Working Paper

Solving a Class of LP Problems with a Primal-Dual Logarithmic Barrier Method

Jacek Gondzio, Marek Makowski

WP-93-02
January 1993

Solving a Class of LP Problems with a Primal-Dual Logarithmic Barrier Method

Jacek Gondzio, Marek Makowski

WP-93-02
January 1993

Foreword

This Working Paper describes the implementation of an LP solver at IIASA. The primary motivation for this research was to provide an efficient and distributable solver for optimization problems frequently generated in collaborative studies of the Food and Agriculture Project with other organizations.

Such optimization problems are often large LP problems and therefore the solver should be both efficient, robust and royalty free for distribution. After the analysis of the characteristics of the LP problems and of the above stated requirements, the HOPDM code, developed at the Systems Research Institute of the Polish Academy of Sciences, has been selected for application.

In order to make the computations easier, a general purpose library for data interchange between the problem generator, the report writer (both developed by the FAP Project) and the solver has been implemented. By using this library one can avoid the MPS format files, which, especially for large problems, require substantial amount of computer resources for generation and reading of respective input and output files.

Both the solver and the library is available in two environments, namely on Sun (under Sun OS 4.1.) and on personal computers running under MS-DOS. The software can be easily used for other applications that require solving medium or large size LP problems.

Contents

Solving a Class of LP Problems with a Primal-Dual Logarithmic Barrier Method

Jacek Gondzio[*], *Marek Makowski*[**]

Abstract

Applying a higher order primal-dual logarithmic barrier method for solving large real-life linear programming problems is addressed in this paper. The efficiency of interior point algorithm on these problems is compared with the one of the state-of-the-art simplex code MINOS version 5.3. Based on such experience, a wide class of LP problems is identified for which logarithmic barrier approach seems advantageous over the simplex one. Additionally, some practical rules for model builders are derived that should allow them to create problems that can easily be solved with logarithmic barrier algorithms.

Keywords: Linear Programming, Primal-Dual Method, Agriculture, Applications.

1 Introduction

The objectives of the research reported in this paper are two-fold. First, to apply the HOPDM (higher order primal-dual method cf [1, 2]), which is an efficient implementation of the primal-dual interior point method, for solving large real-life linear programming (LP) problems. Second, to provide the Food and Agriculture Project (FAP) at IIASA with a robust and efficient solver that could be used and distributed for solving optimization problems frequently generated in collaborative studies with other organizations, related to land use and development planning.

FAP was established in 1976 as an international research program which resulted in the development of a linked system of national models to analyze the implications of agricultural policies on world food supply, demand and trade (cf Fischer et al. [5]). FAP also embarked in research on land resources assessment and development planning. Some of these research activities require solving large LP models. Models that resulted from one of these assessments (cf Section 2) have been selected for a pilot study aimed at application of the HOPDM, a primal-dual code due to Altman and Gondzio (cf [1, 2]).

The primal-dual method applies logarithmic barrier functions for handling inequality type constraints. Although the use of such functions was studied in detail by Fiacco and McCormick [4] in 1968, this approach was never made operational and competitive in the context of large scale LP until the mid-eighties. Soon after Karmarkar's [11] publication of

[*]Systems Research Institute, Polish Academy of Sciences, Newelska 6, PL-01-447 Warsaw, Poland.
[**]International Institute for Applied Systems Analysis (IIASA), A-2361 Laxenburg, Austria. On leave from the Systems Research Institute, Warsaw.

the new polynomial-time linear programming algorithm, Gill et al. [10] established (under some restrictive assumptions) the equivalence of this new interior point method and the logarithmic barrier approach.

Megiddo [15] proposed applying logarithmic barriers to both primal and dual problems at a time, which led several authors to practical (implementable) methods. This primal-dual algorithm proved to be the most efficient variant of the interior point methods. There exist several efficient implementations of it, e.g. OSL of Forrest and Tomlin [9], OB1 of Lustig et al. [13], the code of Mehrotra [16], and the earlier mentioned HOPDM of Altman and Gondzio [1, 2]. Independently, great progress has been made in the simplex implementations (see e.g. CPLEX of Bixby [3], OSL of Forrest and Tomlin [8], and MINOS of Murtagh and Saunders [17]).

