STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–16.
Published online in International Academic Press (www.IAPress.org)

A Parametric Hybrid-Direction Algorithm for Bounded-Variable Linear Programming

Khalil Djeloud*

Department of Mathematics-Higher Normal School of Laghouat, Algeria

Abstract This paper proposes a parametric hybrid-direction algorithm for bounded-variable linear programming. The method is based on an odd integer parameter ν, which defines a family of search directions within a unified framework. In particular, when ν= -1, the proposed approach reduces to a special case corresponding to previously studied hybrid-direction methods [1, 2, 5, 7], whose effectiveness has been demonstrated in connection with the simplex algorithm of the open-source solver GLPK [15]. Computational experiments on randomly generated problems show that the algorithm is efficient and robust, with stable performance in terms of iteration counts and CPU time. Among the tested values ν=1 provides the best overall results, offering a good compromise between convergence speed and computational cost.

Keywords Linear Programming, Adapted Method, Feasible Solution, Hybrid Direction, Optimality Estimation, long Step Rule, Numerical Experiments.

AMS 2010 subject classifications 90C05, 65K05

DOI: 10.19139/soic-2310-5070-3554

1. Introduction

Linear programming (LP) is a central field in applied mathematics, with wide-ranging applications in industry, finance, planning, scheduling, and optimal resource allocation. Over the past decades, numerous solution methods have been developed to address LP problems efficiently. Among the most prominent are the simplex method [8, 15], which remains a cornerstone in practical optimization, and interior-point methods [11, 12, 15], known for their strong theoretical complexity and effectiveness on large-scale problems. In addition, support and adapted direction methods [9, 10] have been proposed to enhance the search process by exploiting problem structure.

More recently, hybrid-direction approaches have emerged as an effective class of methods for bounded-variable linear programming [1, 2, 5, 6, 7]. These methods construct the search direction by combining components of adapted directions with terms depending on reduced costs, thereby achieving improved flexibility and computational performance. The success of these approaches demonstrates the benefit of integrating multiple directional strategies within a unified iterative framework.

Despite these advances, existing hybrid-direction methods remain largely problem-specific and lack a unified parametric formulation that systematically captures and generalizes their underlying principles. To the best of our knowledge, no comprehensive framework currently exists that allows these methods to be viewed as instances of a broader parametric family.

In this work, we address this limitation by introducing a parametric hybrid-direction scheme based on an odd integer parameter ν. The proposed approach generates a family of algorithms, denoted HDA-(ν), in which the parameter ν directly controls the structure of the search direction. This framework not only recovers several existing

*Correspondence to: Khalil Djeloud (Email: k.djeloud@ens-lagh.dz). Department of Mathematics-Higher Normal School of Laghouat, Box 4033 Station post, avenue of Martyrs, 03000; Laghouat, Algeria.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

methods as special cases but also enables the construction of new variants with potentially improved convergence properties for bounded-variable linear programming problems.

The remainder of this paper is organized as follows. Section 2 formulates the bounded-variable linear programming problem and introduces the notation and fundamental concepts used throughout the paper. Section 3 develops the proposed parametric hybrid-direction scheme, including the definition of the search direction, the description of the iterative procedure, and the main theoretical properties such as feasibility, optimality and suboptimality conditions, and update rules. Section 4 presents the proposed algorithm in a structured form. Section 5 provides a numerical example to illustrate the proposed approach. Finally, Section 6 reports the numerical results, including test problem generation, performance evaluation in terms of iterations and CPU time, and a comparative analysis for different values of ν.

2. Problem Formulation and Preliminaries

We consider the following linear programming problem with bounded variables in standard form:

$$\begin{aligned} &\max\ z = c^T x \\ &\text{s.t. } x \in \Gamma = \{\, t \in \mathbb{R}^n : At = b,\ l \le t \le u\}, \end{aligned} \tag{1}$$

where c, l and $u \in \mathbb{R}^n$ are finite vectors, $b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m\times n}$ with $\operatorname{rank}(A) = m < n$.

Basic Definitions

1. Any vector $x \in \Gamma$ is called a *Feasible Solution* (FS) of problem (1).
2. A FS x^0 is *optimal* if $z(x^0) \ge z(x), \quad \forall x \in \Gamma$.
3. A solution x^ϵ is called ϵ-*optimal* if $z(x^0) - z(x^\epsilon) \le \epsilon$, where x^0 is an optimal solution and $\epsilon \ge 0$.

Index Sets and Decomposition

Let $I = \{1, 2, \ldots, m\},\ J = \{1, 2, \ldots, n\}$, and partition J as $J = J_B \cup J_N,\ J_B \cap J_N = \emptyset,\ |J_B| = m$.

Accordingly, vectors c, x, l, u and matrix A are partitioned as

$$v = (v_B, v_N), \quad v_\varsigma = (v_j,\ j \in J_\varsigma),$$

$$A = (A_B, A_N), \quad A_\varsigma = (a_{ij},\ i \in I, j \in J_\varsigma).$$

Support, Feasible Solutions, and Optimality Conditions

1. The set J_B is called a *support* if $\det(A_B) \ne 0$.
2. The pair $\{x, J_B\}$, where x is a FS and J_B is a support, is called a *Support Feasible Solution* (SFS).
3. An SFS $\{x, J_B\}$ is *non-degenerate* if $l_j < x_j < u_j,\ \forall j \in J_B$.
4. The vector of reduced costs is defined by $\Delta^T = \pi^T A - c^T,$ where $\pi^T = c_B^T A_B^{-1}$.
5. The quantity
$$\beta = \beta(x, J_B) = \sum_{j\in J_N,\ \Delta_j>0} \Delta_j(x_j - l_j) + \sum_{j\in J_N,\ \Delta_j<0} \Delta_j(x_j - u_j) \tag{2}$$
is called the *suboptimality estimate* [9].

The following results are taken from [9].

Theorem 1
Let $\{x, J_B\}$ be an SFS of problem (1). The conditions

$$\begin{aligned} \Delta_j &\geq 0 \quad \text{if } x_j = l_j, \\ \Delta_j &\leq 0 \quad \text{if } x_j = u_j, \\ \Delta_j &= 0 \quad \text{if } l_j \leq x_j \leq u_j, \quad j \in J_N, \end{aligned}$$

are sufficient for optimality of x. They are also necessary when the SFS is non-degenerate.

