175 Years of Linear Programming

3. Pune's Gift

Vijay Chandru and M R Rao

The announcement, by L G Khachiyan, of the polynomial solvability of linear programming by the ellipsoid method, followed by the probabilistic analyses of the simplex method in the early 1980's left researchers in linear programming with a dilemma. We had one method that was good in a theoretical sense but poor in practice and another that was good in practice (and on average) but poor in a theoretical worst-case sense. This left the door wide open for a method that was good in both senses. In 1984, Narendra K Karmarkar, a 28 year-old computer scientist based at AT&T Bell Laboratories, a former prodigy of Pune raised by a family of mathematicians, closed this gap with a breathtaking new projective scaling algorithm.

In retrospect, the new algorithm has been identified with a class of nonlinear programming methods known as logarithmic barrier methods. Implementations of primal-dual variants of the logarithmic barrier method have proven to be the best approaches at present. It is a version of this method that we describe in this part of the series on linear programming.

Vijay Chandru is a Professor of computer science & automation at the Indian Institute of Science. His research interests are in optimisation, geometry, logic and computer graphics.

M R Rao is Director and Professor of quantitative methods and information systems at the Indian Institute of Management, Bangalore. His research interests include optimisation and risk management in the context of managerial decision making.

Shrinking Ellipsoids

As we saw in the second article of this series (Pivots in Column Space, *Resonance* January 1999), V Klee and G L Minty constructed exponential examples for the simplex method. However, both empirical and probabilistic analyses indicate that the 'average' number of iterations of the simplex method is just slightly more than linear in the dimension of the polyhedron.

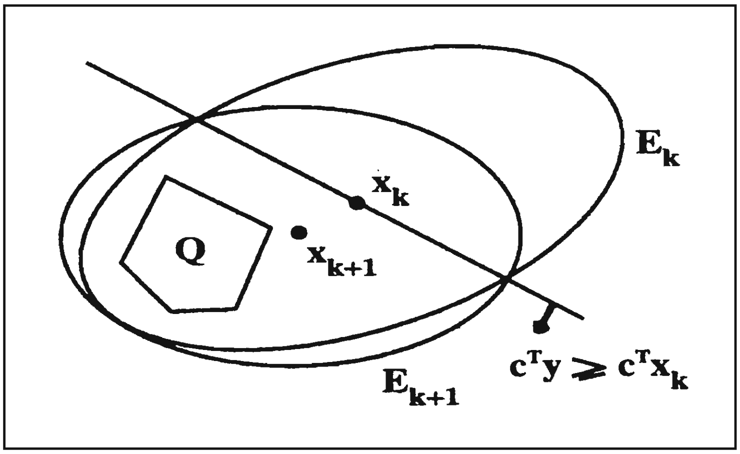

Figure 1. Shrinking ellipsoids.

The ellipsoid method was devised to overcome poor scaling in convex programming problems and therefore turned out to be the natural choice of an algorithm to first establish polynomial-time solvability of linear programming, that is, testing if a polyhedron $\mathcal{Q} \in \Re^d$, defined by linear inequalities, is non-empty. For technical reasons let us assume that $\mathcal{Q}$ is rational, i.e. all extreme points and rays of $\mathcal{Q}$ are rational vectors or equivalently that all inequalities in some description of $\mathcal{Q}$ involve only rational coefficients.

The ellipsoid algorithm initially chooses an ellipsoid large enough to contain a part of the polyhedron $\mathcal{Q}$ if it is non-empty. The centre of the ellipsoid is in $\mathcal{Q}$ if all the inequalities defining $\mathcal{Q}$ check out. Else, an inequality of $\mathcal{Q}$ separates the centre point of the ellipsoid from the polyhedron $\mathcal{Q}$. We translate the hyperplane defined by this inequality to the centre point. The hyperplane slices the ellipsoid into two halves, one of which can be discarded. The algorithm now creates a new ellipsoid that is the minimum volume ellipsoid containing the remaining half of the old one. The algorithm now questions if the new centre is feasible and so on. The key is that the new ellipsoid has substantially smaller volume than the previous one. When the volume of the current ellipsoid shrinks to a sufficiently small value, we are able to conclude that $\mathcal{Q}$ is empty. This fact is used to show the polynomial time convergence of the algorithm.

