc$.
    (projection) Calculate the projection matrix $P$ onto $Null(A^k)$.
    Set $c_p := Pc^k$, $a_p := Pa^k$.
    (internal algorithm)
        Calculate a lower bound $u$ such that $v^k \leq u \leq v$.
        Set $\bar{c} := c_p - ua_p$.
        Define the Internal Problem
            minimize $g(y) := n \log \bar{c}'y - \sum_{j=1}^{n} \log y_j$          (Pi)
            subject to $A^k y = 0$
                       $y \geq 0$
                       $a^{k\prime} y > 0$
        Use a non-linear programming algorithm to find a feasible point $y^*$ for (Pi) such that
        $g(y^*) < g(e) - \alpha$.
        (conical projection) calculate
            $\bar{y} := \frac{1}{a^{k\prime} y^*} y^*$          (4)
    (back to the original space) Set
      $x^{k+1} := D\bar{y}$
      $v^{k+1} := u$
        $k := k + 1$
```

The internal problem

To each internal problem (Pi) we shall associate the sets S^k, C^k, D^k, defined as in (1), corresponding to the feasible set, the cone and subspace generated by it and the origin.

The internal problem (Pi) is not well defined, since for y near the origin, $g(y)$ can assume any value. We shall nevertheless keep this formulation with the understanding that the desired result is a truncated sequence of points with decreasing values of the objective function.

The feasible set for (Pi) is the cone C^k. This set has the complicating constraint $a'x > 0$, but it will be irrelevant whenever we can guarantee that $D^k \subset C^k$, like in the compact case or as a result of the second manipulation (see lemma 2.1). In this section we shall assume that, at least in the first iteration of the internal algorithm, the search directions will not cross the null space of a' in the first orthant. We stress the fact that this is guaranteed in the two cases mentioned above, and we shall prove that this is also true in the non-compact cases to be studied in section 5.

The actual feasible set for the internal problem is then D^k, simply the intersection of $Null(A^k)$ and the interior of the first orthant. C^k (or D^k) is interesting for two reasons: first, because it is easy to use search methods on it, since the projection matrix is known; secondly, because it is a cone with the property that the potential function is constant on each of its rays, as we formalize in the following lemma:

3.2 *Lemma*: For all $y \varepsilon C^k$, for any $\lambda > 0$,

$$g(\lambda y) = g(y) \ .$$

Proof: by direct substitution in the definition of $g(.)$.

Consequently, to each feasible point $x > 0$ for (P), a ray $\{\lambda D^{-1}x \mid \lambda > 0\}$ of constant potential in (Pi) is associated. Conversely, given a point $y^* \varepsilon C^k$ resulting from the internal algorithm, the point $\bar{y} \varepsilon C^k$ given by the expression (4) satisfies $g(\bar{y}) = g(y^*)$ and $a^{k\prime}\bar{y} = 1$, as can be trivially verified. $\bar{y}$ is the conical projection of y^* onto S^k , i.e. the intersection of the ray through y^* and $e + Null(a^{k\prime})$. The existence of the conical projection is guaranteed by $y^* > 0$ and the assumption that $a^{k\prime}y > 0$ discussed above. The subject of conical projections will be further studied in section 5.1 .

If $v^k = v$, then the internal problem is equivalent to problem (P), and the problem can in principle be solved by a single application of a good non-linear programming algorithm. The potential function has a bizarre behavior near the boundaries of the orthant: it tends to $+\infty$ near any non-optimal boundary point $x \neq 0$, to $-\infty$ for a sequence convergent to an optimal solution, and to any number for sequences convergent to 0 . This can lead to the conclusion that the potential function has an ill-conditioned Hessian matrix near an optimal solution, but it is not necessarily the case

Iri [7] shows that when restricted to S^k and under suitable compactness conditions, the potential function in form (3) is strictly convex and well conditioned on S^k . This reference shows that a Newton method to minimize $h(x)$ on S has asymptotic superlinear convergence, and discusses several examples.

An affine Newton method on S is scale invariant, thus making irrelevant the master algorithm, but it is expensive and has no guarantee of descent in the beginning iterations (only asymptotic convergence is assured). The use of conical projections and scaling associated to an internal algorithm with good initial behavior (like conjugate gradients) remedy these problems. not guaranteed near optimal solutions. Such non-linear programming methods can be very effective in reducing the value of the potential function while far from the boundary of C^k , and this can be done at a low cost since no projection matrices must be calculated while iterating on the internal problem.

The internal algorithm iterates while a substantial decrease in the potential function is obtained. It will be shown that if the problem is compact and its first iteration is a steepest descent search, then in this iteration the potential function $g(.)$ drops by at least a fixed constant $\alpha > 0$. This choice for the first iteration is actually equivalent to the line search done in Karmarkar's method, and α is the constant found in his work [9].

Before we state the internal algorithm, the following lemma resumes useful properties of projections and scalings.

3.3 *Lemma*: Let y and x be related by $Dy = x$ in an iteration k of the algorithm. If $A^k y = 0$ then $c_p{}'y = c'x$, $a_p{}'y = a^{k\prime}y = a'x$. Furthermore, if $a^{k\prime}y = 1$ then $\bar{c}'y = c'x - u$.

Proof: Immediate consequences of the definitions of a_p and c_p , since for any vector $z \varepsilon R^n$, for $y \varepsilon Null(A^k)$, $z'y = (Pz)'y$. The last equality is a consequence of lemma 2.4 .

