A Verification Method for Solutions of Linear Programming Problems

Ismail I. Idriss

Institute for Applied Research
University of Applied Sciences / FH Konstanz
D-78405 Konstanz, Germany
idriss@fh-konstanz.de

Abstract. This paper is concerned with the verification of a solution of a linear programming problem obtained by an interior-point method. The presented method relies on a reformulation of the linear programming problem as an equivalent system of nonlinear equations and uses mean value interval extension of functions and a computational fixed point theorem. The designed algorithm proves or disproves the existence of a solution on a computer and, if it exists, encloses this solution in narrow bounds.

Keywords: interior-point algorithms, linear programming, interval arithmetic, verified computation.

1 Introduction

Linear programming problems have a wide range of practical applications that arise in such diverse areas as economics, computer science, operations research, medicine, finance, mathematics, as well as many branches of engineering.

There are a number of general algorithms for the solution of the linear programming problem. It can be solved very efficiently by recently developed interior-point methods. An interior-point algorithm is an iterative technique that approaches the optimal solution by generating a sequence of points in the interior of the feasible region.

The modern era of interior-point methods dates to 1984, when Karmarkar [7] proposed his new polynomial-time algorithm for linear programming. After the publication of Karmarkar's algorithm, various interior point methods were developed which are based on it. However, until now these algorithmic developments have always aimed at increasing the runtime efficiency and limiting the computational effort, not at improving the accuracy and reliability of the computational results.

Infeasible interior-point algorithms are known as the most efficient computational methods for solving linear programming problems. But these algorithms sometimes compute an approximate solution with duality gap less than a given tolerance even when the problem may not have a solution [14,17].

One approach for proving or disproving the existence and possibly also uniqueness of a solution on computers is the use of interval arithmetic, e.g., [1,8,12]. This enables data uncertainties to be considered. Moreover, by performing outward rounding all rounding errors appearing during the computation can be taken into account. As a consequence, one

J. Dongarra, K. Madsen, and J. Waśniewski (Eds.): PARA 2004, LNCS 3732, pp. 132–141, 2006.

works with intervals of floating point numbers instead of with the numbers themselves. The computation of intervals is performed according to the rules of interval arithmetic. An algorithm implemented with interval arithemtic delivers (often tight) upper and lower bounds on the exact solution which are guaranteed also in the presence of rounding errors.

The use of interval arithmetic enables a rigorous verification of the existence and uniqueness of a solution and the computation of a guaranteed enclosure [8]. This verification is an application of Brouwer's fixed-point theorem which requires that a continuous function maps an interval vector onto itself. Such conditions can easily be tested on computers using fundamental properties of interval arithmetic. Enclosures with narrow bounds usually cannot be obtained by naively replacing real numbers by intervals and every arithmetic operation by the corresponding interval operation. One approach to improve enclosures of a differentiable function evaluated with interval arguments is to use the mean value theorem for differentiable functions in connection with an interval extension of the derivative.

The paper is organized as follows. In Section 2 we consider an infeasible interior-point algorithm for solving linear programming problems. In Section 3 we propose an algorithm that tests for the existence of a solution. Numerical results are presented in Section 4. We conclude with an empirical evaluation of our algorithm and some suggestions for further research.

In this paper, vectors are usually denoted by lower case letters, and matrices by upper case letters. Intervals and interval vectors are usually denoted by boldface lower case letters. It should be clear from the context, whether the letter denotes an interval or an interval vector. We denote the set of real numbers by $\mathbb{R}$ and the set of the compact nonempty real intervals by $\mathbb{IR}$.

2 Infeasible Interior-Point Algorithm

We consider the linear programming problem in standard form, which is generally referred to as the primal problem

$$\begin{aligned} &\text{Minimize} && c^\top x \\ &\text{subject to} && Ax = b, \\ & && x \geq 0 \end{aligned} \tag{P}$$

and its dual problem after adding the slack variable z to convert it to equality form

$$\begin{aligned} &\text{Maximize} && b^\top y \\ &\text{subject to} && A^\top y + z = c, \\ & && z \geq 0 \end{aligned} \tag{D}$$

where $c, x, z \in \mathbb{R}^n$, $b, y \in \mathbb{R}^m$ are vectors and $A \in \mathbb{R}^{m \times n}$ is a matrix. Every feasible solution for one of these two problems gives considerable information about the feasibility and optimality of the other. Moreover, it can easily be shown that for a primal feasible x and dual feasible y and z,

$$c^\top x - b^\top y = x^\top z,$$

holds true. The left hand side is called the duality gap and provides an excellent and easily obtainable measure of closeness of the computed solution to the optimal one.