175 Years of Linear Programming

Vijay Chandru and M R Rao

1.The French Connection. Fourier's Algorithm and LP Duality, Vol.3, No.10, 1998.
2. Pivots in Column Space. The Simplex Method, Vol.4, No.1, 1999.
3. Pune's Gift.Karmarkar's Projective Scaling Method.
4. Minimax and Cake.von Neumann's Minimax Theorem.
5. MaxFlow equals MinCut. The Theory of Network Flows.

Computational experience with the ellipsoid algorithm, however, showed a disappointing gap between the theoretical promise and practical efficiency of this method in the solution of linear programming problems. The need for high precision as well as the slow average-case convergence properties are the reasons most often cited for this behaviour of the ellipsoid algorithm.

The early 1980's left scholars of linear programming in a dilemma. We had one method that was good in a theoretical sense but poor in practice and another that was good in practice (and on average) but poor in a theoretical worst-case sense. This left the door wide open for a method that was good in both senses.

Look Before You Leap

Narendra Karmarkar's proposal, in 1983, to drive through the interior of the feasible region, of a linear program, to an optimal solution seemed at first to be naíve. It is well known that navigating through the interior of the feasible region of a linear program using the gradient of the objective function, as the improvement direction, often runs into trouble because of getting 'jammed' into corners. This is because, in high dimensions, corners make up most of the interior of a polyhedron.

Here is a simple thought experiment to illustrate the idea. Consider a uniform box (a square, a perfect cube, a uniform hypercube) and inscribe the largest possible sphere in the box. Let us denote the volume of the box, excluding the sphere, as the corner volume. The ratio of the corner volume to that of the box ($(1-\pi/4)$ in 2D, $(1-\pi/6)$ in 3D etc rapidly approaches one with increasing dimension. Thus, high dimensional polyhedra are naturally 'angular'.

This jamming of interior methods can be overcome if the optimizing direction is balanced with a 'centering' direction.

Box 1. Linear Programming Headlines

A fast way to solve hard problems' wrote Gina Kolata in the Research News column of *Science* in September 1984. James Gleick called it a 'Breakthrough in problem solving' in the *New York Times* on November 19, 1984. There was a uniform agreement that the young lad had done something spectacular. His boss at Bell Labs, the eminent combinatorialist Ronald Graham said, "*Breakthrough is one of the most abused words in science. But this is one situation where it is truly appropriate.*"

While there was no question that young Karmarkar had created a new intellectual era of mathematical programming, the commercial interests in the intellectual property that the new algorithm represented led to some ugly twists to the story. Early to catch this angle was Jerry Bishop of the *Wall Street Journal*, who wrote on November 26,1984 that "*AT&T maintains secrecy on details in touting problem-solving invention.*" The Mathematical Programming Society meeting in the summer of 1985 saw acrimonious exchanges between champions and detractors and the Boston Globe screamed, "*Founding father or just a footnote*".

As usual, the lawyers had the final say. The spectre of corporate patents for 'mathematical algorithms' had loomed. AT&T obtained US Patent Number 4744028 in 1988 for 'Methods and apparatus for efficient resource allocation' and the *Wall Street Journal* rejoiced with 'AT&T markets problem solver, based on math whiz's find, for $8.9 Million'.

Karmarkar showed that a delicate balancing act could be performed by using an ingenious fractional linear transform to pull the current iterate towards the centre of the feasible region. Using the cross-ratio invariant of projective geometry, Karmarkar formulated a potential function that provided the substrate for proving polynomial time convergence. His heuristic of creating an algorithm for linear programming which averted the two devices of the Klee–Minty construction – scaling and projection – conjured up a winner (see *Box* 1).

