NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents

POLYNOMIAL CONVERGENCE IN INTERIOR METHODS FOR LINEAR PROGRAMMING PROBLEMS

by

John Edward Hutchings

PhD, University of British Columbia, 1973

A THESIS SUBMITTED IN PARTIAL FUFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in the department of

Computing Science

SIMON FRASER UNIVERSITY

December 1989

ISBN 0-315-59359-8

APPROVAL

NAME: John Edward Hutchings

DEGREE: Master of Science

TITLE OF THESIS: Polynomial convergence in interior methods for linear programming problems

EXAMINING COMMITTEE:

Chairman: Dr Thomas Sherme

Dr Pavol Hell
Senior Supervisor

Dr ~~Louis~~ Hafer

Dr Joseph Peters

Dr Robert Russell
External Examiner
Professor, Dept of Mathematics & Statistics, SFU

DATE APPROVED: December 21, 1989

PARTIAL COPYRIGHT LICENSE

I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

Polynomial Convergence in Interior Methods for Linear Programming Problems.

Author: _____

(signature)

John Edward Hutchings

(name)

December 21, 1989.

(date)

ABSTRACT

This thesis describes the ideas underlying modern interior methods for the solution of linear programming problems which find the solution in the worst case in polynomial time. Much of the thesis is taken up by an exposition of Karmarkar's algorithm; we have attempted to give an intuitive description of the working of this algorithm, whose standard exposition can be quite difficult to read. The thesis ends with a description of recent results in the field.

TABLE OF CONTENTS

0. INTRODUCTION

This thesis describes algorithms for the solution of Linear Programming Problems (LPP's) by so-called interior methods. These include the LPP algorithms with the best known worst-case complexity. Although I offer no new algorithm, the thesis attempts to present the main ideas in an intuitive and "motivational" way. The original idea of describing "the state of the art" was lost sight of rather quickly because of the rapid advance of the field; however I include some discussion of interesting results as of Sept 1988 [8],[9],[10].

0.1. Linear Programming problems.

We consider only LPP's with integer coefficients. The theory of LPP's with integer coefficients is appropriate to the solution of LPP's on digital computers (any fixed precision problem can be scaled to such a problem), and the various polynomial time LPP algorithms which have appeared since 1975 – Khachian's and Karmarkar's algorithms, path-following methods, etc. – apply to this class of LPP.

We also assume that a first feasible point exists whenever this is convenient, since in principle finding one feasible point is straightforward, cf. §1.3. This paper takes a very geometric approach. We avoid the (non-trivial) matter of applications entirely (see Chapters 11 to 15 of [4] for a discussion of applications) and consider an LPP to be an objective vector $\mathbf{c} \in E_n$ plus a constraint space, which is a subset of the positive orthant of E_n . The problem is to find a point in the constraint space which maximizes (or minimizes) $\mathbf{cx}$. In this paper we always write an LPP in the slack variable or affine space form

$$\begin{aligned}
&\text{Maximize } \lambda(\mathbf{x}) = \mathbf{cx} \\
&\qquad \text{under the constraints} \\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\
&\quad \vdots \qquad \vdots \qquad\qquad \vdots \qquad \vdots \\
&\quad \vdots \qquad \vdots \qquad\qquad \vdots \qquad \vdots \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_n \\
&\mathbf{x} \in P_+
\end{aligned} \tag{0.1.1}$$

where $\lambda(\mathbf{x})$ is a linear objective function whose value is $\mathbf{cx}$,

and P_+ is the positive orthant, i.e. the set $\{\mathbf{x} : x_i \geq 0;\ i=1,2,\ldots,n\}$. The set of solutions of the linear system

$$\Sigma_{j=1}^{n} a_{ij} x_j = b_i \quad ;\ i = 1,\ldots,\ m \qquad (0.1.2)$$

is called an affine space (or translated vector space; see the next section). Evidently an affine space is the intersection of a finite number of hyperplanes of E_n . The part of this intersection which lies in P_+ is the constraint space. In the sequel the affine space determined by (0.1.2) will be called Ω . Thus this constraint could have been written "$\mathbf{x} \in \Omega \cap P_+$". Fig. 1 shows $\mathbf{c}$ and $\Omega \cap P_+$ for one possible LPP (0.1.1) in 3 variables.

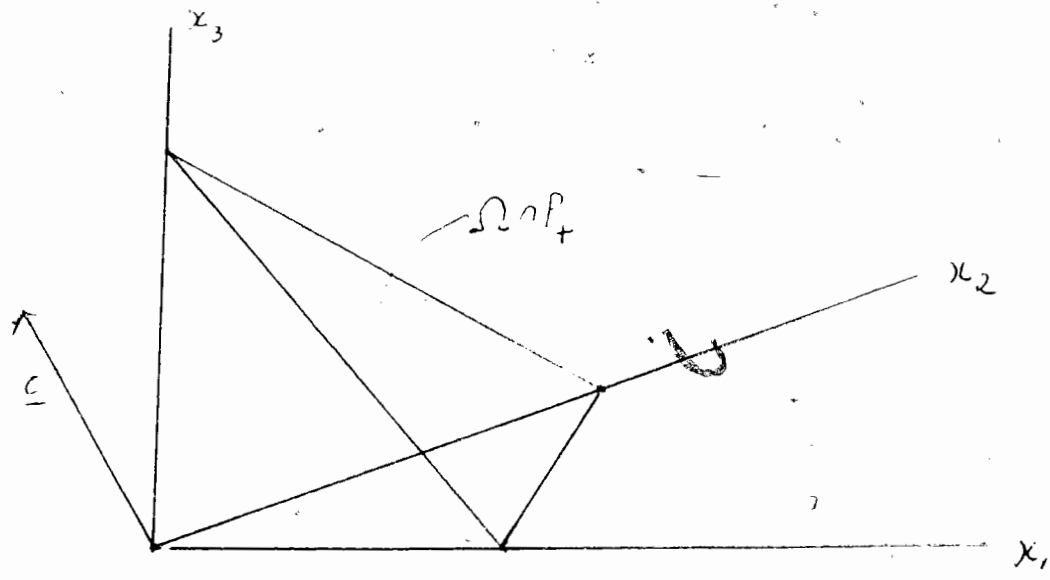

Fig. 1

The space Ω looks like a vector subspace of E_n except that it

need not hit the origin (see §0.3). Finally, we usually write the linear equations in LPP (0.1.1) in the matrix form

$$\mathbf{Ax} = \mathbf{b}$$

where A is the matrix $[\, a_{ij} \,]$. One also meets a form of LPP which is identical to (0.1.1) except that some of the equations that define the constraint space are replaced by inequalities. Thus

$$\Sigma_{j=1}^{n} a_{ij} x_j \leq b_i \ .$$

instead of

$$\Sigma_{j=1}^{n} a_{ij} x_j = b_i \ .$$

This form is likely to appear when applications of LPP's are considered. The reader is probably familiar with the technique called 'addition of slack variables', which converts the form with inequalities to the form (0.1.1).

0.2. Geometric aspects of LPP's.

This section summarizes some ideas that arise naturally when one employs the strongly geometric view of LPP's that is essential here. These will be used in the sequel without further comment.

(0.2.1) If $\mathbf{c}$ is an objective vector, then for $\mathbf{x} \in E_n$, the objective value $\mathbf{cx}$ is $\|\mathbf{c}\|$ times the length of $\mathbf{x}$ projected onto $\mathbf{c}$ (see Fig 2). Hence for all vectors $\mathbf{x}$ of length $\|\mathbf{x}\| = \rho$, the maximal value of $\mathbf{cx}$ is $\rho\|\mathbf{c}\|$ and occurs when the direction of $\mathbf{x}$ is the direction of $\mathbf{c}$.

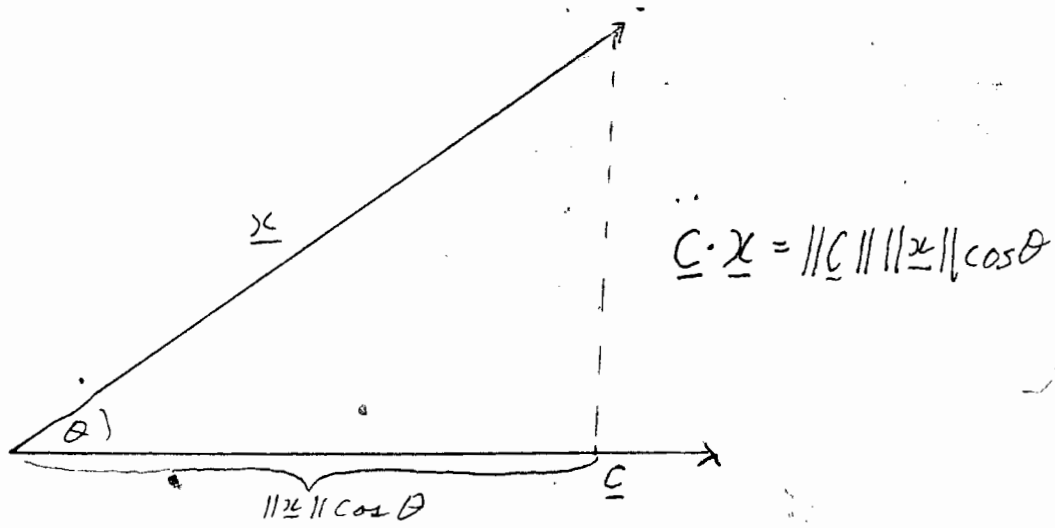

Fig. 2

(0.2.2) DEFINITION. If $F : S \longrightarrow R$ is a mapping (R is the set of reals) and $S \subset E_n$, we let $\mathbf{m}_{S,F}$ denote a maximal point of S under F, i.e. a point $\mathbf{m}$ with $F(\mathbf{m}) \geq F(\mathbf{x})$ for $\mathbf{x} \in S$.

Evidently $\mathbf{m}_{P_+ \cap \Omega, \lambda}$ is a solution of LPP (0.1.1). If F is the objective function $\lambda(\mathbf{x}) = \mathbf{cx}$, we frequently write just $\mathbf{m}_S$. Obviously $\mathbf{m}_{S,F}$ and $\mathbf{m}_S$ may not be unique. Minimal points $\mu_{S,F}$ and μ_S are defined analogously.

(0.2.3) Let

$$\Sigma_i^n \frac{(x_i - p_i)^2}{\alpha_i^2} = 1$$

be the equation of an ellipsoid axially aligned in E_n with principal axes $\alpha_1, \alpha_2, \ldots, \alpha_n$, and centre $(p_1, p_2, \ldots p_n)$. Then a mapping ϕ of E_n onto itself which carries the unit ball onto the ellipsoid is given by

$$\phi(\mathbf{x}) = D\mathbf{x} + \mathbf{p}$$

where D is the matrix $\begin{bmatrix} \alpha_1 & & 0 \\ & \alpha_2 & \\ 0 & & \alpha_n \end{bmatrix}$ and $\mathbf{p}$ is $\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}$.

The presence of so much linearity makes it tempting to look for linear mappings which 'map the LPP' in some sense while preserving the solution. This is dangerous because linear mappings do not preserve inner products, and hence may not preserve the value of the objective function. Nevertheless in §1.4 we will have to look at mapping of LPP's since this idea is fundamental to Karmarkar's algorithm.

0.3. Affine spaces and affine maps.

An affine space Ω can be considered either as a translated vector space or as the space of solution vectors $\mathbf{x}$ of the system $A\mathbf{x} = \mathbf{b}$, where A is an n×m matrix and $\mathbf{b}$ an n-vector. Let $\mathbf{q}$ be any solution of system $A\mathbf{x} = \mathbf{b}$. By linearity of matrix multiplication, if $\mathbf{y}$ is in the null space

of A (i.e. if $\mathbf{Ay} = \mathbf{0}$ is true) then $\mathbf{q} + \mathbf{y}$ is a solution of $\mathbf{Ax} = \mathbf{b}$. Hence $\Omega = \mathbf{q} + V$, where $V = \{\mathbf{x} : \mathbf{Ax} = \mathbf{0}\}$. Evidently $\mathbf{q}$ can be any point in Ω , and in the case of an LPP a convenient base is the first feasible point.

Affine mappings. A mapping T of E_n to itself is called an <u>affine mapping</u> if $T = \theta\tau$ where θ is a linear mapping and τ is a translation given by $\tau(\mathbf{x}) = \mathbf{x} + \mathbf{q}$. Alternatively one can write T in the equivalent form $T = \tau'\theta$ (for the same mapping θ but a new translation τ'), since

$$\theta\tau(\mathbf{x}) = \theta(\mathbf{x}+\mathbf{q}) = \theta(\mathbf{x}) + \theta(\mathbf{q}) = \tau'\theta(\mathbf{x}) ,$$

where $\tau'(\mathbf{y}) = \mathbf{y} + \theta(\mathbf{q})$. The mapping ϕ of §0.2.3 which carries the unit sphere onto an ellipsoid is an affine mapping. It follows from the preceding paragraph that T will carry an affine space onto an affine space. The composition of two affine mappings, $T_i = \theta_i\tau_i$, i = 1,2, is an affine mapping since

$$T_1T_2(\mathbf{x}) = \theta_1[T_2(\mathbf{x})+\mathbf{q}_1] = \theta_1\theta_2(\mathbf{x}) + \theta_1\theta_2(\mathbf{q}_2) + \theta_1(\mathbf{q}_1)$$

which is clearly affine. In this essay I'll consider only non-singular affine mappings. The important property of an affine mapping T, usually called simply "the affine property", is

(0.3.1) Affine Property of the mapping T : for vectors $\mathbf{x}_i$, i = 1, … 4, and scalar α ,

$\alpha(T(\mathbf{x}_1) - T(\mathbf{x}_2)) = T(\mathbf{x}_3) - T(\mathbf{x}_4)$ is true iff

$\alpha(\mathbf{x}_1 - \mathbf{x}_2) = \mathbf{x}_3 - \mathbf{x}_4$ is true.

Proof :

$\alpha(\mathbf{x}_1 - \mathbf{x}_2) = \mathbf{x}_3 - \mathbf{x}_4$ ⟷

$\alpha\theta(\mathbf{x}_1 - \mathbf{x}_2) = \theta(\mathbf{x}_3 - \mathbf{x}_4)$ ⟷

$\alpha\theta(\mathbf{x}_1 - \mathbf{x}_2 + \mathbf{q} - \mathbf{q}) = \theta(\mathbf{x}_3 - \mathbf{x}_4 + \mathbf{q} - \mathbf{q})$ ⟷

$\alpha[\ \theta(\mathbf{x}_1+\mathbf{q}) - \theta(\mathbf{x}_2+\mathbf{q})\ ] = \theta(\mathbf{x}_3+\mathbf{q}) - \theta(\mathbf{x}_4+\mathbf{q})$ ⟷

$\alpha[\ T(\mathbf{x}_1) - T(\mathbf{x}_2)\ ] = T(\mathbf{x}_3) - T(\mathbf{x}_4)$

It is important that there are non-affine mappings which carry affine spaces onto affine spaces; we will meet one in §2.4.1.

1. INTERIOR METHODS FOR THE SOLUTION OF LPP'S

These are simply methods which seek the solution point of an LPP by directing a 'search path' through the interior of the constraint polyhedron. This is to be contrasted with the simplex method, which searches through a sequence of vertices along a path in the edges of the constraint

polyhedron. An interior method attacks the LPP by beginning at a feasible point $\mathbf{q}$ and tracing a path $\mathbf{x}(t)$, $t \geq 0$ in Ω such that $\mathbf{x}(0) = \mathbf{q}$ and objective value $\mathbf{cx}(t)$ continually improves as t increases. How should $\mathbf{x}(t)$ be specified?

$\Omega \cap P_+$ is convex, so conceivably a search path could trace a straight line from the terminal point of $\mathbf{q}$ to the terminal point of $\mathbf{m}_{P_+ \cap \Omega}$, i.e. we could set $\mathbf{x}(t) = \mathbf{q} + t\mathbf{z}$ where $\mathbf{z}$ is a unit vector. From (0.2.1), the obvious choice of direction for $\mathbf{z}$ would be c itself if it were not that $\mathbf{c}$, whose direction is arbitrary, is likely to point out of the space Ω . The best direction turns out to be $\mathbf{c}_\pi$, the <u>projection of vector $\mathbf{c}$ into affine space Ω</u>.

