Noname manuscript No.
(will be inserted by the editor)

A Review of Computation of Mathematically Rigorous Bounds on Optima of Linear Programs

Jared T. Guilbeau · Md. Istiaq Hossain · Sam D. Karhbet · Ralph Baker Kearfott · Temitope S. Sanusi · Lihong Zhao

Received: date / Accepted: date

Abstract Linear program (LP) solvers sometimes fail to find a good approximation to the optimum value, without indicating possible failure. However, it may be important to know how close the value such solvers return is to an actual optimum, or even to obtain mathematically rigorous bounds on the optimum. In a seminal 2004 paper, Neumaier and Shcherbina, propose a method by which such rigorous lower bounds can be computed; we now have significant experience with this method. Here, we review the technique. We point out typographical errors in two formulas in the original publication, and illustrate their impact.

Separately, implementers and practitioners can also easily make errors. To help implementers avoid such problems, we cite a technical report where we explain things to mind and where we present rigorous bounds corresponding to alternative formulations of the linear program.

1 Introduction

Linear program solvers play an important role throughout scientific computation, operations research, and commercial computation. Although their quality has steadily improved, they still may fail or even return non-optimal solutions, without warning. Additionally, in various contexts, such as in branch-and-bound or branch-and-cut algorithms for continuous global optimization or mixed integer linear programs, it is important to have mathematically rigorous lower bounds on the optimum. In [25], Neumaier et al propose a simple idea to obtain a mathematically rigorous lower bound on the optimum of a linear program (posed as a

J. T. Guilbeau: E-mail: jtg7268@louisiana.edu
Md. I. Hossain: E-mail: mhh9786@louisiana.edu
S. D. Karhbet: E-mail: sdk6173@louisiana.edu
R. B. Kearfott: E-mail: rbk@louisiana.edu
T. S. Sanusi: E-mail: tss9743@louisiana.edu
Lihong Zhao: E-mail: lxz6134@louisiana.edu
University of Louisiana
P.O. Box 4-1010
Lafayette, Louisiana 70504-1010
USA

minimization problem), given an approximation to the dual variables. This lower bound becomes more accurate, the more accurate the dual variables are.

The paper [25] provides valuable observations and developments, enabling mathematical rigor in constrained global optimization. However, there are minor errors in a couple of the formulas used to compensate for roundoff error. These errors are not apparent if the incorrect formulas are implemented, since the upper and lower bounds of intervals computed with these erroneous formulas will be correct to within roundoff error. However, within branch and bound algorithms, use of the formulas with the errors will lead to incorrect conclusions or contradictions. In fact, we discovered the errors after observing such contradictory behavior within our Globsol package [20], eliminating alternative sources for the error after extensive study.

To illustrate the difficulty of finding this error, we observe that [25] has been extensively cited, in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35], etc., and these works themselves have significant citations, but the error has not been reported. These numerous references both illustrate the importance of the basic ideas in [25] and the ease with which the errors we have discovered can be overlooked. In all probability, these papers have merit, despite their possible use of the erroneous formulas, although readers of these papers may want to double-check purported mathematically rigorous conclusions based on use of the formulas in [25] in a branch-and-bound algorithm.

Here, we review the ideas and arguments in [25], consider pitfalls, report errors, and cite a technical report containing related formulas that are convenient to use in the context of various LP software packages.

2 Errors and Their Impact

Formulas for safe lower bounds on the optimum were presented in [25] for two formulations. We first point out errors that appeared in publication in those formulas, then explain the desirability of utilizing the basic ideas behind these formulas to present variations.

2.1 The Errors

The first formulation is

$$\begin{aligned}
&\text{Primal:} \begin{cases} \text{minimize} & c^T x, & c \text{ and } x \in \mathbb{R}^n \\ \text{subject to} & A_{\mathrm{e}} x = b_{\mathrm{e}}, & A_{\mathrm{e}} \in \mathbb{R}^{m_{\mathrm{e}} \times n},\ b_{\mathrm{e}} \in \mathbb{R}^{m_{\mathrm{e}}}, \\ & 0 \le x. & \end{cases} \\
&\text{Dual:} \quad \begin{cases} \text{maximize} & b_{\mathrm{e}}^T y, \\ \text{subject to} & A_{\mathrm{e}}^T y \le c. \end{cases}
\end{aligned} \tag{1}$$

In [25], rigorous upper bounds $x \le \overline{x}$ are assumed (even though such bound constraints are not assumed to be a part of the problem). It is then assumed that an arbitrary approximation λ to the optimal value of the dual variables y has been computed, and an upper bound r, $r \geq A_e^T y - c$ on the dual infeasibility has been computed. A line of simple algebra then gives

$$c^T x \geq y^T b - \max\{r, 0\}^T \overline{x}, \tag{2}$$

where the lower bound in (2) can be computed using either interval arithmetic or by careful directed rounding[1]. In any case, without caution, use of the implicit upper bounds $\overline{x}$ without including them as bound constraints in the optimization problem can easily lead to an incorrect conclusion; see Section 2.3 and [16].

