1
Simplex algorithms

István Maros

Brunel University, Uxbridge UB8 3PH, England
Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Hungary

Gautam Mitra

Brunel University, Uxbridge UB8 3PH, England

1 Introduction

Linear inequality systems were studied by Fourier, Motzkin and Farkas (see [14]) but only in the late 1940s did the work of Dantzig in the USA and Kantorovich in the former USSR make linear programming a leading research topic with wide ranging applications to problems of planning and scheduling as found in business, industry and government [14]. Dantzig developed the celebrated *simplex method* (in the tableau form) for finding the maximum or minimum of a linear function subject to linear restrictions. The resulting optimization problem because of its many planning applications was called linear programming (LP). Since that time there has been considerable curiosity driven research, as well as applicable research, on the topic of LP.

In Table 1 we have highlighted the major landmarks in theoretical developments. In Table 2 we have summarized some of the major computational developments.

We would, however, like to outline how a number of important research topics sprung out of LP and the simplex method and then became mature in their own right. All through the 1960s and 1970s the industrial application of equation solving technologies of large sparse systems continued to gain importance and applicability. The basis factorization (inversion) strategies of the simplex method were studied with considerable interest. A series of special conferences on the topic of sparse matrices and sparse equation solving methods was started during this period (Willoughby *et al.* [63, 69], Reid [60], Bunch and Rose [10], Duff and Stewart [19]). Today there is a wide body of literature which covers these developments. This has made a major contribution to the now established field of computational methods for large sparse systems where data structures, software implementation and exploitation of machine architecture continue to be leading research issues.

It can be claimed that the study of computational complexity grew out of

Table 1 Landmark contributions in theoretical developments of the simplex method

	Topic Area	Main Researchers	Date
1.	Primal simplex method	Dantzig	1947
2.	Duality theory and dual simplex algorithm	Lemke	1954
3.	Degeneracy and cycling in LP	Beale	1955
	Lexicographic rules for anti-degeneracy	Dantzig, Orden, Wolfe	1955
		Wolfe	1963
	Other approaches	Bland	1977
4.	Complexity of simplex algorithms		
	Worst case behaviour	Klee, Minty	1972
	Probabilistic/average behaviour	Borgwardt, Smale	1982
5.	Crossover: recovering an optimal basis from a pair of primal and dual optimal solutions	Megiddo	1991

the intractability of integer linear programs in contrast to linear programs. The algorithmic behaviour of the simplex has been well understood. Its average behaviour and worst case behaviour have been studied and explained by Borgwardt [6] and Klee and Minty [44], respectively. Karp in his Turing award lecture in [42] explained the central role occupied by LP and its solution methods in the area of computational complexity or in more practical terms the scalability of algorithms.

Indeed Khachiyan [43] was motivated to find a (worst case) polynomial algorithm for LP, however, computationally it performed very poorly. Karmarkar's work [41] and the subsequent development of the interior point method (IPM) resulted from his motivation to find a method which can dominate the simplex method computationally both in its average behaviour and in the worst case. The overwhelming computational success of IPM in turn strongly challenged the simplex algorithm designers who responded with a whole series of extensions and improvements. As a result the simplex algorithm continues to be the chosen method within software systems designed to solve large LP problems of widely varying structures. We believe simplex will continue to be widely used for the following reasons:

(1) There are many industrial applications in which "warm start" from an earlier solution of a slightly perturbed neighbouring LP problem provides considerable computational advantage. In these cases IPM performs poorly in contrast to simplex. Successive linear programming to solve non-linear programming problems in the oil industry is an example of such an application. Solution of subproblems in integer programming and post optimality analysis of LPs provide other examples of repeated use of warm start.

