Author : P.A. Bruijs
Address : Gershwinstraat 1 / 2807 SG / Gouda / The Netherlands
Phone : +31182689052
E-mail : p.a.bruijs@wxs.nl
Formerly : Dr Neher Laboratories of the Dutch PTT

Peter A. Bruijs

How bad is a gradient algorithm for linear programming?

April 20, 2011

Abstract. In their 1972 paper 'How good is the simplex algorithm ?' Klee and Minty present a class of problems the simplex algorithm for linear programming (LP) is not able to solve in a polynomial way. Later developments have resulted in algorithms by Khachiyan and Karmarkar that do solve LP in a polynomial way, although the bounds they generate are not exclusively based on LP's problem dimensions *m* and *n*.
Until now it is an open question whether there exists a strongly polynomial algorithm solving LP in O(*m*,*n*) operations.
In this paper we try to identify characteristics of present algorithms that might hinder in attaining such a result, and look at the possibilities a strongly gradient oriented algorithm might offer in this respect.

Keywords. linear programming - simplex algorithm - ellipsoid method - interior-point methods - gradient methods - computational complexity

1. Introduction

Although the definition of the Linear Programming problem (LP) lies some 75 years behind us, and although several prolific algorithms are available to solve the problem in most cases, there is still no algorithm solving the problem generally in polynomial time measured in the problem's dimensions: number of constraints *m* and number of variables *n*.
The first very successful attack to the problem, the simplex algorithm, has for some decades been on top, and still has a very strong position, based on the clear and insightful approach it takes, and last but not least on the practical performance of the algorithm.
However, scientific research, e.g. see Klee and Minty's paper "How good is the simplex algorithm ?" [2], revealed weaknesses in special cases, where the algorithm takes an exponential number of iterations, measured in terms of *m* and *n*, to reach the optimal solution.
At that moment the complexity of solving LP has become a prominent issue, and as such it did attract attention, resulting in new algorithms with their own strengths and weaknesses.
The extreme point approach of Dantzig's simplex algorithm was followed by completely different approaches, taken first by Khachiyan, and later by Karmarkar.
However, whereas they present polynomial algorithms, the measure of complexity of their algorithms does not relate exclusively to *m* and *n*, and so it is still an interesting question whether a strongly polynomial algorithm for LP exists at all . . .

Since Dantzig 1950 [1], Khachiyan 1979 [3], and Karmarkar 1982 [4] a vast and impressive body of literature on the problem has been generated. Thereby as well practical as theoretical issues have been discussed. In his 2001 paper "The many facets of linear programming" [5] Todd gives an historic overview of these developments offering an excellent basis for the present analysis.

Large parts of Todd's overview are concerned with the simplex algorithm, and ellipsoid and interior-point methods. 'Gradient-like methods' however are covered in a tiny section 6.1 of his overview that states:

'These try to follow projected gradient directions from one feasible point to another. Since this seems like such a natural strategy, and apparently so obviously superior to edge-following methods, such methods have been proposed numerous times in slightly varying guises.'

On the basis of this statement it is not quite clear in which way the gradient-like algorithms differ from the simplex algorithm, and the polynomial ellipsoid and interior point methods. Don't simplex and interior point methods too follow a path of feasible points, and don't they too use some gradient-related directions ?
Why do they seem to offer 'such a natural strategy' ?
Also, what would make such an approach 'so obviously superior to edge-following' ?
Or are on the contrary the qualifications in the above statement not at all adequate in the end ?

Furthermore and interestingly it may be observed that apparently no complexity result for these methods can be given.

In the sequel the questions posed will be treated, leading to the presentation of an almost completely greedy gradient-like algorithm, with characteristics strongly different from as well the simplex algorithm as the ellipsoid and interior point methods.

2. Some characteristics of well-known successful approaches to LP

In this paragraph the aim is to look for some leading characteristics of the approaches taken by currently used algorithms, thereby assuming basic knowledge on problem LP and related terminology.