Its Always so Simple, in Retrospect!

In retrospect, the centering direction in Karmarkar's algorithm has been shown to be based on the *analytic center* y_c (see *Box* 2) of a full dimensional polyhedron $\mathcal{D} = \{x : A^T y \leq c\}$ which is the unique optimal solution to

$$\max\{\sum_{j=1}^{n} \ln(z_j) \; : \; A^T y + z = c\}.$$

Box 2. Analytic Centres and Central Trajectories

The analytic centre of a convex polytope is the unique interior point that maximizes the product of the distances to the hyperplanes that define the facets of the polytope. Consider the following construction. First fix an objective direction and take cross-sections of the polytope with hyperplanes orthogonal to this objective. Now find the analytic centre of each cross section. If we string these centres together we obtain trajectories (as illustrated in *Figure 2*). These are called central trajectories and have been studied as solutions to ordinary differential equations and proved results about their algebraic geometry. Karmarker posed the following intriguing question about the Riemannian geometry of central trajectories: "*Given a Riemannian metric defined on the interior of a polytope, what is the integral of the scalar curvature of the central trajectory*?" An answer would resolve a nagging question about Karmarkar's algorithm – *why is the number of iterations needed by the algorithm so small in practice?*

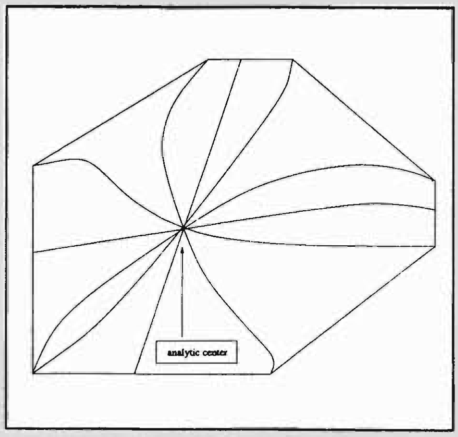

Figure 2. Central trajectories.

Note that the logarithms in the objective force us to keep the slack variables at a positive level, i.e., they model a barrier function known as the logarithmic barrier.

Recall the primal and dual forms of a linear program may be taken as

$$(P)\min\{cx : Ax = b,\, x \geq 0\}$$

$$(D)\max\{b^T y : A^T y \leq c\}.$$

The logarithmic barrier formulations of the primal (P) and dual (D) are

$$(P_\mu)\min\{c^T x - \mu \sum_{j=1}^{n} \ln(x_j) : \; Ax = b,\; x \geq 0\}$$

$$(D_\mu)\max\{b^T y + \mu \sum_{j=1}^{n} \ln(z_j) : \; A^T y + z = c\}.$$

Notice that $(P_\mu)/(D_\mu)$ converge to $(P)/(D)$ respectively, as $\mu \to 0^+$ The first-order optimality KKT (Karush–Kuhn–

Box 3. Newton's Method

Newton's method is usually easily understood in the context of finding the roots of an equation $f(x)=0$ of a scalar x. As indicated in *Figure* 3 the simple iteration $x_{k+1}=x_k-(f(x_k)/f'(x_k))$ converges locally to a root. In the vector case, the iteration is the vector equivalent, i.e., $x_{k+1}=x_k-J^{-1}(x_k)f(x_k)$ where $J(x)$ is the Jacobian of $f(x)$. The first derivative of a function f: $\mathbf{R}^n\rightarrow\mathbf{R}^n$ is called the Jacobian. It is an $n\times n$ matrix.

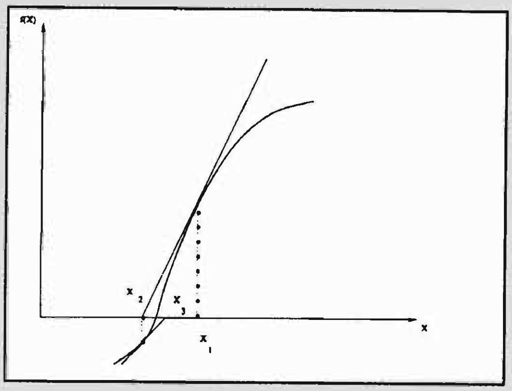

Figure 3. Newton's method.