The two approaches for solving LP problems compete with each other and there is no other way to state definite superiority of any of them than making a fair comparison on a wide class of tests. Although absolute superiority of one approach will probably never be established, it is possible to identify applications (or classes of LP problems) for which one of the methods is better suited than the other. For example, the need of having vertex solution and/or performing post-optimal analysis favorizes simplex method. On the other hand, the requirement of having a strictly interior solution (some LP constraints might be formulated as strong inequalities) clearly favorizes the logarithmic barrier method. In the majority of applications, however, the most important criterion is the performance of the method (assuming reliability of solution, of course).

In the authors opinion, a very large size of the problem being solved should favorize using an interior point based solver unless the linear program has some special structure (e.g. network) for which dedicated simplex implementations exist.

In this paper, we are concerned with an application of a primal-dual logarithmic barrier method for solving large real-life FAP problems. We thus first describe briefly in Section 2 the class of problems we are dealing with.

In Section 3, we recall the basic ideas of logarithmic barrier method and its modification applying higher order trajectory approximation due to Mehrotra [16]. Next, in Section 4, we compare the performance of two LP codes: MINOS 5.3 (which is reported to be a great improvement over the earlier version 5.1) simplex implementation of Murtagh and Saunders [17] and HOPDM implementation of the predictor-corrector primal-dual method by Altman and Gondzio [1, 2]). In particular, we demonstrate high efficiency of the technique of dropping inactive constraints which lies behind the success of HOPDM code and is mutual for logarithmic barrier algorithm, especially its primal-dual variant.

In conclusion, we will identify a wide class of LP problems that are "easy" for the primal-dual logarithmic barrier method. We end the paper addressing those who use LP solvers in their Operations Research (OR) applications and give a few practical rules concerning the building of an LP model and a choice of a most suitable solver.

2 The AEZ Land Resources Assessment for Development Planning

Many of the research activities of the FAP Project, which are performed in cooperation with the FAO (Food and Agriculture Organization) and with research and governmental agencies in the countries involved, require solving different medium- or large-size LP problems. Therefore the FAP Project can easily provide good testing examples for LP solvers and solving such examples also have practical meaning. One of the current activities of the FAP Project at IIASA is aimed at modeling agro-ecological land resources assessment for agricultural development planning. For this purpose both the relevant sets of data and the software for generation of mathematical programming problems have been developed (cf Fisher et. al.[6]).

The research which provided testing examples discussed in this paper is described by Fischer and Shah in [7], who summarize the problem in the following way: "Food, crop and livestock production in most West African countries has in recent years not expanded fast enough to keep up with population growth. The main contribution of this study relates to a quantitative assessment of the potential role of forage legumes in contributing to livestock supplies on a seasonal and spatial basis together with their importance in conserving soil as well as improving soil fertility. The approach, based on Agro-Ecological Zones (AEZ) land resources inventory, is aimed at specifying national level development scenarios, integrating crop and livestock sectors to appraise the scope for food self-sufficiency in thirteen West African countries." A more detailed description of this research is beyond the scope of this paper. The FAP Project has selected from this study eighteen examples of models whose characteristics are listed in Table 1 in Section 4. The name of each problem reported contains both the name of a country for which the model was generated and a two- (or three)-character acronym which identifies the kind of scenario tested for this particular model. However, we would like to stress, that the examples were not selected for illustrating the research reported in [7] but rather for providing different (from the numerical point of view) examples that represent well the corresponding class of LP problems.

3 Primal-Dual Method

We shall present in this section the basic ideas of a primal-dual logarithmic barrier method. For a detailed discussion of its theoretical properties the reader is referred to Kojima et al. [12] and Megiddo [15].

The method deals at the same time with a dual pair of LP problems:

$$\text{minimize } c^t x, \tag{1a}$$

subject to:

$$Ax = b, \tag{1b}$$

$$x + s = u, \tag{1c}$$

$$x, s \geq 0, \tag{1d}$$

where $c, x, s, u \in \Re^n, b \in \Re^m, A \in \Re^{m \times n}$ and

$$\text{maximize } b^t y - u^t w, \tag{2a}$$

subject to:

$$A^t y + z - w = c, \tag{2b}$$

$$z, w \geq 0, \tag{2c}$$

where $y \in \Re^m$ and $z, w \in \Re^n$.