2.1 Simplex algorithm

In a comic way it can be said that the simplex algorithm is basically based on basic solutions. And there is a clear argument to do so. For, as is well-known from the theory of linear optimization, an optimum of the problem can be found in an extreme point of the convex constraint set which is a basic solution. Depending on the specific problem data one may encounter the situation that there are multiple optima implying that there are not only several optimal basic solutions, but apart from that there are also optimal non-basic solutions.
The algorithm essentially searches for the best basic solution in the set of all basic solutions. As extensively discussed in the literature several pivot selection criteria can be used for the selection of an improved basic solution, each criterion having its own merits.
However, the sequence of intermediate basic solutions before reaching the optimum may be quite long as demonstrated by Klee and Minty in their paper "How good is the simplex algorithm ?" [2]. And in fact this leads to the conclusion that there are problems for which this sequence has an exponential length measured in terms of *m* and *n*, thereby refuting strong polynomial complexity of the simplex algorithm, independent of the pivot selection criterion chosen.

In practice the expected performance of the algorithm proves to be quite good. But, although the optimal solution is a basic solution, in the light of the theoretical complexity it does not seem wise in every instance to constrain the solution sequence exclusively to basic solutions. And in principle there is no need to do so: the set of basic solutions is only a small subset of the complete convex space of feasible solutions to the problem.

2.2 Polynomial algorithms: the ellipsoid and interior point methods

The approaches taken by ellipsoid and interior point methods are quite different in the sense that they originate from the realm of nonlinear convex optimization.
They share an interesting characteristic in the fact that they are based, as to the complexity result, on the assumption that the problem data can be represented by an L-bit sequence.
The complexity bound derived proves to be a function of the number of variables n and the length L of this sequence, the number of constraints m does not play a part in this measure.

The ellipsoid algorithm is based on the fact that it is possible to construct from the problem data an ellipsoid enclosing all solutions so including the optimum solution. In successive steps a sequence of shrinking ellipsoids is constructed. In every step a separating hyperplane can be constructed dividing the present ellipsoid in a part that does contain the optimum and a part that doesn't. At some step it becomes possible to decide the optimum has been reached.
In the process $O(n^4L)$ operations on $O(L)$-digit numbers are involved.

An interior point method tries to converge to the optimum from the inside of the feasible region. On the basis of an available solution a transformation is made positioning this solution 'in the centre', meaning 'with equal distance' to the constraints, thereby providing room for improvement of the objective advancing through the inside of the feasible region.
In this case $O(n^{3,5}L)$ operations equally on $O(L)$-digit numbers are involved.

It is clear that essentially these methods are approximation methods with a sequence of enclosing ellipsoids or explicit feasible non-basic solutions both converging to the optimum.

3. An appreciation

The question now is: what can be derived from the characteristics described in the foregoing ? Well first of all, the complexity of all approaches so far to solve LP does not offer a strong polynomial bound.

For the simplex algorithm what hinders is the fact that the algorithm reduces the space of candidate solutions to the set of basic solutions, the cardinality of which grows exponentially with the problem dimensions. And in this way it is possible to imagine, and proves to be possible to construct, problems for which specific versions of the simplex algorithm generate solution sequences of super-polynomial length.

For the ellipsoid method the essential tool is a separating hyperplane dividing the current ellipsoid into a 'successful' and an 'unsuccessful' part. However this hyperplane does not have a direct relation to the constraints defining the problem at hand.

For interior point methods the essential ingredient is the fact that intermediate solutions are far from the constraints hopefully leading to far steps in the interior of the feasible region of

the problem. However also in this situation there is no direct confrontation with the constraints defining the problem.

For the polynomial methods as it seems the available information on the problem is not fully exploited, the complexity measure remarkably being independent of *m*.

What does this mean in relation to the questions formulated earlier for gradient-like methods. A real gradient-oriented method might be defined as a method that maximally moves in the direction of the gradient without violating any constraint.

What is special in the case of LP is, that the gradient is a unique and constant direction, and that furthermore there are only *m* linear constraints defining a convex solution space.

Aren't these the crucial ingredients to be fully exploited ?
So does the following statement constitute a naïve 'natural strategy' as meant by Todd ?

> Follow the unique direction of the gradient, only and minimally deviating when a constraint would get violated, bearing in mind that there are only *m* constraints for which a deviation may be necessary, and the fact that the solution space is convex.

Does the superiority to an 'edge-following' strategy in this case lie in the fact that no more than strictly necessary concessions are done in deviating from the gradient ?