An infeasible interior-point algorithm solves a primal-dual pair starting from an arbitrary infeasible point that satisfies the positivity constraints, but does not necessarily satisfy the equality constraints. Then the algorithm usually generates a sequence of iterates of feasible or infeasible solutions that approach feasibility and optimality. In the limit the iterates will converge to an approximate optimal solution with duality gap less than a given tolerance. The algorithm can be derived by applying the logarithmic barrier method to the primal problem (P), or alternatively, to the dual problem (D). Here we consider the dual problem (D) which is transformed to

$$\begin{aligned} &\text{Maximize} && b^\top y - \mu \sum_{j=1}^{n} \ln z_j \\ &\text{subject to} && A^\top y + z = c, \end{aligned} \tag{D_μ}$$

where $\mu > 0$ is a barrier parameter. For more details on the relation between (D) and (D_μ), see [2], where the logarithmic barrier method is introduced and [3], where it is developed. The Lagrangian for (D_μ) is

$$L(x, y, z, \mu) = b^\top y - \mu \sum_{j=1}^{n} \ln z_j - x^\top (A^\top y + z - c)$$

and the first order conditions for (D_μ) are

$$\begin{aligned} \nabla_x L &= c - z - A^\top y = 0, \\ \nabla_y L &= b - Ax = 0, \\ \nabla_z L &= \mu Z^{-1} e - X = 0, \end{aligned} \tag{2.1}$$

where X and Z are $n \times n$ diagonal matrices with elements x and z, respectively, and e is the n-vector of all ones. Newton's method can be applied to (1) in order to compute a search direction and then to determine a better estimate of the solution of the primal-dual pair.

Furthermore, Mehrotra [10] proposed a predictor-corrector method as an efficient strategy to compute the search direction. In particular, the method can be derived directly from the optimality conditions (2.1). Substituting (x, y, z) by $(x + \Delta x, y + \Delta y, z + \Delta z)$ leads to the implicit equations

$$\begin{aligned} A^\top \Delta y + \Delta z &= c - z - A^\top y, \\ A\Delta x &= b - Ax, \\ X\Delta z + Z\Delta x &= \mu e - XZe - \Delta X \Delta Z e, \end{aligned} \tag{2.2}$$

where ΔX and ΔZ are $n \times n$ diagonal matrices with elements equal to Δx and Δz, respectively.

Mehrotra proposed to first solve the affine system

$$\begin{aligned} A^\top \Delta\hat{y} + \Delta\hat{z} &= c - z - A^\top y, \\ A\,\Delta\hat{x} &= b - Ax, \\ X\Delta\hat{z} + Z\Delta\hat{x} &= -XZe, \end{aligned} \tag{2.3}$$

and then substitute the vectors $\Delta\hat{x}$ and $\Delta\hat{z}$ computed from (2.3) in the term $\Delta X \Delta Z e$ on the right-hand side of (2.2). Moreover, he suggested testing the reduction in the complementarity $(x + \alpha_P \Delta\hat{x})^\top (z + \alpha_D \Delta\hat{z})$, where α_P and α_D are the largest step lengths in the primal and dual variables which ensure $x > 0$ and $z > 0$. If we let

$$\hat{\mu} = (x + \alpha_P \Delta\hat{x})^\top (z + \alpha_D \Delta\hat{z}),$$

then a good estimate for the barrier parameter μ is

$$\mu = \left(\frac{\hat{\mu}}{x^\top}\right)^2 \left(\frac{\hat{\mu}}{n}\right).$$

This generates μ to be small when the affine direction produces a large decrease in complementarity and large otherwise.