Theorem 2
Let $\{x, J_B\}$ be an SFS of problem (1) and let $\epsilon \geq 0$. If $\beta \leq \epsilon$, then x is ϵ-optimal.

2.1. The Dual Problem

The dual problem associated with (1) is given by [3, 5, 13]:

$$\min \; L(\lambda) = b^T y - l^T v + u^T w, \tag{3}$$

$$A^T y - c - v + w = 0, \tag{4}$$

$$v \geq 0, \; w \geq 0, \; y \in \mathbb{R}^m. \tag{5}$$

1. A vector $\lambda = (y, v, w) \in \mathbb{R}^{m+2n}$ is called a *Dual Feasible Solution* (DFS) if it satisfies (4)–(5).
2. A DFS λ^0 is *optimal* if $L(\lambda^0) \leq L(\lambda)$, for all DFS λ.
3. The pair $\{\lambda, J_B\}$ where λ is a DFS and J_B is a support is called a *Support Dual Feasible Solution* (SDFS).
4. Let $\{x, J_B\}$ be an SFS of (1). Define

$$y = \pi, \quad v_j = \max(0, \Delta_j), \quad w_j = \max(0, -\Delta_j), \quad j \in J. \tag{6}$$

Then $\lambda = (y, v, w)$ is a DFS, and

$$\begin{aligned} \beta &= \sum_{\Delta_j > 0} \Delta_j (x_j - l_j) + \sum_{\Delta_j < 0} \Delta_j (x_j - u_j) \\ &= \Delta^T x - l^T v + u^T w \\ &= (\pi^T A - c^T) x - l^T v + u^T w \\ &= b^T \pi - l^T v + u^T w - c^T x \\ &= L(\lambda) - z(x). \end{aligned}$$

3. An Iteration of the Hybrid-Direction Algorithm

Let $\{x, J_B\}$ be an SFS of problem (1), and let $\eta > 0$ and $\nu \in 2\mathbb{Z} + 1$[†]. We partition the index set J_N into the following subsets :

$$\begin{aligned} J_0 &= \{\, j \in J_N \;:\; \Delta_j = 0 \,\}, \\ J_1 &= \{\, j \in J_N \;:\; 0 < \Delta_j \leq \eta (x_j - l_j)^{-\nu} \,\}, \\ J_2 &= \{\, j \in J_N \;:\; \eta (x_j - u_j)^{-\nu} \leq \Delta_j < 0 \,\}, \\ J_3 &= \{\, j \in J_N \;:\; \Delta_j > \eta (x_j - l_j)^{-\nu}, \; x_j > l_j \,\}, \\ J_4 &= \{\, j \in J_N \;:\; \Delta_j < \eta (x_j - u_j)^{-\nu}, \; x_j < u_j \,\}, \\ J_5 &= \{\, j \in J_N \;:\; \Delta_j > 0, \; x_j = l_j \,\}, \\ J_6 &= \{\, j \in J_N \;:\; \Delta_j < 0, \; x_j = u_j \,\}. \end{aligned} \tag{7}$$

[†]The set of odd integers.

These sets form a partition of J_N, i.e., $J_N = \bigcup_{i=0}^{6} J_i, \quad J_i \cap J_j = \emptyset \ \text{ for } i \neq j.$

Lemma 1
Let $\{x, J_B\}$ be a SFS of problem (1). Then the following assertions hold:

$$\begin{aligned} &\text{if } j \in J_1 \cup J_3, \text{ then } x_j > l_j \text{ and } \Delta_j > 0, \\ &\text{if } j \in J_2 \cup J_4, \text{ then } x_j < u_j \text{ and } \Delta_j < 0. \end{aligned}$$

Lemma 2
Let $\{x, J_B\}$ be an SFS of problem (1). If $J_N = J_0 \cup J_5 \cup J_6$, then x is an optimal solution of problem (1).

We now define the *optimality estimate* associated with the SFS $\{x, J_B\}$:

$$\gamma_\nu^\eta = \gamma_\nu^\eta(x, J_B) = \sum_{j \in J_3 \cup J_4} \Delta_j \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} + \sum_{j \in J_1} \Delta_j (x_j - l_j) + \sum_{j \in J_2} \Delta_j (x_j - u_j). \tag{8}$$

Next, define

$$\begin{aligned} \mu_\nu^\eta = \mu_\nu^\eta(x, J_B) &= \operatorname{sign}(\nu) \left[\sum_{j \in J_3} \Delta_j \left(x_j - l_j - \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} \right) + \sum_{j \in J_4} \Delta_j \left(x_j - u_j - \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} \right) \right] \\ &= \operatorname{sign}(\nu) \big(\beta - \gamma_\nu^\eta\big). \end{aligned} \tag{9}$$

Remark 1
If $J_3 \cup J_4 = \emptyset$, then

$$\beta = \gamma_\nu^\eta \quad \text{and} \quad \mu_\nu^\eta = \emptyset.$$

Lemma 3
For all $\nu \in 2\mathbb{Z} + 1$ and $\eta > 0$, we have

$$\gamma_\nu^\eta \geq 0, \quad \mu_\nu^\eta \geq 0, \quad \text{and} \quad \beta = \gamma_\nu^\eta + \operatorname{sign}(\nu)\, \mu_\nu^\eta.$$

Proof
We examine each component.

1. For $j \in J_1$, $\Delta_j > 0$ and $x_j > l_j$, hence $\Delta_j(x_j - l_j) > 0$.

2. For $j \in J_2$, $\Delta_j < 0$ and $x_j < u_j$, thus $\Delta_j(x_j - u_j) > 0$.

3. For $j \in J_3$, $\Delta_j > 0$ and $\frac{\eta}{\Delta_j} < (x_j - l_j)^\nu$. Since ν is odd, it follows that

$$\Delta_j \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} > 0, \quad \operatorname{sign}(\nu)\, \Delta_j \left(x_j - l_j - \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} \right) > 0.$$

4. For $j \in J_4$, $\Delta_j < 0$ and $\frac{\eta}{\Delta_j} > (x_j - u_j)^\nu$, which implies

$$\Delta_j \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} > 0, \quad \operatorname{sign}(\nu)\, \Delta_j \left(x_j - u_j - \left(\frac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}} \right) > 0.$$

Conclusion. All terms are nonnegative, hence $\gamma_\nu^\eta \geq 0$ and $\mu_\nu^\eta \geq 0$. The identity follows from (9). □

Lemma 4
For $\eta > 0$ and $\nu \in 2\mathbb{Z} + 1$, the following equivalence holds:

$$\bigcup_{i=1}^{4} J_i = \emptyset \iff \gamma_\nu^\eta = 0.$$

Proof

Assume first that $\bigcup_{i=1}^{4} J_i = \emptyset$. Then $J_N = J_0 \cup J_5 \cup J_6$, and all the terms appearing in (8) are equal to zero. Hence $\gamma_\nu^\eta = 0$, and thus $\beta = 0$ by Remark 1.

