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.

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.
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 $Q \subseteq \mathbb{R}^d$, defined by linear inequalities, is non-empty. For technical reasons let us assume that $Q$ is rational, i.e. all extreme points and rays of $Q$ are rational vectors or equivalently that all inequalities in some description of $Q$ involve only rational coefficients.

The ellipsoid algorithm initially chooses an ellipsoid large enough to contain a part of the polyhedron $Q$ if it is non-empty. The centre of the ellipsoid is in $Q$ if all the inequalities defining $Q$ check out. Else, an inequality of $Q$ separates the centre point of the ellipsoid from the polyhedron $Q$. 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 $Q$ is empty. This fact is used to show the polynomial time convergence of the algorithm.