The Internal Algorithm

The Internal Algorithm performs three operations: calculation of a new lower bound, calculation of a sub-optimal solution for the Internal Problem defined in the master algorithm 3.1, and conical projection. The calculation of a new lower bound is done by a simple algorithm to be presented now; the proofs and motivation will be the subject of section 4. In all the development to follow we use the notation introduced in the master algorithm.

3.4 *Algorithm*: Calculation of a new lower bound.

```
    If $c_p - v^k a_p$ has a non-positive component, then $u := v^k$
    Else $u := \min_{j=1,2,\ldots,n} \{ \frac{c_{p_j}}{a_{p_j}} \mid a_{p_j} > 0 \}$
```

As we shall prove in the next section, this algorithm finds a lower bound for v, and guarantees that the cost vector $\bar{c}$ used by the internal algorithm has at least one non-positive component. An equivalent result can be found in [13], obtained by a different approach.

3.5 *Algorithm*: (model) given a precision $\delta > 0$

```
$i := 0$ ; $y^0 := e$
Repeat
    Calculate $P \nabla g(y^i) := \frac{n}{\bar{c}' y^i} \bar{c} - P [ (y_1^i)^{-1} (y_2^i)^{-1} \cdots (y_n^i)^{-1} ]'$
    Choose a descent direction $h \in Null(A^k)$ such that $h' P \nabla g(y^i) < 0$ .
    Find $\bar{\lambda} > 0$ such that $g(y^i + \bar{\lambda} h) = \min \{ g(y^i + \lambda h) \mid \lambda > 0 , y^i + \lambda h > 0 \}$
    $y^{i+1} := y^i + \bar{\lambda} h$
    $i := i + 1$
Until $g(y^{i-1}) - g(y^i) < \delta$
$y^* := y^i$
```

The algorithm above is no more than the general model for a non-linear programming feasible directions search, for a special case in which the result of the line searches is guaranteed to lie in the interior of the feasible set. The choice of the descent directions was not specified and can be the result of any strategy like conjugate gradients or variable metric methods [5]. The line search is well defined, since the potential function grows indefinitely near the frontier of the feasible set (except for the lucky case of hitting an optimal solution for $u = v$). It has been proved in [13] that the potential function restricted to the line $y^i + \lambda h$ is unimodal, and reference [10] gives a method for the search.

We shall then assume hereafter that the line search in 3.5 always has a unique solution $\bar{\lambda}$.

4. Determination of lower bounds for the value of an optimal solution

In the beginning of iteration k of the master algorithm, a feasible solution x^k is known. A linear transformation on the original problem (P) brings this point to the vector e in the new coordinates. Using the notation introduced in Algorithm 3.1, a Linear Programming Problem can be written in the new coordinates:

$$\begin{aligned} &\textit{minimize} \quad c_p' y && \\ &\textit{subject to} \quad A^k y = 0 \\ &\qquad\qquad\quad a_p' y = 1 \\ &\qquad\qquad\quad\ \ y \geq 0 \end{aligned} \qquad \text{(Pk)}$$

This problem is equivalent to (P), in the sense that each feasible point x for (P) is univocally associated to a point $y = D^{-1} x$ feasible for (Pk), and their costs are related by $c_p' y = c' x$, as was shown in lemma 3.3. The value of an optimal solution for (Pk) is then equal to the value v of an optimal solution for (P).

In the beginning of iteration k a value $v^k \le v$ is known. If $\bar{c} = c_p - v^k a_p$ has a non-positive component, then a reduction of at least α for the potential function will be guaranteed by the results in section 6. If this is not the case, a new lower bound for v must be found, so that the new $\bar{c}$ has a non-positive component: in this section we show that this is always possible, and it is accomplished by algorithm 3.4, repeated below:

Algorithm: Calculation of a new lower bound.

If $c_p - v^k a_p$ has a non-positive component, then $u := v^k$

$$\text{Else } u := \min_{j=1,2,\ldots,n} \left\{ \frac{c_{p_j}}{a_{p_j}} \mid a_{p_j} > 0 \right\} \tag{5}$$

4.1 *Theorem*: Algorithm 3.4 generates a lower bound u for v, and $c_p - u a_p$ has a non-positive component.

Proof: If $c_p - v^k a_p$ has a non-positive component, then the result is trivial. Suppose then that $c_p - v^k a_p > 0$.

Consider the following relaxed version of (Pk):

$$\begin{aligned} &\textit{minimize} \quad c_p{}'y \\ &\textit{subject to} \quad a_p{}'y = 1 \\ &\qquad\qquad\qquad y \ge 0 \end{aligned} \tag{6}$$

The dual of this Linear Programming problem is given by:

$$\begin{aligned} &\textit{maximize} \quad z \qquad\qquad , \quad z \in R \\ &\textit{subject to} \quad z\, a_p \le c_p \end{aligned} \tag{7}$$

By hypothesis, $v^k a_p < c_p$, and consequently (7) is feasible. It follows that (6) has an optimal solution y^*, (7) has an optimal solution $u > v^k$, and $u = c_p{}'y^*$.

Since (6) was obtained by relaxing (Pk), u is a lower bound for v.

(7) can be rewritten as

$$\begin{aligned} &\textit{maximize} \quad z \\ &\textit{subject to} \quad z \le \frac{c_{p_j}}{a_{p_j}} \qquad j = 1,\ldots,n \qquad \textit{such that} \quad a_{p_j} > 0 \end{aligned}$$

The solution for this problem is given by (5), and for some index k, $u = \dfrac{c_{p_k}}{a_{p_k}}$. It follows that $c_{p_k} - u\, a_{p_k} = 0$, completing the proof.

