RAIRO-Oper. Res. 56 (2022) 3643–3658
https://doi.org/10.1051/ro/2022165

RAIRO Operations Research
www.rairo-ro.org

LINEAR PROGRAMMING WITH A FEASIBLE DIRECTION INTERIOR POINT TECHNIQUE FOR SMOOTH OPTIMIZATION

Angélica Miluzca Victorio Celis[1,*] and José Herskovits Norman[2,†]

Abstract. We propose an adaptation of the Feasible Direction Interior Points Algorithm (FDIPA) of J. Herskovits, for solving large-scale linear programs. At each step, the solution of two linear systems with the same coefficient matrix is determined. This step involves a significant computational effort. Reducing the solution time of linear systems is, therefore, a way to improve the performance of the method. The linear systems to be solved are associated with definite positive symmetric matrices. Therefore, we use Split Preconditioned Conjugate Gradient (SPCG) method to solve them, together with an Incomplete Cholesky preconditioner using Matlab's ICHOL function. We also propose to use the first iteration of the conjugate gradient, and to presolve before applying the algorithm, in order to reduce the computational cost. Following, we then provide mathematica proof that show that the iterations approach Karush–Kuhn–Tucker points of the problem under reasonable assumptions. Finally, numerical evidence show that the method not only works in theory but is also competitive with more advanced methods.

Mathematics Subject Classification. 90C05, 90C51, 90C06.

Received September 1, 2022. Accepted September 21, 2022.

1. Introduction

Linear programming today is widely used in decision making in finance, economics, engineering, as well as a tool for solving integer and nonlinear programming problems.

There are two kinds of methods to solve linear programming (LP) problems. One of the first proposed algorithms is a simplex method proposed by Dantzig [1, 2]. This algorithm searches all the edge of the feasible region (which is generated by the constraints). In the worst case, it tests all the polytop's vertices and possesses an exponential complexity. Later, to avoid this complexity, several researchers, such as Von Neumann [3], and Hoffman *et al.* [4] proposed algorithms that traverse the interior of the viable region, but in practice they saw

Keywords. Interior point, feasible direction, linear programming.

[1] Systems Engineering and Computer Science, COPPE, Federal University of Rio de Janeiro, P.O. Box 68511, Rio de Janeiro, RJ 21945-970, Brazil.

[2] Mechanical Engineering Program, COPPE, Federal University of Rio de Janeiro, P.O. Box 68503, CEP 21945-970, CT, Cidade Universitéria, Ilha do Fundão, Rio de Janeiro, Brazil.

[†]This work is dedicated to the memory of our advisor José Herskovits, who passed away while this document was being written. José Herskovits was a brilliant researcher and a dedicated adviser.

[*]Corresponding author: miluzcav@cos.ufrj.br

that it was not competitive with the simplex method because of the possibility of numerical instability in the calculations and the expensive computational steps they require.

The second method is the interior point method that arises with the Karmarkar's algorithm [5], which starts from a strict interior point and applies repeated projective transformations on an inscribed sphere to create a sequence of points that converges to the optimal solution with polynomial complexity. From the Karmarkar algorithm, different types of interior point algorithms have been generated, depending on the type of transforms and search directions to be used. For example, the affine scaling method studied by Dikin [6], Barnes [7], Cavalier and Soyster [8], the path following method by Gonzaga [9, 10] and Ye potential reduction method [11]. For a better understanding of interior point theory, reading the books [12, 13] is suggested.

In this article, the Algorithm FDIPA proposed by Herskovits [14] for nonlinear programming problems will be adapted to solve large-scale linear programming problems (FDIPA-LP) efficiently. Starting from a strictly feasible initial point, FDIPA performs a Newton-like iteration to solve the Karush–Kuhn–Tucker (KKT) optimality condition. During this procedure a descent direction for the objective function is obtained. A perturbation is introduced for the complementary condition so that a feasible descent direction is obtained. This algorithm ensures the primary feasibility and updating the dual variable ensures its dual feasibility. This is relevant, especially in engineering, economics or finance, where physically non-admissible solutions can not be accepted.

FDIPA has already been adapted to various types of problems, see *e.g.,* [15–20] with various applications, especially in mechanical engineering, aerodynamic and electromagnetism; proving to be an easy code to implement, strong and efficient.