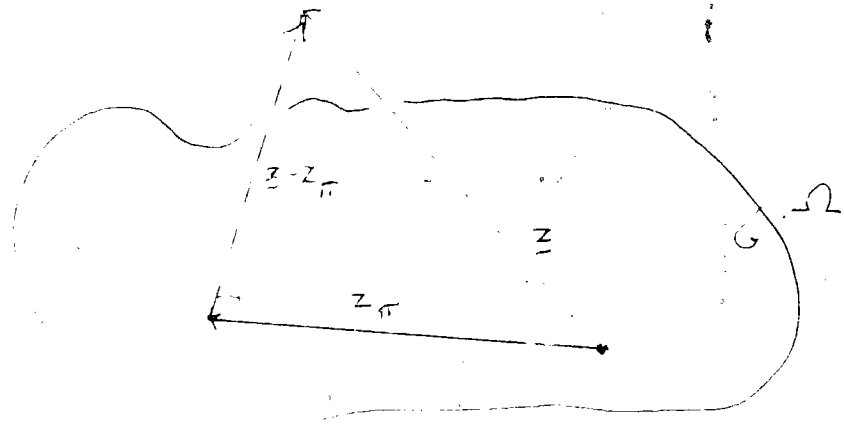

Fig. 3

(1.0.1) DEFINITION. Vector z_π is the projection of vector z into affine space Ω if for any coordinate system such that the origin $0 \in \Omega$

i) z_π lies in Ω ,
ii) $(z - z_\pi)x = 0$ for all $x \in \Omega$.

Fig. 3 shows the situation for a 2-dimensional Ω in 3-space. Vector z is shown as a vector with initial point in the plane Ω ; then $z - z_\pi$ is represented by a perpendicular dropped from the terminal point of z to Ω .

Once we know the value cq for the first feasible point $q \in \Omega$, then for computing the objective value of any point $x \in \Omega$, the projection c_π is as good as c because

$$\begin{aligned} cx &= cq + c(x-q) \\ &= cq + c_\pi(x-q) + c_\perp(x-q) \\ &= cq + c_\pi(x-q) + 0 \quad \text{(from the definition of } c_\pi\text{).} \end{aligned}$$

Vector c_π is found using methods described in the Appendix. It is possible for c_π to be 0 ; this simply means that c is orthogonal to all of Ω and thus any feasible point is optimal. In the sequel I'll often ignore this trivial case.

We can now write down an easy interior method which sometimes provides a useful approximation of the solution of LPP (0.1.1):

- the input is objective vector $\mathbf{c}$, which we take to be a unit vector, and feasible point $\mathbf{q} \in \text{Int}(\Omega \cap P_+)$.
- find $\mathbf{c}_\pi$, the projection of objective vector $\mathbf{c}$ into the affine space Ω . We assume that $\mathbf{c}_\pi$ is non-zero.
- travel from the feasible point $\mathbf{q}$ in the direction of $\mathbf{c}_\pi$ (that is, set $\mathbf{x}(t) = \mathbf{q} + t\mathbf{c}_\pi$ and trace the path $\mathbf{x}(t)$ with t increasing from 0) until the boundary of P_+ is encountered at point $\mathbf{q}'$ which is the output.

(1.0.2)

This method is sometimes called "crashing", i.e. the path 'takes off' from $\mathbf{q}$ and collides simple-mindedly with the boundary. Its virtue is that it always maximises the rate $\frac{d}{dt}\lambda(\mathbf{x})$. Its defect is that while it improves the objective value, it usually gets the wrong answer, as can be seen by trying some experimental values of $\mathbf{c}$ and $\mathbf{q}$ on the example in fig. 1. Considered as a method of approximating the solution point $\mathbf{m}_{P_+ \cap \Omega}$, this method demonstrates two problems typical of interior methods: things get much more complicated if $\mathbf{q}$

lies on the boundary of P_+; and the improvement in the objective value may be arbitrarily close to zero for unlucky pairs **q** and c .

1.1. Interior methods and approximation.

All interior methods I have encountered have the property that they approximate the solution but nevertheless obtain an exact answer. The reason that this can happen was first stated explicitly by Khachian in 1978 [11].

(1.1.1) GRANULARITY THEOREM. Let L be the number of bits required to encode all constants in the LPP (Since we assume integer coefficients, L will be finite). Let an approximate solution Y of the LPP lie within a distance $2^{-(L+1)}$ of the exact solution Z. Then value Y, rounded to a precision of L bits, is exactly Z .

This idea was 'around' in a general sort of way in the '60s (its background is discussed in [4]), but it was not until Khachian's paper [11] that it was realized that approximation methods could be a good place to look for polynomial time solutions to LPP's. Some (non-Russian) scientists could be heard grumbling that the whole thing was obvious and should have been seen through much earlier. Typically an interior method using approximation needs to begin by knowing a (probably large) bounded subset C of E_n in

which the solution point of the LPP is guaranteed to lie. Such a set is not obvious, since the LPP (0.1.1) may have maximal points arbitrarily far from the origin, (e.g. let $\Omega = E_2$, $\mathbf{c}^T = [-1, 0]$, then any point $\begin{bmatrix} 0 \\ y \end{bmatrix}$ for $y \geq 0$ is maximal). Karmarkar's algorithm provides a special solution to the problem of finding C , but the usual answer is provided by the

(1.1.2) RANGE THEOREM. A solution point of the LPP must lie in the cube C given by $\{\mathbf{x} : 0 \leq x_i \leq 2^L\}$ if any solution point exists. Furthermore $\|\mathbf{cx}\| < 2^L$ for any $\mathbf{x} \in C$. Here L is the encoding length defined above.

Hence once it is known that a solution of LPP (0.1.1) exists, an approximation of this solution begins by looking in a large cube of known size and then finds successive approximations to the answer until it finds one with an absolute error less than $2^{-O(L)}$. This final approximation, rounded to L bits, is the exact solution. Proofs for the Granularity and Range theorems are easy corollaries of Lemma 1 of [3].

1.2. Local and Global Approximation. Assume that the solution of an LPP is being approximated by a sequence of points $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_r$ which converges to the maximal point $\mathbf{m}_{P_+ \cap \Omega}$ of $P_+ \cap \Omega$. A method of global approximation is one in which the

procedure which produces $\mathbf{q}_{i+1}$ from $\mathbf{q}_i$ considers all points in the constraint space, whereas in a local approximation some points of the space (say those remote from $\mathbf{q}$) tend to be left out of consideration. At the moment we lack the background to give an example of a method of local approximation (one will appear in §1.4.1) and must make do with examples which do not use approximating sequences. Algorithm (1.0.2) is a local method which takes a single coarse approximating step. Beginning with point $\mathbf{q} \in \mathrm{Int}(P_+) \cap \Omega$, algorithm (1.0.2) will produce feasible point $\mathbf{q}' \in \mathrm{Bd}(P_+)$ with $\lambda(\mathbf{q}') > \lambda(\mathbf{q})$ (assuming the LPP is not unbounded). Point $\mathbf{q}'$ depends only on $\mathbf{q}$, $\mathbf{c}_\pi$ and the nearest coordinate hyperplane, and will be a poor choice for some locations of $\mathbf{q}$. The simplex algorithm for LPP (0.1.1) is a good example of a local method: essentially it adopts the "best strategy for the next pivot", which may be a poor global strategy (the well-known Klee-Minty examples demonstrate this – a good brief description is in Ch 16 of [4]; the paper itself is [5]). Methods of global approximation for LPP's are rare: Karmarkar's algorithm and related methods are the only successful ones. I'll give an example of a successful global method (not an approximation) which, unfortunately, does not

solve LPP (0.1.1) but instead solves this special problem:

Maximize objective value $\mathbf{cx}$
under the constraints
$\mathbf{Ax} = \mathbf{b}$, or equivalently $\mathbf{x} \in \Omega$ for some affine space Ω .
$\mathbf{x} \in \{\mathbf{x} : \|\mathbf{x}-\mathbf{p}\| \leq \rho\}$
where $\mathbf{p}$ is a point in Ω , ρ a real number
(1.2.1)

Here $\mathbf{x}$, which may be of any dimension, is constrained to lie in an n-ball of radius ρ.

(1.2.2) Theorem I. Let $\mathbf{c}_\pi$ be the projection of objective vector $\mathbf{c}$ into affine space Ω . Then the solution of LPP (1.2.1) is found by beginning at the feasible point $\mathbf{p}$ and travelling in the direction of $\mathbf{c}_\pi$ for a distance equal to the radius ρ .

The easy proof follows from (0.2.1).

1.3. First feasible points and infeasibility

LPP (0.1.1) may be infeasible, a case of both practical and theoretical importance. It would be nice to detect this by a cheap method that precedes and may make unnecessary the main problem solving operation. Additionally, methods to detect infeasibility usually report feasibility by producing a feasible point, which may be essential to the main method. A two-phase method of solution of LPP (0.1.1) consists of a

first phase which analyses the problem and either detects infeasibility or finds a first feasible point, followed if necessary by a second phase which takes the objective value of the first feasible point and applies some form of progressive optimization. The two-phase method is standard in practical implementations of the simplex method.

The operation of the first phase is straightforward. To simplify things I'll use the form of LPP in which equations are replaced by inequalities. If the constraint space $\Omega \cap P_+$ of the LPP is given by

$$\Sigma_{j=1}^{n} a_{ij} x_j \leq b_i \quad ; \; i = 1, \ldots, m$$

$$\mathbf{x} \geq \mathbf{0} \; ,$$

then to find some point in $\Omega \cap P_+$ we can define the artificial variable x_0 and solve the new problem

Minimise x_0

under the constraint

$$-x_0 + \Sigma_{j=1}^{n} a_{ij} x_j \leq b_i \quad ; \; i = 1 \ldots, m$$

$$x_j \geq 0 \; ; \; j = 0, \ldots, n \; .$$

(1.3.1)

Problem (1.3.1) has an obvious feasible point: the point whose every coordinate is zero except x_0 which is set so large that $-x_0$ is less than any b_i . We can solve (1.3.1)

because we know how to do the second phase; hence we can obtain the minimum value of x_0 . If this minimum is 0, then any point in the constraint space of (1.3.1) which satisfies $x_0 = 0$ can be interpreted as a feasible point in LPP (0.1.1). If the minimum is greater than zero, then evidently LPP (0.1.1) is infeasible.

It is also possible to solve LPP (0.1.1) by a 'one-phase approach'. Approximation methods for (0.1.1) usually require the approximating sequence to consist of feasible points. However one can imagine at least in a vague way an approach in which this requirement is removed and no first feasible point is needed. If we know how to approximate the solution of (0.1.1) by a feasible sequence $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_r, \ldots$, then we can begin at an arbitrary point $\mathbf{q}_1'$, not necessarily in Ω , and construct a sequence $\mathbf{q}_1', \mathbf{q}_2', \ldots, \mathbf{q}_r', \ldots$ using rules like those for constructing the $\mathbf{q}_i$ except that the locations of the points $\mathbf{q}_i'$ are perturbed so that the $\mathbf{q}_i'$ lie continually closer to Ω . For methods which are good at keeping the approximating sequence inside P_+ (the 'barrier methods', see § 3) this is a practical procedure, and something like it is used in Karmarkar's algorithm [1].

Approximation methods which start at an infeasible point all seem to have the defect that their usual method of detecting infeasibility is to converge, perhaps slowly,

to an infeasible point. In fact there is some indication that finding first feasible points and detecting infeasibility may not be easy in practical algorithms based on Karmarkar's method. In a conversation with one of the authors of [6] I was told that the system described in [6], which is a state-of-the-art implementation of path-following methods, was a two-phase method which was using far more time to establish feasibility and provide a first feasible point than it was using to solve the rest of the problem.

1.4. Good and bad transformations of LPP's.

As remarked earlier, a transformation of an LPP, say in the form of a non-singular linear transformation of $\mathbf{c}$ and $\Omega \cap P_+$ cannot be expected to preserve solution points $\mathbf{m}_{\Omega \cap P_+}$; in fact the only linear transformations that preserve the solution point are transpositions and rotations. However it is still possible to use transformations to solve LPP's.

Suppose we wish to solve the problem

$$\text{Maximize } \lambda(\mathbf{x}) = \mathbf{cx}$$
$$\text{under the constraint}$$
$$\mathbf{x} \in \Omega$$
$$\mathbf{x} \in E \qquad (1.4.1)$$

where **c**, E, Ω are the vector, elliptical disk, and line shown in fig. 4. Note that the origin **0** is feasible.

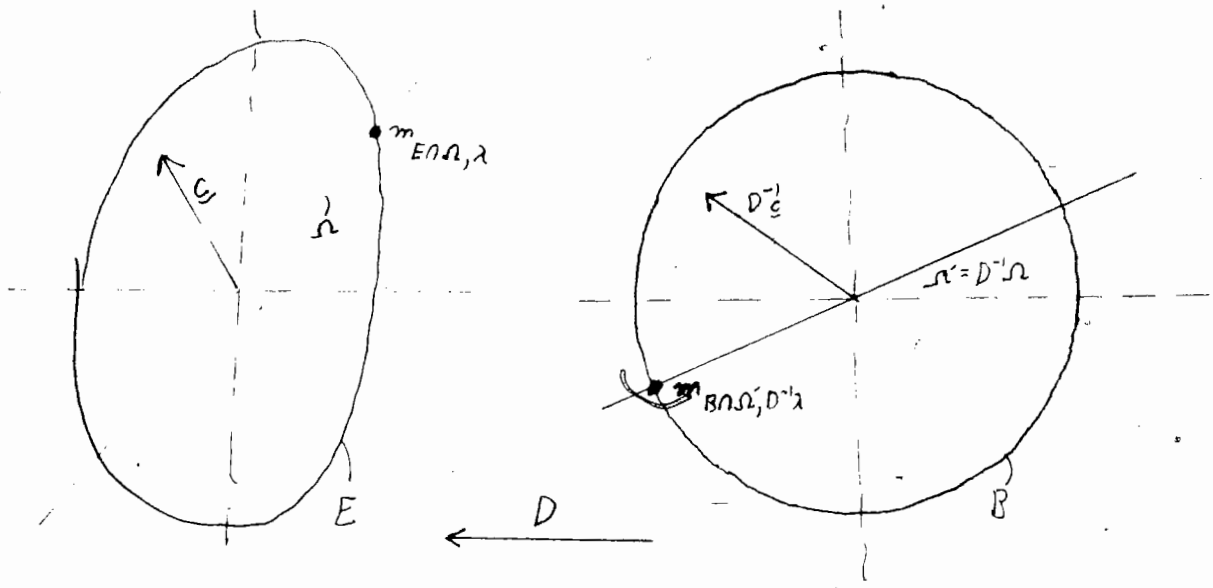

Fig. 4

We decide to find the solution to this LPP by transforming **c**, Ω and E to the vector **c**′, line Ω′ and unit circle B shown to the right in fig. 4. If E has major and minor axes of length 2 and 1 and centre **0** as shown in the figure, then the non-singular linear transformation that carries circle B onto

E can be written in matrix form as $\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} = D$.

Ignoring the warning in §0.2, we might hope that the mapping $\mathbf{x}' = D^{-1}\mathbf{x}$ can be used to transform the LPP on the left of fig. 4 to the 'transformed LPP' on the right, and might then invent the following method to solve LPP (1.4.1) (writing D for both the matrix and the transformation)

- Solve the LPP

  Maximize $D^{-1}\mathbf{c}\mathbf{x}'$

  under the constraint

  $\mathbf{x}' \in D^{-1}(\Omega) \cap D^{-1}(E) = \Omega' \cap B$

- If $\mathbf{z}'$ is a solution to the transformed LPP then hope that $\mathbf{z} = D\mathbf{z}'$ is a solution to the original problem.

But, as fig. 4 shows, this doesn't work, since the maximal point $\mathbf{z}'$ of the transformed LPP does not map back to $\mathbf{m}_{E\cap\Omega,\lambda}$. The reason is that D^{-1}, which does not preserve inner products, has altered the angle between $\mathbf{c}$ and Ω so that $D^{-1}\mathbf{c}$ projects onto $D^{-1}(\Omega)$ in exactly the wrong direction. The way to solve the problem in Fig. 4 is:

– Solve the LPP

Maximize $cDx' = \lambda'(x')$

under the constraint

$x' \in D^{-1}(\Omega) \cap B$

– if z' is the solution to the transformed problem, then the solution to the original problem is Dz' .

Here c is mapped under D rather than D^{-1} to get the 'compensating function' λ' (it would be more correct to write $\lambda' = c^{T}D$ but I think the meaning is clear). The reason this works is that mapping points $x \in \Omega$ under D^{-1} and objective vector c under D preserves the value of the objective function, i.e.

$$\begin{aligned} \lambda'(x') &= cDx' \\ &= cDD^{-1}x \\ &= cx \\ &= \lambda(x) \ . \end{aligned}$$

So if the transformed problem has a solution point $z' \in D^{-1}(\Omega) \cap B$ with objective value cDz' , then LPP (1.4.1) has a feasible point $z = Dz'$ with the same objective value, which must be maximal since the transformation $\lambda(x) = \lambda'(x')$ clearly preserves maximality.