4.2 *Remark*: The following consequence of the definition of u will be useful later: for any $z \ge u$, $c_p - z\, a_p$ has a non-positive component.

5. Descent directions for the internal algorithm

We now describe several possible choices for the descent directions used by the internal algorithm. This section will be dedicated to the description of the procedures, and the next one will concentrate all convergence proofs for these procedures. We begin by showing that due to the homogeneity of $g(.)$, each possible direction belongs to a class of equivalent directions that lead to the same reduction in $g(.)$.

5.1. Equivalent Directions

Each iteration of the internal algorithm starts with a point y^i and performs a line search along a direction h, resulting in a point $y^{i+1} = y^i + \overline{\lambda} h$. We will show that due to the 0-degree homogeneity of $g(.)$, there exists a cone of directions leading to the same decrease in $g(.)$ as h, and the minimizers for all these directions lie on the same ray.

Consider a set T in the first orthant of R^n. The *cone generated by* T is defined as $K(T) = \{\alpha y \varepsilon R^n \mid \alpha > 0, y \varepsilon T\}$.

Since the potential function $g(.)$ is 0-degree homogeneous, i.e, $g(\lambda y) = g(y)$ for any $\lambda > 0$, it follows that

$$\inf_{y \varepsilon T} g(y) = \inf_{y \varepsilon K(T)} g(y)$$

If two sets T, T_1 in the first orthant generate the same cone, then $\inf_{y \varepsilon T} g(y) = \inf_{y \varepsilon T_1} g(y)$.

This shows that the minimization of $g(.)$ along two different lines lead to the same optimal value whenever both lines generate the same cone.

We shall say that two directions h^1, h^2 are *equivalent from* $y > 0$, whenever minimizers y^1, y^2 of $g(.)$ respectively along h^1 and h^2 in the first orthant exist and lie on the same ray (and consequently $g(y^1) = g(y^2)$). Some facts are straightforward:

5.1.1 *Lemma*: Let $\overline{h}$ be such that $y^i + \overline{h} > 0$ in some iteration of the internal algorithm, and that the line search from y^i results in $\overline{\lambda} < 1$. Then for any $\alpha > 0$, the direction $h = \alpha(y^i + \overline{h}) - y^i$ is equivalent to $\overline{h}$ from y^i.

Proof: It is sufficient to see that the line segments $\{y^i + \lambda \overline{h} \mid \lambda \varepsilon [0,1]\}$ and $\{y^i + \lambda h \mid \lambda \varepsilon [0,1]\}$ generate the same cone, defined by the extreme rays $\{\delta y^i \mid \delta > 0\}$ and $\{\delta(y^i + h) \mid \delta > 0\}$.

5.1.2 *Lemma*: If two directions are equivalent from $y > 0$, then positive multiples of these directions are also equivalent from y.

Proof: Straightforward, since scaling the directions does not change the result of the line search.

5.1.3 *Lemma*: Let $\overline{h}$ be as in lemma 5.1.1. Then for any $\mu \varepsilon (-\infty, 1)$, the direction $h = \overline{h} + \mu y^i$ is equivalent to $\overline{h}$ from y^i.

Proof: By lemma 5.1.1, for any $\alpha > 0$ the direction $\alpha \overline{h} + (\alpha - 1) y^i$ is equivalent to $\overline{h}$ from y^i. By lemma 5.1.2, we can multiply this direction by α^{-1}, obtaining $h = \overline{h} + (1 - \frac{1}{\alpha}) y^i$, $\alpha > 0$. Setting $\mu = 1 - \frac{1}{\alpha}$, the directions are defined by $h = \overline{h} + \mu y^i$, $\mu \varepsilon (-\infty, 1)$, completing the proof.

In particular, given any direction h from $y \varepsilon S^k$ such that $y + h \varepsilon C^k$, its conical projection $K_y(h)$ onto the feasible set S^k is given by

$$K_y(h) = \frac{y+h}{a^{k\prime}(y+h)} - y = \frac{1}{1+a^{k\prime}h}\,(h - a^{k\prime}h\,y) \tag{8}$$

This direction is of special interest: it is a search direction in S^k equivalent to h, and the points $y+h$ and $y+K_y(h)$ lie on the same ray.

5.2. Steepest Descent Directions : Karmarkar's algorithm

The first natural choice for the descent directions is given by the projected gradient direction,

$$h = -P\,\nabla g(y^i) = -\frac{n}{\bar{c}'y^i}\,\bar{c} + P\,[(y_j^i)^{-1}] \tag{9}$$

In particular, the first iteration uses (since $P\,e = e$)

$$h^0 = -\frac{n}{w}\,\bar{c} + e \qquad , \qquad \text{with } w := \bar{c}'e \ .$$

The line defined by h^0 lies on the unit simplex, defined by the equation $e'x = n$. This is an immediate consequence of the zero degree homogeneity of $g(.)$: since $g(\lambda e)$ is constant for $\lambda > 0$, necessarily $e'\nabla g(e) = 0$.

By lemma 5.1.3, in the first iteration, any direction of the form

$$-\frac{n}{\bar{c}'e}\,\bar{c} + e + \mu e = -\frac{n}{\bar{c}'e}\,\bar{c} + \nu e \qquad , \quad \nu \in (-\infty, 2) \tag{10}$$

is equivalent to h^0.

This set of directions includes for instance $h = -\bar{c}$, which consequently is as good as the gradient direction as a search direction.