Finally, the most current interior-point codes are based on the predictor–corrector technique [10]. Implementations of the algorithm also incorporate other heuristics that are essential for robust behaviour on many practical problems, including the determination of a starting point, the choice of step length, updating the barrier parameter μ, and the computing of the predictor–corrector direction. More details and further information on implementation techniques and the most relevant issues of infeasible interior–point methods are given in [17].

3 A Verification Method

In this section, we develop a practical verification method via reformulation of the linear programming problem as an equivalent system of nonlinear equations.

We define the function $F : \mathbb{R}^{2n+m} \to \mathbb{R}^{2n+m}$ by

$$F(x, y, z) = \begin{pmatrix} Ax - b \\ A^\top y + z - c \\ xZe - \mu e \end{pmatrix}.$$

Then we can rewrite conditions (2.1) in the form $F(x, y, z) = 0$. The Jacobian of F is given by

$$J_F(x, y, z) = \begin{pmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ Z & 0 & x \end{pmatrix},$$

which is nonsingular for all $(x, z) > 0$.

By using the notation

$$u = (x, y, z),$$
$$\mathbf{u} = \{(u_1, u_2, \ldots, u_{2n+m})^\top \in \mathbb{R}^{2n+m} \mid \underline{u}_i \leq u_i \leq \overline{u}_i,\ 1 \leq i \leq 2n+m\},$$

the problem is then to verify that there exists a $u^* \in \mathbf{u}$ such that $F(u^*) = 0$. Interval arithmetic in combination with the mean value extension and Brouwer's fixed point theorem can be used to verify the existence and uniqueness of a solution to $F(u) = 0$ (see [8]).

The mean value extension can be constructed using both the mean value theorem and the interval extension of the derivative.

Theorem 1 (Mean value extension). *Let $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function and let the gradient ∇f have an interval extension for all $\mathbf{u}$ contained in the domain of f. Then*

$$f(u) \in f(\tilde{u}) + \nabla f(\mathbf{u})\,(\mathbf{u} - \tilde{u}) \quad \textit{for all} \quad u, \tilde{u} \in \mathbf{u}.$$

Proof. For a proof see, for example, [1].

A computationally verifiable sufficient condition is given by Krawczyk [9] for the existence of a solution to a system of nonlinear equations using an interval version of Newton's method.

Definition 1. *Let $F : D \subseteq \mathbb{R}^n \to \mathbb{R}^n$ be a continuously differentiable function and let $\mathbf{u}$ be an interval vector in $\mathbb{IR}^n$ with $\mathbf{u} \subseteq D$. Then shape Krawczyk's operator is defined as:*

$$K(\mathbf{u}, \tilde{u}) = \tilde{u} - R \cdot F(\tilde{u}) + (I - R \cdot J_F(\mathbf{u}))(\mathbf{u} - \tilde{u}),$$

where $\tilde{u}$ is a vector in $\mathbf{u}$, $J_F(\mathbf{u})$ is an interval evaluation of the Jacobian of F, R is a real nonsingular matrix approximating $(J_F(\tilde{u}))^{-1}$ and I is the $n \times n$ identity matrix.

A verification method can be derived by iteratively applying Krawczyk's operator until the conditions of the following theorem are satisfied:

Theorem 2 (Existence and uniqueness). *If $K(\mathbf{u}, \tilde{u}) \subseteq \mathbf{u}$, then F has a zero u^* in $K(\mathbf{u}, \tilde{u})$ and therefore also in $\mathbf{u}$.*

Proof. For a proof see, for example, [9] or [11].