In general, to attack the optimization problem

$$\begin{aligned} &\text{Maximize } F(\mathbf{x}) \\ &\quad \text{under the constraint} \\ &\qquad \mathbf{x} \in S \qquad \text{where } S \text{ is some} \\ &\qquad\qquad\qquad \text{constraint space} \end{aligned}$$

by the method of transformation means first to specify a non-singular mapping T of S to a space S′ and a 'compensating function' F′ such that $F'(T(\mathbf{x})) = F(\mathbf{x})$, i.e. $F' = FT^{-1}$. Then, taking $\mathbf{x}' = T(\mathbf{x})$, try to solve the transformed problem

$$\begin{aligned} &\text{Maximize } F'(\mathbf{x}') \\ &\quad \text{under the constraint} \\ &\qquad \mathbf{x}' \in S' \end{aligned}$$

which we hope is more tractable than the original. If we find $\mathbf{m}_{S',F'}$ then we know that $\mathbf{m}_{S,F} = T^{-1}(\mathbf{m}_{S',F'})$. Furthermore the maximal objective value of the original problem is $F(\mathbf{m}_{S,F}) = F'(\mathbf{m}_{S',F'})$

This broad definition allows for the fact that the transformed problem might not be a LPP, although in all the cases we will meet space S′ will be affine. The method of transformation works well in nice cases where T is an affine mapping and can also be made to work when T is non-affine. Additionally in nice cases the transformed objective function will be linear. Soon we will meet Karmarkar's algorithm,

which is not one of the nice cases.

1.4.1. An approximation using ellipsoids. We will attempt to solve LPP (0.1.1) using a sequence E_0, E_1, E_2, ... of approximating ellipsoids. The sequence of the E_i approaches a solution point of (0.1.1) in a way that the reader will probably find 'pictorialy convincing' (see fig. 5);

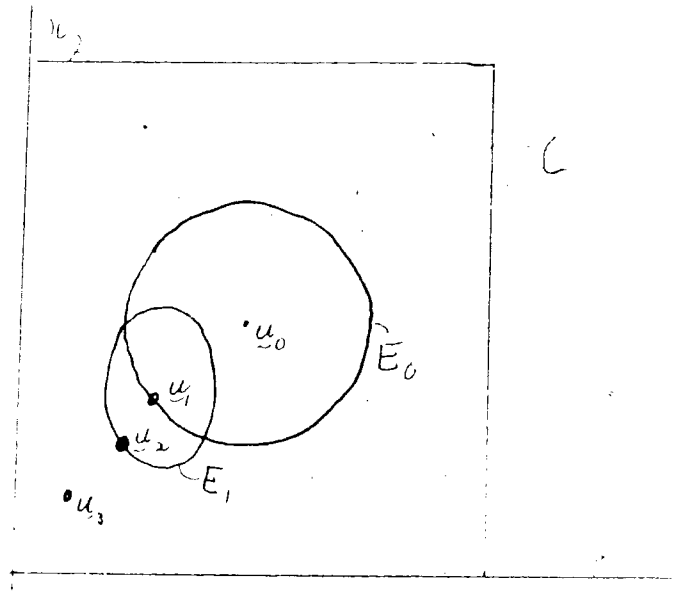

Fig. 5

however we will not give a proof that the E_i converge to a solution since we do not know a short proof of this and since the method given in this section is useful only as an introduction to Karmarkar's algorithm (whose rate of convergence will be analysed at some length). Some fussy details will be skipped, in particular we assume that the LPP is not unbounded and that objective vector c is not

orthogonal to Ω . Fig. 5 shows P_+ for n = 2. The range theorem provides that the maximal point of $\Omega \cap P_+$ lies in an n-cube C, which is also shown. Rescale everything so that a side of C has length 2; thus point $\mathbf{u}_0$ in fig. 5 is $\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$.

We also assume that there is a feasible point in Int(C), and that $\mathbf{u}_0$ is feasible (although the argument works after a fashion for any first feasible point in Int(C)).

Step one: Construct ellipsoid E_0 as a ball concentric with the inscribed ball B of C but with half the radius of B. Find the maximal point of $E_0 \cap \Omega$ using Theorem I (1.2.2) and the fact that $\mathbf{u}_0$ is feasible. Call this maximal point $\mathbf{u}_1$. The first stage is easy since E_0 is a ball.

Step two: $\mathbf{u}_1$ is not optimal since it lies in Int(C) (assuming that $\mathbf{c}$ is not orthogonal to Ω) so we look for a point with a better objective value. Let E_1' be the largest ellipsoid lying in C having centre $\mathbf{u}_1$ and axes parallel to the coordinate axes (the unnecessary 'parallel' requirement simplifies the example). Let E_1 be an ellipsoid concentric with E_1' but with one-half the diameter. Find the maximal point $\mathbf{u}_2$ of $E_1 \cap \Omega$ (see fig. 5). Since $\mathbf{u}_1 \in \text{Int}(E_1 \cap \Omega)$ and $\mathbf{c}_\pi$ is not zero by assumption, $\lambda(\mathbf{u}_2) > \lambda(\mathbf{u}_1)$.

Step three, four, etc. Evidently we could continue in this way to define points $\mathbf{u}_3$, $\mathbf{u}_4$, ... with monotonically increasing values of $\lambda(\mathbf{u}_i)$. From fig. 5 it is plausible (and in fact true) that the E_i converge to the solution of the LPP.

This method requires us to find the maximal points of the ellipsoids in fig. 5. We find $\mathbf{m}_{E_1 \cap \Omega}$ using the method of transformation. Begin by constructing an affine mapping ϕ which carries the inscribed unit ball B of C onto E_1 . From (0.2.3) if ellipsoid E_1 is given by

$$\Sigma_i^n \frac{[x_i - (\mathbf{u}_0)_i]^2}{\alpha_i^2} = 1$$

then, allowing for the fact that the centre of B is not the origin but $\mathbf{u}_0$, ϕ is defined by

$$\phi(\mathbf{x}) = D\mathbf{x} + \mathbf{u}_1 - \mathbf{u}_0$$

where D is  . To apply the method of transformation we need $\phi^{-1}(\mathbf{x}) = D^{-1}(\mathbf{x} - \mathbf{u}_1 + \mathbf{u}_0)$. This is an affine map and $\Omega' = \phi^{-1}(\Omega)$ is an affine space. (The reader may prefer $E_1 \longrightarrow$ B as the natural direction of ϕ ; however I use the direction $\phi: B \longrightarrow E_1$ because it will turn out to agree with Karmarkar's usage in [1].)

Now let $\mathbf{x}' = \phi^{-1}(\mathbf{x})$, and transform the problem

$$\begin{aligned} &\text{Maximize } \lambda(\mathbf{x}) = \mathbf{cx} \\ &\quad \text{under the constraint} \\ &\quad\quad \mathbf{x} \in \Omega \cap E_1 \end{aligned}$$

to the problem

$$\begin{aligned} &\text{Maximize } \lambda'(\mathbf{x}') = \mathbf{c}\phi(\mathbf{x}') = \mathbf{cDx}' + \mathbf{c}(\mathbf{u}_1 - \mathbf{u}_0) \\ &\quad \text{under the constraint} \\ &\quad\quad \mathbf{x}' \in \phi^{-1}(E_1 \cap \Omega) \\ &\quad\quad\quad = B \cap \Omega' \end{aligned}$$

Déspite the extra constant term in the objective function, this can obviously be solved by Theorem I since Ω' is affine. It is easy to check that objective value $\lambda'(\mathbf{x}') = \lambda(\mathbf{x})$, since

$$\lambda'(\mathbf{x}') = \mathbf{c}\phi(\mathbf{x}') = \mathbf{c}\phi\phi^{-1}(\mathbf{x}) = \mathbf{cx} = \lambda(\mathbf{x}) \ .$$

Hence if $\mathbf{z}'$ is $\mathbf{m}_{B\cap\Omega',\lambda'}$, then $\phi(\mathbf{z}') = \mathbf{z}$ is $\mathbf{m}_{E_1\cap\Omega,\lambda}$.

Evidently all the points $\mathbf{u}_2$, $\mathbf{u}_3$, etc. can be obtained in this way. This method converges poorly to $\tilde{\mathbf{u}}$. Intuitively the E_i 'get too small too fast' so that the value $\mathbf{cu}_i$ may increase arbitrarily slowly, and it was more or less assumed that there was no way to prevent this behaviour until the surprising appearance of Karmarkar's algorithm, which is a very sophisticated ellipsoidal approximation with polynomial time convergence.

2. KARMARKAR'S ALGORITHM

Karmarkar's algorithm, which appeared in 1980 [1], was the first algorithm to solve LPP's in reasonably fast ($\leq n^4L$) polynomial time. The worst case time of the version presented here is O(nL) iterations each of time $O(n^3)$, where L is the number of bits required to encode all the constants in LPP (0.1.1). The $O(n^3)$ time of each iteration is due to the fact that in each iteration an $n \times n$ matrix must be inverted. In [1] Karmarkar decribes a variation of the algorithm in which the sequence of inverted matrices is produced by 'updating' – each matrix in the sequence is used to generate an approximation of the next by "rank one modification" [2]. This approach reduces the worst case complexity of each iteration to $O(n^{2.5})$. Thus Karmarkar's algorithm can be made to run in time of $O(n^{3.5}L)$ operations. Karmarkar's algorithm is still considered fast: the current best time for the solution of the LPP (0.1.1) (see [9]) is $O(n^3)$ operations.

Before describing how the algorithm works, there is a technical detail to attend to: Karmarkar's approach requires

the reader to visualize the LPP not in the form (0.1.1) but in terms of a whole new picture which involves replacing our familiar $\mathbf{c}$, Ω and P_+ with new objects $\mathbf{c}'$, Ω', Σ . A feature of the new form is the fact that the positive orthant P_+ is replaced by an n-simplex Σ which lies obliquely in the space. This new "Karmarkar standard form" will be described in the next section.

2.1. This section describes the new Karmarkar standard form of the LPP which replaces the form (0.1.1). In this new form, the affine space Ω of (0.1.1) is replaced by a new affine space Ω' which is actually a vector space (i.e. meets the origin) and the positive orthant P_+ in E_n is replaced by a 'unit' n-simplex Σ embedded in E_{n+1} ('unit' meaning that the vertices of Σ are $(1,0,0, \ldots ,0)$, $(0,1,0, \ldots ,0)$, $(0,0,1, \ldots ,0)$, ... $(0,0,0, \ldots ,0,1)$). Thus although the simplex Σ is n-dimensional, there are n+1 variables. We will often refer to the centre $\mathbf{p}$ of Σ , where

$$\mathbf{p} \text{ is } \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} /(n+1) \ .$$

The definition of the new form is

Minimize $\mathbf{c}'\mathbf{x}$

under the constraint

$\mathbf{x} \in \Omega'$, where Ω' is the affine space defined by $A'\mathbf{x} = \mathbf{0}$ (here A' is a matrix of dimension $m \times (n+1)$)

$\mathbf{x} \in \Sigma$, where the n-simplex Σ is defined by

$$\Sigma_{i=1}^{n+1} x_i = 1 \ , \ \mathbf{x} \geq \mathbf{0} \ , \tag{2.1.1}$$

We illustrate this in fig. 6, in which $n = 2$, Σ is 2-dimensional, and Ω' is a two dimensional space. In fig. 6 the constraint space is a line segment.

In the sequel, let J stand for the n-plane $\Sigma_{i=1}^{n+1} x_i = 1$.

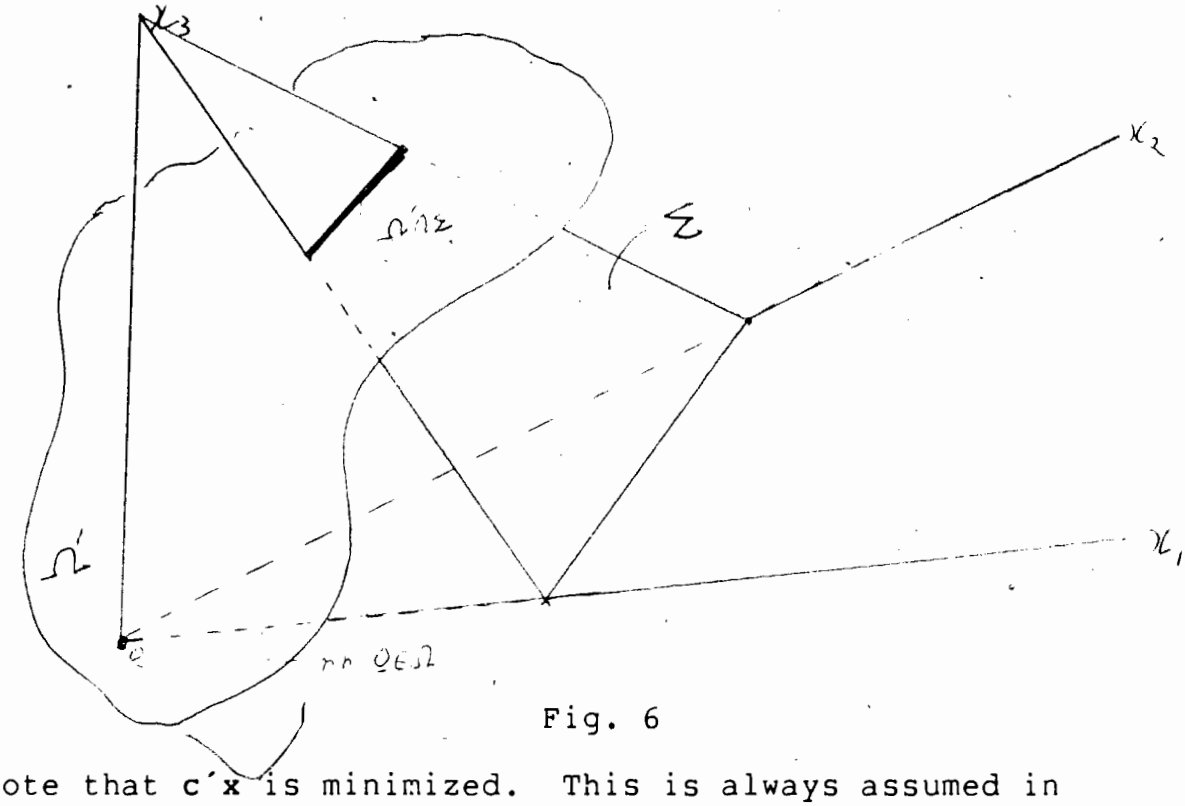

Fig. 6

Note that $\mathbf{c}'\mathbf{x}$ is minimized. This is always assumed in discussions of Karmarkar's algorithm. Recall that μ_S or $\mu_{S,\lambda}$ stands for a minimal point of the set S .

This new form is unexpected; the reader may doubt that applications will arise naturally in form (2.1.1). I'll describe how to transform a LPP in form (0.1.1) to one in form (2.1.1). That is, suppose there is an LPP in form (0.1.1) specified by objective vector **c**, m×n matrix A, and m-vector **b** . I will show how to construct vector **c**′, and m×(n+1) matrix A′ so that with these values LPP (2.1.1) corresponds to LPP (0.1.1). 'Corresponds' means that there will be a mapping Ξ of E_n into E_{n+1} which carries the constraint space of form (0.1.1) into that of form (2.1.1)

and preserves optimal points and the value of the objective function.