h^0 is actually the Karmarkar direction, with well known properties that will be discussed in section 6 . The fact that it lies on the unit simplex seems to attach a great importance to this set. In reality the properties of h^0 are by no means connected to the simplex, since by (10) there is an infinite set of directions that share the same properties. The only advantage that seems to arise from the fact that h^0 lies on the unit simplex is in the ease with which the guaranteed decay of $g(.)$ is proved: direct proofs for other equivalent directions may not be so elegant. The most important directions equivalent to the projected gradient for the first iteration are summarized in the following lemma:

5.2.1 *Lemma*: The following directions are equivalent in the first iteration of the internal algorithm:

(i) $\quad -\nabla g(e) = -\frac{n}{w}\,\bar{c} + e$

(ii) $\quad -\bar{c}$

(iii) $\quad -\bar{c} + a^{k\prime}\bar{c}\,e \qquad$ (conical projection onto S^k)

Furthermore, $\nabla g(e)$ is proportional to the conical projection of $-\bar{c}$ onto the unit simplex, defined by $\frac{e'y}{n} = 1$.

Proof: (i) to (iii) are direct consequences of (9) and (10). To prove the last assertion, use (8) with $a^k = \frac{e}{n}$

$$K_{e/n}(-\bar{c}) \approx -\bar{c} - \frac{e'}{n}(-\bar{c})\,e = -\bar{c} + \frac{w}{n}\,e \qquad , \text{ proportional (i)} .$$

The discussion above is useful for two reasons: first because it indicates that no improvement in the convergence rates can be obtained by trying other directions generated by linear combinations of

$\bar{c}$ and e (or h and $[(y_j^i)^{-1}]$ in the general case); the second reason is more appealing: the line searches can be executed in the original space, by projecting the search direction *conically* onto S^k, by the expression in lemma 5.2.1 .

5.3. Directions of non-increasing costs

In each iteration of the internal algorithm the function $g(.)$ is reduced by a search along a direction h . The point resulting from each search corresponds to a vector in S^k with identical value of $g(.)$ by conical projection, and there is no guarantee that the cost decreases from the first to the second point in S^k . Monotonically decreasing algorithms were studied in [11] and [2], and we now show a procedure that is closely related to the results in this last reference.

Assume that $y^i \varepsilon S^k$ (otherwise, set $y^i := \frac{y^i}{a^{k\prime} y^i}$), let h^0 be the steepest descent direction (9). Its conical projection onto S^k is proportional to

$$\bar{h} = h^0 - a_p' h^0 y^i \qquad \text{, by (8) ,} \tag{11}$$

and the variation in cost in this direction is given by

$$\bar{c}'\bar{h} = (\bar{c} - w^i a_p)' h^0 \qquad , \qquad \text{with} \quad w^i = \bar{c}' y^i \tag{12}$$

This expression defines a hyperplane

$$Q = \{ h \mid d'h = 0 \} \quad \text{, where} \tag{13}$$
$$d = \bar{c} - w^i a_p \quad .$$

To guarantee non-increasing costs, it is sufficient to choose the descent direction by the procedure:

5.3.1. Search direction for non-decreasing costs:

Set $h^0 = -P\, \nabla g(y^i)$ (or an equivalent direction).
If $(\bar{c} - w^i a_p)' h^0 \le 0$ then $h^1 := h^0$
Else set h^1 equal to the (orthogonal) projection of h^0 onto Q .

Notice that this is simply a projection onto a hyperplane with known normal d , with little computation involved. To show that the direction is well defined, we must show that $y^i + \lambda h^1$ does not leave C^k in the first orthant. Two things must be proved: $A^k h^1 = 0$ and $a^{k\prime}(y^i + \lambda h^1) > 0$ for all $\lambda > 0$ such that $y^i + \lambda h^1 > 0$. The first one is easy: the projection onto Q has the form

$$h^1 = h^0 - \gamma(\bar{c} - w^i a_p) \tag{14}$$

Since all vectors involved in (14) are in the null space of A^k , then so is h^1 . The second one is not immediate, and the cases in which it is true depend on the following lemma:

5.3.2 *Lemma*: Let h be a direction such that $y^i + h \,\varepsilon\, D^k$ and assume that one of the following conditions is satisfied:
(i) S^k *is compact.*
(ii) $d'h < 0$
(iii) $d'h \le 0$ *and the optimal set for (P) is compact*. Then $y^i + h \,\varepsilon\, C^k$.

Proof:
By contradiction to all cases , suppose that $a^{k\prime}(y^i + h) \le 0$.
Since $a^{k\prime}(y^i + \lambda h) = 1$ for $\lambda = 0$, there must exist a $\hat{\lambda}$ such that $a^{k\prime}(y^i + \hat{\lambda} h) = 0$.
Let $\bar{h} = h - a_p' h\, y^i$ be the direction on S^k constructed as in (11) . Then $\bar{h}$ (conical projection of h) is equivalent to h . To each point $y = y^i + \lambda h$ with $\lambda \,\varepsilon\, [0, \hat{\lambda})$ corresponds a point $\bar{y} = K(y) = \frac{y}{a^{k\prime} y}$ on the line generated by $\hat{h}$. It follows that when $\lambda \to \hat{\lambda}$, $\| K(y^i + \lambda h) \to \infty$
We are ready to examine each alternative hypothesis:

(i) Ready, since the argument above contradicts the compactness of S^k .
(ii) If $(\bar{c}-w^i a_p)'h < 0$ then by (12) $\bar{c}'\bar{h} < 0$, and the problem is unbounded, contradicting the general hypotheses for (P).