Tucker) conditions (necessary and sufficient) for the nonlinear convex programming problem (D_μ) are given by

$$\begin{aligned} D_xD_ze &= \mu e \\ Ax &= b \\ A^Ty+z &= c, \end{aligned}$$

where D_x and D_z denote $n\times n$ diagonal matrices whose diagonals are x and z respectively. Notice that if we set μ to 0, the above conditions are precisely the primal-dual optimality conditions; complementary slackness, primal and dual feasibility of a pair of optimal (P) and (D) solutions.

The problem has been reduced to solving the above equations in x, y, z. The classical technique for solving equations is Newton's method (see *Box* 3) which prescribes the directions

$$\begin{aligned} \Delta y &= -(AD_xD_z^{-1}A^T)^{-1}AD_z^{-1}(\mu e-D_xD_ze) \\ \Delta z &= -A^T\Delta y \\ \Delta x &= D_z^{-1}(\mu e-D_xD_ze)-D_xD_z^{-1}\Delta z. \end{aligned} \tag{1}$$

Notice that these directions will maintain feasibility of (x,y,z) in (P) and (D), provided 'small' steps are taken.

The strategy is to take one Newton step, reduce μ and iterate until the optimization is complete. The criterion for stopping can be determined by checking if the duality gap $(x^t z)$ is close enough to 0. We are now ready to describe the algorithm.

```
Procedure: Primal-Dual Interior

  0. Initialize:
        • $x_0 > 0$, $y_0$ feasible in $(P)$ and $(D)$, $z_0 > 0, \mu_0 > 0, \epsilon > 0, \rho > 0$
        • $k := 0$
  1. Iterative Step:
        do
        Stop if $x_k^T z_k \leq \epsilon$.
        $x_{k+1} \leftarrow x_k + \alpha_k^P \Delta x_k$
        $y_{k+1} \leftarrow y_k + \alpha_k^D \Delta y_k$
        $z_{k+1} \leftarrow z_k + \alpha_k^D \Delta z_k$
        /* $\Delta x_k$,  $\Delta y_k$,  $\Delta z_k$ are the Newton directions from (1) */
        $\mu_{k+1} \leftarrow \rho \mu_k$
        $k := k + 1$
        od
  2. End
```

Remarks:

1. The primal-dual algorithm has been used in several large-scale implementations. For appropriate choice of parameters, it can be shown that the number of iterations in the worst-case is $O(\sqrt{n} \log(\epsilon_0/\epsilon))$ to reduce the duality gap from ϵ_0 to ϵ. While this is sufficient to show that the algorithm is polynomial time, it has been observed that the 'average' number of iterations is more like $O(\log n \log(\epsilon_0/\epsilon))$: How-

ever, unlike the simplex method we do not yet have a satisfactory theoretical analysis to explain this observed behaviour (see *Box* 2).

2. The stepsizes α_k^P and α_k^D are chosen to keep x_{k+1} and z_{k+1} strictly positive. The ability in the primal-dual scheme to choose separate stepsizes for the primal and dual variables is a major computational advantage that this method has over the pure primal or dual methods. Empirically this advantage translates to a significant reduction in the number of iterations.

3. An important assumption made in the algorithm is that we are given a pair of feasible, interior solutions to the linear programs (P) and (D) to start with. If such solutions are not evident, two possible strategies are to introduce a Phase I formulation using artificial variables or to introduce a corrector term in the Newton direction that includes a penalty for infeasibility. The latter approach has been proved more practical.

4. The stopping condition essentially checks for near complementary slackness. Exact complementary slackness is not possible with interior solutions. In any case, when the algorithm terminates with an interior solution, a post-processing step is usually invoked to obtain optimal extreme point solutions for the primal and dual. This is usually called the *purification* of solutions.