The second form in [25] is

$$\begin{array}{ll}
\text{Primal:} \left\{ \begin{array}{lll} \text{minimize} & c^T x, & c \text{ and } x \in \mathbb{R}^n \\ \text{subject to} & \underline{b} \leq Ax \leq \overline{b}, & A \in \mathbb{R}^{m\times n},\ \underline{b}, \overline{b} \in \mathbb{R}^m. \end{array} \right. \\
\\
\text{Dual:} \quad \left\{ \begin{array}{ll} \text{maximize} & \underline{b}^T y - \overline{b}^T z, \\ \text{subject to} & A^T(y - z) = c, \\ & y \geq 0 \quad \text{and} \quad z \geq 0. \end{array} \right.
\end{array} \tag{3}$$

In the formula in [25] for the rigorous lower bound for (3), rigorous two-sided bounds $\underline{x} \leq x \leq \overline{x}$ are assumed, even though such bound constraints do not occur explicitly in the primal. As with the case of the one-sided implicit bounds for (1), this can lead to an incorrect lower bound, as we show in §2.3.

In [25], there is an inconsistency in whether $\lambda = y - z$ or $\lambda = z - y$. This manifests itself as two small errors in the lower bound formulas in [25] that, if implemented as is, would render the lower bounds computed with them incorrect. First, in the line above [25, Formula (8)], an approximate multiplier $\lambda \approx z - y$ is assumed to be returned by an approximate solver. Interval bounds on the optimal objective value are then given in [25] as $c^T x \in \lambda^T \boldsymbol{b} - \boldsymbol{r}^T \boldsymbol{x}$, where $\boldsymbol{r} = [\underline{r}, \overline{r}]$ consists of interval bounds on the dual residual $A^T\lambda - c$, $\boldsymbol{x} = [\underline{x}, \overline{x}]$, and $\boldsymbol{b} = [\underline{b}, \overline{b}]$. However, if the solver returns approximate Lagrange multipliers $\lambda_{\underline{b}} \approx y$ and $\lambda_{\overline{b}} \approx z$, the correct λ in these formulas would be $\lambda \approx \lambda_{\underline{b}} - \lambda_{\overline{b}}$, not $\lambda_{\overline{b}} - \lambda_{\underline{b}}$; that is, it should be $\lambda \approx y - z$, not $\lambda \approx z - y$.

The second error in the computational scheme for the lower bound on the $c^T x$ objective in (3) over the feasible points is in the scheme for computing the mathematically rigorous lower bound. The computational scheme (11) in [25] is

[1] A scheme using directed rounding is given in [25, Equation (5)]. However, it may be simpler to use interval arithmetic, now more widely available than when [25] appeared.

given incorrectly in pseudo-code as

$$
\text{Incorrect scheme: } \left\{ \begin{array}{l} \texttt{rounddown}; \\ \mu = \lambda_+^T \underline{b}; \\ r = A^T\lambda + c; \\ \texttt{roundup}; \\ r = \max\{-r, A^T\lambda + c\}; \\ \mu = \lambda_-^T \overline{b} - \mu + r^T \max\{-\underline{x}, \overline{x}\}; \\ \mu = -\mu; \end{array} \right.
$$
$$
\text{Corrected scheme: } \left\{ \begin{array}{l} \texttt{rounddown}; \\ \mu = \lambda_+^T \underline{b}; \\ r = A^T\lambda - c; \\ \texttt{roundup}; \\ r = \max\{-r, A^T\lambda - c\}; \\ \mu = \lambda_-^T \overline{b} - \mu + r^T \max\{-\underline{x}, \overline{x}\}; \\ \mu = -\mu; \end{array} \right. \tag{4}
$$

(In (4), $\lambda_+ = \max\{\lambda, 0\}$ and $\lambda_- = \max\{-\lambda, 0\}$.) Note that the errors occured on lines 3 and 5 of the pseudo-code, where the incorrect $A^T\lambda + c$ should have been $A^T\lambda - c$.