Furthermore in the realm of nonlinear convex optimization a saying is:

> Separation = Optimization

So one can wonder what separation device to use in constructing and selecting the optimum. And in the case of LP what would be a more natural choice than for the hyperplanes defined by the constraints to LP itself ? They in fact do cut off those parts of the problem space where the optimum cannot lie, and one needs the gradient direction starting from the current solution as a guide to assess which hyperplanes should be used for separation.

An interesting further, but clearly rhetoric, question, with the implicit suggestion about a possible number of separations, is in fact:

> How many separating hyperplanes are available anyway ?

Furthermore in nonlinear optimization in every iteration and every intermediate solution point we have to solve a direction-finding problem. In LP on the contrary one could say that we are confronted with a direction-following problem: the direction is there we just have to follow n the best way.

With the description of characteristics of available methods to solve LP and the appreciation added we have arrived at a point where an idea for an alternative method can be put forward.

4. An idea for a gradient algorithm

So essentially the alternative approach proposed here to solve LP uses the information of the unique gradient to a maximum extent, bearing in mind that, starting from a feasible solution, one should not violate any of the constraints. And there are no more than these constraints that can hinder in proceeding in directions following the gradient, that is with positive projection on the gradient.

An informal introduction to the alternative method can be given in the following way.

Let problem LP be

Max $c^T x$
Sub $A\,x <= b$

Where $x = (x_1, \ldots, x_n)$
$c = (c_1, \ldots, c_n)$
$b = (b_1, \ldots, b_m)$
A has row-vectors a_i, $i = 1, \ldots, m$
$a_i = (a_{i1}, \ldots, a_{in})$

Let us assume now that we know an initial feasible solution $x(0)$.

(Such a solution by the way can be computed gradient-based in an appropriate phase I procedure whenever the algorithm for phase II starting from $x(0)$ proves to be successful.)

The first simple question to be asked seems to be how far the solution $x(0)$ can be improved into $x(1)$ simply, greedily, by proceeding for a step-size $s(1)$ in the direction of the gradient as specified by the problems object-vector c.

So

$$x(1) = x(0) + s(1)\, c$$

where $x(1)$ too has to be a feasible solution,
so,

$$a_i^{\,T} x(1) <= b_i \qquad \text{all } i$$

$$a_i^{\,T} x(0) + s(1)\, a_i^{\,T} c <= b_i \qquad \text{all } i$$

$$s(1) <= (b_i - a_i^{\,T} x(0)) / a_i^{\,T} c \qquad \text{all } i$$

so the maximum step without violating any constraint is

$$\min_i (b_i - a_i^{\,T} x(0)) / a_i^{\,T} c$$

But by then there is at least one constraint $a_i(1)$ that is binding,

$$a_i(1)^{\,T} x(1) = b_i(1)$$

so there is no more slack that can be used in the direction *c* of the original gradient.
In the meantime it is clear that for the computation of an improving step one only has to consider those constraints for which one has in this first step

$$a_i^{\,T} c > 0$$

because a feasible solution is already available so only they are candidate to become binding.
In order not to violate the constraint $a_i(1)$ in further steps of our computation, one has to exclude every solution in the non-feasible half-space related to this binding constraint.
For the best direction that is left one has to exclude from the gradient *c* exactly and exclusively the component directing into this half-space.
The resulting feasible gradient $fg(1)$ in $x(1)$ may then be written as

$$fg(1) = c - \{ c^{\,T} a_i(1) / a_i(1)^{\,T} a_i(1) \} a_i(1)$$

and locally in $x(1)$ this is truly the best feasible direction known so far.
In the next iteration in an analogous way one looks for the best value of

$$x(2) = x(1) + s(2)\, fg(1)$$

thereby determining a new binding constraint $a_i(2)$.
In further iterations one will have to exclude the non-admissible half-spaces related to $a_i(1)$ and $a_i(2)$.

It will be seen later to be convenient to determine an orthonormal basis consisting of vectors $aon(1)$, $aon(2)$, etc to span the combined non-admissible space related to binding constraints $a_i(1)$, $a_i(2)$, etc.