5. Instead of using Newton steps to drive the solutions to satisfy the optimality conditions of (D_μ), Mehrotra suggested a predictor-corrector approach based on power series approximations. This approach has the added advantage of providing a rational scheme for reducing the value of μ. It is the predictor-corrector based primal-dual interior method that is considered the current winner in interior point methods. The OB1 code of Lustig, Marsten and Shanno is based on this scheme. CPLEX and IBM's OSL, the two leading general purpose linear (and integer) programming solvers, also contain implementations of interior point methods.

6. Saltzman describes a parallelization of the OB1 method

Suggested Reading

[1] D A Bayer and J C Lagarias, The nonlinear geometry of linear programming, Parts I–IV, *Transactions of the American Mathematical Society*, 314, 2, 499–581, 1989.

[2] D den Hertog, *Interior point approach to linear, quadratic and convex programming*, Kluwer, 1994.

[3] N K Karmarkar, A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica*, 4, 373–395, 1984.

[4] N K Karmarkar, Riemannian geometry underlying interior-point methods for linear programming, *Contemporary Mathematics*, 114, 51–75, 1990.

[5] I J Lustig, R E Marsten, and D F Shanno, Interior Point Methods for Linear Programming: Computational State of the Art, *ORSA J. on Computing*, Vol. 6, No. 1, 1–14, 1994.

[6] S Mehrotra, On the implementation of a primal-dual interior point method, *SIAM Journal on Optimization*, Vol.2, No.4, 575–601, 1992.

[7] R Saigal, *Linear Programming: A modern integrated analysis*, Kluwer, 1995.

[8] M J Saltzman, Implementation of an interior point LP algorithm on a shared-memory vector multiprocessor, in *Computer Science and Operations Research*, edited by O Balci, R Sharda and S A Zenios, Pergamon Press, 87–104, 1992.

[9] S J Wright, Primal-Dual Interior-Point Methods, SIAM Press, 1997.

to run on shared-memory vector multiprocessor architectures. Recent computational studies of parallel implementations of simplex and interior point methods on the SGI Power Challenge (SGI R8000) platform indicate that on all but a few small linear programs in the NETLIB linear programming benchmark problem set, interior point methods dominate the simplex method in run times. New advances in handling Cholesky factorizations in parallel are apparently the reason for this exceptional performance of interior point methods.

7. As in the case of the simplex method, there are a number of special structures in the matrix A which can be exploited by interior point methods to obtain improved efficiencies. Network flow constraints, generalized upper bounds (GUB) and variable upper bounds (VUB) are structures that often come up in practice and which can be useful in this context.

8. Interior point methods, like ellipsoid methods, do not directly exploit the linearity in the problem description. Hence they generalize quite naturally to algorithms for semidefinite and convex programming problems. Karmarkar has also proposed an interior-point approach for integer programming problems. The main idea is to reformulate an integer program as the minimization of a quadratic energy function over linear constraints on continuous variables. Interior-point methods are applied to this formulation to find local optima.

Conclusion

The 1980's were an exciting period for linear programmers. The polynomial time-complexity of linear programming had just been established. A healthy competition between the simplex and Karmarkar's interior methods ensued which ultimately led to rapid improvements in both technologies. This combined with remarkable advances in computing hardware and software have brought powerful linear programming tools to the electronic desktop of the 1990's.

Web Sources

The linearprogrammingFAQ (frequently asked questions) facility is maintained at http://www.mcs.anl.gov/home/otc/Guide/faq/

To have a linear program solved over the internet check the following locations.
http://www.mcs.anl.gov/home/otc/Server/
http://www.mcs.anl.gov/home/otc/Server/lp/

Address for Correspondence
Vijay Chandru
Computer Science and Automation, IISc
Bangalore 560 012, India.
Email:chandru@csa.iisc.ernet.in
M R Rao
Director
Indian Institute of Management
Bannerghatta Road
Bangalore 560 076, India.