I'll give the construction of the mapping Ξ first; then $\mathbf{c}'$ and A' are easily found. The range theorem (1.1.2) provides that we can consider the constraint space of (0.1.1) to be $C \cap \Omega$, where Ω is defined in (0.1.1) and C is the cube $\{\mathbf{x} : 0 \leq x_i \leq 2^L\} \subset E_n$. Rescale $C \cap \Omega$ by dividing every vector in the space by the scalar $n2^L$ (L is the precision number from (1.1.2)) so that the constraint space becomes $C^* \cap \Omega^*$ with $C^* = \{\mathbf{x} : 0 \leq x_i \leq \frac{1}{n}\}$ and Ω^* given by $A\mathbf{x} = \mathbf{b}/n2^L$, where A, $\mathbf{b}$, Ω are taken from (0.1.1) (n.b. $n2^L$ not $(n+1)2^L$, even though we intend to map to E_{n+1}). Obviously minimal points and objective values are preserved, allowing for the scale factor $n2^L$. The mapping Ξ is a composition of mappings Ξ_1, Ξ_2. Mapping Ξ_1 embeds C^* into the $(x_{n+1}=0)$-hyperplane of E_{n+1}; thus for $\mathbf{x} \in C^*$, define $\Xi_1(\mathbf{x}) = \begin{bmatrix} \mathbf{x} \\ \cdots \\ 0 \end{bmatrix} \in E_{n+1}$, Fig. 7 shows this for $n = 2$. As suggested by fig. 7 we easily check that $\Xi_1(C^*)$ lies entirely

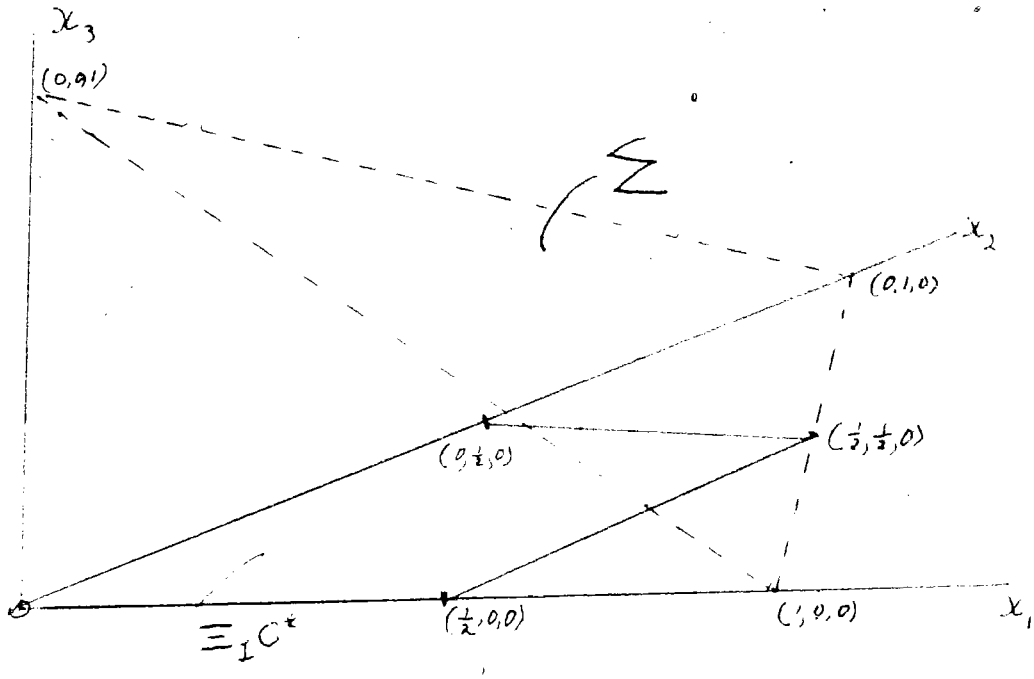

Fig. 7

in the positive orthant of E_{n+1} and entirely on the origin side of the unit n-simplex Σ except for a single point in Σ. Next define Ξ_2, whose action will be to project $\Xi_1(C^*)$ in the direction of the positive x_{n+1}-axis into J (recall that J is the hyperplane containing Σ). Thus if $\Xi_1(\mathbf{x}) = \begin{bmatrix}\mathbf{x}\\ \cdots \\ 0\end{bmatrix}$, then define $\Xi_2\Xi_1(\mathbf{x}) = \Xi_2\begin{bmatrix}\mathbf{x}\\ \cdots \\ 0\end{bmatrix} = \begin{bmatrix}\mathbf{x}\\ \cdots \\ k\end{bmatrix}$, where k is uniquely chosen so that $\begin{bmatrix}\mathbf{x}\\ \cdots \\ k\end{bmatrix} \in J$. Check that when $\Xi_1(C^*)$ is projected in this way by Ξ_2, the image remains in the positive orthant of E_{n+1}; hence $\Xi(C^*) \subset \Sigma$ and $\Xi(C^* \cap \Omega^*) \subset \Sigma$.

Now that we have the mapping Ξ we can describe the affine space Ω', which corresponds to Ω in form (0,1,1). We will find a determining linear system $A'\mathbf{y} = \mathbf{0}$ for Ω' so that $\Omega' \cap J = \Xi(\Omega^*)$. Note first that if $\mathbf{x} \in \Omega^*$, where Ω^* is

given by $\mathbf{Ax} = \mathbf{b}/n2^L$, and $\mathbf{y} = \Xi(\mathbf{x}) \subset J$, then $[\mathbf{A} \vdots \mathbf{0}]\mathbf{y} = \mathbf{b}/n2^L$. This would be the system we want except that it is not homogeneous. To bring the system $[\mathbf{A} \vdots \mathbf{0}]\mathbf{y} = \mathbf{b}/n2^L$ to the homogeneous form required by form (2.1.1), simply rewrite it with an extra m+1'st equation that says that $\mathbf{y} \in J$, i.e. that $\Sigma_{i=1}^{n+1} y_i = 1$. In matrix form this is

$$\begin{bmatrix} A & 0 \\ 1\ 1 \cdots 1 & \end{bmatrix} y = \begin{bmatrix} b/n2^L \\ 1 \end{bmatrix}$$

Now apply row operations to equations 1 ... m , that is add $-b_1/n2^L$ times the m+1'st equation to equation 1, etc., so as to get

$$\begin{bmatrix} A' \\ 1\ 1 \cdots 1 \end{bmatrix} y = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

where A' is the $m \times (n+1)$ matrix we want. Points $\mathbf{y}$ which satisfy both $[\mathbf{A} \vdots \mathbf{0}]\mathbf{y} = \mathbf{b}/n2^L$ and $\Sigma_{i=1}^{n+1} y_i = 1$, that is points of $\Xi(\Omega^*)$, continue to satisfy the new system. Strip off the m+1'st row to get $\mathbf{A'y} = \mathbf{0}$ which defines the affine space Ω' .

We can now give the transformation of the form (0.1.1) into the form (2.1.1): if the original LPP is

minimise **cx**

under the constraints

$\mathbf{Ax} = \mathbf{b}$

$\mathbf{x} \geq \mathbf{0}$

and objective vector $\mathbf{c}^T$ is $[c_1, c_2, \ldots c_n]$, then the transformed objective vector $\mathbf{c}'^T$ is $(n2^L)[c_1, c_2, \ldots c_n, 0]$, and the transformed LPP, is

minimise $\mathbf{c}'\mathbf{y}$

under the constraint

$\mathbf{A}'\mathbf{y} = \mathbf{0}$

$\mathbf{y} \in \Sigma$

where $\mathbf{y} = \Xi(\mathbf{x})$ and Σ is defined as in (2.1.1). It is straightforward to check that this transformation preserves minimal points since it obviously preserves the objective value under Ξ and since boundary points of P_+ are carried to boundary points of Σ .

In the remainder of this section 2, I'll forget the primes on A, Ω, and **c** ; thus **c**, Ω, $\mathbf{Ax} = \mathbf{0}$ will always refer to the objective vector and affine space associated with the Karmarkar standard form (2.1.1). I'll use **p** for the centre of simplex Σ, B for the inscribed n-dimensional ball of Σ (i.e. B lies in the plane J) and B* for the escribed n-dimensional ball.

2.2. The Basic Idea of the Algorithm.

In Sections 2.2, 2.3, 2.4 I'll describe the ideas that motivate Karmarkar's algorithm. We will use the Karmarkar standard form, but for much of the time will look only at the simplex Σ . Karmarkar's algorithm can be thought of as a kind of ellipsoidal approximation method. I'll begin by describing roughly how one would approximate the solution of LPP (2.1.1) in a naive way using a sequence of ellipsoids in Σ , rather like the sequence in § 1.4.1. Next I'll give an argument – which is Karmarkar's first important idea – leading to a hope that an approximation of this kind might converge in polynomial time. It will turn out that the naive approach doesn't work; my reason for spending time on it is that Karmarkar's method, which does work, is simply the naive method with its faults repaired, unfortunately in a sophisticated manner which is not easy to describe. I'll analyse the weaknesses of the naive method at length in order to motivate Karmarkar's use of non-linear functions – the famous 'projective mapping' and 'potential function' – in order to convert the naive method into one which converges in polynomial time.

2.2.1. Consider LPP (2.1.1) with the additional assumption that the centre $\mathbf{p}$ of the simplex Σ is known to be feasible. We

will attempt to solve this problem by an ellipsoidal approximation like the one in §1.4.1. The description will be sketchy in places since our construction resembles that of fig. 5. Fig. 8 shows the first three stages of an

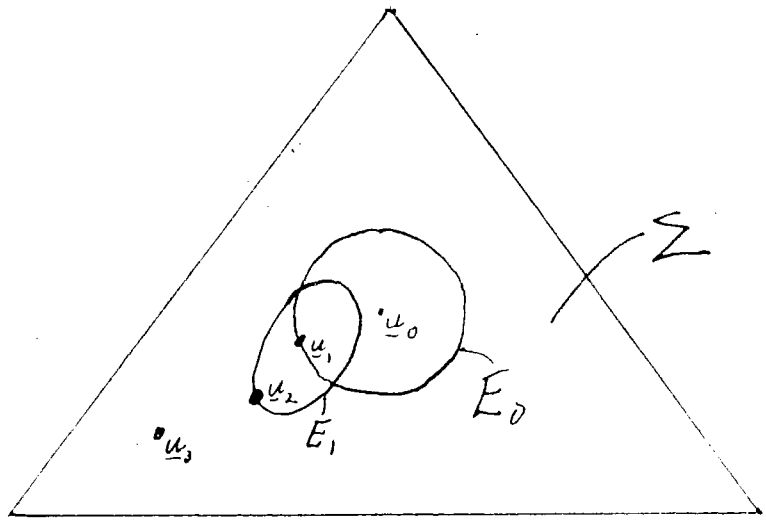

Fig. 8

ellipsoidal approximation of the minimal point $\mu_{\Sigma\cap\Omega}$. Although the ellipsoids in fig. 5 were axially aligned, we don't require this in fig. 8. We define an approximating sequence $\mathbf{u}_0 = \mathbf{p}, \mathbf{u}_1, \mathbf{u}_2, \ldots$ and ellipsoids $E_i \subset \Sigma$ whose centres are the $\mathbf{u}_i$. B is the inscribed ball of Σ with centre $\mathbf{u}_0$. Ball E_0 is half the radius of B but with the same centre. Point $\mathbf{u}_1$ is the minimal point $\mu_{E_0\cap\Omega,\lambda}$, which is easily found. Ellipsoid E_1' is an ellipsoid with centre $\mathbf{u}_1$ which is as large as possible but still fits into Σ. E_1 is an ellipsoid concentric with E_1' and half the diameter (E_1' is

not shown in fig. 8). Let $\mathbf{u}_2$ be $\mu_{E_1\cap\Omega,\lambda}$. Let E_2' be the largest ellipsoid with centre $\mathbf{u}_2$ that fits in Σ and let E_2 be concentric with and half the size of E_2' (the vagueness about how the 'largest ellipsoid' is fitted into Σ will not harm the discussion in this section).

It's easy to find point $\mathbf{u}_1$ using Theorem I (1.2.2). To find $\mathbf{u}_2 = \mu_{E_1\cap\Omega}$ requires the method of transformation; in view of the resemblance of fig. 8 to fig. 5, I'll give only a sketch of this. Define an affine mapping ϕ_1 which carries E_0 onto E_1 and a compensating objective function $\lambda_1' = \lambda\phi_1$. Mapping ϕ_1^{-1} is defined as $\phi_1^{-1}(\mathbf{x}) = D_1^{-1}(\mathbf{x}-\mathbf{u}_1+\mathbf{u}_0)$, where D is the matrix that maps ball E_0 onto the translated ellipsoid $\{\mathbf{y} : \mathbf{y} = \mathbf{x}-\mathbf{u}_1+\mathbf{u}_0 \ , \ \mathbf{x} \in E_1\}$. (Matrix D probably won't be diagonal this time since the 'fit' of E_1 into Σ is likely to require rotation of E_1 , but for the present purposes we can ignore this). Let $\Omega' = \phi_1^{-1}(\Omega)$. Transform the problem in $E_1 \cap \Omega$ to one in $E_0 \cap \Omega'$, and map the point $\mu_{E_0\cap\Omega',\lambda'}$ back to E_1 to get $\mathbf{u}_2$.

Evidently we can continue in this way to find $\mathbf{u}_3 = \mu_{E_2\cap\Omega}$ using a mapping ϕ_2^{-1} and compensating function λ_2' defined analogously to ϕ_1^{-1} and λ_1' , then define E_3 , $\mathbf{u}_4$, etc. As in §1.4.1 'it looks as though' the $\mathbf{u}_i$ converge to a point $\tilde{\mathbf{u}} = \mu_{\Sigma\cap\Omega}$, but this time we will look seriously at the rate of convergence.

2.2.2. I'll examine the convergence of the first few points $\mathbf{u}_i$ in fig. 8 in terms of the differences $\mathbf{cu}_i - \mathbf{c\tilde{u}}$ of the objective values, that is, we are looking at the convergence of objective values $\mathbf{cu}_i$ to $\mathbf{c\tilde{u}}$ rather than convergence of locations $\mathbf{u}_i$ to $\tilde{\mathbf{u}}$. A sequence of points $\mathbf{u}_i$ which converges in this weak sense might not converge in the usual sense; however convergence of objective values is good enough to find some $\mu_{\Sigma\cap\Omega}$ (this does mean though that there is little to be gained from drawing "the point $\tilde{\mathbf{u}}$ " in these figures).

According to his paper [1], the first thing Karmarkar noticed was that for the first easy step, i.e. finding the minimal point of the ball E_0 , the inequality

$$\frac{\lambda(\mathbf{u}_1) - \lambda(\tilde{\mathbf{u}})}{\lambda(\mathbf{u}_0) - \lambda(\tilde{\mathbf{u}})} \leq 1 - \frac{1}{2n} \qquad (2.2.1)$$

where n is the dimension of Σ , gives the improvement of the 'objective distance' $\lambda(\mathbf{u}_1) - \lambda(\tilde{\mathbf{u}})$ over objective distance $\lambda(\mathbf{u}_0) - \lambda(\tilde{\mathbf{u}})$.

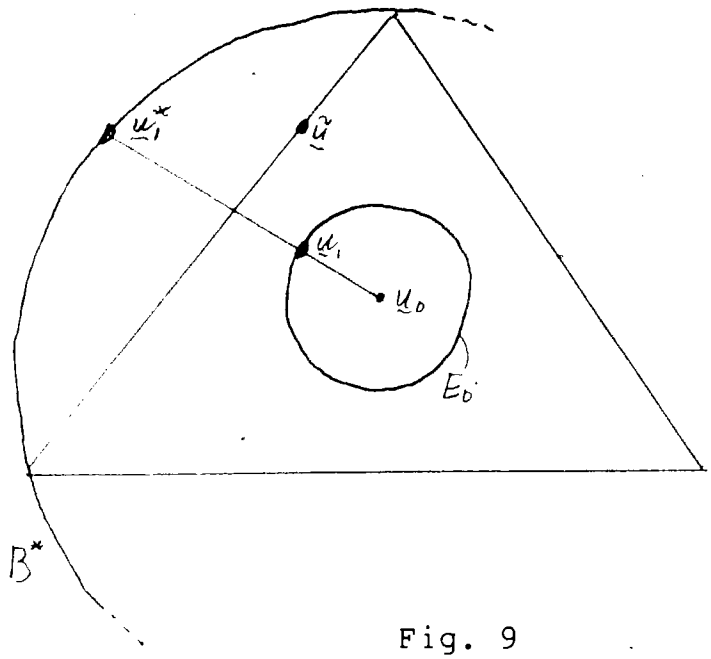

Fig. 9

Proof of inequality (2.2.1): define E_0^* to be the escribed ball of Σ . E_0^* is taken to be a subset of the hyperplane J . The number 2n appears in (2.2.1) because 2n is the ratio $\dfrac{\text{radius}(E_0^*)}{\text{radius}(E_0)}$ (this is easily computed from the geometry). Find $\mathbf{u}_1 = \mu_{E_0 \cap \Omega}$, $\mathbf{u}_1^* = \mu_{E_0^* \cap \Omega}$ from Theorem I (1.2.2). These points are feasible, and Theorem I implies that the terminal points of $\mathbf{u}_0$, $\mathbf{u}_1$, $\mathbf{u}_1^*$ are collinear as shown in fig. 9. Now fig. 9 shows that

$$\mathbf{u}_0 - \mathbf{u}_1^* = \frac{\text{radius}(E_0^*)}{\text{radius}(E_0)} (\mathbf{u}_0 - \mathbf{u}_1) = 2n(\mathbf{u}_0 - \mathbf{u}_1) \tag{2.2.2}$$

and by the linearity of λ

$$\lambda(u_0) - \lambda(u_1^*) = 2n(\lambda(u_0) - \lambda(u_1)) \ . \qquad (2.2.3)$$