This may be done by means of the Gram-Schmidt procedure in the following way.

$$aon(1) = a_i(1) / a_i(1)^{\,T} a_i(1)$$

$$aon(2) = \{ a_i(2) / a_i(2)^{\,T} a_i(2) \} - \{ a_i(2)^{\,T} aon(1) / a_i(2)^{\,T} a_i(2) \} aon(1)$$

In this way one has

$$fg(1) = c - \{ c^{\,T} aon(1) \} aon(1)$$

$$fg(2) = c - \{ c^{\,T} aon(1) \} aon(1) - \{ c^{\,T} aon(2) \} aon(2)$$

It will not be surprising that in step $k+1$ of the algorithm this will result in

$$fg(k) = c - \sum_{1<=j<=k} \{ c^{\,T} aon(j) \} aon(j)$$

As is easily seen the number of operations involved in the computations so far is relatively small and always polynomially related to *m* and *n*, the numbers of variables and constraints.

So by now it becomes interesting to see how many iterations are needed, and also to know whether the optimum will be reached at all, for if this would be the case one might even hope for a strongly polynomial algorithm to solve LP this way.

5. A proposed gradient algorithm

To be able to give specific arguments the proposed algorithm has to be presented first.

The presentation in the preceding material has been rather loose, so apart from the definition of the problem as given earlier, a more precise and complete specification of the algorithm is needed.

5.1 The proposed gradient algorithm

In case a feasible solution is at hand one proceeds as follows :

(step 1) set k=0, a feasible solution is $x(0)$, the feasible gradient is $fg(0) = c$

(step 2) identify the following sets of binding and non-binding constraints:

$$B(k) = \{ a_i \mid a_i^T x(k) = b_i \, , \, a_i^T fg(k) > 0 \}$$

$$NB(k) = \{ a_i \mid a_i \text{ not in } B(k) \}$$

(step 3) compute the orthonormal set $BON(k)$ for the binding constraints by means of the Gram-Schmidt procedure, so

$$BON(k) = \{ aon(i) \mid a_i, a_j \text{ in } B(k) \; ; \; aon(i)^T aon(j) = 0, j \neq i; \; aon(i)^T aon(i) = 1 \}$$

(step 4) compute the actual feasible gradient

$$fg(k+1) = c - \sum \{ aon(j) \text{ in } BON(k) \} \{ c^T aon(j) \} aon(j)$$

(step 5) if $fg(k+1) = 0$: an optimality check must be performed,

so compute the multipliers from the Kuhn Tucker condition

- if all multipliers > 0 : the optimal solution has been reached

- if not, remove the constraints with negative multiplier from $B(k)$ and go to step3

(step 6) compute the maximum feasible step from $x(k)$ along $fg(k+1)$

$$s(k) = \min_{\{ a_i \text{ in } NB(k) \, , \, a_i^T fg(k+1) > 0 \}} \{ b_i - a_i^T x(k) \} / a_i^T fg(k+1)$$

if $s(k)$ is infinite : LP is unbounded

(step 7) $x(k+1) = x(k) + s(k)\ fg(k+1)$

(step 8) set k = k+1 and got to step2

5.2 Remarks

Remark 1
The computation of a starting solution, if not available, can be done by the same algorithm by means of an appropriate object-vector based on the violated constraints starting from an arbitrary starting solution, analogous to the phase I procedure of the simplex algorithm.

Remark 2
Until an iteration is reached where $fg = 0$ every iteration embodies a greedy but also forced move, whereby one is not confronted with the choice for one or another pivot selection rule !
It is not correct, as supposed by several authors, that under the condition $fg = 0$ the optimum has been reached.

Remark 3
The computation of the multipliers in (step 5) in the test for optimality is derived from the fact that, according to the Kuhn Tucker condition, in the optimum *c* may be written as a positive linear combination of the normal vectors to the binding constraints

$$c = \sum \{\ i \mid a_i \text{ in } B(\text{step 5})\ \}\ \mu(i)\ a_i \qquad \mu(i) > 0,\ a_i \text{ in } B(\text{step 5})$$

or, as can be shown to be essentially equivalent

$$c = \sum \{\ i \mid aon(i) \text{ in } BON(\text{step 5})\ \}\ \mu on(i)\ aon(i) \qquad \mu on(i) > 0,\ aon(i) \text{ in } BON(\text{step 5})$$