(iii) If $(\bar{c}-w^i a_p)'h \le 0$ then the level set $\{ y \in S^k \mid \bar{c}'y \le w^i \}$ is unbounded. By a known result in Convex Analysis [12] , if a level set in a convex programming problem is unbounded then so are all non-empty level sets. In particular, the optimal set $\{ y \in S^k \mid \bar{c}'y \le \bar{c}\hat{y} \}$ must be unbounded, where $\hat{y}$ is an optimal solution to (Pi).
This completes the proof that $y^i + h \in C^k$ with contradictions established for all cases.

It is now clear that $y^i + \lambda h$ does not cross the boundary of c^k in the first orthant. If conditions (i) or (iii) in lemma 5.3.2 is satisfied, then there are only two possible outcomes for the line search along h : either the potential function grows indefinitely when the boundary of C^k is approached and there exists a unique minimizer $\bar{\lambda}$, or the search finds a global optimum for (P), and the potential function decreases indefinitely.

Returning to the study of h^1 , it follows from lemma 5.3.2 that if the optimal set is bounded, then $a^{k\prime}(y^i + \lambda h^1) > 0$ for all $\lambda > 0$ such that $y^i + \lambda h^1 > 0$, and we conclude that in this case the search direction is well defined and the line search in (Pi) has a unique solution.

In the next section we shall prove that the resulting direction has the same properties as h^0 with respect to the decrease in the potential function, and this completes our derivation. It is nevertheless interesting to compute an explicit formula for h^1 in the first iteration of the internal algorithm, since it has a nice interpretation.

Explicit computation of h^1

The first thing to notice is that any among the equivalent directions to h^0 can be projected onto Q instead of h^0 . In particular, projecting $-\bar{c}$ produces simpler formulas.
Let $h = -\bar{c}$, and assume that $h'd > 0$. Then

$$h^1 = -\bar{c} - \gamma d \quad , \text{ where } \gamma = \frac{-\bar{c}'d}{d'd} > 0$$

$$h^1 = -(1+\gamma)\bar{c} + \gamma w\, a_p \quad , \text{ or}$$

$$\frac{1}{1+\gamma} h^1 = -(\bar{c} - z\, a_p) \quad , \text{ with } \quad z = \frac{\gamma}{1+\gamma} < 1$$

Substituting the value of γ in the expression above and simplifying, we get

$$z = \frac{\bar{c}'d}{a_p'd} = \frac{\bar{c}'(\bar{c} - wa_p)}{a_p'(\bar{c} - wa_p)} \quad , \quad 0 < z < w \tag{15}$$

This result is summarized in the following lemma:

5.3.3 *Lemma*: In the first iteration of the internal algorithm, if the gradient direction leads to increasing costs in S^k , then the following directions are equivalent , guarantee non-increasing costs and have the same convergence properties as $-\nabla g(e)$:
(i) $-c_z := -(\bar{c} - z\, a_p)$, with z given by (15).
(ii) $-\frac{n}{c_z'e}\, c_z + e$
(iii) The projection of $-\nabla g(e)$ onto Q
Furthermore, directions (ii) and (iii) are proportional to the conical projection of direction (i) onto the unit simplex.

Proof: The equivalence between (i) and (ii) stems directly from lemma 5.2.1, with c_z substituting $\bar{c}$. To see that (iii) is proportional to (ii), compute the conical projection $\bar{h}$ of h^1 , proportional to $-c_z$, onto the unit simplex:

$$\bar{h} = -\bar{c} - \gamma d - \frac{e'}{n}(-\bar{c} - \gamma d)\, e$$

But $e'd = e'\bar{c} - e'a_p w = 0$, since $e'a_p = 1$ and $e'\bar{c} = w$. Consequently,

$$\bar{h} = -\bar{c} - \gamma d + \frac{w}{n} e$$

It follows that $\bar{h} = h^1 + \frac{w}{n} e$. is in Q , since $d'e = 0$

Since this last expression can be regrouped as $\bar{h} = (-\bar{c} + \frac{w}{n} e) - \gamma d$, it must be the projection of $-\bar{c} + \frac{w}{n} e = \frac{w}{n}(-\nabla g(e))$ onto Q .

Our final conclusion is that to achieve non-increasing costs, the direction to be followed must be equal to the steepest descent direction that would be associated to the cost

$$c_z = \bar{c} - z\, a_p = Pc^k - (u+z) a_p$$

This can also be interpreted as the steepest descent direction associated to an estimate $u+z$ for the lower bound to v in iteration k of the master algorithm.

This result is closely related to the result obtained in [2], and expression (15) relaxes a similar formula presented in that reference.

We are now ready to prove important properties of h^0 and h^1 for the first iteration of the internal algorithm:

5.3.4 *Lemma*: Let $h^0 = -\nabla g(e)$ and consider the direction h^1 defined in 5.3.1 . Then $\|h^0\| \geq 1$, $\|h^1\| \geq 1$, and $h^0 \frac{h^1}{\|h^1\|} \geq 1$.

Proof: Initially, note that by lemmas 5.2.1 and 5.3.3, both h^0 and h^1 have the form

$$-\frac{n}{c_z' e}\, c_z + e \quad , \text{ with } \quad z \geq u \tag{16}$$

By remark 4.2, c_z has a non-positive component, and consequently (16) has a component greater than 1 . This implies in $\|h^0\| \geq 1$ and $\|h^1\| \geq 1$.
Since h^1 is the projection of h^0 onto Q , $h^{0\prime} h^1 = \|h^1\|^2$.
Consequently, $h^{0\prime} \frac{h^1}{\|h^1\|} = \|h_1\| \geq 1$.