Then, since $\lambda(\tilde{u}) \geq \lambda(u_1^*)$ (because $\tilde{u}$ is in the ball E_0^* with minimal point u_1^*), we have

$$\begin{aligned} \lambda(u_0) - \lambda(\tilde{u}) &\leq \lambda(u_0) - \lambda(u_1^*) \\ &= 2n(\lambda(u_0) - \lambda(u_1)) \ . \end{aligned} \qquad (2.2.4)$$

Now (2.2.1) follows from the computation

$$\lambda(u_0) - \lambda(\tilde{u}) \leq 2n(\lambda(u_0) - \lambda(u_1))$$

$$\frac{1}{2n} \leq \frac{\lambda(u_0) - \lambda(u_1)}{\lambda(u_0) - \lambda(\tilde{u})}$$

$$-\frac{1}{2n} \geq \frac{\lambda(u_1) - \lambda(u_0)}{\lambda(u_0) - \lambda(\tilde{u})}$$

$$1 - \frac{1}{2n} \geq \frac{\lambda(u_1) - \lambda(u_0) + \lambda(u_0) - \lambda(\tilde{u})}{\lambda(u_0) - \lambda(\tilde{u})}$$

$$= \frac{\lambda(u_1) - \lambda(\tilde{u})}{\lambda(u_0) - \lambda(\tilde{u})} \qquad (2.2.5)$$

This idea applies so far only to ball E_0 ; but we have the method of transformation, which makes ellipsoids behave in some ways like balls. We could be forgiven for wishful thinking along these lines: could we use the above argument plus the method of transformation to show that an inequality like (2.2.1) governs the convergence of the sequence $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots$ at each point $\mathbf{u}_i$? By this vague phrase I mean something like the following:

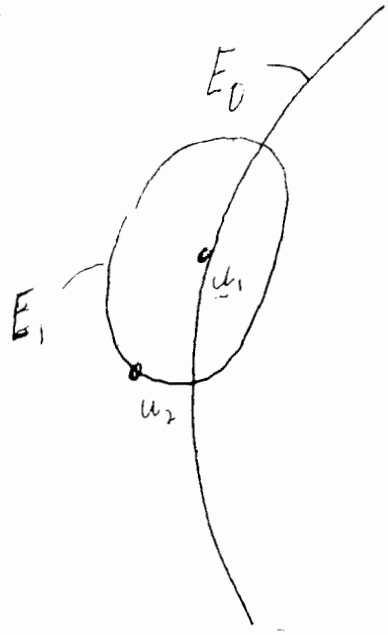

Fig. 10

Look at fig. 10, which shows E_1, $\mathbf{u}_1$, $\mathbf{u}_2$, etc. Let $\phi_1^{-1}(\mathbf{x}) = \mathbf{x}'$; Could we use the method of transformation to prove the analogue of (2.2.1) for E_1 ? i.e. can we prove

$$\frac{\lambda(u_2) - \lambda(\tilde{u})}{\lambda(u_1) - \lambda(\tilde{u})} \leq 1-\frac{1}{2n} \tag{2.2.7}$$

by using ϕ_1^{-1} to transform the problem from E_1 to E_0 as suggested by fig. 11, letting $\lambda_1' = \lambda\phi$, and proving

$$\frac{\lambda_1'(u_2') - \lambda_1'(\tilde{u}')}{\lambda_1'(u_1') - \lambda_1'(\tilde{u}')} \leq 1-1/2n \quad ? \tag{2.2.8}$$

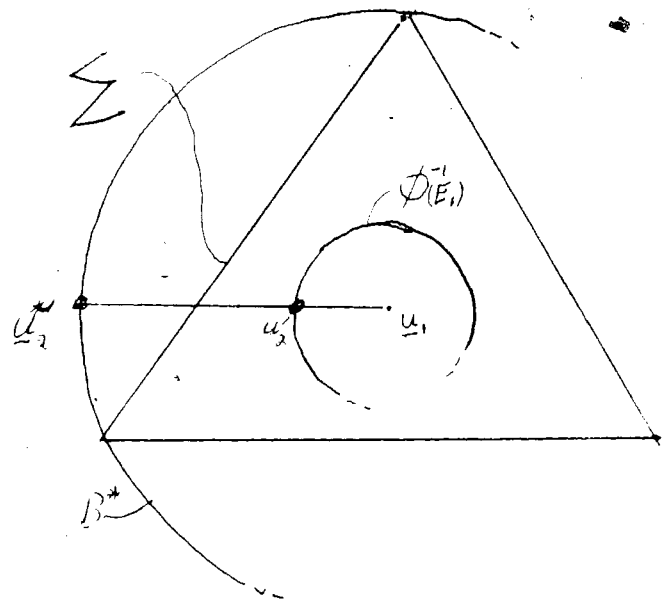

Fig. 11

Then, since $\lambda_1'(x') = \lambda(x)$, (2.2.8) would imply (2.2.7) by the usual argument. The fact that fig. 11 looks a great deal like fig. 9 adds plausibility to this idea. If this approach turned out to work, might we even hope to get the uniform ratio

$$\frac{\lambda(u_i) - \lambda(\tilde{u})}{\lambda(u_{i-1}) - \lambda(\tilde{u})} \leq 1-1/2n \qquad (2.2.9)$$

for each point in the sequence $u_1, u_2, \ldots$? The answer is no, but the idea is attractive because a nice thing that would happen if (2.2.9) were true is that then we could write

$$\frac{\lambda(u_k) - \lambda(\tilde{u})}{\lambda(u_0) - \lambda(\tilde{u})}$$

$$= \frac{\lambda(u_1) - \lambda(\tilde{u})}{\lambda(u_0) - \lambda(\tilde{u})} \times \frac{\lambda(u_2) - \lambda(\tilde{u})}{\lambda(u_1) - \lambda(\tilde{u})} \times \ldots$$

$$\ldots \times \frac{\lambda(u_k) - \lambda(\tilde{u})}{\lambda(u_{k-1}) - \lambda(\tilde{u})}$$

$$\leq (1-1/2n)^k \,, \qquad (2.2.10)$$

with the following happy result: a standard argument shows that $(1-1/A)^A < 1/2$ for any positive integer A , i.e. $(1-1/2n)^{2n} < 1/2$; so if k is taken to be $5nL$, then

$$\frac{\lambda(u_k) - \lambda(\tilde{u})}{\lambda(u_0) - \lambda(\tilde{u})} < (1/2)^{2.5L} \,,$$

and

$$\lambda(\mathbf{u}_k) - \lambda(\tilde{\mathbf{u}}) \leq 2^{-2.5L}(\lambda(\mathbf{u}_0) - \lambda(\tilde{\mathbf{u}}))$$

$$< 2^{-2.5L}(2^L + 2^L) \quad \text{(Range theorem)}$$

$$< 2^{-L}$$

(assuming a version of the Range theorem appropriate to the n-simplex Σ). This means that if (2.2.9) were true and $k = 3nL$, then $\lambda(\mathbf{u}_k)$ would be so close to the solution $\lambda(\tilde{\mathbf{u}})$ that by the Granularity Theorem, $\lambda(\mathbf{u}_k)$ would be exactly $\lambda(\tilde{\mathbf{u}})$ and $\mathbf{u}_k$ would be a minimal point of the simplex Σ . This would mean that the approximating sequence $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \ldots$ finds a solution to (2.1.1) in k steps, i.e. in a number of steps polynomial in n and L .

2.3. Failure of inequality (2.2.9)

Unfortunately (2.2.9) isn't true if we use the method of transformation as described in §1.4.1, that is if we find the successive points $\mathbf{u}_{i+1} = \mu_{E_i}$ by defining affine mappings ϕ_i and compensating functions λ_i' so that $\phi_i^{-1}(E_i) = E_0$ and $\lambda_i' = \lambda\phi_i$. In the speculative and incorrect argument we gave in the previous section, the important flaw is in the alleged inequality (2.2.8). However Karmarkar altered the method of transformation by changing ϕ and λ' so that something like (2.2.8) is true. The method is complicated; to see why such

elaborate means are necessary I'll look at the reason why (2.2.8) fails if we use the method of transformation as we understand it so far. If (2.2.8) were true in fig. 11, then there would be essentially no barrier to proving (2.2.7) and the more general (2.2.9) etc., leading to the desired polynomial time convergence of the u_i .

Why does (2.2.8) fail? Suppose that we have the situation in fig. 11. Define the mappings ϕ_i^{-1} in the usual way. We will find that the ϕ_i must satisfy two seemingly contradictory requirements if we want to prove (2.2.8) and make the 'wish list' (2.2.7), (2.2.9) (2.2.10) come true. We will find that the ϕ_i must map E_0 into (small) approximating sets E_i but simultaneously map E_0^* into very large sets.

2.3.1. Hard questions about the affinity of ϕ .

The alleged inequality (2.2.8) looks a lot like the true statement (2.2.5), so we will look at fig. 11 and try to prove (2.2.8) by imitating the reasoning which led to statement (2.2.5). To prove statement (2.2.5) we first had to prove

$$u_0 - u_1^* = \frac{\text{radius}(E_0^*)}{\text{radius}(E_0)} (u_0 - u_1) = 2n(u_0 - u_1) \quad (2.2.2)$$

$$\lambda(u_0) - \lambda(u_1^*) = 2n(\lambda(u_0) - \lambda(u_1)) . \quad (2.2.3)$$

$$\lambda(u_0) - \lambda(\tilde{u}) \le \lambda(u_0) - \lambda(u_1^*)$$
$$= 2n(\lambda(u_0) - \lambda(u_1)) \ . \qquad (2.2.4)$$

To attack (2.2.8) we will let $\Omega' = \phi_1^{-1}(\Omega)$, $\lambda_1' = \lambda\phi_1$, $u_1' = \phi_1^{-1}(u_1) = u_0$, $u_2' = \mu_{E_0 \cap \Omega', \lambda'}$, $u_2^{*'} = \mu_{E_1^{*'} \cap \Omega', \lambda'}$ where $E_1^{*'} = E_0^*$, and look at the analogous statements

$$u_1' - u_2^{*'} = 2n(u_1' - u_2') \qquad (2.3.1.1)$$

$$\lambda_1'(u_1') - \lambda_1'(u_2^{*'}) = 2n(\lambda_1'(u_1') - \lambda_1'(u_2')) \qquad (2.3.1.2)$$

which are like (2.2.2) and (2.2.3) and are proved in the same way (n.b. (2.3.1.2) uses the affine property of $\lambda_1' = \lambda\phi$). Now move everything to ellipsoid E_1 . Since $\lambda_1'(x') = \lambda_1(x)$,

$$\lambda(u_1) - \lambda(u_2^*) = 2n(\lambda(u_1) - \lambda(u_2)) \qquad (2.3.1.3)$$

which is analogous to (2.2.3). Note that we must define u_2^* to be $\phi_1(u_2^{*'})$. From previous arguments we see that u_2^* is the minimal point of $E_1^* = \phi_1(E_0^*)$.

The trouble appears when we try to prove an inequality analogous to (2.2.4), viz.

$$\lambda(u_1) - \lambda(\tilde{u}) \le \lambda(u_1) - \lambda(u_2^*)$$
$$= 2n(\lambda(u_1) - \lambda(u_2)) \ . \qquad (2.3.1.4)$$

Looking at the way (2.2.4) was proved we see that to prove

(2.3.1.4) we need

$$\lambda(\tilde{\mathbf{u}}) \geq \lambda(\mathbf{u}_2^*) \ ,$$

that is, there must be some minimal point $\tilde{\mathbf{u}}$ in E_1^* . But fig. 12 shows that (2.3.1.4) could be false.

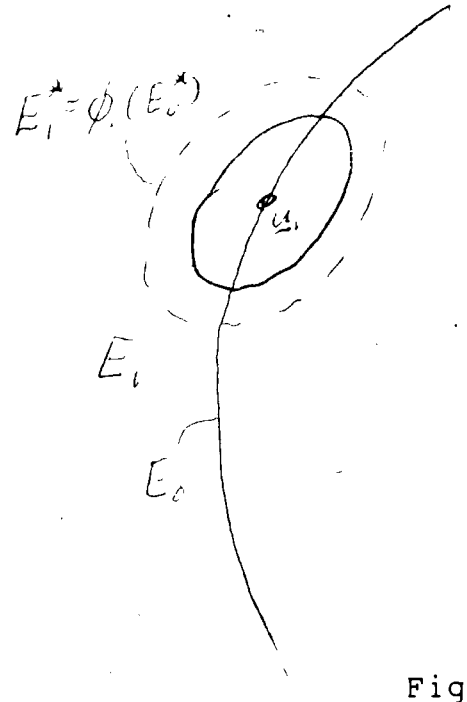

Fig. 12

When we proved (2.2.4) we used the fact that $\tilde{\mathbf{u}} \in E_0^*$, which is true because E_0^* contains all of Σ . But $E_1^* = \phi_1(E_0^*)$ could be small and miss every minimal point $\tilde{\mathbf{u}}$.

There are strong arguments that this difficulty is wedded to the fact that ϕ_1 is an affine map (essentially, if ϕ_1 is affine and $\phi_1(E_0) = E_1$, then $\phi_1(E_0^*) = E_1^*$ can be nothing else but the E_1^* shown in fig. 12). Karmarkar solved the problem boldly by changing ϕ_i to a <u>non-affine</u> mapping Φ_i which maps E_0^* to an image $\Phi_i(E_0^*) = \Gamma_1^*$ which contains every

minimal point (and in fact all of Σ); then he had to fix problems caused by the definition of the new mapping Φ_i; this was done mostly by using an ingenious variation of the method of transformation. Altering the mapping ϕ so that the equivalent of E_1^* in fig. 12 contains all of Σ as Karmarkar did is extremely audacious. One would wonder what kind of sequence of mappings Φ_i could map E_0 to a sequence $\Phi_1(E_0)$, $\Phi_2(E_0)$, ... which approximates $\mu_{\Sigma\cap\Omega}$ while at the same time mapping E_0^* so that every $\Phi_i(E_0^*)$ is bigger than Σ ! For the rest of this section 2 we describe Karmarkar's algorithm. The general form of the algorithm is like that of the naive ellipsoidal approximation just described. We concentrate on the problem of obtaining an inequality analogous to (2.2.4) – once this is done we quickly get the desired fast convergence. But matters are complicated by the details of the mappings Φ_i and by the fact that to use the Φ_i we must introduce a downright tricky version of the method of transformation.

2.4. Description of Karmarkar's algorithm: the pseudo-ellipsoid series.

For the purposes (only) of motivating the appearance of the unusual features of Karmarkar's algorithm, we can imagine that Karmarkar set out to patch the flawed proof of the wish list (2.3.7),(2.3.9),(2.3.10) (in actual fact, of course, I

have no right to pretend to know what Karmarkar was thinking). Karmarkar got the wish list, or something very like it, to work by introducing two new ideas into the situation in fig. 8.

The first idea was to replace the ellipsoids E_i in fig. 8 by pseudo-ellipsoids Γ_i which will be described in the next sections. The Γ_i are subsets of Σ that act like the E_i in fig. 8. Γ_0 is the inscribed ball B of Σ . For $i > 0$ the Γ_i are defined by the pseudo-elliptical mappings Φ_i , thus $\Gamma_i = \Phi_i(\Gamma_0)$. The Φ_i act like the ϕ_i in §2.2 to define an approximating sequence $\Gamma_0, \Gamma_1, \ldots$ but additionally they always map Γ_0^* , which is just B*, so that the image contains Σ and hence every minimal point $\mu_{\Sigma\cap\Omega}$. The Φ_i are (and must be) non-affine and can behave in an extremely non-linear fashion. Karmarkar's second important idea is a new way to apply the method of transformation. This will be described in §2.7 et seq.

We now begin the construction of Karmarkar's algorithm, using the same general approach as that used in §2.2.1. Given LPP (2.1.1.) we will define an approximating sequence of points of Σ , viz. $v_0, v_1, v_2, v_3, \ldots$ with $v_i \in \mathrm{Int}(\Sigma)$. We also construct a sequence of pseudo-ellipsoids $\Gamma_0, \Gamma_1, \Gamma_2, \ldots$. Begin by taking $\mathbf{p} = v_0$ and $\Gamma_0 = B$, the inscribed ball. We easily find the point

$\mu_{\Gamma_0 \cap \Omega, \lambda}$ using Theorem I (1.2.2). Then v_1 will be a point located at a certain fraction $\alpha < 1$ of the distance along the line connecting v_0 and $\mu_{\Gamma_0 \cap \Omega}$. The parameter α is needed later for the convergence arguments and immediately to ensure that $v_1 \in \mathrm{Int}(\Sigma)$. Eventually α will be something like 1/4. See fig. 13.