This is intuitively clear because we have to be sure that we do not deny any directions for use when excluding the space spanned by the constraints in the binding set. But the same space is spanned by the orthonormal set *aon(i)* derived directly from the *a(i)* in *B*, their relation being defined by

$$aon(i) = a(i) / a(i)^T a(i) - \sum \{\ j < i\ \} \{\ a(i)^T aon(j) / a(i)^T a(i)\ \}\ aon(j)$$

From this one has

$$\begin{aligned} c^T aon(im) &= \sum \{\ i \mid aon(i) \text{ in } BON(\text{step 5})\ \}\ \mu on(i)\ aon(i)^T aon(im) \\ &= \mu on(im)\ aon(im)^T aon(im) \\ &= \mu on(im) \end{aligned}$$

because from orthonormality it follows that

$aon(i)^T aon(i) = 1$ all *i*
$aon(i)^T aon(j) = 0$ *i* not equal *j*

Remark 4
If in (step 6) the minimum for *s(*k*)* is not unique one will have to select as the new binding constraint the one *ib* for which *a(ib)* ^{T} *c* has the smallest value. This is the 'most constraining' constraint in the set of newly found binding constraints.

Remark 5
It should be noted that the description of the algorithm given here obviously has not been optimized for practical implementation .

As stated earlier it is now crucial to see how many iterations are to be made, and also to know whether the optimum will be reached at all.

To begin with a discussion of the latter point :

Isn't there a situation in which the algorithm gets stuck, more or less analogous to the effect of cycling for the simplex algorithm ?

And indeed this is the case, for normally when LP is bounded the algorithm will encounter an iteration where

*fg(*k*)* = 0

At such a point one is left with the question : has the solution been reached or hasn't it ?
To this end one has the check on the multipliers resulting from the Kuhn Tucker condition. The computation of the multipliers proves to be rather straightforward as shown in Remark 2, and the resulting values of the multipliers indicate whether the optimum solution has been reached.

So when one or more of the multipliers is negative the conclusion must be that the optimum has not yet been reached.

In this situation the Kuhn Tucker relation reads

$$c = \sum \{ \text{ i} \mid aon(\text{i}) \text{ in } BON(\text{step 5}) \} \mu on(\text{i})\, aon(\text{i})$$

now suppose that for some constraint ineg one finds $\mu on(\text{ineg}) < 0$.

The meaning of this fact can be derived quite easily as follows

$$aon(\text{ineg})^{T} c = \sum \{ i \mid aon(\text{i}) \text{ in } BON(\text{step 5}) \} \mu on(\text{i})\, aon(\text{ineg})^{T}\, aon(\text{i})$$

but from the orthonormality of the *aon(i)* it follows that this reduces to

$$aon(\text{ineg})^{T} c = \mu on(\text{ineg}) < 0$$

indicating that in the current solution *x(*k*)* the constraint *a(*ineg*)* does no more fulfil the necessary condition to be considered binding.

Furthermore by the confrontation of the orthonormal direction *aon(*ineg*)* with a negative multiplier with the direction of the feasible gradient one gets proof of the linear independence of the two in every iteration in which constraint ineg is in *BON(step 5)*

$aon(\text{ineg})^T fg(\text{k}) =$
$aon(\text{ineg})^T \{ c - \sum \{ j<=k \} (aon(\text{j})^T c) aon(\text{j}) \} =$
$aon(\text{ineg})^T c - aon(\text{ineg})^T \sum \{ j<=k \} (aon(\text{j})^T c) aon(\text{j}) =$
$aon(\text{ineg})^T c - aon(\text{ineg})^T c = 0$

Additionally by the confrontation of the orthonormal direction *aon(*ineg*)* with negative multiplier with the direction filtered out of the initial gradient *c* to get the feasible gradient *fg* one gets proof of the fact that *aon(*ineg*)* cannot be considered binding and so possibly hinders in improving the objective.