5.4. Strictly decreasing costs

The former approach can only guarantee non-decreasing costs, and will actually generate directions of constant cost whenever $d'h^0 \geq 0$. In this case, if the optimal set is unbounded (or very large), the potential function can be unbounded (or decrease very much) along h^1 , without any improvement in cost.

Trying to decrease the cost instead of the potential function can, on the other hand, lead to directions that approach the frontier of S^k too fast, with slow convergence. We now present what seems to be the best of two worlds: a direction of steepest descent for the cost subject to a minimum guaranteed decrease for the potential function.

Steepest descent direction for the cost

Given any point $y \varepsilon C^k$, the cost associated to the corresponding point on S^k, $K(y) = \frac{y}{a_p'y} \varepsilon S^k$ is given by $p(y) := \frac{\bar{c}'y}{a_p'y}$. The steepest descent direction in C^k from a point $y^i \varepsilon S^k$ is given by the gradient of this function:

$$\nabla p(y^i) = \frac{1}{(a_p'y)^2} (a_p'y\bar{c} - \bar{c}'y\, a_p) \approx \bar{c} - w^i a_p = d \tag{17}$$

We conclude that $-d$ is the steepest descent direction for the cost in C^k.

Notice that this is in general not equivalent to the orthogonal projection of $-\bar{c}$ onto S^k. The projected cost gives the steepest descent direction among the directions in the intersection of S^k and a ball B centered in e ; $-d$ is the best among the directions in a larger region $B \cap C^k$, and thus is more promising as a descent direction.

A simple procedure to decrease the cost while assuring good convergence properties is obtained by the following algorithm:

```
5.4.1: Search direction for decreasing costs:
    Set $h^0 := -P\nabla g(y^i)$ , $d := \bar{c} - w^i a_p$
    If $h^0 \frac{d}{\|d\|} \le -0.5$ then $h^2 := -d$
    Else set
        $\lambda := \max\{\nu > 0 \mid -h^{0\prime} \frac{h^0 - \nu d}{\|h^0 - \nu d\|} \le -0.5\}$
        $h^2 := h^0 - \lambda d$
```

To show that the algorithm is well defined, use lemma 5.3.4 : $\|h^0\| \ge 1$ and $-h^{0\prime} \frac{h^1}{\|h^1\|} \le -1$. It is then immediate that the value of λ is always greater than the value of the same variable corresponding to h^1 (14). This means that whenever h^0 leads to an increase in cost we now cross the hyperplane Q and plunge as deep as possible while keeping a substantial decrease in $g(.)$, by assuring $-h^{0\prime} \frac{h^2}{\|h^2\|} \le -0.5$ (see 6.1.4 in the next section).

The direction h^2 is then always a direction of strict decrease in cost, and by lemma 5.3.2 it is always feasible in D^k. It corresponds to the direction of steepest descent for the cost, subject to a guaranteed substantial decrease for the potential function.

6. Convergence of the algorithms

Our aim is to prove that all the algorithms proposed are linearly convergent in the following sense: the sequence of values $c'x^k - v$ generated by any of the algorithms is dominated by a sequence that decreases by a constant ratio at each iteration. The sequence $(c'x^k)$ is not necessarily monotonically decreasing, and its values can increase in the beginning or when the lower bound v^k changes.

This will be done in two steps: we first show what behavior can be expected in each iteration of the algorithms, and then use these results to prove the overall convergence properties.

6.1. Variation of the potential function in the internal algorithm

We shall consider only the first iteration of the internal algorithm, since afterwards $g(.)$ can only decrease. The most important properties of the potential function were shown in Karmarkar [9] , and will be reviewed and extended here. The first lemma resumes properties of the logarithm function.

6.1.1 *Lemma* (Karmarkar [9]):
a) The logarithm function $x \varepsilon R^+ \to \log x$ is strictly concave and differentiable, and for any λ such that $|\lambda| < 1$,

$$\log(1+\lambda) = \lambda - o(\lambda) \ , \tag{18}$$

where $o(\lambda) \le \frac{\lambda^2}{2}\frac{1}{1-|\lambda|}$, $o(\lambda) \ge 0$.
b) Given a direction $h \ \varepsilon \ R^n$ such that $\|h\| = 1$, the function

$$x > 0 \ \to \ g_2(x) := \sum_{j=1}^{n} \log x_j \tag{19}$$

satisfies (with $o(.)$ defined above)

$$g_2(e+\lambda h) = \lambda \nabla g_2(e)'h + o(\lambda) = \lambda e'h - o(\lambda)$$

Proof:
The results were proved in [9], in different formulations. (b) is obtained directly from (a) by performing the summation and using:

$$\sum_{j=1}^{n} o(\lambda h_j) \le \sum_{j=1}^{n} \frac{(\lambda h_j)^2}{2}\frac{1}{1-|\lambda h_j|}$$

The fact that $|\lambda h_j| \le |\lambda| \, \|h\|$ for $j = 1, \cdots ,n$ leads immediately to the expression in (a).

6.1.2 *Lemma*: Consider a direction $h \ \varepsilon \ R^n$ such that $\|h\| = 1$ and $\lambda \ \varepsilon \ [0,1)$. Then

$$g(e+\lambda h) - g(e) \le \lambda \nabla g(e)'h + \frac{\lambda^2}{2}\frac{1}{1-|\lambda|} \tag{20}$$

Proof: Let $g(y) = g_1(y) - g_2(y)$, where $g_1(y) := n \log \bar{c}'y$, $g_2(y) := \sum_{j=1}^{n} \log y_j$.