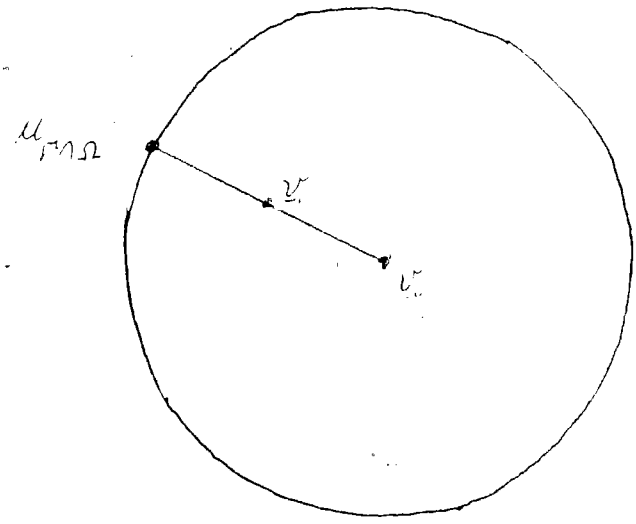

Fig. 13

Point v_1 will be the 'pseudocentre' of the next pseudoellipse Γ_1 in a sense to be explained. In general, each v_{i+1} is a point in the interior of Γ_i ; then v_{i+1} is the pseudocentre of $\Gamma_{(i+1)}$ .. v_{i+1} is a point located at a fraction $\alpha < 1$ of the distance along the line connecting the pseudocentre v_i to a point on the boundary of Γ_i which for the moment we can think of as being $\mu_{\Gamma_i \cap \Omega}$; later we will see that the 'new way to apply the method of transformation' causes this not to be

quite true. It is important that $v_i \in \mathrm{Int}(\Gamma_i) \subset \mathrm{Int}(\Sigma)$ since our proofs often require $v_i > 0$. The pretty unsatisfactory fig. 14 attempts to suggest that the Γ_i can be thought of as distorted ellipses which 'direct' the v_i so that they approach $\tilde{v} = \mu_{\Sigma\cap\Omega}$ just as the E_i direct the u_i toward $\tilde{u}$ in fig. 9. Fig. 14 is not supposed to help the reader visualize the Γ_i , which are extraordinarily hard to 'see', but merely to suggest that the Γ_i are 'something like ellipses', that they lie entirely in Σ and that the pseudocentres v_i are not 'in the middle of the Γ_i' in any obvious way.

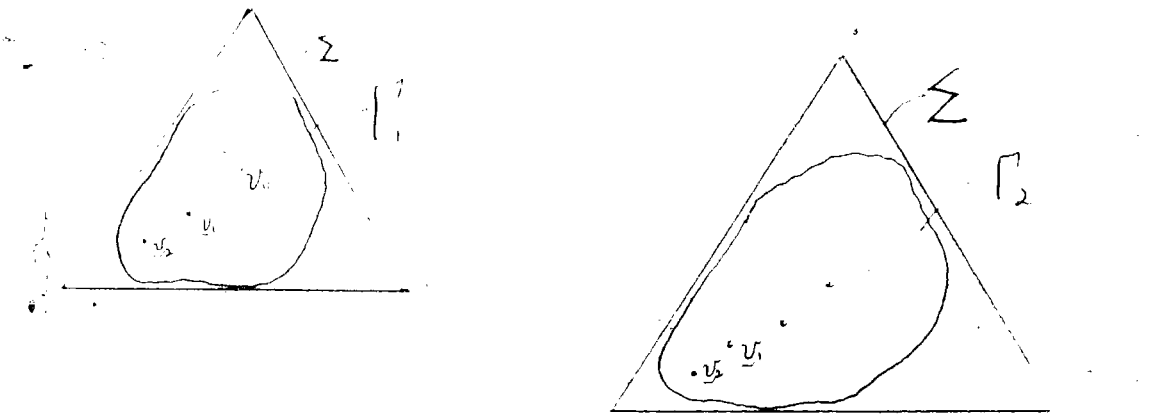

Fig. 14

2.5. Pseudo-elliptical mappings and pseudo-ellipsoids. At the ith stage of the approximation we will have constructed points v_0, v_1, v_2, ... v_i . We now want to construct the pseudo-ellipsoid Γ_i of which v_i is the pseudocentre. The

pseudo-ellipsoid Γ_i is the image of the inscribed ball B of Σ under a non-singular mapping Φ_i which carries the centre **p** of Σ onto $\mathbf{v}_i$, fixes every vertex of Σ , and fixes the whole of Σ setwise. This behaviour requires Φ_i to be non-affine, since Φ_i must map n+2 points onto prescribed locations in E_{n+1} , that is, $\Phi_i(\mathbf{v}_i) = \mathbf{p}$ and at the same time Φ_i must fix the n+1 vertices of Σ . We define $\Gamma_i = \Phi_i(B)$. The inverse mapping Φ_i^{-1} is used like ϕ_i^{-1} in fig 8, i.e. we use the fact that $\Phi_i^{-1}(\Gamma_i) = B$ just where we used $\phi_i^{-1}(E_i) = E_0$ in fig. 8.

The definition of Φ_i depends strongly on $\mathbf{v}_i$. If $\mathbf{x} \in J$, $\Phi_i(\mathbf{x})$ is the composition of a linear mapping L_i of **x** into the larger space E_{n+1} with a nonlinear mapping M which squashes the image in E_{n+1} back into Σ . Definition of L_i : if $\mathbf{x} \in J$, then $L_i\mathbf{x} = D_i\mathbf{x}$, where D_i is the diagonal matrix

$$D_i = \begin{bmatrix} d_1 & & 0 \\ & d_2 & \\ & & \ddots \\ 0 & & d_{n+1} \end{bmatrix} .$$

The diagonal elements d_j of D_i are just the $(\mathbf{v}_i)_j$, where $(\mathbf{v}_i)_j$ stands for the jth component of $\mathbf{v}_i$. Thus D_i maps (n+1)**p** onto $\mathbf{v}_i$.

Definition of the mapping M : if $\mathbf{y} = D_i\mathbf{x}$ then $M(\mathbf{y}) = \mathbf{y}/\Sigma_j(\mathbf{y})_j$. For $\mathbf{x} \in J$, $\Phi_i(\mathbf{x}) = M(L_i\mathbf{x})$. If $\mathbf{e} = [1,1,1, \ldots ,1]^T$, then

$\Phi(\mathbf{x})$ can be written as $\dfrac{D_i\mathbf{x}}{\mathbf{e}^T D_i\mathbf{x}}$. We must show that the denominator $\mathbf{e}^T D_i\mathbf{x}$ is non-zero. The reason is that $\mathbf{e}^T D_i\mathbf{x}$ is a sum of terms $(\mathbf{x})_j(v_i)_j$ (from the definition of D_i), which is not zero because $v_i > \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$ with some component positive.

2.5.1. Assume that Φ_i has been defined with respect to some $v_i \in \text{Int}(\Sigma)$. The important properties of Φ_i are

1) $\Phi_i(\mathbf{p}) = v_i$ (easily computed).

2) Φ is non-singular considered as a mapping of J onto itself, and its inverse is given by

$$\Phi_i^{-1}(\mathbf{x}) = \frac{D_i^{-1}\mathbf{x}}{\mathbf{e}^T D_i^{-1}\mathbf{x}}$$

(can be checked straightforwardly)

3) $\Phi_i(\Sigma) = \Sigma$.

Proof: let $\mathbf{x} \in \Sigma$. We know $v_i > \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$ with some $(\mathbf{x})_j > 0$. Thus $\mathbf{e}^T D_i\mathbf{x} > 0$, $D_i\mathbf{x} \geq \mathbf{0}$, and $\dfrac{D_i\mathbf{x}}{\mathbf{e}^T D_i\mathbf{x}} = \Phi_i(\mathbf{x}) \geq \mathbf{0}$, which means $\Phi_i(\mathbf{x}) \in P_+$.

From the definition all components of $\Phi_i(\mathbf{x})$ sum to 1; therefore $\Phi_i(\mathbf{x})$ lies in $P_+ \cap J = \Sigma$.

This shows that $\Phi_i(\Sigma) \subset \Sigma$. A similar argument shows that $\Phi_i^{-1}(\Sigma) \subset \Sigma$. Hence, since Φ_i is non-singular, $\Phi_i(\Sigma) = \Sigma$.

4) All of $\Gamma_i = \Phi_i(B)$ lies in Σ . All of Σ lies in $\Phi_i(B^*)$. All minimal points in LPP (2.1.1) map under Φ_i^{-1} into B^*. Here B (resp. B^*) is the inscribed (escribed) n-ball of Σ .
(Three corollaries to 3) above).

5) Φ_i carries affine spaces onto affine spaces.
Proof: I'll just prove this for the case that will be needed later, viz. that if $\mathbf{x} \in \Sigma \cap \Omega$, then Φ_i^{-1} carries $\mathbf{x}$ into an affine space of the same dimension intersected with Σ . Specifically, I'll show that if Ω is given by $A\mathbf{x} = \mathbf{0}$ and $\mathbf{x} \in \Omega \cap \Sigma$, then $\mathbf{x}' = \Phi_i^{-1}(\mathbf{x}) \in \Omega' \cap \Sigma$, where Ω' is given by $AD_i\mathbf{x} = \mathbf{0}$. This follows from the computation

$$\begin{aligned} \mathbf{0} = A\mathbf{x} &= AD_iD_i^{-1}\mathbf{x} \\ &= AD_iD_i^{-1}\mathbf{x}/\mathbf{e}^TD_i\mathbf{x} \qquad \text{(check that denom.}\neq 0\text{)} \\ &= AD_i\mathbf{x}'. \end{aligned}$$

Thus $\mathbf{x}'$ is in the null space of AD_i , which has

the same dimension as the null space of A because matrix D is nonsingular. Finally we know from 3) that $\mathbf{x}' \in \Sigma$.

2.6. The transformation method refined: two assumptions.

In my discussion of 'where (2.2.4) went wrong' in §2.3, I concentrated on the fact that although the mappings ϕ_i map E_0 into a sequence $\phi_1(E_0)$, $\phi_2(E_0)$, $\phi_3(E_0)$, ... which approximates some $\tilde{\mathbf{u}}$, they also do something we don't want, which is to map E_0 (the escribed ball of Σ) so that the image is small and could miss every $\tilde{\mathbf{u}}$. The failure occurred in (2.3.1.4), whose proof will not work unless there is a $\tilde{\mathbf{u}}$ in every $\phi_i(E_0)$. Now if we replace the ϕ_i by the pseudo-elliptical mappings Φ_i , then (I won't give the detailed argument) the $\Phi_i(E_0) = \Gamma_i$ will approximate $\tilde{\mathbf{u}}$ and also each $\Phi_i(E_0^*)$ will contain a $\tilde{\mathbf{v}}$; so in this respect the Φ_i succeed where the ϕ_i fail. We could ask if we can now prove the wish list (2.3.7), (2.3.9), (2.3.10) taking into account the change from ϕ_i to Φ_i etc., but the answer is : not without altering the method of transformation. In §2.3 we looked carefully at (2.3.1.1),(2.3.1.2), etc. Notice how the affinity of ϕ_i was used in (2.3.1.2) to change the ratio

$$\frac{\|\mathbf{u}_1' - \mathbf{u}_2^{*\prime}\|}{\|\mathbf{u}_1' - \mathbf{u}_2'\|} = 2n$$

into a ratio of objective values

$$\frac{\lambda_1'(\mathbf{u}_1') - \lambda_1'(\mathbf{u}_2^{*\prime})}{\lambda_1'(\mathbf{u}_1') - \lambda_1'(\mathbf{u}_2')} = 2n .$$

Since Φ_i is non-affine, this part of the argument will not work if Φ_i replaces ϕ_i . Karmarkar made something just as good work by introducing an ingenious variation of the method of transformation which uses properties of Φ_i to transform the ratios. I'll give a brief outline of this variation (more detail in §2.4): it's easy to find $\mu_{B\cap\Omega,\lambda}$, and it's clearly true that

$$n(\mathbf{p} - \mu_{B\cap\Omega',\lambda}) = \mathbf{p} - \mu_{B^*\cap\Omega,\lambda} .$$

We want to preserve this ratio in something involving $\lambda_i' = \lambda\Phi_i$, $\Omega_i = \phi_i(\Omega)$. It would be nice to prove

$$\frac{\lambda'(\mathbf{p}) - \lambda'(\mu_{B\cap\Omega_i,\lambda'})}{\lambda'(\mathbf{p}) - \lambda'(\mu_{B^*\cap\Omega_i,\lambda'})} \leq 1/n ,$$

which is like (2.2.3), but it's not clear how since λ' is not affine. In the variant of the method which we are about to introduce, we replace λ_i' by λD_i and get interested in

$$\frac{\lambda D_i\mathbf{p} - \lambda D_i\mu_{B\cap\Omega',\lambda D_i}}{\lambda D_i\mathbf{p} - \lambda D_i\mu_{B^*\cap\Omega',\lambda D_i}} \leq 1/n$$

which is true since λD_i is affine. The reason for the appearance of λD_i at this point isn't obvious.

However now we introduce a whole new (and complicated) function f , whose properties, speaking very roughly, allow us to use λ and λD_i as if they were a 'compensating pair', i.e as if $\lambda D_i \Phi_i^{-1}(\mathbf{x}) = \lambda(\mathbf{x})$. Then a logarithmic version of (2.2.9) and in fact of the whole wish list turns out to be true.

In §2.7 I'll define the new function f and describe the analytic ideas that make f central to the operation of the algorithm. But before doing so, I need to mention that to make the analysis work Karmarkar has to impose two assumptions that he later shows how to remove. The first assumption is that the first feasible point is $\mathbf{p}$ = center of Σ . In [1] Karmarkar eventually removes this assumption by showing how to start his algorithm at the centre $\mathbf{p}$ of Σ regardless of the feasibility of $\mathbf{p}$ and without having obtained a first feasible point, however I will omit this part of the algorithm and assume that a first feasible point has been provided by the first phase of a two phase method.

The second assumption is that the minimal objective function value $\lambda(\tilde{\mathbf{v}})$ must be a known constant – for our purposes zero. This almost sounds like a promise that our complex machinery will solve the LPP provided that we know the answer to begin with!! The reason that this assumption does no harm is that if we know how to construct a sequence to approximate $\lambda(\tilde{\mathbf{v}})$ in the case where we know that $\lambda(\tilde{\mathbf{v}})$ is a given constant, we can find the answer for the case where we don't know $\lambda(\tilde{\mathbf{v}})$ by the following procedure which I will only sketch roughly: guess a value MIN for $\lambda(\tilde{\mathbf{v}})$ and approximate $\lambda(\tilde{\mathbf{v}})$ assuming that $\lambda(\tilde{\mathbf{v}})$ = MIN. If we have made a wrong guess then the approximation $f(\mathbf{v}_i) \longrightarrow \lambda(\tilde{\mathbf{v}})$ doesn't converge to MIN, and the approximation arrow points in the direction of a better $\lambda(\tilde{\mathbf{v}})$. This direction is used to 'aim' the next stage of a binary search over possible values of $\lambda(\tilde{\mathbf{v}})$.

2.7. <u>The new objective function f ; origin and definition</u>.

Assume the LPP in the Karmarkar Standard Form (2.1.1), with $\lambda(\mathbf{x}) = \mathbf{cx}$. Let $\mathbf{x} \in \mathrm{Int}(\Sigma)$ and write $(\mathbf{x})_j$ for the jth component of $\mathbf{x}$. Then we define a new objective function f by

$$f(\mathbf{x}) = \Sigma_j \log \frac{\lambda(\mathbf{x})}{(\mathbf{x})_j}$$

The function f looks odd, with its repetition of $\lambda(\mathbf{x})$ n times and its division by the product of all the components of $\mathbf{x}$.

Function f could also be written as

$$(n+1)\ \log\ \lambda(\mathbf{x})\ -\ \Sigma_j \log\ (\mathbf{x})_j\ .$$

Recall that $\lambda(\mu_{\Sigma\cap\Omega}) = 0$ by the special (and essential) assumption. This means that $f(\mu_{\Sigma\cap\Omega})$ is undefined, or $-\infty$ if you wish. Points $\mu_{\Sigma\cap\Omega}$ behave like 'poles', and f is a potential function on the field containing the points $\mu_{\Sigma\cap\Omega}$. The way the function f is used is that the sequence $\{\mathbf{v}_i\}$ (all the $\mathbf{v}_i$ will be in $\mathrm{Int}(\Sigma)$) will be defined by a procedure like the one in fig. 13 but involving the special objective function f ; then it will turn out that the values $f(\mathbf{v}_i)$ are pushed progressively more negative; in fact we will get

$$f(\mathbf{v}_{i+1})\ -\ f(\mathbf{v}_i)\ <\ -\ \frac{1}{8}\ . \qquad (2.7.1)$$

uniformly for every $i = 1, 2, 3, \ldots$. Thus $f(\mathbf{v}_i) \longrightarrow -\infty$, which drives $\lambda(\mathbf{v}_i)$ toward zero, since

$$f(\mathbf{x}) = \Sigma_j \log\ \frac{\lambda(\mathbf{x})}{(\mathbf{x})_j}$$

$$= (n+1)\ \log\ \lambda(\mathbf{x})\ -\ \Sigma_j \log\ (\mathbf{x})_j$$

$$\geq (n+1)\ \log\ \lambda(\mathbf{x})\ .$$

(To check the last line note that $0 < (\mathbf{x})_j < 1$ because

$\mathbf{x} \in \text{Int}(\Sigma)$). Thus if $f(\mathbf{x}) \longrightarrow -\infty$, then $\log \lambda(\mathbf{x}) \longrightarrow -\infty$ and $\lambda(\mathbf{x}) \longrightarrow 0$.