$aon(\text{ineg})^T \{ c - fg(\text{k}) \} =$
$aon(\text{ineg})^T \{ c - \{ c - \sum \{ j<=k \} (aon(\text{j})^T c) aon(\text{j}) \} \} =$
$aon(\text{ineg})^T \sum \{ j<=k \} (aon(\text{j})^T c) aon(\text{j}) =$
$aon(\text{ineg})^T c = \mu on(\text{ineg}) < 0$

After finding *fg* = 0 for the feasible gradient, and after the conclusion that the optimum has not yet been reached, it is the question whether there is a way out of this situation that is as greedy, clear and forcing, as in the iterations before the point where *fg* = 0 has been reached

Because the present solution is feasible only two outcomes are to be identified

- it is possible to improve the objective function after redefining *B(*k*)* as specified in the algorithm by removing those constraints with a negative multiplier
- it is not possible to improve the objective any further

Looking at the latter outcome the conclusion is, that the value of the objective is optimal, but the binding set *B(*k*)* does not have the right elements. A strict formulation says that there is apparently another constraint, presently not in *B(*k*)*, that is more relevant than one of the constraints that are in *B(*k*)*. However this contradicts the fact that in the algorithm in every step the most constraining one constraint is added to *B(*k*)*.

(This touches upon the concept of degeneracy as in the simplex algorithm, although there it is not clear which variable to introduce in the basis, whereas in the gradient algorithm on the contrary it is quite clear which constraint to introduce in the binding set.)

The foregoing implicates that in case *fg* = 0 and at least one of the multipliers is negative, it is possible to improve the objective. After the modification of *B(*k*)* it becomes possible to recompute *fg* - this time with $fg \neq 0$ as a result - as well as *BON(*k*)*, and *aon(*i*)* for i=1,…,k.

By then however we can show that resulting from the convexity of the feasible region the constraints removed from the previous binding set will not re-enter forthcoming binding sets.

For now the position is that one has a feasible point *x* in iteration k, one will get a feasible point *y* in iteration k3 > k+1, while in both binding sets one would have a constraint *a(*i**)* that has been discarded in an iteration $k \leq k2 < k3$, and such that

$a(\text{i}^*)^T x = b(\text{i}^*)$
$a(\text{i}^*)^T y = b(\text{i}^*)$

with

$$c^T y > c^T x$$

because the algorithm did not stop.
So

$$c^T (y - x) > 0$$

while from the convexity of the feasible region one derives that all points

$$x + \theta (y - x) \qquad 0 < \theta < 1$$

are points in the interior or on the convex hull of the feasible region.
And from the definition of the algorithm it is clear that none of these points has been excluded, one excludes in fact only non-feasible points as candidates for the optimum.
So this is in contradiction with the fact that in x the direction $(y - x)$ with a positive projection on c has been identified as an infeasible direction.

In this way one has a guarantee that the proposed algorithm is finite, for either

- LP is found to be unbounded, or,
- there are only m constraints that can enter a binding set, and all in all no more than m constraints may be discarded from, and (this is the crucial part) will not reenter in, binding sets for LP

6. Additional characteristics

It is well known that in LP there are weaknesses especially related to the simplex algorithm that may hinder the performance of the algorithm in practice : cycling and degeneracy.
One may wonder whether these phenomena are intrinsic to LP, to the problem as such.

6.1 Cycling

As is well known the simplex algorithm exhibits the phenomenon of cycling on certain problems and one wonders, the algorithm proposed here, would it cycle too ?
As the algorithm explores all slack in all feasible directions, only excluding directions, and so steps, that would lead to infeasible solutions, it will not do so.
Actually every intermediate solution is completely and uniquely determined by a specific set of binding constraints (and the initial solution).

And as an incidental test the cycling problem defined by Beale has been solved by the proposed algorithm without any problem. It is however good to state that this is incidental, for, in the fashion of Popper, one does not need an example to prove success, but an example to disprove the hypothesis at stake.

6.2 Degeneracy

Some (classes of) LP problems exhibit the phenomenon of degeneracy, in which the choice of the locally best basis is problematic. In the simplex algorithm this may result in possibly many ineffective iterations.
The algorithm proposed here, how would it manage in degenerate problems ?
As the algorithm explores all slack in all feasible directions, only excluding directions, and so steps, that would lead to infeasible solutions, it will not have to cope with non-uniqueness of the locally best next step to make.
Actually it is perfectly possible to encounter, even a succession of, zero step-lengths, however these are meaningful steps in that a meaningful constraint is added to the binding set.