Its key idea is to handle inequality type constraints by means of logarithmic barrier functions, which leads to the following Lagrangian for (1)-(2)

$$L(x,s,y,w,\mu) = c^t x - \mu \sum_{i=1}^{n} ln(x_i) - \mu \sum_{j=1}^{n} ln(s_j) - y^t(Ax-b) - w^t(u-x-s) \qquad (3)$$

where μ denotes the barrier parameter.

The first order conditions necessary for the point $(x^*, s^*, y^*, z^*, w^*)$ to be optimal for (3) are

$$Ax = b, \qquad (4a)$$

$$x + s = u, \qquad (4b)$$

$$A^t y + z - w = c, \qquad (4c)$$

$$XZ = \mu e, \qquad (4d)$$

$$SW = \mu e, \qquad (4e)$$

$$x, s, z, w > 0, \qquad (4f)$$

where $X, S, Z,$ and W are diagonal matrices with the elements x_j, s_j, z_j and w_j, respectively and $e \in \Re^n$ is a vector of ones.

The basic primal dual algorithm takes one step of the Newton method to find an approximate solution to (4) for a current μ and modifies (usually decreases) μ accordingly to the progress made in reduction of the duality gap. The algorithm terminates when the duality gap is reduced to a predetermined tolerance.

Newton's direction for any of the variables considered x, s, y, z and w may be decomposed into two parts

$$\Delta = \Delta_a + \Delta_c, \qquad (5)$$

affine scaling one Δ_a and centering one Δ_c. The first one is supposed to improve the objective (reduce duality gap) while the second one points to the analytical center of the simplex which in practice means keeping current iterate away from the boundary of a feasible region. The presence of Δ_c component in (5) ensures practical fast convergence of the algorithm as it prevents the trajectory from approaching those constraints which are not binding at the optimum. It is a useful property of the primal-dual algorithm as it allows elimination of blocks of inactive constraints in its successive iterations.

Basic algorithm applies the first order Newton's method to find corrections of a current iterate x, s, y, z, w. Mehrotra proposed a computationally attractive modification that incorporates higher order information into the primal-dual method, namely, computes the k-th order approximation of the optimal trajectory that starts at a given point and leads to the optimum. The number of primal-dual iterations can thus be reduced significantly, which, as long as the order of the Taylor polynomial is not too large, gives remarkable computational savings. Efficiency of the Mehrotra's predictor-corrector technique results from the fact that it reduces the number of very expensive factorizations of $A\theta A^T$ matrix and better exploits the information from a single factorization (solves more equations with the same decomposition). It thus reduces the computation time considerably as the cost of solves with triangular factors of $A\theta A^T$ is at least an order of magnitude smaller than that of the factorization.

It is not in the scope of this paper to discuss theoretical aspects of the primal-dual method. An interested reader is referred to paper of Mehrotra [16]. Instead, we shall address a very promising computational technique that can naturally be embedded into any of its implementations. This technique allows early identification and elimination of inactive inequality-type constraints. To our knowledge, HOPDM is the only primal-dual code with such a technique incorporated, and as a result of our experience gained on FAP problems, we want to recommend it as an attractive option to be included in any primal-dual implementation.

We exploit the structural property of the primal-dual logarithmic barrier algorithm that follows optimal trajectory leading from a given point to the optimum of a linear program (see e.g. Megiddo [15]). What is even more important is that this trajectory is supposed to lie deeply in the interior of the feasible region and not to approach those constraints which are not binding at the optimum. It is a specific feature of the method resulting from the presence of logarithmic barriers in Lagrangian and, consequently, the presence of a centering term in the Newton's direction (5). Hence it is possible to formulate a practical rule for early elimination of inactive inequality-type constraints.