Let a continuously differentiable function F and an interval vector $\mathbf{u}$ be given. The midpoint of $\mathbf{u}$ is defined as $\mathrm{mid}(\mathbf{u}) = (\underline{\mathbf{u}} + \overline{\mathbf{u}})/2$. A verification algorithm for testing the existence or nonexistence of a solution of $F(u) = 0$ in $\mathbf{u}$ is based on a sequence of iterates of the form:

$$\begin{aligned} \mathbf{u}^0 &= \mathbf{u}, \\ \tilde{u}^k &= \mathrm{mid}(\mathbf{u}^k), \\ \mathbf{u}^{k+1} &= K(\mathbf{u}^k, \tilde{u}^k) \cap \mathbf{u}^k, \quad k = 0, 1, 2, \ldots \end{aligned}$$

Our numerical experiences show that during the first iteration it can often be decided whether there is a zero of F in $\mathbf{u}$, if $\mathbf{u}$ is sufficiently small. Sharper interval bounds may

be obtained by continuing the iteration. Moreover, an important and powerful result is that we do not need to solve a system of linear equations to find an enclosure of zeroes of a system of nonlinear equations. More details and further information on this algorithm can be found in [9], where the method was first introduced, and in [11], where the power of the method as a computational fixed point theorem was first analyzed.

Furthermore, the following properties of this algorithm are remarkable, see also [8,12], where its theory and practice have been expounded. Here it is assumed that $\mathrm{mid}\big(J_F(\mathbf{u}^k)\big)$ is nonsingular.

1. Every zero of F in $\mathbf{u}$ can be computed and correctly bounded.
2. If there is no zero of F in $\mathbf{u}$, the algorithm will automatically prove this fact in a finite number of iterations.
3. The proof of existence or nonexistence of a zero of F in a given interval vector $\mathbf{u}$ can be delegated to the computer. If $K(\mathbf{u}, \tilde{u}) \overset{\circ}{\subset} \mathbf{u}$, then there exists a unique zero of F in $\mathbf{u}$. If $K(\mathbf{u}, \tilde{u}) \cap \mathbf{u} = \emptyset$, then there is no zero of F in $\mathbf{u}$. Here, the relation $\overset{\circ}{\subset}$ denotes inner inclusion, i.e., $K(\mathbf{u}, \tilde{u}) \overset{\circ}{\subset} \mathbf{u}$ states that the interval vector $K(\mathbf{u}, \tilde{u})$ is contained in the interior of $\mathbf{u}$.
4. In practice, the algorithm converges quadratically when $\tilde{u}$ is taken to be the midpoint vector of $\mathbf{u}$, R is an appropriate approximation to the midpoint inverse, the interval extensions are sufficiently sharp, and the widths of the components of $\mathbf{u}$ are sufficiently small.

Now, the general algorithm for verification of approximate solutions of linear programming problems can be stated as follows:

ALGORITHM (General verification algorithm)

Step 0. Let a primal problem (P) and its dual (D) be given.

Step 1. Compute an approximate solution $\tilde{u} = (\tilde{x}, \tilde{y}, \tilde{z})$ for the primal problem (P) and its dual (D) by running an infeasible interior-point algorithm.

Step 2. Construct a small box $\mathbf{u} = (\mathbf{x}, \mathbf{y}, \mathbf{z})$ around the approximate solution by using epsilon-inflation [13]

$$\mathbf{u} = \tilde{u} + \epsilon\,[-e, e], \tag{3.4}$$

where ϵ is a given small positive machine number and e is the $(2n + m)$-vector of all ones.

Step 3. Perform a verification step:

Step 3.1. Prove the existence or nonexistence of a zero by applying Krawczyk's method to the nonlinear system $F(u) = 0$.

Step 3.2. Enclose the optimal solution value by the interval

$$c^\top \mathbf{x} \cap b^\top \mathbf{y}. \tag{3.5}$$

We conclude with some practical remarks on the algorithm. If the optimality conditions (2.1) hold and the existence of an optimal feasible solution in the constructed box is proved, then bounds on the exact optimal objective value are provided by (3.5). However, the approximate solution $\tilde{u}$ should be near an exact solution. Thus, if the tolerance of the stopping criterion in the approximate algorithm is set smaller than ϵ (3.4), then

the verification step will succeed. Finally, if the infeasible interior-point algorithm fails to compute a solution to the primal-dual pair, then the verification algorithm requires an initial interval vector approximation to be prescribed by the user. It then manages to contract this initial interval vector.