And as incidental tests several assignment and transportation problems have been solved by the proposed algorithm without any problem.

7. An in between

From the history of LP it is clear that the major steps in the solution of the problem have until now been made by Dantzig, Khachiyan and Karmarkar. An ultimate step however still could be made granting LP the status of strong polynomial solvability.
By now it is clear that the view presented here is an attempt to make this step, or at least to show the possible attractiveness of a gradient approach in this respect.
For sure there have been earlier attempts to make this step but it is hard to discover the nature of the algorithms involved and or the arguments refuting the claims for their complexity.
However as far as the present author is aware all attempts involve in one way or another either simplex-like or interior-point concepts, thereby adopting characteristics as discussed in sections 2 and 3.
Recently Ping-Qi Pan[6] has presented an algorithm in which he uses a simplex-like approach as a ground pattern but where he does not use a full, but a deficient basis, thereby generating some freedom relative to an algorithm using the full basis concept. The freedom generated is then used to compute directions of improvement that are better than those you get when moving strictly from basic solution to basic solution, from vertex to vertex.
And indeed results reported show a better performance than for the simplex approach.

However Ping-Qi Pan's approach seems to prove how hard it is psychologically to leave a proven paradigm behind one could say.

8. Conclusion

In the conclusion of his review [5] mentioned earlier, Todd presents questions as to the expectations he has, and one might have, as to future developments in the approach to LP.
It might be called remarkable to see that he does not have explicit hopes for developments in the area 'gradient-like methods', until now seemingly hardly researched for their complexity.

Here as a result of reflections on present approaches to problem LP we have

- discussed the source of the non-polynomial character of the simplex algorithm, and concluded that a possibly unnecessary extra constraint has been introduced by the assumption that intermediate solutions should have to be basic solutions

- discussed available problem information not used in ellipsoid and interior point methods, and concluded that it may be unwise to avoid the direct confrontation with available problem constraints
- presented an alternative algorithm to solve LP that is explicitly based on the convexity of the feasible region and uses only purely linear concepts, as opposed to ellipsoid and interior-point methods, in line with LP's definition
- an algorithm leading to computations
    - that are clearly polynomial in effort per iteration
    - with a maximum number of iterations that is bounded by the fact that the binding set cannot grow beyond *m* constraints, meaning that the number of iterations is also clearly polynomially related to the number of constraints
- an algorithm leading to the LP-optimum as a seemingly strongly polynomial alternative to the simplex algorithm and to ellipsoid and interior point methods.

The use of the qualification 'seemingly' is inspired by G.B. Dantzig remarking on several occasions[5] that a geometric view of the problem, clearly underlying the approach presented, can be profoundly misleading! It is Todd who remarks in a personal communication that :

'The combinatorics and geometry of high-dimensional polyhedra is often counter-intuitive'.

These warnings are greatly appreciated, however, in the approach presented here the essential elements in problem LP are treated in a geometrically completely transparent way, and as LP is a continuous optimization problem, treated here as such, one does not expect combinatorial issues to be involved.

The presentation of a theoretical view is the main purpose of the present paper, and although computational evidence with the algorithm is available, for the moment the focus is left on the theoretical aspects.

Still it is interesting to find that Ping-Qi Pan[6] reports computational results indicating a better than simplex performance in iterations and computation times for his partial simplex, partial gradient approach.

Further investigation of the theoretical and practical impact of the present findings might be interesting.

8. Acknowledgement

Early discussions around 1980 with Erik van Doorn, as well as recent discussions with Kees Roos and Michael Todd have been helpful in the development of the ideas presented.

References

1 G.B. Dantzig. *Linear Programming and extensions*. Princeton University Press, Princeton, NJ, 1963
2 V. Klee and G.J. Minty. How good is the simplex algorithm ? In O. Shisha, editor, *In equalities III*, 159-175, Academic Press, 1972
3 L.G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady* 20:191-194, 1979
4 N.K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373-395, 1984
5 M.J. Todd The many facets of linear programming. *Mathematical Programming*, 91:417-436, 2002
6 Ping-Qi Pan. A face algorithm for linear programming. *Optimization online*, 2007