In any case, within present computing environments, we recommend using the interval arithmetic computation of the lower bound μ on c^Tx directly, namely,

$$
\mu = \inf(\lambda^T \boldsymbol{b} - \boldsymbol{r}^T \boldsymbol{x}) \qquad \text{(Formula (10) in [25])}, \tag{5}
$$

since (5) is both simpler (and hence less error prone) and sharper, and since high-quality packages for interval arithmetic are now widely available in various environments.

Confusion over the algebraic signs of z, y, and λ can easily arise due to inconsistencies between how approximate dual variables are reported by software. For instance, if the primal problem is simply to minimize c^Tx subject to $A_\mathrm{e}x = b_\mathrm{e}$, dual variables λ are sometimes viewed (that is, defined) as solving the equation $A_\mathrm{e}^T\lambda = c$ and in other cases are defined as solving $c - A_\mathrm{e}^T\lambda = 0$.

2.2 Impact

To investigate the impact of the errors we describe in §2.1, we implemented the calculations in Matlab. We refer to these calculations as the *safe bounds algorithm*. One test problem, an unpublished problem written by Anthony Holmes as a simple example of minimizing risk in a stock portfolio, is given by

$$
\begin{array}{r}
\min \varphi(x) = 10x_1 + 3.5x_2 + 4x_3 + 3.2x_4 \\
\text{such that} \\
0 \le 100x_1 + 50x_2 + 80x_3 + 40x_4 \le 200000, \\
18000 \le 12x_1 + 4x_2 + 4.8x_3 + 4x_4 \le 36000, \\
0 \le 100x_1 \le 100000, \\
0 \le 50x_2 \le 100000, \\
0 \le 80x_3 \le 100000, \\
0 \le 40x_4 \le 100000.
\end{array}
$$

Matlab's linear solver `linprog` was used, with default settings, to calculate the dual variables. `linprog` returned an approximate optimum function value of $\varphi(x) \approx 14,666.67$ almost the exact optimal value of $\frac{44,000}{3}$, while the safe bounds algorithm offered a mathematically rigorous lower bound on that optimum of $\mu \approx -30,333.34$. Intuition suggested this was a poor bound on such a simple problem.

For the related problem in which we replace $10x_1 + 3.5x_2 + 4x_3 + 3.2x_4$ by $-(10x_1 + 3.5x_2 + 4x_3 + 3.2x_4)$, `linprog` gave an approximate optimum of $-18,000$, which is also the exact optimum. The safe bounds algorithm then offered a lower bound of $\mu \approx -3,022.57$, which is not compatible with the exact optimum.

The corrected version of the safe bounds algorithm was then implemented, and returned lower bounds of $\mu = 14,526.39$ $\mu = -18,962.81$ for the original and negative versions, respectively. These bounds agree with the exact optima described above.

Next, we quantify the difference between the correct and incorrect schemes. Begin with two observations: First, writing $\lambda \approx z - y$ instead of $\lambda \approx y - z$ means the incorrect scheme actually uses $-\lambda$. Any appearance of λ, λ_+, or λ_- can be replaced with $-\lambda$, λ_-, and λ_+, respectively.

Second,

$$\begin{aligned}\max\big\{-(A^T(-\lambda)+c), A^T(-\lambda)+c\big\} &= \max\big\{-\big[-(A^T(\lambda)-c), -(A^T(\lambda)-c)\big)\big]\big\} \\ &= \max\big\{(A^T(\lambda)-c), -(A^T(\lambda)-c)\big\}\end{aligned}$$

Thus, if u_i and u_c represent the bound returned by the incorrect and correct schemes, respectively, then the difference between the two, without accounting for rounding error, is given by

$$\begin{aligned}|\mu_i - \mu_c| = \big|&\big[-(\lambda_+^T\overline{b} - \lambda_-^T\underline{b} + \max\{-(A^T(-\lambda)+c), (A^T(-\lambda)+c)\}\big] - \\ &\big[-(\lambda_-^T\overline{b} - \lambda_+^T\underline{b} + \max\{-(A^T(\ \ \lambda)-c), (A^T(\ \ \lambda)-c)\}\big] \quad \big| \\ = |&-(\lambda_+^T\overline{b} - \lambda_-^T\overline{b} - \lambda_-^T\underline{b} + \lambda_+^T\underline{b})| \\ = |&-(\lambda_+^T - \lambda_-^T)(\overline{b} + \underline{b})| \\ = |&-\lambda^T(\overline{b} + \underline{b})|\end{aligned}$$

Calculating this term for the two problems above gives approximately 44,859.72 and 15,940.25, which agrees with the observed differences between the correct and incorrect schemes.