We will also need the compensating function $f_i' = f\Phi_i$, which is

$$f_i'(\mathbf{x}) = \Sigma_j \log \frac{\lambda D_i \mathbf{x}}{(\mathbf{x})_j} - \Sigma_j \log d_j$$

where the d_j are the diagonal entries $d_1, d_2, \ldots d_{n+1}$ of the diagonal matrix D_i . There is a different f_i' for each D_i . Let $\Phi_i^{-1}(\mathbf{x}) = \mathbf{x}'$. As an example of computing with f', I'll show that $f_i'(\mathbf{x}') = f(\mathbf{x})$:

$$\begin{aligned} f_i'(\mathbf{x}') &= f_i'(\Phi_i^{-1}(\mathbf{x})) \\ &= \Sigma_j \log \frac{\lambda D_i \Phi_i^{-1}(\mathbf{x})}{(\Phi_i^{-1}(\mathbf{x}))_j} - \Sigma_j \log d_j \\ &= \Sigma_j \log \frac{\lambda D_i D_i^{-1}\mathbf{x}/\mathbf{e}^T D_i^{-1}\mathbf{x}}{(D_i^{-1}\mathbf{x})_j/\mathbf{e}^T D_i^{-1}\mathbf{x}} - \Sigma_j \log d_j \\ &= \Sigma_j \log \frac{\lambda(\mathbf{x})/\mathbf{e}^T D_i^{-1}\mathbf{x}}{(D_i^{-1}\mathbf{x})_j/\mathbf{e}^T D_i^{-1}\mathbf{x}} - \Sigma_j \log d_j \\ &= \Sigma_j \log \frac{\lambda(\mathbf{x})}{(D_i^{-1}\mathbf{x})_j} - \Sigma_j \log d_j \end{aligned}$$

Now since D_i^{-1} is diagonal, each quotient $(D_i^{-1}\mathbf{x})_j$ above is just $\frac{1}{d_j}(\mathbf{x})_j$, and the whole expression becomes

$$\Sigma_j \log \left[\frac{\lambda(\mathbf{x})}{(\mathbf{x})_j} \cdot d_j\right] - \Sigma_j \log d_j$$

$$= \Sigma_j \log \frac{\lambda(\mathbf{x})}{(\mathbf{x})_j} + \Sigma_j \log d_j - \Sigma_j \log d_j$$

$$= \Sigma_j \log \frac{\lambda(\mathbf{x})}{(\mathbf{x})_j}$$

which is exactly $f(\mathbf{x})$. This is true for any positive integer i.

I'll now describe the ideas that motivate the appearance of the admittedly odd-looking function f . Go back to fig. 13 and imagine that we are trying to construct the sequence $\{\mathbf{v}_i\}$ and have produced the easy point $\mathbf{v}_1$, which is near (and can be assumed for now to be) $\mu_{B\cap\Omega,\lambda}$. We have constructed $\mathbf{v}_2$, $\mathbf{v}_3$, ... $\mathbf{v}_i$ successfully and we now want $\mathbf{v}_{i+1}$ which we would like to be a point such that

$$\frac{c\mathbf{v}_{i+1} - c\tilde{\mathbf{v}}}{c\mathbf{v}_i - c\tilde{\mathbf{v}}} \leq 1-1/n$$

The discussion in (2.3.1), (2.3.2) has shown that the

construction of a v_{i+1} with the above property is the crux of the problem. Karmarkar found v_{i+1} using a variation on the method of transformation.

Let $\Phi_i^{-1}(\Omega) = \Omega_i$ and look at $\Phi_i^{-1}(\Gamma_i \cap \Omega) = B \cap \Omega_i$ using a special objective function λD_i , which is just part of the function f' . Check that $\mathbf{p}$ is feasible in the image $B \cap \Omega_i$ using properties of Φ_i^{-1} . Remember that $\lambda D_i \Phi_i^{-1}(\mathbf{x}) = \lambda(\mathbf{x})$ is not true since these mappings are not compensating pairs. We easily find $\mu_{B\cap\Omega_i, \lambda D_i}$ from Th. I (1.2.2) since λD_i is a nice linear function. Let $\hat{\mathbf{v}}$ be the point at a fraction α of the distance from $\mathbf{p}$ in the direction of $\mu_{B\cap\Omega_i, \lambda D_i}$. In the sequel I'll assume that α, the steplength parameter, is always 1/4. Karmarkar gives this as a practical value for α ; I won't discuss the choice of values for the steplength parameter since this is an element in a hard piece of analysis in [1] which I intend to omit. Evidently another way to define $\hat{\mathbf{v}}$ would be as $\mu_{B(\mathbf{p},r/4)\cap\Omega_i, \lambda D_i}$, where $B(\mathbf{p},r/4)$ stands for the sub-n-ball of B with centre $\mathbf{p}$ and radius r/4. This will be needed when we refer in detail to [1]. We also easily get $\hat{\mathbf{v}}^* = \mu_{B^*\cap\Omega_i, \lambda D_i}$.

Now we pull the rabbit out of the hat, because point $\Phi_i(\hat{\mathbf{v}})$ is going to be v_{i+1} . The rest of this section describes the analytic argument which justifies this assertion. The description will establish (2.7.1) and will

be complete except that I'll omit one passage of lengthy analysis. (2.7.1) is the theorem that makes everything work. In the formal description of Karmarkar's algorithm which follows this section immediately, (2.7.1) establishes fast convergence in about a page.

Following earlier computations in §2.3 and making some obvious changes,

$$4n(\mathbf{p} - \hat{\mathbf{v}}) = \mathbf{p} - \hat{\mathbf{v}}^*$$

(the '4n' coming from the fact that $\alpha = 1/4$). By the affine property of λD_i

$$4n(\lambda D_i \mathbf{p} - \lambda D_i \hat{\mathbf{v}}) = \lambda D_i \mathbf{p} - \lambda D_i \hat{\mathbf{v}}^* \qquad (2.7.2)$$

Now for some $\tilde{\mathbf{v}}$ let ξ be $\Phi_i^{-1}(\tilde{\mathbf{v}})$. By property 4) of §2.5.1, Φ_i^{-1} carries $\tilde{\mathbf{v}}$ into B^* . The fact that Φ_i does this is the point of the whole elaborate definition of Φ_i . It's not known that ξ has any minimal property with respect to λD_i , but we can still say

$$\lambda D_i \xi \geq \lambda D_i \hat{\mathbf{v}}^*$$

since $\xi \in B^*$, and go on to compute

$$\frac{\lambda D_i \hat{\mathbf{v}} - \lambda D_i \xi}{\lambda D_i \mathbf{p} - \lambda D_i \xi} \leq 1 - \frac{1}{4n} \qquad (2.7.3)$$

using a computation modelled exactly on the one which

produced (2.2.1) from (2.2.4). Now $\lambda D_i \xi = 0$, since

$$\begin{aligned} 0 &= \lambda(\tilde{v}) \\ &= \lambda D_i D_i^{-1} \tilde{v} \\ &= \lambda D_i D_i^{-1} \tilde{v} / e^T D_i^{-1} \tilde{v} \qquad \text{(check that denom.} \neq 0) \\ &= \lambda D_i \Phi_i^{-1}(\tilde{v}) \\ &= \lambda D_i \xi \end{aligned}$$

Hence we can replace $\lambda D_i \xi$ by 0 in (2.7.3) and get

$$\frac{\lambda D_i \hat{v}}{\lambda D_i p} \leq 1 - 1/4n \ . \qquad (2.7.4)$$

Notice that everything would now work if λD_i were the compensating function of λ with respect to the mapping Φ_i^{-1} ; because then $\lambda D_i \Phi_i^{-1}(x) = \lambda(x)$ would be true and at the ith stage, $p = v_i$, $\hat{v} = v_{i+1}$; so we could say

$$\frac{\lambda(v_{i+1})}{\lambda(v_i)} \leq 1 - 1/4n$$

which is just what we want, i.e. it's a version of (2.2.9) in the wish list; (note also that it's like (2.7.1) if you allow for the logs). However λD_i isn't the correct compensating function, so we use a trick. We will alter λD_i so that it looks like the compensating function of f w.r.t. Φ_i^{-1} and still preserves (2.7.4). First make (2.7.4) into a

logarithmic ratio; thus (2.7.4) becomes

$$\log \lambda D_i \hat{\mathbf{v}} - \log \lambda D_i \mathbf{p} \leq \log (1-1/4n) \ .$$

Make the LHS look more like f':

$$(n+1)\log \lambda D_i \hat{\mathbf{v}} - (n+1)\log \lambda D_i \mathbf{p} \leq (n+1)\log (1-1/4n) = \log (1-1/4n)^{n+1} \qquad (2.7.5)$$

Prove that $\log (1 - 4/n)^{n+1} < -1/4$:

Proof: By looking at the graphs it's clear that $\log (1-x) \leq -x \quad 0 < x < 1$. Then

$$\log (1-1/4n) \leq -\frac{1}{4n} \qquad \text{since } n > 0$$

$$n \log (1-1/4n) \leq -\frac{1}{4}$$

$$\log (1-1/4n)^{n+1} \leq \log (1-1/4n)^{n} \leq -\frac{1}{4}$$

So the RHS of (2.7.5) can be replaced by $-1/4$. It's encouraging that this inequality is uniform over every ith step of the construction of the $\{\mathbf{v}_i\}$. Now add some odd bits:

$$(n+1)\log \lambda D_i \hat{\mathbf{v}} - (n+1)\log \lambda D_i \mathbf{p} - \Sigma_j \log \frac{(\hat{\mathbf{v}})_j}{(\mathbf{p})_j} \leq -\frac{1}{8} \qquad (2.7.6)$$

where $(\mathbf{x})_j$ means the jth component of $\mathbf{x}$. In (2.7.6) we have added a term which 'trims' the R.H.S. to $-1/8$. Justification of (2.7.6) takes a long passage of analysis (several pages) in [1] and I'm going to omit it in this paper. Rearrange

(2.7.6) to get

$$\Sigma_j \log \frac{\lambda D_i \hat{v}}{(\hat{v})_j} - \Sigma_j \log \frac{\lambda D_i p}{(p)_j} < -1/8$$

$$\Sigma_j \log \frac{\lambda D_i \hat{v}}{(\hat{v})_j} - \Sigma_j \log d_j$$

$$- \Sigma_j \log \frac{\lambda D_i p}{(p)_j} + \Sigma_j \log d_j < -1/8$$

$$f_i'(\hat{v}) - f_i'(p) < -1/8 \qquad (2.7.7)$$

In this analysis we always assume that the parameter α used to define $\hat{v}$ is 1/4, i.e. that $\hat{v}$ is the minimal point with respect to λD_i of Ω_i intersected with a ball having centre p and radius equal to a quarter of the radius of B .

It's now only a short step to (2.7.1), for in the construction of any v_{i+1} given v_i , p will be v_i' and v_{i+1}' will be defined to be $\hat{v}$, i.e. v_{i+1} will be $\Phi_i(\hat{v})$. Then we use the basic equation $f_i'(x') = f_i(x)$ to get (2.7.1). In the formal statement of the algorithm which follows immediately after this section, the logarithmic inequality (2.7.1) will be shown to be as good as (2.2.9) for the purpose of driving the $\lambda_i(v_i)$ to the optimal value in polynomial time. Since I

have omitted some of the details of the proof of (2.7.7), I'll quote the result from [1] that establishes (2.7.7), making some obvious notational adjustments. Recall that Ω_i is $\Phi_i^{-1}(\Omega)$, $\mathbf{p}$ is the centre of the inscribed n-ball B of Σ , J is the plane containing Σ and B .

(2.7.8) Theorem 4 of [1]. Let r be the radius of B . Let $\hat{\mathbf{v}}$ be $\mu_{B(\mathbf{p},r/4)\cap\Omega_i,\lambda D_i}$, where $B(\mathbf{p},r/4) \subset J$ stands for the sub-n-ball of B with radius = r/4 and centre $\mathbf{p}$. Then if $\lambda(\hat{\mathbf{v}})$ is not zero (i.e. if we do not obtain the answer to LPP (2.1.1) immediately), we have

$$f_i'(\hat{\mathbf{v}}) - f_i'(\mathbf{v}_i) \leq -\frac{1}{8}$$

This inequality is true for any index i.

We have now described the various ideas which occur in Karmarkar's algorithm. It only remains to give a formal statement of the algorithm.

2.8. Formal statement of Karmarkar's algorithm. Given the "Karmarkar standard form" we assume that $\mu_{\Sigma\cap\Omega,\lambda} = 0$ and that $\mathbf{p}$ = centre of B is feasible. We will construct a sequence of feasible points $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_i$ such that $\lambda(\mathbf{v}_i) \longrightarrow 0$ as i increases. The $\mathbf{v}_i$ have been described in §2.5 as the 'pseudocentres' of the pseudoellipses $\Gamma_1, \Gamma_2, \Gamma_3, \ldots$ but the Γ_i will not be explicitly constructed. By the

granularity theorem there is an integer k such that $\lambda(\mathbf{v}_k) = 0$, and it will turn out that k can be taken to be $= O(nL)$.

2.8.1. Starting the approximating sequence. The first point $\mathbf{v}_0$ of the sequence will be the feasible point $\tilde{\mathbf{p}}$, the centre of Σ . Assume that $\mathbf{v}_i$ has been constructed. The next section describes the construction of $\mathbf{v}_{i+1}$.

2.8.2. Inductive construction of the sequence. The construction of point $\mathbf{v}_{i+1}$ takes three steps, using a variant of the method of transformation.

Step One: construct the transformed affine space $\Omega_i \cap J$ required by (2.7.8). Space Ω is given by

$$A\mathbf{x} = 0$$

and Ω_i is given by

$$AD_i\mathbf{x} = 0$$

(see property 5 in §2.5.1). The intersection with J imposes the additional constraint that feasible points must lie in J . This is done by adding an additional equation to the system $AD_i\mathbf{x} = 0$ which says that the coefficients of $\mathbf{x}$ add to 1. Thus the final representation of the transformed affine space $\Omega_i \cap J$ is

$$\begin{bmatrix} AD_i \\ \hline 1\ 1\ \cdots\ 1 \end{bmatrix} \underline{x} = \begin{bmatrix} \underline{0} \\ \hline 1 \end{bmatrix} .$$

Step two: find the point $\hat{\mathbf{v}}$ described in §2.7. In §2.7, $\hat{\mathbf{v}}$ was defined with respect to a parameter α ; we continue to take $\alpha = 1/4$. The objective function λ in (2.1.1) is given by $\lambda(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$. Define the transformed objective function λ_i' by $\lambda_i'(\mathbf{x}) = \lambda D_i\mathbf{x} = \mathbf{c}^T D_i\mathbf{x}$. Thus the transformed objective vector is $\mathbf{c}' = D_i\mathbf{c}$. This choice of λ_i' is the result of the special form of the method of transformation used here. To find $\hat{\mathbf{v}}$, project the vector $\mathbf{c}'$ into $\Omega_i \cap J$ using the method in the Appendix. Call the projected vector $\mathbf{c}_\pi'$. If r is the radius of ball B, then let $\hat{\mathbf{v}}$ be the point obtained by starting at centre $\mathbf{p}$ and travelling in the direction of $\mathbf{c}_\pi'$ for distance r/4. By Theorem I, plus the fact that $\mathbf{c}_\pi'\mathbf{x} = \mathbf{c}'\mathbf{x}$ for $\mathbf{x} \in J$, $\hat{\mathbf{v}}$ is $\mu_{B(\mathbf{p},r/4)\cap\Omega_i,\lambda D_i^{-1}}$ (for $B(\mathbf{p},r/4)$ see the statement of Theorem 4 of [1] quoted in the previous section.)

Step three: The required point $\mathbf{v}_{i+1} = \Phi_i(\hat{\mathbf{v}})$, taking Φ_i as defined in §2.5.