Conversely, assume that $\gamma_\nu^\eta = 0$. By Lemma 3, all terms in the definition of γ_ν^η are nonnegative; hence, each of them must be equal to zero. It follows that no index belongs to $J_1, \ldots, J_4$, and therefore

$$\bigcup_{i=1}^{4} J_i = \emptyset.$$

□

3.1. Optimality criterion

Theorem 3
Let $\{x, J_B\}$ be an SFS of problem (1), with $\eta > 0$ and $\nu \in 2\mathbb{Z} + 1$. The condition

$$\bigcup_{i=1}^{4} J_i = \emptyset$$

is sufficient for the optimality of x, and it is also necessary when the SFS $\{x, J_B\}$ is non-degenerate.

Proof
Sufficiency. Let $\bar{x}$ be any FS of (1). Then

$$z(\bar{x}) - z(x) \leq \beta.$$

If $\bigcup_{i=1}^{4} J_i = \emptyset$, then by Lemma 4 we have $\gamma_\nu^\eta = 0$, and hence $\beta = 0$. Therefore,

$$z(\bar{x}) \leq z(x),$$

and x is optimal.

Necessity. Assume that x is a non-degenerate optimal FS. Then

$$z(\bar{x}) \leq z(x), \quad \forall \bar{x} \in \Gamma. \tag{10}$$

Suppose, by contradiction, that

$$\bigcup_{i=1}^{4} J_i \neq \emptyset.$$

By Lemma 4, this implies $\gamma_\nu^\eta > 0$.

Define $\bar{x} = x + \theta d$, where $\theta > 0$ and d is defined by

$$d_j = \begin{cases} l_j - x_j, & \text{if } j \in J_1, \\ u_j - x_j, & \text{if } j \in J_2, \\ -\dfrac{\eta}{\Delta_j}, & \text{if } j \in J_3 \cup J_4, \\ 0, & \text{if } j \in J_0 \cup J_5 \cup J_6, \end{cases} \qquad d_B = -A_B^{-1} A_N d_N.$$

Then $Ad = 0$, and hence $A\bar{x} = b$.
To ensure feasibility, $\bar{x}$ must satisfy

$$l - x \leq \theta d \leq u - x,$$

that is,

$$l_j - x_j \leq \theta d_j \leq u_j - x_j. \tag{11}$$

Since $\{x, J_B\}$ is non-degenerate, we have

$$l_j - x_j < 0 < u_j - x_j, \quad j \in J_B.$$

Moreover, by Lemma 1, for all $j \in \bigcup_{i=1}^{4} J_i$, we have

$$\begin{aligned} l_j - x_j < 0, \text{ if } \Delta_j > 0, \\ u_j - x_j > 0, \text{ if } \Delta_j < 0. \end{aligned}$$

Thus, for $\theta > 0$ sufficiently small, $\bar{x}$ is feasible.
Finally, using the increase formula, we obtain

$$\begin{aligned} z(\bar{x}) - z(x) &= \theta \left(\sum_{j \in J_3 \cup J_4} \eta + \sum_{j \in J_1} \Delta_j (x_j - l_j) + \sum_{j \in J_2} \Delta_j (x_j - u_j) \right) \\ &= \theta \gamma_\nu^\eta(x, J_B) > 0. \end{aligned}$$

This contradicts (10). Hence,

$$\bigcup_{i=1}^{4} J_i = \emptyset.$$

□

3.2. The New Feasible Solution

Let $\{x, J_B\}$ be an SFS of problem (1), with $\eta > 0$ and $\nu \in 2\mathbb{Z} + 1$. Assume that

$$\bigcup_{j=1}^{4} J_j \neq \emptyset.$$

We define a direction $d \in \mathbb{R}^n$ by

$$d_j = \begin{cases} l_j - x_j, & \text{if } j \in J_1, \\ u_j - x_j, & \text{if } j \in J_2, \\ -\left(\dfrac{\eta}{\Delta_j}\right)^{\frac{1}{\nu}}, & \text{if } j \in J_3 \cup J_4, \\ 0, & \text{if } j \in J_0 \cup J_5 \cup J_6, \end{cases} \qquad d_B = -A_B^{-1} A_N d_N. \tag{12}$$

It follows from the definition of d that $Ad = 0$.
To preserve the bound constraints at the next iterate, we define the maximum admissible step length as

$$\theta^0 = \min\{1, \theta_{j_1}, \theta_{j_2}\}, \quad \theta_{j_1} = \min_{j \in J_B} \theta_j, \quad \theta_{j_2} = \min_{j \in J_3 \cup J_4} \theta_j, \tag{13}$$

where, for each $j \in J$,

$$\theta_j = \begin{cases} \dfrac{u_j - x_j}{d_j}, & \text{if } d_j > 0, \\ \dfrac{l_j - x_j}{d_j}, & \text{if } d_j < 0, \\ +\infty, & \text{if } d_j = 0. \end{cases}$$

The new iterate is defined by

$$\bar{x} = x + \theta^0 d. \tag{14}$$

By the definition of θ^0, the step length is chosen to preserve all bound constraints; hence, the new iterate satisfies $\bar{x} \in \Gamma$.

We now show that d is an ascent direction. Indeed,

$$z(\bar{x}) - z(x) = -\theta^0 \sum_{j \in J_N} \Delta_j d_j \quad = \theta^0 \left(\sum_{j \in J_1} \Delta_j (x_j - l_j) + \sum_{j \in J_2} \Delta_j (x_j - u_j) + \sum_{j \in J_3 \cup J_4} \Delta_j \left(\frac{\eta}{\Delta_j} \right)^{\frac{1}{\nu}} \right) = \theta^0 \, \gamma_\nu^\eta(x, J_B).$$

Since $\theta^0 \geq 0$ and $\gamma_\nu^\eta(x, J_B) \geq 0$, it follows that $z(\bar{x}) \geq z(x)$.