2.3 Pitfalls

Linear programs can in principle be formulated in various equivalent ways; we reviewed the two such ways given in [25] in Section 2.1 of this note. However, depending on the actual problem being solved, certain formulations can be misleading, or are more prone to errors when implemented. That is, although particular formulations are general and mathematically correct, naive or inattentive practitioners may incorrectly interpret the results of the computation or incorrectly reformulate the problem. To make it easier for such practitioners, we expand on the seminal work in [25] in a technical report [16], where we point out certain pitfalls and use the ideas in [25] to give analogues to (5) for various reformulations.

3 Summary

We have pointed out small, correctable, but hard-to-recognize errors in the formulas in [25]. These errors can have consequences in branch and bound algorithms meant to provide mathematically rigorous bounds, and can result in puzzling but difficult-to-pinpoint behavior in such algorithms.

We have also mentioned a separate issue, difficulties of implementing and using particular reformulations of linear programs for particular problems, when the goal is obtaining mathematically rigorous bounds on solutions. To mitigate these difficulties, we have cited our technical report [16], where we discuss pitfalls and present formulas for mathematically rigorous bounds when using alternative formulations.

Acknowledgment

We wish to thank all of the referees and editors for their patience and thought. Not only did they correct errors, but they formulated wise suggestions that made the purpose and exposition clearer.

References

1. Ernst Althaus and Daniel Dumitriu. Certifying feasibility and objective value of linear programs. *Operations Research Letters*, 40(4):292 – 297, 2012.
2. David L. Applegate, William Cook, Sanjeeb Dash, and Daniel G. Espinoza. Exact solutions to linear programming problems. *Operations Research Letters*, 35(6):693 – 699, 2007.
3. Charles Audet, Pierre Hansen, Frédéric Messine, and Jordan Ninin. The small octagons of maximal width. *Discrete and Computational Geometry*, 49(3):589–600, 2013.
4. Ali Baharev, Lubomir Kolev, and Endre Rév. Computing multiple steady states in homogeneous azeotropic and ideal two-product distillation. *AIChE Journal*, 57(6):1485–1495, 2011.
5. Ali Baharev and Endre Rév. Reliable computation of equilibrium cascades with affine arithmetic. *AIChE Journal*, 54(7):1782–1797, 2008.
6. Ralph Baker Kearfott. Discussion and empirical comparisons of linear relaxations and alternate techniques in validated deterministic global optimization. *Optimization Methods and Software*, 21:715–731, October 2006.
7. Frédéric Benhamou and Laurent Granvilliers. Chapter 16 - Continuous and interval constraints. In Peter van Beek Francesca Rossi and Toby Walsh, editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 571 – 603. Elsevier, 2006.
8. Maurizio Boccia, Antonio Sforza, Claudio Sterle, and Igor Vasilyev. A cut and branch approach for the capacitated p-median problem based on Fenchel cutting planes. *Journal of Mathematical Modelling and Algorithms*, 7(1):43–58, 2008.