2.8.3. Termination: For $K = O(nL)$, $\lambda(\mathbf{v}_K)$ is the optimal point of LPP (2.1.1).

2.9. The complexity of the algorithm. Construction of $\hat{\mathbf{v}}$ as $\mu_{B(\mathbf{p},r/4)\cap\Omega_i,\lambda D_i^{-1}}$ satisfies the requirements of Theorem 4 of [1] quoted as (2.7.8). Hence we now know that

$$f_i'(\hat{\mathbf{v}}) - f_i'(\mathbf{v}_i') \leq -\frac{1}{8}$$

where f and f_i' are defined as in §2.7. But by construction $\mathbf{v}_{i+1}$ is $\Phi_i(\hat{\mathbf{v}})$, so taking $\mathbf{v}_{i+1}' = \Phi_i^{-1}(\mathbf{v}_{i+1})$ we can write

$$f_i'(\mathbf{v}_{i+1}') - f_i'(\mathbf{v}_i') \leq -\frac{1}{8}$$

and the equation $f(\mathbf{x}) = f_i'(\mathbf{x}')$ immediately gives

$$f(\mathbf{v}_{i+1}) - f(\mathbf{v}_i) < -\frac{1}{8}$$

uniformly in i, which says that each time we define a new pseudocentre $\mathbf{v}_{i+1}$, we drive the value of the 'pseudo-objective function' f negative by at least 1/8.

Evidently we can define an induction on the sequence $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \ldots$ with $f(\mathbf{v}_{i+1}) - f(\mathbf{v}_i) \leq -1/8$ (the logs in the computation will not work if any $\lambda(\mathbf{v}_i) = 0$; but in this case just let the optimal point be $\mathbf{v}_i$). Then

$$f(\mathbf{v}_i) - f(\mathbf{v}_0) \leq -i/8.$$

As remarked previously this is good enough to give polynomial time convergence. The computation is:

$$f(v_i) \leq f(v_0) - i/8$$

$$\Sigma_j \log \frac{c^T v_i}{(v_i)_j} \leq \Sigma_j \log \frac{c^T v_0}{(v_0)_j} - \frac{i}{8}$$

$$(n+1) \log \frac{c^T v_i}{c^T v_0} \leq \Sigma_j \log (v_i)_j - \Sigma_j \log (v_0)_j - \frac{i}{8} .$$

But since $v_i \in \text{Int}(\Sigma)$, $0 < (v_i)_j < 1$, which means that $\Sigma_j \log (v_i)_j < 0$. Also each $(v_0)_j = 1/(n+1)$ since v_0 is the centre of Σ, so $\Sigma_j \log (v_0)_j$ is just $-(n+1) \log (n+1)$. Thus the inequality becomes

$$(n+1) \log \frac{c^T v_k}{c^T v_0} < 0 + (n+1) \log (n+1) - \frac{k}{8} ,$$

$$\log \frac{c^T v_k}{c^T v_0} < \log (n+1) - \frac{k}{8(n+1)} .$$

where integer k counts the number of iterations, i.e. the number of times we construct a new v_i. Iterate K times where

$K = 40(n+1)(L + \log (n+1))$, then $K = O(nL)$, assuming that L dominates log n+1. If we set $k = K$, we get

$$\log \frac{c^T v_K}{c^T v_0} < -5L$$

and

$$\frac{c^T v_K}{c^T v_0} < 2^{-5L}.$$

which means, by an argument given earlier at the end of §2.2, that v_K is the exact solution of LPP (2.1.1). The time of each iteration is dominated by the time needed to project cD_i into Ω_i which is $O(n^3)$ operations. Hence the algorithm as a whole takes $O(n^4 L)$ operations. As remarked previously, Karmarkar reduced this time to $O(n^{3.5} L)$.

3. OTHER INTERIOR METHODS. FURTHER DEVELOPMENTS TO 1988.

In this last section we consider the standard LPP to be

$$\begin{aligned} &\text{Minimize } \lambda(\mathbf{x}) = \mathbf{cx} \\ &\text{under the constraints} \\ &\qquad \mathbf{x} \in \Omega \\ &\qquad \mathbf{x} \in P_+ \end{aligned} \tag{3.1.1}$$

where Ω, P_+ are defined as usual, i.e. as in (0.1.1), including the fact that Ω is defined by the linear system $\mathbf{Ax} = \mathbf{b}$. Evidently (3.1.1) is just (0.1.1) using minimization rather than maximization.

3.1. Barrier Methods.

Assume that we intend to solve (3.1.1) by some search procedure (eventually we will construct a sequence $\{v_i\}$ which approximates $\mu_{P_+ \cap \Omega}$). A <u>barrier method</u> for the solution of (3.1.1) is a method which establishes a 'barrier' on the boundary of P_+ which prevents the search procedure from leaving the positive orthant P_+. In the example barrier

method which we will describe, the objective function $\lambda(\mathbf{x})$ is replaced by a non-linear function

$$F_\mu(\mathbf{x}) = \lambda(\mathbf{x}) - \mu\Sigma_j \log(\mathbf{x})_j \qquad (3.1.2)$$

where $(\mathbf{x})_j$ is the jth component of $\mathbf{x}$ and μ is a real positive parameter. Function F_μ is just λ with the addition of $-\mu\Sigma_j \log(\mathbf{x})_j$, which is one of many possible barrier functions. Intuitively if $\mathbf{x} \in P_+$ approaches the boundary of P_+ then at least one component of $\mathbf{x}$ will contribute a positive number which increases without bound to the value of F_μ. Thus F_μ will have a minimal point in $P_+ \cap \Omega$ if $\mu > 0$ although this point will probably not be $\mu_{P_+\cap\Omega,\lambda}$. We would hope that as μ approaches 0 the sequence $\mu_{P_+\cap\Omega,F_\mu}$ approaches $\mu_{P_+\cap\Omega,\lambda}$, and this turns out to be true (there is a proof in [7]).

In [8], Gill et al. describe a straightforward barrier method algorithm which constructs an approximating sequence $v_0, v_1, v_2, \ldots$ which converges to the minimal point of $P_+ \cap \Omega$ in LPP (3.1.1) beginning with a strictly feasible point v_0 ('strictly feasible' means that $v_0 \in \text{Int}(P_+) \cap \Omega$). If point v_i has been constructed, then v_{i+1} is found by the following procedure: set $\mu_i = 1/i$ and write the truncated Taylor approximation of F_{μ_i} as

$$F_{\mu_i}(v_i+\mathbf{x}) = F_{\mu_i}(v_i) + \nabla F_{\mu_i}(v_i)\mathbf{x} + \tfrac{1}{2}\mathbf{x}^T\nabla^2 F_{\mu_i}(v_i)\mathbf{x} .$$

Then if $\mathbf{v}_i + \mathbf{x}_0$ is the minimal point of $F_{\mu_i}(\mathbf{x})$ in Ω (and necessarily also in P_+), $F_{\mu_i}(\mathbf{v}_i+\mathbf{x}_0)$ is approximated by

$$F_{\mu_i}(\mathbf{v}_i) + \nabla F_{\mu_i}(\mathbf{v}_i)\mathbf{x}_0 + \tfrac{1}{2}\mathbf{x}_0{}^T\nabla^2 F_{\mu_i}(\mathbf{v}_i)\mathbf{x}_0 \ .$$

and the vector $\mathbf{x}_0$ points from $\mathbf{v}_i$ to the minimal point of the ith subproblem, although not precisely. In [8], the vector $\mathbf{x}_0$, the Newton search direction, is found as the solution of the problem

$$\text{Minimize } \nabla F_{\mu_i}(\mathbf{v}_i)\mathbf{x} + \tfrac{1}{2}\mathbf{x}^T\nabla^2 F_{\mu_i}(\mathbf{v}_i)\mathbf{x} \ .$$

$$\text{under the constraint } A\mathbf{x} = \mathbf{0} \ .$$

This is solved using Lagrange multipliers. Then $\mathbf{v}_{i+1}$ is defined to be $\mathbf{v}_i + \alpha\mathbf{x}_0$ where $\alpha > 0$ is a steplength parameter which may be varied in the course of empirical tests of the method. The typical experience (as described in, say, [6]) is that a small value of α makes for an elegant convergence proof, but in practical applications a much larger value gives increased speed without evident harm to the operation of the method.

3.2. Gill et al. on Karmarkar's algorithm as a particular barrier method.

The authors of [8] showed that with some reservations concerning the sign of the parameter μ, Karmarkar's algorithm is equivalent to the logarithmic barrier method described in §3.1, provided that the additional constraint

$$\Sigma_j x_j = 1 \; ; \; x_j \text{ a component of } \mathbf{x} \in E_n$$

is added to the problem (3.1.1), thus effectively converting (3.1.1) to the Karmarkar standard form. Gill et al. describe a detailed implementation of the algorithm in §3.1 and present numerical results obtained by applying the algorithm to various standard LPP examples. They conclude that a general barrier method can be comparable in speed to the simplex algorithm. The paper [8] does not attempt to prove polynomial complexity for any version of the barrier method not equivalent to Karmarkar's algorithm.

3.3. An Exchange in SIAM News concerning Karmarkar's algorithm

The newspaper SIAM News in [12] and following issues carried an exchange of views on whether Karmarkar's algorithm was in fact a practical advance over the work on barrier methods which had been going on at least as early as 1957. The originality of Karmarkar's convergence proof was

acknowledged, but there was some expression of doubt whether his claimed experimental results, which were exceedingly optimistic, were likely to be confirmed. A group from Stanford University – which I take to be more or less the authors of [8] – were of the opinion that all the desirable properties of Karmarkar's algorithm are to be found in general barrier methods. It was noticeable that this group reported experimental results which were much more modest than Karmarkar's; for example, while Karmarkar claimed that an implementation of his method was superior to the simplex method on every problem tested, the Stanford group simply reported that their experiments showed barrier methods to be sometimes better than the simplex method and sometimes not. Correspondents in issues subsequent to [12] complained that Karmarkar was unhelpful to researchers trying to confirm his results, which were after all over four years old, and that he was reluctant to acknowledge that his method belongs to the class of barrier methods. Other writers provided a rebuttal: the difficulty of communication could be explained by the fact that Karmarkar's employer clearly intended to patent the method. As regards the relation of Karmarkar's work to work on barrier methods, Karmarkar's reluctance to acknowledge this was indefensible; however his work was not

subsumed in what was known about barrier methods but was in fact an important contribution to it.

3.4. Recent progress to 1988

Gill at al. had presented empirical evidence that barrier methods could compete with the simplex algorithm and presumably with the canonical form of Karmarkar's algorithm; however [8] left open the question of the worst case complexity of barrier methods, especially those which did not use the Karmarkar standard form. Several papers by Clovis Gonzaga (cited in [6]) establish that the barrier method can be applied to the problem (3.1.1) (i.e. not in Karmarkar standard form) so as to have polynomial worst case complexity. I'll violate proper practice here by citing a paper I have not seen and have been unable to obtain: reference [9] by Gonzaga seems to establish the best known worst case complexity for the solution of LPP (3.1.1) at $O(n^3L)$ operations. This is a small but significant improvement on Karmarkar's time. The only paper by Gonzaga I have looked at in which he gives complexity arguments is [10], in which he proves convergence in $O(n^{3.5}L)$ operations for a path following method which solves the standard LPP (0.1.1). The complexity arguments in [10] usually do not prove polynomial convergence; where they do, they depend on Karmarkar's methods which they refer to in an extremely

general way, inviting the reader to "repeat the analysis of Karmarkar" in [1]. It must be borne in mind that Gonzaga's construction of an algorithm which solves (0.1.1) and is amenable to Karmarkar's analysis is a quite significant advance. Nevertheless it appears that Karmarkar's methods in [1] are still basic to complexity proofs in this area. The most recent paper I have seen is [6] which is uninterested in the question of worst case complexity. It describes a state of the art implementation of a primal-dual method due originally to Montiero and Adler, who showed that their method has worst case time of $O(n^3L)$ operations.

APPENDIX

This appendix describes how to project the objective vector $\mathbf{c}$ into the affine space Ω . To project $\mathbf{c}$ into Ω means to find a vector $\mathbf{c}_\pi$ such that if $\mathbf{0} \in \Omega$, then $\mathbf{c}_\pi \in \Omega$ and $\mathbf{c} - \mathbf{c}_\pi$ is orthogonal to all of Ω . Let Ω be the affine space in E_n determined by the system of linear equations $A\mathbf{x} = \mathbf{b}$ where A is an m×n matrix. Assume that $\mathbf{0} \in \Omega$, or more precisely translate the origin to a point in Ω ; the determining equation of Ω will then be $A\mathbf{x} = \mathbf{0}$, i.e. Ω is the null space of A. We want to write $\mathbf{c}$ as a sum $\mathbf{c} = \mathbf{c}_\pi + \mathbf{c}_\perp$ where $\mathbf{c}_\pi \in \Omega$ and $\mathbf{c}_\perp\mathbf{x} = 0$ for all $\mathbf{x} \in \Omega$.

What we will actually do is project $\mathbf{c}$ into the row space $R = \{\mathbf{x} : \mathbf{x} = A^T\mathbf{h};\ \mathbf{h} \in E_n\}$, that is, we want the unique vector $\mathbf{z} \in R$ such that $(\mathbf{c}-\mathbf{z})\mathbf{x} = 0$ for $\mathbf{x} \in R$. Since R is the orthogonal complement of Ω (which is the null space of A), $\mathbf{c}_\pi$ is evidently $\mathbf{c} - \mathbf{z}$. To find $\mathbf{z}$, write any point of R as $A^T\mathbf{h}$ for any $\mathbf{h} \in E_n$, and $\mathbf{z}$ as $A^T\mathbf{h}_0$ for a particular vector $\mathbf{h}_0$. Then $\mathbf{c} - A^T\mathbf{h}_0$ is orthogonal to every point of R, that is

$$0 = (\mathbf{c} - A^T\mathbf{h}_0)^T A^T\mathbf{h}$$

$$= (\mathbf{c}^T A^T - \mathbf{h}_0^T AA^T)\mathbf{h}$$

for every $\mathbf{h} \in E_n$; which means that $\mathbf{c}^T A^T - \mathbf{h}_0^T AA^T$ is the zero vector. Now compute

$$Ac - AA^T h_0 = 0$$

$$AA^T h_0 = Ac$$

$$h_0 = (AA^T)^{-1} Ac$$

which gives us the unique vector h_0 ; and

$$z = A^T h_0 = A^T (AA^T)^{-1} Ac \ .$$

In the last part of the argument we need $(AA^T)^{-1}$. It is a standard theorem that $(AA^T)^{-1}$ exists if A is row-independent. This means that we must assume in LPP (2.1.1) that matrix A has been brought to row-independent form by some convenient method.

As remarked earlier, if z is the projection of c into space R, then the projection c_π of c into Ω is $c - z$; that is

$$c_\pi = c - A^T (AA^T)^{-1} Ac = [I - A^T (AA^T)^{-1} A]c \ .$$

Because of the need to invert AA^T, the task of projecting c into Ω takes $O(n^3)$ operations.

LIST OF REFERENCES

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

[2] Cuthbert, Thomas R. Jr. Optimization with Applications to Electrical Networks. John Wiley and Sons. 1987.

[3] Aspvall, B. and Stone, R.S.
Khachiyan's Linear Programming Algorithm.
J. Algorithms **1** (1980) pp 1-13.

[4] Chvatal, V.
Linear Programming.
New York: W.H. Freeman and Co. 1980.

[5] Klee, V. and Minty, G.J.
How Fast is the Simplex Algorithm?
in Shisa, O. ed. Inequalities-III, pp 159-175.
New York: Academic Press 1972.

[6] McShane, K. Monma C.L. and Shanno, D.
An Implementation of a Primal-Dual Interior Point Method for Linear Programming.
1988 (to appear).

[7] Luenberger, David G.
Linear and Nonlinear Programming, 2nd Ed.
Reading, Mass.: Addison-Wesley 1984.

[8] Gill, Phillip E., Murray, Walter, Saunders, Michael A., Tomlin, J.A., Wright, Margaret H.
On projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method.
Mathematical Programming **36** (1986) pp 183-200.

[9] Gonzaga, C.C.
An algorithm for solving linear programming in $O(n^3L)$ operations.
Technical Report UCB/ERL M87/10,Electronics Research Laboratory, University of California, Berkely, CA 94720, March 1987.

[10] Gonzaga, C.C.
A conical projection algorithm for linear programming.,
Mathematical Programming **43** 1987.

[11] Khachian, L.G.
A polynomial algorithm in linear programming [in Russian].
Doklady Academiia Nauk SSSR **244** pp 1093-1096.
[English translation: Soviet Mathematics Doklady **20** pp 191-194.]

[12] SIAM News: The bimonthly newspaper of the SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS.
18 NUMBER 8/NOVEMBER 1985.