4 Numerical Results

In this section, we evaluate the practical efficiency of the presented algorithm. To demonstrate the effects of our verification algorithm, we have compared it against the software package LOQO [15] which is a linear programming solver based on an infeasible primal-dual interior-point algorithm. The computations are performed on a personal computer using FORTRAN–XSC [16]. In all our examples the quantity ϵ in (3.4) is set to 0.25. For more details on the efficient use of epsilon–inflation in verification methods, refer to [13].

Several software tools have been developed over the last decades to improve the accuracy of computer arithmetic. They include the library FORTRAN–XSC for interval arithmetic which provides reliable arithmetic by producing two values for each result. These two values correspond to the endpoints of an interval such that the exact result is guaranteed to be contained within it. Moreover, FORTRAN–XSC provides special notations and data types for interval arithmetic and an exact dot product which is particularly suited to the design of algorithms delivering automatically verified results with high accuracy. For more details, see [16], where the concepts and the important features are described.

Example 1. Consider the following linear programming problem

$$\begin{array}{lrl} \text{Minimize } -50x_1 - 9x_2 + 10^{-40}x_3 & & \\ \text{subject to} & x_1 & \leq 50 \\ & x_2 & \leq 200 \\ & 100x_1 + 18x_2 & \leq 5000 \\ & x_1, x_2 & \geq 0. \end{array}$$

This example is taken from [4], page 221, where the exact optimal value enclosure,

$$[-2.500000000000000 \cdot 10^3, -2.500000000000000 \cdot 10^3],$$

is computed by using the verified simplex method from [6]. The results achieved by running the infeasible interior-point algorithm [15] for this problem show that the primal problem and its dual are infeasible. Our verification algorithm proves that this problem has a solution. The optimal value is enclosed in the interval

$$[-2.5000000000000019 \cdot 10^3, -2.4999999999999981 \cdot 10^3].$$

A comparison between the results obtained for this test problem indicates that the approximate result from LOQO is quantitatively, but also qualitatively incorrect.

Example 2. Consider the following linear programming problem given in the form (P) with

$$A = \begin{pmatrix} -2 & -1 & -2 \\ -1 & -2 & -2 \\ -1 & -3 & -1 \end{pmatrix}, \quad b = \begin{pmatrix} -3 \\ -1 \\ -3 \end{pmatrix}, \quad \text{and} \quad c = \begin{pmatrix} 2 \\ 5 \\ 6 \end{pmatrix}.$$

This linear programming problem has no feasible solution, and this fact is proved by our verification algorithm. LOQO computes after 9 iterations an approximate optimal solution of this problem

$$= \begin{pmatrix} 2.6 \\ 0.6 \\ -1.4 \end{pmatrix}$$

with the optimal objective values,

$$\begin{aligned} \text{primal objective} &= -0.2000000028 \\ \text{dual objective} &= -0.1999999998\,. \end{aligned}$$

The infeasibility of this solution is not recognized by this algorithm. By constructing a small box $\mathbf{u}$ around the obtained approximate solution using epsilon-inflation and performing a verification step, we obtain that

$$K(\mathbf{u}, \tilde{u}) \cap \mathbf{u} = \emptyset.$$

Hence, the linear programming problem has no solution in $\mathbf{u}$ in spite of the fact that an approximate optimal solution with optimal objective values has been computed by LOQO. Therefore, our verification method provides a safety valve for the infeasible interior-point algorithm.

Example 3. Consider the following linear programming problem

$$\begin{aligned} &\text{Minimize} \quad 2x_1 + x_2 \\ &\text{subject to} \quad -x_1 - x_2 = 10^{-6}, \\ &\qquad\qquad x_1, x_2 \geq 0. \end{aligned}$$

When our algorithm attempts to verify the existence of a solution for this problem, it affirms that the problem is infeasible. In contrast, LOQO terminated after 254 iterations with the result that the primal problem and its dual are infeasible. This result from LOQO typically specifies that a maximum number of iterations is performed and no optimal solution has been computed.

5 Conclusions

We conclude with some observations and propose directions for future research.

The main goal was to develop an effective verification method for solutions of linear programming problems which computes enclosures for the solutions also in the presence of roundoff errors.

In combination with the fundamental properties of interval arithmetic, other tools such as Brouwer's fixed point theorem are used to automatically compute rigorous bounds on exact solutions.