9. Leonid Chindelevitch, Jason Trigg, Aviv Regev, and Bonnie Berger. An exact arithmetic toolbox for a consistent and reproducible structural analysis of metabolic network models. *Nature Communications*, 5:4893:1–10, October 2014.
10. William Cook, Sanjeeb Dash, Ricardo Fukasawa, and Marcos Goycoolea. Numerically safe gomory mixed-integer cuts. *INFORMS Journal on Computing*, 21(4):641–649, 2009.
11. William Cook, Thorsten Koch, Daniel E. Steffy, and Kati Wolter. A hybrid branch-and-bound approach for exact rational mixed-integer programming. *Mathematical Programming Computation*, 5(3):305–344, 2013.
12. Gérard Cornuéjols, François Margot, and Giacomo Nannicini. On the safety of Gomory cut generators. *Mathematical Programming Computation*, 5(4):345–395, 2013.
13. Ferenc Domes and Arnold Neumaier. Rigorous filtering using linear relaxations. *Journal of Global Optimization*, 53(3):441–473, 2012.
14. Ricardo Fukasawa. Gomory cuts. In James J. Cochran, Louis A. Cox, Pinar Keskinocak, Jeffrey P. Kharoufeh, and J. Cole Smith, editors, *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley and Sons, Inc., 2011.
15. M. Gouttefarde, D. Daney, and J.-P. Merlet. Interval-analysis-based determination of the wrench-feasible workspace of parallel cable-driven robots. *IEEE Transactions on Robotics*, 27(1):1–13, Feb 2011.
16. Jared T. Guilbeau, Md. Istiaq Hossain, Sam D. Karhbet, Ralph Baker Kearfott, Temitope S. Sanusi, and Lihong Zhao. Advice for mathematically rigorous bounds on optima of linear programs. Technical report, University of Louisiana at Lafayette, 1964. `http://interval.louisiana.edu/preprints/reformulations-for-rigorous.pdf`.
17. C. Jansson. Rigorous lower and upper bounds in linear programming. *SIAM Journal on Optimization*, 14(3):914–935, 2004.
18. C. Jansson, D. Chaykin, and C. Keil. Rigorous error bounds for the optimal value in semidefinite programming. *SIAM Journal on Numerical Analysis*, 46(1):180–200, 2008.
19. Christian Jansson. On verified numerical computations in convex programming. *Japan Journal of Industrial and Applied Mathematics*, 26(2-3):337–363, 2009.
20. Ralph Baker Kearfott. GlobSol user guide. *Optimization Methods and Software*, 24(4-5):687–708, August 2009.
21. Ralph Baker Kearfott, Jessie Castille, and Gaurav Tyagi. A general framework for convexity analysis in deterministic global optimization. *Journal of Global Optimization*, 56(3):765–785, 2013.
22. Y. Lebbah, C. Michel, M. Rueher, D. Daney, and J. Merlet. Efficient and safe global constraints for handling numerical constraint systems. *SIAM Journal on Numerical Analysis*, 42(5):2076–2097, 2005.
23. Yahia Lebbah, Claude Michel, and Michel Rueher. A rigorous global filtering algorithm for quadratic constraints. *Constraints*, 10(1):47–65, 2005.

24. François Margot. Testing cut generators for mixed-integer linear programming. *Mathematical Programming Computation*, 1(1):69–95, 2009.
25. Arnold Neumaier and Oleg Shcherbina. Safe bounds in linear and mixed-integer programming. *Math. Prog.*, 99(2):283–296, March 2004.
26. Arnold Neumaier, Oleg Shcherbina, Waltraud Huyer, and Tamás Vinkó. A comparison of complete global optimization solvers. *Math. Prog.*, 103(2):335–356, 2005.
27. Jordan Ninin, Frédéric Messine, and Pierre Hansen. A reliable affine relaxation method for global optimization. *4OR*, 13(3):247–277, 2015.
28. J.M. Porta, L. Ros, and F. Thomas. A linear relaxation technique for the position analysis of multiloop linkages. *IEEE Transactions on Robotics*, 25(2):225–239, April 2009.
29. Ionela Prodan and Enrico Zio. A model predictive control framework for reliable microgrid energy management. *International Journal of Electrical Power and Energy Systems*, 61(0):399 – 409, 2014.
30. S. Ratschan and Z. She. Providing a basin of attraction to a target region of polynomial systems by computation of Lyapunov-like functions. *SIAM Journal on Control and Optimization*, 48(7):4377–4394, 2010.
31. Andrew Paul Smith. Fast construction of constant bound functions for sparse polynomials. *Journal of Global Optimization*, 43(2-3):445–458, 2009.
32. Ronald R. van Nooijen and Alla Kolechkina. Speed of discrete optimization solvers for real time sewer control. *Urban Water Journal*, 10(5):354–363, 2013.
33. Xuan-Ha Vu, Djamila Sam-Haroud, and Boi Faltings. Enhancing numerical constraint propagation using multiple inclusion representations. *Annals of Mathematics and Artificial Intelligence*, 55(3-4):295–354, 2009.
34. Yu Xiang and Tian Lan. *Smart Pricing of Cloud Resources*, pages 453–476. John Wiley and Sons, Inc., 2014.
35. Yi Xu, Shunze Wu, Hongkuan Zang, and Guiguang Hou. An interval joint-probabilistic programming method for solid waste management: a case study for the city of Tianjin, China. *Frontiers of Environmental Science and Engineering*, 8(2):239–255, 2014.