Since g_1 is concave,

$$g_1(e+\lambda h) - g_1(e) \le \lambda \nabla g_1(e)'h$$

Using lemma 6.1.1 ,

$$-(g_2(e+\lambda h) - g_2(e)) \le -\lambda \nabla g_2(e)'h + \frac{\lambda^2}{2}\frac{1}{1-|\lambda|}$$

The proof is completed by adding the two last inequalities.

Consider now a direction h , and assume that the line search along $e+\lambda h$ has a unique solution $\bar{\lambda}$. Since the right hand side of (20) depends only on the value of $\nabla g(e)'h$, the following is true:

6.1.3 *Lemma*: Given any fixed number $\mu < 0$, there exists a fixed number $\alpha > 0$ such that if $\nabla g(e)' \frac{h}{\|h\|} \le \mu$ then $g(e+\bar{\lambda} h) \le g(e) - \alpha$.
In particular, for $\mu = -1$, $\alpha = 0.26$ satisfies the inequality; for $\mu = -0.5$, $\alpha = 0.086$.

The lemma above shows that a fixed descent in the potential function is achieved in the first iteration of each application of the internal algorithm whenever the search direction has $\nabla g(e)'h$ bounded away from 0 by a negative constant μ . (we then say that " h has a guaranteed descent of α ") We are now ready to show that this is true for the directions h^0 , h^1 , h^2 defined in section 5 , under suitable compactness conditions.

6.1.4 *Theorem*: Let $h^0 = -\nabla g(e)$ and consider the directions h^1 and h^2, defined respectively in 5.3.1 and 5.4.1 . There exists a fixed number $\alpha \geq 0.086$ such that:
(i) If the feasible set S for (P) is compact, then h^0 has a guaranteed descent of α .
(ii)If the optimal set for (P) is compact, then both h^1 and h^2 have a guaranteed descent of α .

Proof: The compactness conditions in both cases guarantee that the line search has a unique solution, as was shown in lemma 5.3.3. To prove the hypotheses in lemma 6.1.3 , we shall use lemma 5.3.4 :
(i) $-h^{0\prime}\dfrac{h^0}{\|h^0\|} = -\|h^0\| \leq -1$.
(ii) $-h^{0\prime}\dfrac{h^1}{\|h^1\|} < -1$, by lemma 5.3.4 . By construction, $-h^{0\prime}\dfrac{h^2}{\|h^2\|} < -0.5$.
We conclude that all directions in consideration satisfy the hypotheses of lemma 6.1.3 with $\mu \leq -0.5$, completing the proof.

6.2. Proof of Linear Convergence

We begin by stating without demonstration two well known lemmas. Proofs can be found in [9] and [6].

6.2.1 *Lemma*: For any $x \varepsilon C$, for $r \leq s \leq v$, $f_s(x) \leq f_r(x)$.

6.2.2 *Lemma*: At any iteration k , if $g(y^*) \leq g(e) - \alpha$ then

$$f_{v_{k+1}}(x^{k+1}) \leq f_{v_k}(x^k) - \alpha \quad . \tag{21}$$

This lemma associated to theorem 6.1.4 show that under the compactness conditions in that theorem all our descent directions generate sequences $(f_{v_k}(x^k))$ that decrease at each iteration by at least α . The sequence $(f(x^k))$ is dominated by it, as a direct consequence of lemma 6.2.1. It is generally not true that the objective function of (P) decreases at each iteration, but the following results guarantee the overall convergence.

First case: S compact

Let

$$\gamma := \max \left\{ \sum_{j=1}^{n} \log x_j \mid x \varepsilon S \, , x > 0 \right\} \quad . \tag{22}$$

γ is well defined, since S is compact, the argument is continuous in its interior and decreases indefinitely near its frontier. If x^* is a maximizer of this expression, then it can be thought of as a "point of minimum potential in S ".
Define $K = \exp\left(\dfrac{f_{v_0}(x^0)+\gamma}{n}\right)$.

6.2.3 *Theorem*: At any iteration k of the algorithm 3.1,
$c'x^k - v \leq K \exp\left(-\dfrac{k\alpha}{n}\right)$.

Proof: Using lemma 6.2.2 , at an iteration k , $f(x^k) \leq f_{v_k}(x^k) \leq f_{v_0}(x^0) - k\,\alpha$. Using the definition of $f(.)$,

$$n \log(c'x^k - v) \le f_{v0}(x^0) + \sum_{j=1}^{n} \log x_j - k\,\alpha$$

$$\le f_{v0}(x^0) + \gamma - k\,\alpha \ ,$$

by definition of γ. Now, removing the logarithm,

$$c'x^k - v \le \exp\left(\frac{f_{v0}(x^0)+\gamma}{n} - \frac{k\alpha}{n} \right)$$

and the final result is obtained by using the definition of K .

Second case: non-increasing costs and compact optimal set

If the costs are non-increasing, all action takes place in the set $\bar{S} = \{ x \in S \mid c'x \le c'x^0 \}$. This is a level set for the convex programming problem, and is compact as a consequence of the compactness of the optimal set. The analysis is then reduced to the case already studied, with

$$\gamma := \max \left\{ \sum_{j=1}^{n} \log x_j \mid x \in \bar{S} \ , x > 0 \right\}$$

The discussion above shows that the algorithms are linearly convergent, with a convergence speed equal to $\exp(\alpha/n)$

Polynomial convergence

To show that the algorithms converge polynomially, the only necessary step is to notice that K is polynomially bounded in relation to the input length L , and then repeat the analysis done by Karmarkar [9]. For compact feasible sets, $x_j < \exp(L)$ in (22) , and consequently $\gamma \le nL$. Similarly, we conclude that $f_{v0}(x^0) \le nL$, and finally $K \le \exp(2L)$.