Remark 2
If $\nu \geq 1$ or $J_3 \cup J_4 = \emptyset$, then $\theta_{j_2} = +\infty$.
Moreover, if $\nu \leq -1$ and $\theta^0 = 1$, then $\gamma_\nu^\eta = \beta$.

Lemma 5
The following update formula holds:

$$\bar{\beta} = \beta(\bar{x}, J_B) = \beta - \theta^0 \gamma_\nu^\eta \leq \beta.$$

Proof
Using $\bar{x} = x + \theta^0 d$, we have

$$\begin{aligned} \bar{\beta} &= \sum_{j \in J_3} \Delta_j (\bar{x}_j - l_j) + \sum_{j \in J_4} \Delta_j (\bar{x}_j - u_j) + \sum_{j \in J_1} \Delta_j (\bar{x}_j - l_j) + \sum_{j \in J_2} \Delta_j (\bar{x}_j - u_j) \\ &= (1 - \theta^0)\beta + \theta^0 \left[\sum_{j \in J_3} \Delta_j (x_j - l_j + d_j) + \sum_{j \in J_4} \Delta_j (x_j - u_j + d_j) \right]. \end{aligned}$$

By the definition of d_j for $j \in J_3 \cup J_4$,

$$d_j = -\left(\frac{\eta}{\Delta_j} \right)^{\frac{1}{\nu}},$$

hence

$$\begin{aligned} \bar{\beta} &= (1 - \theta^0)\beta + \theta^0 \left[\sum_{j \in J_3} \Delta_j \left(x_j - l_j - \left(\frac{\eta}{\Delta_j} \right)^{\frac{1}{\nu}} \right) + \sum_{j \in J_4} \Delta_j \left(x_j - u_j - \left(\frac{\eta}{\Delta_j} \right)^{\frac{1}{\nu}} \right) \right] \\ &= (1 - \theta^0)\beta + \theta^0 \, \mu_\nu^\eta = \beta - \theta^0 \gamma_\nu^\eta. \end{aligned}$$

Since $\theta^0 \geq 0$ and $\gamma_\nu^\eta \geq 0$, it follows that $\bar{\beta} \leq \beta$. □

Lemma 6
Let $\{x, J_B\}$ be a non-optimal SFS of problem (1). If $\theta^0 = \dfrac{\beta}{\gamma_\nu^\eta}$, then the solution $\bar{x}$ is optimal.

Proposition 1
The following results summarize the consequences of the previous analysis:

(i) From Remark 1, if $J_3 \cup J_4 = \emptyset$, then we obtain the update formula for the suboptimality estimate [9].

(ii) From Remark 2, if $\nu \leq -1$ and $\theta^0 = 1$, then the FS $\bar{x}$ is optimal.

(iii) From Lemma 5, for $\nu \geq 1$, if $\theta^0 = 1$ and $\mu_\nu^\eta \leq \varepsilon$, then the FS $\bar{x}$ is ε-optimal.

3.3. The New Support

We define the n-vector κ and the scalar α_0 as follows:

$$\kappa = x + d \quad \text{and} \quad \alpha_0 = \kappa_{j_1} - \bar{x}_{j_1},$$

where j_1 is the leaving index computed using (13).

The dual direction t is defined by:

$$t_{j_1} = -\text{sign}(\alpha_0), \quad t_j = 0 \text{ for } j \neq j_1,\ j \in J_B, \quad t_N^T = t_B^T A_B^{-1} A_N. \tag{15}$$

Next, we introduce the following index sets:

$$J_0^+ = \{j \in J_0 \ : \ t_j > 0\}, \quad J_0^- = \{j \in J_0 \ : \ t_j < 0\}. \tag{16}$$

We also define the scalar quantity:

$$\alpha = -|\alpha_0| + \sum_{j \in J_0^+ \cup J_3} t_j(\kappa_j - l_j) + \sum_{j \in J_0^- \cup J_4} t_j(\kappa_j - u_j). \tag{17}$$

The updated reduced cost vector and the new support are given by:

$$\bar{\Delta} = \Delta + \sigma^0 t, \quad \bar{J}_B = (J_B \setminus \{j_1\}) \cup \{j_0\},$$

where

$$\sigma^0 = \sigma_{j_0} = \min_{j \in J_N} \{\sigma_j\}, \tag{18}$$

and

$$\sigma_j = \begin{cases} \dfrac{-\Delta_j}{t_j}, & \text{if } \Delta_j t_j < 0, \\ 0, & \text{if } j \in J_0^- \text{ and } \kappa_j \neq u_j, \\ 0, & \text{if } j \in J_0^+ \text{ and } \kappa_j \neq l_j, \\ \infty, & \text{otherwise.} \end{cases} \tag{19}$$

Remark 3
If $\sigma^0 = \infty$, then problem (1) is infeasible. Hence, in the remainder, we assume that $\sigma^0 < \infty$.

The suboptimality estimate associated with the SFS $\{\bar{x}, \bar{J}_B\}$ is given by:

$$\bar{\bar{\beta}} = \beta(\bar{x}, \bar{J}_B) = \sum_{j \in \bar{J}_N \ \bar{\Delta}_j > 0} \bar{\Delta}_j(\bar{x}_j - l_j) + \sum_{j \in \bar{J}_N \ \bar{\Delta}_j < 0} \bar{\Delta}_j(\bar{x}_j - u_j).$$

Proposition 2
The suboptimality estimate $\bar{\bar{\beta}}$ satisfies

$$\bar{\bar{\beta}} = \bar{\beta} + \sigma^0 \alpha.$$

Proof
We partition the index set J according to the signs of $\bar{\Delta}_j$ and Δ_j as follows:

$$\begin{aligned} J^{++} &= \{j \in J : \bar{\Delta}_j \geq 0,\ \Delta_j > 0\}, & J^{-+} &= \{j \in J : \bar{\Delta}_j < 0,\ \Delta_j > 0\}, \\ J^{+-} &= \{j \in J : \bar{\Delta}_j \geq 0,\ \Delta_j < 0\}, & J^{--} &= \{j \in J : \bar{\Delta}_j < 0,\ \Delta_j < 0\}, \\ J^{+0} &= \{j \in J : \bar{\Delta}_j \geq 0,\ \Delta_j = 0\}, & J^{-0} &= \{j \in J : \bar{\Delta}_j < 0,\ \Delta_j = 0\}. \end{aligned}$$