The infeasible interior-point method and the stopping criteria have been realized in floating point arithmetic. The verification algorithm has been implemented using FORTRAN–XSC. The potential of this approach has been documented by comparing the results for three linear programming problems. Our results for many other larger problems up to a hundred variables indicate that the presented algorithm is robust [5]. A reliable result was obtained for all test problems.

We outline some directions for further research. It is essential for the infeasible interior-point algorithm to have a good starting point. The choice of starting point has a strong influence on the number of iterations, and a bad choice may lead to divergence or cycling so that a solution can never be reached. The best choice for a default starting point is still very much an open question. This problem concerns not only the verification algorithm but is more generally a problem of any infeasible interior-point algorithm. Further research for algorithmic improvements should be aimed at accelerating the verification process. In particular, the computation of an approximation to the midpoint inverse usually requires a lot of runtime. Finally, although interior-point methods have been extended to more general classes of problems, verification algorithms for these have not yet appeared.

Acknowledgment

The author would like to thank Jürgen Garloff and Andrew P. Smith for the careful reading of the manuscript and numerous suggestions for improvements in the presentation. Support from Damascus University, the Ministry of High Education of Syria, and the Ministry of Education and Research of Germany under contract no. 1705803 is gratefully acknowledged.

References

1. G. E. ALEFELD and J. HERZBERGER, Introduction to Interval Computations, Academic Press, New York and London, 1983.
2. R. K. FRISCH, The logarithmic potential method of convex programming, Memorandum, University Institute of Economics, Oslo, 1955.
3. A. V. FIACCO and G. P. MCCORMICK, Nonlinear Programming: Sequential Unconstraint Minimization Techniques. SIAM Classics in Applied Mathematics, vol. 4 (1990), Philadelphia.
4. R. HAMMER, M. HOCKS, U. KULISCH and D. RATZ, Numerical Toolbox for Verified Computing I, Springer-Verlag, Berlin, 1993.
5. I. I. IDRISS, Verified Linear Programming in Control System Analysis and Design, Dresden University of Technology, Dissertation, 2001.

6. C. JANSSON, Zur Linearen Optimierung mit unscharfen Daten. Dissertation, Universität Kaiserslautern, 1985.
7. N. K. KARMARKAR, A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica 4* (1984) 373-395.
8. R. B. KEARFOTT, Rigorous Global Search: Continuous Problems, Kluwer Academic Publishers, Dordrecht, Netherlands, 1996.
9. R. KRAWCZYK, Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken, *Computing 4* (1969) 187–201.
10. S. MEHROTRA, On the Implementation of a Primal-Dual Interior Point Method, *SIAM Journal on Optimization 2:4* (1992) 575–601.
11. R. E. MOORE, A Test for Existence of Solutions to Nonlinear Systems, *SIAM Journal on Numerical Analysis 14* (1977) 611-615.
12. A. NEUMAIER, Interval Methods for Systems of Equations, Cambridge University Press, London, 1990.
13. S. M. RUMP, Verification Methods for Dense and Sparse Systems of Equations, Topics in Validated Computations — Studies in Computational Mathematics, J. Herzberger , editor, Elsevier Amsterdam, (1994) 63–136.
14. M. J. TODD, Detecting Infeasibility in Infeasible-Interior-Point Methods for Optimization, in: *Foundations of Computational Mathematics*, F. Cucker, R. DeVore, P. Olver and E. Suli, (editors) Cambridge University Press (2004) 157–192.
15. R. J. VANDERBEI, LOQO User's Manual, Version 4.05, Technical Report NO. ORFE-99-307, Department of Operation Research and Financial Engineering, Princeton University, 2000.
16. W. V. WALTER, FORTRAN-XSC: A Portable Fortran 90 Module Library for Accurate and Reliable Scientific Computing, in Validation Numerics — Theory and Application, R. Albrecht, G. Alefeld and H. J. Stetter (eds.), Computing Supplementum 9 (1993) 265–285.
17. S. J. WRIGHT, Primal-Dual Interior-Point Methods, SIAM Publication, Philadelphia, 1997.