From the first order optimality conditions for a given point to be optimal we get

$$y_i(Ax-b)_i = 0, \quad \text{for } i = 1,2,\cdots,m, \tag{6}$$

which for inequality-type constraints translates to the requirement

$$y_i x_i^s = 0, \text{ for } i = 1,2,\cdots,m, \tag{7}$$

where x_i^s is a slack (surplus) variable associated with the i-th constraint. If thus some slack variable x_i^s is bounded away from zero, which in practice gives condition

$$x_i^s \geq \xi,$$

with some predetermined threshold ξ ($\xi = 1$) and dual variable associated with appropriate constraint satisfies

$$|y_i| < \delta$$

for some predetermined tolerance δ ($\delta = 10^{-5}$), then we assume that at the optimum

$$x_i^s > 0 \text{ and } y_i = 0.$$

In other words, i-th constraint is presumed inactive at the optimum and may be removed from the LP problem formulation.

The technique of elimination of inactive constraints proved to be very efficient in practice. Next section addresses its application to solving FAP problems.

4 Test Results

We have run two LP solvers on a sample of 19 problems (cf Section 2). The first solver is a state-of-the-art simplex implementation MINOS 5.3 from Murtagh and Saunders. This version is, up to our knowledge, the newest one available and represents significant improvement over the earlier one (version 5.1) of 1987. The second is a higher order primal-dual code HOPDM of Altman and Gondzio that, when applied to solving problems from the Netlib collection, has been shown to compare favorably with other primal-dual codes available, e.g. OB1 of Lustig et al. [13] and the code of Mehrotra [16].

The HOPDM is coded in fortran-77, and the interface library as well as the dynamic memory allocation is coded in ANSI-C. We have used the standard fortran-77 Sun compiler and the Gnu C compiler, both with the optimization option. The same fortran compiler and option were used for compiling MINOS. Our code can also be compiled by the Microsoft Fortran 5.1 and C ver. 6.0 for PC under DOS.

The tests have been performed with HOPDM and MINOS on a Sun Sparc-2 computer with the Weitek 3170-based FPU run at 33.0 MHz. The values of the SPEC (the System Performance Evaluation Cooperative) indices for this machine are 21.8 and 22.8 for SPECint92 and SPECfp92, respectively. Less informative but more common MIPS and MFLOPS values are 28.5 and 4.3, respectively.

Both solvers were run with the default settings of parameters for controlling stability of respective methods except for one customization done to the HOPDM. Namely, the default starting point has been replaced by the following:

$$x_j = \begin{cases} 0.5 & \text{if } u_j < \infty; \\ 10 & \text{otherwise} \end{cases}$$

This modification was motivated by the presence of a very large number of upper bounds equal to 0.75 or 1.0 in the FAP linear models and resulted in the average 20-50% improvement of the HOPDM efficiency.

Problem		M	N	NZ	MINOS		HOPDM	
					iters	time	iters	time
Gambia	B2	153	203	1604	9	0.1	13	2.1
Gambia	B2	246	1379	9401	746	8	36	46
Liberia	B9	384	1330	10355	898	11	24	27
Liberia	B1	392	1331	10171	1853	24	36	44
Liberia	B7	412	1930	13761	1693	22	27	41
Liberia	B2	420	1931	13085	2332	34	40	67
Liberia	B52	461	2547	18887	2807	45	41	94
Senegal	B7	826	6979	48666	8494	274	53	390
Senegal	B2	833	6980	45487	9410	309	47	362
Burkina	B7	1026	8525	60184	7604	269	49	413
Cameroon	B9	1566	6938	56993	7546	355	38	345
Cameroon	B1	1574	6939	56091	10645	522	44	420
Cameroon	B7	1731	11573	81419	11935	701	49	575
Cameroon	B2	1739	11574	76584	23423	1351	66	753
Nigeria	B2	1846	16422	103580	43723	3009	64	987
Cameroon	B52A	1881	20429	168740	41084	3089	58	1120
Nigeria	C82	1989	31041	267345	19758	1792	61	1568
Nigeria	B52	1992	29614	239862	54304	5102	67	1731
Cameroon	B52	2057	21203	193167	39046	3213	63	1414

Table 1: *Comparison of MINOS 5.3 and HOPDM.*

Table 1 summarizes the results obtained. It contains dimensions of problems solved (M, N and NZ denote, numbers of rows, columns and nonzero elements, respectively) and iteration counts and computational time in seconds for both solvers used.