Let $\lambda = (y, v, w)$ be a DFS, where the components are defined in (6). We define a new vector $\bar{\lambda} = (\bar{y}, \bar{v}, \bar{w})$ by

$$\bar{y} = y + \sigma^0 s, \quad \bar{v} = v + \sigma^0 p, \quad \bar{w} = w + \sigma^0 q,$$

where $s^T = t_B^T A_B^{-1}$ and the vectors p and q are given componentwise by

$$\begin{aligned} &p_j = t_j,\ q_j = 0, \quad j \in J^{++} \cup J^{+0}, \\ &p_j = \frac{\bar{\Delta}_j}{\sigma^0},\ q_j = \frac{\Delta_j}{\sigma^0}, \quad j \in J^{+-}, \\ &p_j = -\frac{\Delta_j}{\sigma^0},\ q_j = -\frac{\bar{\Delta}_j}{\sigma^0}, \quad j \in J^{-+}, \\ &p_j = 0,\ q_j = -t_j, \quad j \in J^{--} \cup J^{-0}. \end{aligned}$$

Since

$$s^T A = t_B^T A_B^{-1} A = (t_B, t_N) = t^T,$$

we have $t = A^T s$, and therefore

$$A^T \bar{y} - c = A^T y - c + \sigma^0 A^T s = \Delta + \sigma^0 t = \bar{\Delta}.$$

It follows by direct verification that, for all $j \in J$,

$$(A^T \bar{y} - c)_j - \bar{v}_j + \bar{w}_j = 0,$$

which shows that $\bar{\lambda}$ is DFS.

We now evaluate the variation of the Lagrangian:

$$\bar{\bar{\beta}} = \bar{\beta} + L(\bar{\lambda}) - L(\lambda),$$

where

$$L(\bar{\lambda}) - L(\lambda) = \sigma^0 \sum_{j \in J} (t_j \kappa_j - p_j l_j + q_j u_j) = \sigma^0 (S_B + S_N).$$

The basic component is expressed as

$$S_B = t_{j_1} \kappa_{j_1} - p_{j_1} l_{j_1} + q_{j_1} u_{j_1},$$

which simplifies to

$$S_B = \begin{cases} t_{j_1}(\kappa_{j_1} - l_{j_1}), & \text{if } j_1 \in J_B^{+0}, \\ t_{j_1}(\kappa_{j_1} - u_{j_1}), & \text{if } j_1 \in J_B^{-0}. \end{cases}$$

Using $\alpha_0 = \kappa_{j_1} - \bar{x}_{j_1}$ and $t_{j_1} = -\operatorname{sign}(\alpha_0)$, we obtain

$$S_B = -|\alpha_0|.$$

Restricting attention to the nonbasic indices, S_N is decomposed according to the above partition, where only the subsets J_N^{++}, J_N^{+0}, J_N^{-0}, and J_N^{--} yield nonzero contributions:

$$S_N^{++} = \sum_{j\in J_N^{++}} t_j(\kappa_j - l_j) = \sum_{j\in J_3} t_j(\kappa_j - l_j),$$
$$S_N^{+0} = \sum_{j\in J_N^{+0}} t_j(\kappa_j - l_j) = \sum_{j\in J_0^{+}} t_j(\kappa_j - l_j),$$
$$S_N^{-0} = \sum_{j\in J_N^{-0}} t_j(\kappa_j - u_j) = \sum_{j\in J_0^{-}} t_j(\kappa_j - u_j),$$
$$S_N^{--} = \sum_{j\in J_N^{--}} t_j(\kappa_j - u_j) = \sum_{j\in J_4} t_j(\kappa_j - u_j).$$

Combining these terms yields S_N. Finally,

$$L(\bar{\lambda}) - L(\lambda) = \sigma^0(S_B + S_N) = \sigma^0\alpha,$$

which proves the result. □

If $\alpha \le 0$, then

$$\bar{\bar{\beta}} \le \bar{\beta}.$$

Otherwise, the parameter η is updated according to the following procedure.

Procedure 1 1. Define the index sets

$$\bar{J}_3 = \left\{ j \in J_N : \Delta_j > \eta(\bar{x}_j - l_j)^{-\nu},\ \bar{x}_j > l_j \right\},$$

$$\bar{J}_4 = \left\{ j \in J_N : \Delta_j < \eta(\bar{x}_j - u_j)^{-\nu},\ \bar{x}_j < u_j \right\}.$$

2. If $\bar{J}_3 \cup \bar{J}_4 = \emptyset$, then set $\bar{\eta} = \eta$. Otherwise, compute

$$\eta_0 = \max_{j\in\bar{J}_3}\ \Delta_j(\bar{x}_j - l_j)^{\nu}, \qquad \eta_1 = \max_{j\in\bar{J}_4}\ \Delta_j(\bar{x}_j - u_j)^{\nu}.$$

3. If $\bar{J}_3 \neq \emptyset$ and $\bar{J}_4 = \emptyset$, then set $\bar{\eta} = \eta_0$.

4. If $\bar{J}_3 = \emptyset$ and $\bar{J}_4 \neq \emptyset$, then set $\bar{\eta} = \eta_1$.

5. If $\bar{J}_3 \neq \emptyset$ and $\bar{J}_4 \neq \emptyset$, then set

$$\bar{\eta} = \max\{\eta_0, \eta_1\}.$$

Thanks to Proposition 2, the long-step rule proposed in [10] can be applied to our method. The corresponding procedure is described below.

Procedure 2 1. Compute σ_j for $j \in J_N$, and determine σ^0 using (18)–(19).