7. Conclusions

We undertook a systematic effort to understand why Karmarkar's algorithm is so efficient, and believe to have found an answer in two fundamental mechanisms: conical projections and barrier functions. Conical projections expand the scope of each search, by trying to reduce an objective (potential function or cost) in a large trust region (the cone generated by a spherical region), instead of the sphere that determines orthogonal projections. Barrier functions - the potential function - avoid approaching too much the boundary of the feasible set, a feature that cannot be reproduced by cost minimizations in those trust regions.

The main features of the resulting algorithms are:

(i) The Linear Programming Problem in its usual formulation can be manipulated into the ideal format for the use of conical projections without any modification of the original problem, by describing the unique positive cone generated by its feasible set. No new constraints are added and the problem dimension is not changed. The resulting projected vectors c_p and a_p are uniquely determined by the geometry of the feasible set.

(ii) The unit simplex is not used at all.

(iii) Each expensive projection computation can be followed by several iterations of the internal algorithm, obtaining a greater reduction of its objective per projection.

(iv) Search directions belong to equivalence classes by conical projections, and Karmarkar's algorithm is reproduced by a single line search in the steepest descent direction for the potential function, or equivalently, by a line search on the direction $-\bar{c}$.

(v) All line searches can be done on the feasible set for the scaled problem by using the conical projections of the search directions, or equivalently on the original feasible set by inverting the scaling transformation. This may be interesting if the LP problem has been obtained by transforming or approximating some parent problem (e.g. in internal or external linearizations, or in dealing with inequality constraints): in these cases it may be possible to further transform the search directions so as to perform the line searches in the feasible set of the parent problem.

(vi) Finally, different options are available for the choice of search directions, as a result of a "goal programming problem" that weights the objectives of decreasing costs and decreasing the potential functions. These objectives are not always conflictive, and the resulting search directions are all equivalent to directions of the form $-\bar{c}_z = -(c_p - (u+z)a_p)$, where $z \geq 0$.

The computational aspects must now be studied in two lines: efficient calculation of the projection matrices, and efficient algorithms for the solution of the internal problem. Much work has already been done in the first direction [7],[13],[3],[1]. These results are directly applicable to the present method.

The second direction consists in trying to reduce the projection computations to the minimum possible number , and the best techniques are still to be detected. We hope to have contributed to this goal by making clearer what the options are.

Acknowlegement

I would like to express my thanks to Prof. E. Polak for his friendly support and encouragement.

Research partly sponsored by CNPq - Brazilian National Council for Scientific and Technological Development, by National Science Foundation grant ECS-857362, Office of Naval Research contract N00014-86-K-0295, and AFOSR grant 86-0116.

References

[1] I. Adler, M. Resende, G. Veiga, "An Implementation of Karmarkar's Algorithm for Linear Programming", Report ORC86-8 , Operations Research Center, University of California, Berkeley, May 1986.

[2] K.Anstreicher, "A Monotonic Projective Algorithm for Fractional Linear Programming", manuscript, Yale School of Organization and Management, New Haven, CT, November 1985.

[3] T. Cavalier, A. Soyster, "Some Computational Experience and a Modification of the Karmarkar Algorithm", Working Paper 85-105, Dept. of Industrial and Management System Eng., Pennsylvania State Univ., Feb. 85.

[4] D. Gay, "A Variant of Karmarkar's Linear Programming Algorithm for Problems in Standard Form", Numerical Analysis Manuscript 85-10, AT&T Bell Laboratories, Murray Hill, NJ, October 1985.

[5] P. Gill, W. Murray, M. Wright, Practical Optimization , Princeton University Press, New York, NY, 1981.

[6] C. Gonzaga, "A Conical Projection Algorithm for Linear Programming", memorandum No. UCB/ERL M85/61, Electronics Research Laboratory, University of California, Berkeley, CA, July

1985.

[7] M.Heath, "Some Extensions of an Algorithm for Sparse Linear Least Squares problems", SIAM Journal on Scientific ad Statistical Computing 3 (1982) 223-237.

[8] M. Iri and H. Imai, "A Multiplicative Penalty Function Method for Linear Programming - Another "New and Fast" Algorithm", Proc. of the 6th. Mathematical Programming Symposium, Tokio, Japan, November, 1985.

[9] N. Karmarkar, "A New Polynomial Time Algorithm for Linear Programming", Combinatorica 4 (1984) 373-395.

[10] W. Murray, M. Wright, "Efficient Linear Search Algorithms for the Logarithmic Barrier Function", Report SOL 76-18, Dept. of Operations Research, Stanford, California, 1976.

[11] M. Padberg, "Solution of a Nonlinear Programming Problem Arising in the Projective Method for Linear Programming", Manuscript, New York University, New York, NY, March 1985.

[12] T. Rockafellar, "Convex Analysis", Princeton Mathematics Ser., Vol. 28, Princeton Univ. Press, 1970.

[13] M. Todd, B. Burrell, "An extension of Karmarkar's Algorithm for Linear Programming Using Dual Variables", Technical Report 648, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, January 85.

[14] Y. Ye, "A Large Group of Projections for Linear Programming", Manuscript, Engineering-Economic System Dept., Stanford University, Stanford, Ca, February 1985.