We report only pure solution time (with excluded time for model input and solution output) for both solvers compared. The omitted time for model input and solution output may reach the level of considerable fraction of the time spent in solver, especially if the input/output format is the popular but inefficient MPS one. For actual implementation of the HOPDM we have applied the concept of MP-DIT (Mathematical Program Data Interchange Tool) from Makowski and Savelsbergh [14]. We have implemented an interface library which can be used by a solver, by a problem generator and by a report writer. The library uses an efficient binary format (which is hidden from a user) for data interchange which makes it possible to replace the need of generating a formatted MPS file with a problem generator and processing such a file by a solver. Such an approach is not only much faster but it is also easier to implement for both a problem generator and a solver. It is presently planned to make the interface library available as a public domain software for non-commercial applications. As a result of using the interface library, the time needed for the model input is much smaller than that required by reading and processing the MPS-format input and it became negligible compared with time required to solve the problem. Such an approach is of course independent of the LP solver chosen, so its contribution is not taken into account in the results collected in Table 1. It is however strongly recommended for any real-life application.

Results collected in Table 1 show that for smaller problems MINOS is faster. HOPDM favorizes a large problem size. The breakpoint (for this particular class of problems) is about M = 1000. In other words, if problems with the number of constraints larger than a thousand are to be solved, then it is advisable to use the logarithmic barrier method for this purpose.

iteration	*M*	$nz(A)$	$nz(AA^T)$	$nz(L)$	*MFLOPS*
1	2057	193167	90878	169823	16.7
32	2057	193167	90878	169823	16.7
34	2048	192759	90247	168169	16.5
36	1978	188734	82419	149923	13.8
38	1974	186010	81279	147124	13.4
40	1951	183381	78979	142995	12.8
42	1930	179076	76751	137332	12.0
44	1907	171873	73777	132433	11.4
46	1893	169734	72265	129866	11.0
48	1875	164149	69751	124992	10.4
50	1853	141389	65943	119891	9.8
52	1808	127424	60066	111487	8.8
54	1754	114436	54405	102889	7.8
56	1684	103674	48384	94452	7.0
58	1659	101290	46483	90746	6.7
60	1639	100344	45876	88019	6.5

Table 2: *Changes of the Cameroon-B52 problem size*

Our next experiment demonstrates advantages of eliminating inactive constraints during the optimization process. Results collected in Table 2 monitor changes of the size of the real problem to be solved for one of the largest linear program from our collection. For subsequent iterations of the primal-dual algorithm it thus reports: number of LP constraints M, number of non-zero elements of still active part of A, number of non-zero elements in the adjacency structure AA^T, number of off-diagonal elements of Cholesky factor L and number of millions of flops (floating point operations) required to compute Cholesky decomposition of $A\theta A^T$ matrix. This last value may be viewed as the approximate cost of a single primal-dual iteration (it is supposed that Cholesky decomposition takes 60-70% of the time of every iteration).

It easily follows from the analysis of Table 2 that primal-dual logarithmic barrier method is able to eliminate a remarkable fraction of LP constraints before reaching optimum, which is not possible in a simplex method. Thus if the linear program has a considerable number of inequality constraints and a supposition exists that many of them will be inactive at the optimum, then it is advisable to apply logarithmic barrier method to solve it. A problem's property of having many inactive inequality constraints can easily be identified in the problem's solution. One can expect that any problem with many more rows than columns should possess such a property. We encourage OR practitioners to try primal-dual logarithmic barrier method in such cases.

The last observation we would like to share is due to specification of large values of lower and/or upper bounds whenever a default bound value should not apply for a particular column. This widely used approach (which is also recommended by some LP solvers) has no negative-side effects for the simplex method. However, it results in a substantial increase of computation time for a logarithmic barrier method. For example, specification of large values of bounds that served for removing default bounds for the FAP examples have resulted sometimes in more than doubling the execution time. Therefore we strongly recommend, for interior point methods, explicit specifications of a lack of a respective bound.