2. Order the indices $j \in J_N : \sigma_j \neq \infty$ in nondecreasing order of σ_j:

$$\sigma_{i_1} \le \sigma_{i_2} \le \cdots \le \sigma_{i_p}, \qquad i_k \in J_N,\ k = \overline{1,p}.$$

3. If $p = 1$, set $i_q = i_1$ and go to Step 8.

4. For each $k = \overline{1,p}$, compute

$$\Delta V_{i_k} = |t_{i_k}|(u_{i_k} - l_{i_k}).$$

5. Compute
$$V_0 = -|\alpha_0| + \sum_{j\in J_0^+\cup J_3} t_j(\kappa_j - l_j) + \sum_{j\in J_0^-\cup J_4} t_j(\kappa_j - u_j).$$

6. Compute the sequence $\{V_{i_k}\}_{k=0}^{p}$ defined by
$$V_{i_0} = V_0, \qquad V_{i_k} = V_{i_{k-1}} + \Delta V_{i_k}, \quad k = \overline{1,p}.$$

7. Select the index i_q such that
$$V_{i_{q-1}} < 0 \quad \text{and} \quad V_{i_q} \geq 0.$$

8. Let j_r be the index satisfying
$$\theta^0 = \theta_{j_1} = \min_{j\in J_B} \theta_j,$$
where θ_j is defined in (13). Set
$$\bar{J}_B = (J_B \setminus \{j_1\}) \cup \{i_q\}, \quad \sigma^0 = \sigma_{i_q}, \quad \bar{\Delta} = \Delta + \sigma^0 t.$$

9. Compute
$$\bar{\bar{\beta}} = \bar{\beta} + \sigma_{i_1} V_0 + \sum_{k=2}^{q} (\sigma_{i_k} - \sigma_{i_{k-1}})\, V_{i_{k-1}}.$$

4. Algorithmic Scheme

Let $\{x, J_B\}$ be an initial SFS of problem (1), and let $\epsilon, \eta > 0$. For $\nu \in 2\mathbb{Z}+1$, the hybrid-direction algorithm for solving bounded linear programs is defined as follows:

1. Compute
$$\pi^T = c_B^T A_B^{-1}, \qquad \Delta_N^T = \pi^T A_N - c_N^T.$$

2. Compute β using (2).

3. **Optimality test.**
   - If $\beta = 0$, then $\{x, J_B\}$ is optimal; **stop**.
   - If $\beta \leq \epsilon$, then $\{x, J_B\}$ is ϵ-optimal; **stop**.

4. Compute the index sets J_i, $i = \overline{0,4}$, using (7).

5. Compute γ_ν^η and μ_ν^η using (8)–(9).

6. Compute the primal search direction d using (12).

7. Compute θ_{j_1}, θ_{j_2}, and the step length θ^0 using (13).

8. Update:
$$\bar{x} = x + \theta^0 d, \qquad \bar{z} = z + \theta^0 \gamma_\nu^\eta.$$

9. **Termination criteria.**
   - If $\theta^0 = \dfrac{\beta}{\gamma_\nu^\eta}$, then $\bar{x}$ is optimal; **stop**.
   - If $\theta^0 = 1$ and $\nu \leq -1$, then $\bar{x}$ is optimal; **stop**.

- If $\theta^0 = 1$ and $\nu \geq 1$:
  - If $\mu_\nu^\eta \leq \epsilon$, then $\bar{x}$ is ϵ-optimal; **stop**.
  - Otherwise, update η using Procedure 1, then set

$$\eta \leftarrow \bar{\eta}, \quad x \leftarrow \bar{x}, \quad z \leftarrow \bar{z}, \quad \beta \leftarrow \text{sign}(\nu)\mu_\nu^\eta,$$

    and go to Step 4.

10. If $\theta^0 < 1$, compute

$$\bar{\beta} = \beta - \theta^0 \gamma_\nu^\eta.$$

11. If $\theta^0 = \theta_{j_2}$, set

$$x \leftarrow \bar{x}, \quad z \leftarrow \bar{z}, \quad \beta \leftarrow \bar{\beta}, \quad J_B \leftarrow \bar{J}_B,$$

    and go to Step 4.

12. Compute

$$\kappa = x + d, \qquad \alpha_0 = \kappa_{j_1} - \bar{x}_{j_1}.$$

13. Compute the dual direction t using (15).

14. Compute the sets J_0^+ and J_0^- using (16).

15. Compute α using (17).

    - If $\alpha < 0$, go to Step 16.
    - Otherwise:
      - If $J_0 \neq \emptyset$, select an index $j_0 \in J_0$, and set

$$\Delta \leftarrow \bar{\Delta}, \qquad \bar{J}_B = (J_B \setminus \{j_1\}) \cup \{j_0\},$$

        then go to Step 17.
      - If $J_0 = \emptyset$, update η using Procedure 1, then set

$$x \leftarrow \bar{x}, \quad z \leftarrow \bar{z}, \quad \beta \leftarrow \bar{\beta},$$

        and go to Step 4.

16. Compute

$$\sigma^0, \quad \bar{\Delta} = \Delta + \sigma^0 t, \quad \bar{J}_B = (J_B \setminus \{j_1\}) \cup \{j_0\},$$

    and compute $\bar{\bar{\beta}}$ using Procedure 2.

17. Update

$$x \leftarrow \bar{x}, \quad J_B \leftarrow \bar{J}_B, \quad z \leftarrow \bar{z}, \quad \beta \leftarrow \bar{\bar{\beta}},$$

    and return to Step 3.

5. Example

In this section, we illustrate the proposed method by solving a bounded variable linear programming problem of the form

$$\max\ z(x) = c^T x$$

subject to

$$Ax = b, \quad l \le x \le u,$$

where

$$A = \begin{pmatrix} 3 & -1 & 1 & 1 & 0 \\ -1 & -4 & 1 & 0 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ -3 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad l = 0_{\mathbb{R}^5}, \quad u = \begin{pmatrix} 1 \\ 4 \\ 5 \\ 5 \\ 19 \end{pmatrix}.$$

Iteration 1

We start with $\eta = 1$ and $\nu = 3$. The initial support is chosen as $J_B = \{4, 5\}, \quad J_N = \{1, 2, 3\}$.

Step 1: Construction of an Initial Feasible Solution. An initial basic feasible solution is obtained by solving the system $A_B x_B = b$, where A_B is the submatrix of A associated with the support J_B. Solving this system yields

$$x = (0, 0, 0, 1, 2)^T.$$

A direct verification shows that this vector satisfies the equality constraints $Ax = b$ as well as the bound constraints $l \le x \le u$. Hence, x is a FS of the problem. The corresponding objective value is given by

$$z(x) = c^T x = 0.$$

Step 2: Reduced costs and suboptimality estimate. The multiplier vector is defined by

$$\pi^T = c_B^T A_B^{-1} = (0, 0).$$

The reduced cost vector is then computed as

$$\Delta^T = \pi^T A - c^T = (-1, 3, -1, 0, 0).$$

To evaluate how far the current solution is from optimality, we compute the suboptimality estimate

$$\beta = \sum_{\Delta_j > 0} \Delta_j (x_j - l_j) + \sum_{\Delta_j < 0} \Delta_j (x_j - u_j) = 6.$$

Step 3: Partition of the Set J_N. The set J_N is partitioned into $J_1, \ldots, J_6$ based on the signs of Δ_j and the positions of x_j relative to their bounds. Thus, $J_2 = \{1\}, \quad J_4 = \{3\}, \quad J_5 = \{2\}$, and all other subsets are empty.