Table 2 Landmark contributions to the simplex-based computational solution methods

	Topic Area	Main Researchers	Date
1.	Tableau simplex method	Dantzig	1947
2.	Revised simplex method	Dantzig,	
		Orchard-Hays, Wolfe	1953/54
3.	Simple upper bound algorithm	Dantzig	1954
4.	Generalized upper bound algorithm	Dantzig, Van Slyke	1967
5.	Basis factorization and the elimination	Markowitz	1954
	form of the inverse (EFI)	Beale	1971
		Hellerman, Rarick	1971/72
6.	Sparse update procedures		
	Bartels, Golub	Bartels, Golub	1969
	Forrest, Tomlin	Forrest, Tomlin	1972
	Reid	Reid	1976
7.	Presolve procedures	Brearley,	
		Mitra, Williams	1975
8.	Composite Phase-I procedures	Wolfe	1965
		Maros	1986
9.	Combined price and pivot choice		
	DEVEX procedure	Harris	1973
	Steepest edge pricing	Goldfarb, Reid	1977
		Forrest, Goldfarb	1992

(2) Shadow prices (costs) play a key role in the descriptive analysis of many economic planning and business applications of LP. In these situations the established theory relating to optimal basis simplex multipliers and dual solution values have wide acceptance. This in turn requires that the simplex method is used to compute these items of (economically) meaningful descriptive information.

The simplex method has undergone very substantial improvement over the past two decades. These improvements are mainly computational and have the sole purpose of making the simplex solution method efficient as well as reliable in respect of solving a wide range of practical problems. This chapter sets out a summary account of these developments.

The rest of the chapter is organized in the following way. In Section 2 we state the linear programming problem and introduce different types of variables. In Section 3 the revised simplex method is set out in a summary form. Section 4 introduces the sparse simplex (SSX) method and the algorithmic developments relating to preprocessing, starting basis, pricing, degeneracy, inverse represen-

Table 3 Types of variables

Feasibility range	Type	Reference
$x_j = 0$	0	Fixed variable
$0 \leq x_j \leq u_j < +\infty$	1	Bounded variable
$0 \leq x_j \leq +\infty$	2	Non-negative variable
$-\infty \leq x_j \leq +\infty$	3	Free variable

tation and sparse updates. In short, we consider the major developments in the algorithmic components of SSX. In Section 5 we cover the computational issues relating to SSX. Thus we outline the information flow and computational use of SSX, we present an analysis of benchmark problems and provide profiling information for benchmark problems in respect of the major algorithmic components. We also set out comparative performance summaries of the two leading solvers, FortMP and MINOS, offered by academic institutions and two full strength commercial solvers, OSL and CPLEX. In Section 6 we summarize the recent extensions of SSX for parallel computing architecture and we also discuss the related difficulties and explain why there is very limited development in this field. We consider the integration of IPM and SSX in Section 7 and outline our view of future research directions in Section 8.

2 Problem statement

2.1 The primal problem

We consider the following primal linear programming (LP) problem:

$$\begin{aligned} &\text{minimize} && c^T x, && \\ &\text{subject to} && Ax = b, && (2.1)\\ & && l \leq x \leq u, && (2.2) \end{aligned}$$

where A is an $m \times n$ matrix, c, x, l, and u are n vectors, and b is an m vector. Some or all components of l and u can be $-\infty$ or $+\infty$, respectively. For computational convenience, it is assumed that all finite lower bounds in (2.2) are translated to zero. A itself contains a unit matrix I, that is, $A = [I, \bar{A}]$, so it is of full row rank. Variables which multiply columns of I transform every constraint to an equation and are often referred to as *logical variables.* Variables which multiply columns of $\bar{A}$ are called *structural variables.*

Based on the relation in (2.2), the variables (whether logical or structural) can be categorized as shown in Table 3 (for further details, see [58]).

An x vector that satisfies (2.1) is called a *solution.* If, additionally, it satisfies (2.2) it is called a *feasible solution.*

LP systems allow for other types of variables and constraints. Namely, a variable can have infinite lower bound and finite upper bound (*minus type variable*). Such variables are transformed during the setup phase into type-2, that is, non-negative variables with zero lower bound. Similarly, a constraint can be of