5 Conclusions and Remarks

We have presented in this paper the performance of two LP solvers: MINOS version 5.3 simplex implementation and HOPDM predictor-corrector primal-dual code applied for solving FAP problems of different sizes (the largest tested example has over 20,000 variables and close to 200,000 non-zero elements). For those particular linear programs growth of the size clearly favorizes primal-dual code, which, as we assume, is typical in general.

We have shown that the ability of early identification (before the optimum is reached) of inactive constraints, that is mutual for logarithmic barrier approach, may remarkably improve its performance. Time savings that results from application of this technique vary from 10% to 30% for FAP problems and we expect similar results on a wide class of LP problems that have remarkably many inactive constraints at their optimum. We have shown the advantages of using for real-life applications an efficient tool for interchange of data between a solver and a problem generator. Finally we pointed out an easy change in the LP problem specification which can result in substantial performance improvement of a solver based on logarithmic barrier method.

References

[1] Altman A., and J. Gondzio. An efficient implementation of a higher order primal-dual interior point method for large sparse linear programs, (to appear in: Archives of Control Sciences).

[2] Altman A., and J. Gondzio. HOPDM–A higher order primal-dual method for large scale linear programming. (to appear in: European Journal of Operational Research).

[3] Bixby R. Implementing the simplex method: the initial basis. ORSA Journal on Computing 4, No 3, pp. 267-284.

[4] Fiacco A.V. and G.P. McCormick. Nonlinear Programming: Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968.

[5] Fischer G. and K. Frohberg, M.A. Keyzer, K.S. Parikh, Linked National Model: A Tool for International Food Policy Analysis, Kluwer Academic Publishers, 1988.

[6] Fisher G.W. and M.M. Shah, A.H. Kassam, H.T. van Velthuizen, System Documentation Guide to Computer Programs for Land Productivity Assessment, Technical Annex 7, Agroecological Land Resources Assessment for Agricultural Development Planning, A Case Study of Kenya, AGL-FAO/IIASA, Rome, 1991.

[7] Fisher G.W. and M.M. Shah, M.A. Saleem, R. von Kaufmsun, Potential for Forage Legumes and Relevance to Food and Livestock Production in West Africa, IIASA/ILCA, Addis Ababa, (forthcoming).

[8] Forrest J.J.H. and J.A. Tomlin. Implementing the Simplex Method for the Optimization Subroutine Library, IBM Systems Journal 31, No. 2 pp. 11-25.

[9] Forrest J.J.H. and J.A. Tomlin. Implementing Interior Point Linear Programming Methods in the Optimization Subroutine Library, IBM Systems Journal 31, No. 2 pp. 26-38.

[10] Gill P.E., W. Murray, M.A. Saunders, J.A. Tomlin, and M.H. Wright. On Projected Newton Barrier Methods for Linear Programming and an Equivalence to Karmarkar's Projective Method, Mathematical Programming 36, pp. 183-209.

[11] Karmarkar N. A New Polynomial-time Algorithm for Linear Programming. Combinatorica 4, pp. 373-395.

[12] Kojima M. and S. Mizuno, and A. Yoshise. A Primal-dual Interior Point Algorithm for Linear Programming. N. Megiddo ed. in: Progress in Mathematical Programming, Springer-Verlag, New York, 1986 pp. 29-48.

[13] Lustig I.J., R. Marsten, and D.F. Shanno. On Implementing Mehrotra's Predictor-corrector Interior Point Method for Linear Programming, SIAM Journal on Optimization 2, pp. 435-449.

[14] Makowski M. and M.W.P. Savelsbergh. MP-DIT Mathematical Programming Data Interchange Tool, (to appear: Mathematical Programming Society COAL Newsletter).

[15] Megiddo N. Pathways to the Optimal Set in Linear Programming. N. Megiddo ed. in: Progress in Mathematical Programming, Springer-Verlag, New York, 1986 pp. 131-158.

[16] Mehrotra S. Higher Order Methods and their Performance. Technical Report 90-16R1, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois, 1991.

[17] Murtagh B.A. and M.A. Saunders. MINOS 5.1 User's guide. Technical Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, California, 1983 (revised 1987).