Step 4: Computation of γ_ν^η and μ_ν^η. Only the indices in J_2 and J_4 contribute to γ_ν^η. Hence,

$$\gamma_\nu^\eta = \Delta_3 \left(\frac{\eta}{\Delta_3} \right)^{\frac{1}{\nu}} + \Delta_1 (x_1 - u_1) = 2.$$

Consequently,

$$\mu_\nu^\eta = \beta - \gamma_\nu^\eta = 4.$$

Step 5: Computation of the Search Direction d. The direction is computed componentwise. For indices in J_N, we have $d_1 = u_1 - x_1 = 1,\ d_2 = 0,\ d_3 = -\left(\frac{1}{\Delta_3}\right)^{1/3} = 1$, hence $d_N = (1,0,1)^T$. For indices in J_B, feasibility requires $d_B = -A_B^{-1}A_N d_N$, which yields the full direction

$$d = (1,0,1,-4,0)^T.$$

Step 6: Step Length Determination. We compute

$$\theta_{j_1} = \min_{j\in J_B} \theta_j = \min\{\theta_4,\theta_5\} = \theta_4 = \tfrac{1}{4}, \quad \theta_{j_2} = \min_{j\in J_3\cup J_4} \theta_j = \tfrac{u_3 - x_3}{d_3} = 5.$$

Thus,

$$\theta^0 = \min\{1,\ \theta_{j_1},\ \theta_{j_2}\} = \min\{1, \tfrac{1}{4}, 5\} = \tfrac{1}{4}.$$

Step 7: Update of the primal solution.

$$\bar{x} = x + \theta^0 d = \left(\tfrac{1}{4}, 0, \tfrac{1}{4}, 0, 2\right)^T,$$
$$z(\bar{x}) = z(x) + \theta^0 \gamma_\nu^\eta = \tfrac{1}{2}.$$

Step 8: Update of the suboptimality estimate.

$$\bar{\beta} = \beta - \theta^0 \gamma_\nu^\eta = \tfrac{11}{2}.$$

Step 9: Dual Step, Entering Index Selection, and Update. We compute

$$\kappa = x + d = (1,0,1,-3,2)^T, \quad \alpha_0 = \kappa_4 - \bar{x}_4 = -3.$$

The dual direction is defined by

$$t_B^T = (t_4, t_5) = (-\mathrm{sign}(\alpha_0), 0) = (1,0),$$
$$t_N^T = (t_1,t_2,t_3) = t_B^T A_B^{-1} A_N = (3,-1,1),$$

hence

$$t = (3,-1,1,1,0)^T.$$

Ordering the step lengths, we obtain

$$\tfrac{1}{3} = \sigma_1 \le \sigma_3 = 1 \le \sigma_2 = 3.$$

Next,

$$V_0 = -|\alpha_0| + \sum_{j\in J_0^+\cup J_3} t_j(\kappa_j - l_j) + \sum_{j\in J_0^-\cup J_4} t_j(\kappa_j - u_j) = -7.$$
$$\Delta V_1 = 3, \quad \Delta V_3 = 5, \quad \Delta V_2 = 4,$$

which gives

$$V_1 = -4, \quad V_3 = 1.$$

Since a sign change is observed, the entering index is 3, with

$$\sigma^0 = \sigma_3 = 1.$$

Finally, we update

$$\bar{\bar{\beta}} = \bar{\beta} + \tfrac{1}{3} V_0 + \left(1 - \tfrac{1}{3}\right) V_1 = \tfrac{1}{2},$$
$$\bar{\Delta} = \Delta + \sigma^0 t = (2,2,0,1,0)^T,$$

and the support becomes

$$\bar{J}_B = \{3,5\}, \quad \bar{J}_N = \{1,2,4\}.$$

Iteration 2

Using the updated support, we have

$$\bar{J}_B = \{3, 5\}, \quad \bar{J}_N = \{1, 2, 4\},$$

and

$$x = \left(\tfrac{1}{4}, 0, \tfrac{1}{4}, 0, 2\right)^T, \quad z(x) = \tfrac{1}{2}.$$

The reduced costs are

$$\Delta = (2, 2, 0, 1, 0)^T,$$

with suboptimality estimate

$$\beta = \tfrac{1}{2}.$$

Since only $J_1 = \{1\}$ is nonempty, it follows that

$$\gamma_\nu^\eta = \tfrac{1}{2}.$$

The search direction is

$$d = (1, 0, 1, -4, 0)^T,$$

and the admissible step length is $\theta^0 = 1$. Thus, the updated solution is

$$\bar{x} = (0, 0, 1, 0, 1)^T, \quad z(\bar{x}) = 1.$$

Since

$$\theta^0 = \frac{\beta}{\gamma_\nu^\eta},$$

the solution is optimal.

6. Experimental Results

We denote by HDA-(ν) the proposed parametric hybrid-direction algorithm, and by HDA-(k) its instance corresponding to a fixed parameter value $\nu = k$. The objective of this section is to provide a rigorous computational assessment of HDA-(ν) (see Section 4), with particular emphasis on quantifying the effect of the parameter $\nu \in 2\mathbb{Z} + 1$ on computational performance, and overall robustness.

The algorithm was implemented in C++ and executed on a workstation equipped with an Intel Core i7-4790 processor (3.60 GHz) and 8 GB of RAM, running under Windows 10. The benchmark set consists of randomly generated bounded linear programming instances constructed to guarantee feasibility and boundedness. For each problem size, 10 independent instances were generated, and all reported results correspond to averaged values in order to mitigate random variability.

The test problems are formulated as

$$\max z = c^T x, \quad \text{s.t. } Ax \le b, \quad l \le x \le u,$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $c, x, l, u \in \mathbb{R}^n$. The constraint matrix A was generated with a sparsity density of 5%, reflecting large-scale structured instances. The entries of A and a reference optimal solution x^0 were randomly sampled as $A_{ij} \in [-100, 100]$ and $x_i^0 \in [1, 100]$, while the vectors b and c were constructed so that x^0 is an optimal solution. In total, 100 instances were considered, including 50 problems of size $m \times m$ and 50 problems of size $m \times 2m$.

The algorithm was evaluated for $\nu \in \{-3, -1, 1, 3\}$, with $\eta = 1$ fixed in all experiments. The initial support was defined as $J_B = \{m + 1, \ldots, 2m\}$ for problems of size $m \times m$, and $J_B = \{2m + 1, \ldots, 3m\}$ for problems of size $m \times 2m$. The performance was assessed using the number of iterations **nit** and the **CPU** time in seconds. The numerical results are reported in Tables 1 and 2, corresponding to the two problem classes.

$m \times m$	HDA-(-3)		HDA-(-1)		HDA-(1)		HDA-(3)	
	nit	**CPU (s)**	**nit**	**CPU (s)**	**nit**	**CPU (s)**	**nit**	**CPU (s)**
200×200	64.3	0.037	63.2	0.036	62.2	0.035	63.3	0.036
400×400	74.8	0.053	73.5	0.052	72.4	0.051	73.6	0.052
600×600	100.3	0.081	98.5	0.079	97.1	0.078	98.6	0.081
800×800	174.4	0.165	171.4	0.161	168.8	0.157	171.4	0.161
1000×1000	218.8	0.338	215.2	0.333	211.8	0.264	215.5	0.27

Table 1. Numerical results for test problems of size $m \times m$.

$m \times 2m$	HDA-(-3)		HDA-(-1)		HDA-(1)		HDA-(3)	
	nit	**CPU (s)**	**nit**	**CPU (s)**	**nit**	**CPU (s)**	**nit**	**CPU (s)**
200×400	67.5	0.057	66.2	0.055	65.2	0.054	66.4	0.055
400×800	108.6	0.086	106.7	0.085	105.2	0.083	106.9	0.085
600×1200	148	0.11	145.3	0.107	143.7	0.106	145.9	0.11
800×1600	211.6	0.234	208.3	0.229	205.1	0.225	208.2	0.231
1000×2000	287.3	0.451	282.6	0.444	278.3	0.438	283.2	0.448

Table 2. Numerical results for test problems of size $m \times 2m$.

The computational results provide several important insights. First, the algorithm exhibits a high degree of robustness with respect to the parameter ν, as the variations in iteration counts and CPU times across all tested values remain marginal. This indicates that the method is inherently stable and does not require delicate parameter tuning. Second, HDA-(1) consistently achieves the best performance, attaining the lowest iteration counts and execution times for nearly all instances. This suggests that $\nu = 1$ induces a hybrid direction that effectively balances convergence speed and feasibility preservation. Third, negative values of ν (namely -1 and -3) tend to produce slightly higher computational costs, which may indicate a less favorable alignment of the corresponding search directions with the geometry of the feasible region. Furthermore, the computational effort increases smoothly with problem size, both in terms of iterations and CPU time, without exhibiting irregular behavior, thereby confirming the scalability of the method. Finally, the same qualitative trends are observed for both $m \times m$ and $m \times 2m$ instances, which highlights the robustness of the algorithm with respect to changes in problem dimension and structure.

Overall, these results demonstrate that HDA-(ν) is a stable and efficient algorithm for bounded linear programming, with $\nu = 1$ emerging as a particularly effective choice in practice.

REFERENCES

1. K. Djeloud, M. Bentobache, and M. O. Bibi, *A new method with hybrid direction for linear programming*, *Concurrency and Computation: Practice and Experience*, vol. 33, no. 1, e5836, 2021.
2. K. Djeloud, *Une nouvelle direction pour l'optimisation linéaire et application au contrôle optimal*, PhD Thesis, 2021.
3. K. Djeloud, *The Dual Simplex Method for Solving Bounded-Variable Linear Programs*, *Statistics, Optimization & Information Computing* Accepted for publication, 2026.
4. K. Djeloud, M. Bentobache, and M. O. Bibi, *A hybrid direction algorithm for solving linear programs with bounded variables*, Proceedings of COSI'2017, University of Bouira, May 14–16, 2017.
5. M. O. Bibi, M. Bentobache, K. Djeloud, and R. Guerbane, *Hybrid Direction Methods for Linear Programming and Applications*, *Advances in Mathematics Research*, Nova Science Publishers, vol. 33, no. 1, pp. 1–77, 2023.

6. M. Bentobache and M. O. Bibi, *Numerical Methods of Linear and Quadratic Programming*, French Academic Press, Germany, 2016 (in French).
7. M. A. Hakmi, M. Bentobache, and M. O. Bibi, *A Hybrid Direction Method for Linear Fractional Programming*, *Statistics, Optimization & Information Computing*, vol. 13, no. 3, pp. 922–948, 2025.
8. G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, 2016.
9. R. Gabasov and F. M. Kirillova, *Methods of Linear Programming, Parts 1–3: Special Problems*, Izdatel'stvo BGU, Minsk, Belarus, 1977, 1978, 1980.
10. R. Gabasov, F. M. Kirillova, and S. V. Prischepova, *Optimal Feedback Control*, Springer, 1995.
11. L. G. Khachiyan, *A Polynomial Algorithm for Linear Programming*, *Doklady Akademii Nauk SSSR*, vol. 244, pp. 1093–1096, 1979.
12. N. Karmarkar, *A New Polynomial-Time Algorithm for Linear Programming*, Proceedings of the 16th Annual ACM Symposium on Theory of Computing, pp. 302–311, 1984.
13. E. Kostina, *The Long Step Rule in the Bounded-Variable Dual Simplex Method: Numerical Experiments*, *Mathematical Methods of Operations Research*, vol. 55, pp. 413–429, 2002.
14. E. R. Bixby, *Implementing the Simplex Method: The Initial Basis*, *ORSA Journal on Computing*, vol. 4, pp. 1–18, 1992.
15. A. Makhorin, *GNU Linear Programming Kit: Reference Manual, Version 4.9*, Draft Edition, 2006. Available at: https://www.gnu.org/software/glpk/glpk.html.