FDIPA already has a first adaptation to solve LP in Tits and Zhou [21], where it has a resemblance to the affine scaling method, demonstrating convergence using as search direction, the direction obtained by solving the first FDIPA system (that is, it is a FDIPA without perturbation), and a step which guarantees strict viability.

We will see that both FDIPA-LP and the unperturbed FDIPA work well in theory. Computationally, FDIPA-LP has to solve two linear systems, which implies a significant computational effort. To reduce the size of these systems, before starting the algorithm, we will use a presolver [22] and scaling techniques. We will also use the Split Preconditioned Conjugate Gradient (SPCG) method to solve them, along with an incomplete Cholesky preconditioner using Matlab's ICHOL function. It was observed, that the iterations generated by SPCG are candidates to be search direction. Therefore, we use only the first iteration of the SPCG.

In the next section, we discuss and present the adaptation of FDIPA algorithm for LP case of inequality constrained optimization. Section 3 studies the convergence, the implementation details are presents in Section 4. In Section 5, we present the results of FDIPA-LP algorithm applied to the problems from NETLIB and we compare the FDIPA performance with and without perturbations following [21]. Finally, in the last section, we present some conclusions.

2. The feasible direction interior points algorithm for linear programming

We consider the linear programming problem with the inequality formulation:

$$\begin{aligned} &\underset{\lambda\in R^m}{\text{minimize}} \quad b^T\lambda \\ &\text{subject to} \quad A\lambda \leq c, \end{aligned} \tag{1}$$

where $A \in R^{n\times m}$ is the constraint's matrix, $c \in R^m$ is the independent vector, $b \in R^n$ is the cost vector and $\lambda \in R^m$ is the variables vector, with $n \geq m \geq 1$ and $Rank(A) = m$.

Let $I = \{1, \ldots, n\}$, for $i \in I$, let a_i^T denotes the ith row of A, c_i is the ith component of c. The constraints function g_i is given by

$$g_i(\lambda) = a_i^T\lambda - c_i.$$

Define $f(\lambda) = b^T\lambda$, where f is the objective function that is to be minimized.

The set of feasible points of (1), denoted by Ω, is defined

$$\Omega = \{\lambda|\ g_i(\lambda) \leq 0,\ i \in I\}.$$

By the interior set of Ω we mean

$$int(\Omega) = \{\lambda |\ g_i(\lambda) < 0,\, i \in I\}. \tag{2}$$

The Lagrangian of problem (1) is

$$L(\lambda, x) = f(\lambda) + \sum_{i=1}^{n} x_i g_i(\lambda). \tag{3}$$

With it gradient given by

$$\nabla_\lambda L(\lambda, x) = b + A^T x. \tag{4}$$

A point $\lambda \in \Omega$ is a stationary point of the problem (1) if there exists $x \in R^n$ such that:

$$A^T x + b = 0, \tag{5}$$

$$G(\lambda)X = 0, \tag{6}$$

where X denotes $diag(x)$ and $G(\lambda)$ denotes $diag(g(\lambda))$, we will refer to such x as a *multiplier vector* associated with λ. If $x \geq 0$, then λ is a Karush–Kuhn–Tucker (KKT) point of problem (1).

Next, we adapt the basic ideas of the interior point method with feasible direction (FDIPA) developed by Herskovits [14].

Let

$$\Omega_{\mathtt{p}} = \{\lambda \in \Omega |\ \ f(\lambda) \leq \mathtt{p}\},$$

where $\mathtt{p}$ is a real number with the interior set $int(\Omega_{\mathtt{p}}) = \{\lambda \in int(\Omega) |\ \ f(\lambda) < \mathtt{p}\}$.

We now make the following assumptions:

Assumption 1. *There exist a real number $\mathtt{p}$ such that $\Omega_{\mathtt{p}}$ is compact and has a nonempty interior $int(\Omega_{\mathtt{p}})$.*

Assumption 2. *(Regularity condition) for all $\lambda \in \Omega$ the vectors $\nabla g_i(\lambda) = a_i$, for $i \in I$, such that $g_i(\lambda) = 0$, are linearly independent.*

FDIPA algorithm looks for a feasible primal and dual solutions, applying a Newton-like iteration to KKT equations in such a way that it has a sequence of strictly feasible points minimizing the objective function.

The Newton-like iteration that solves (5) and (6) is given by the following system

$$\begin{bmatrix} 0 & A^T \\ XA & G(\lambda) \end{bmatrix} \begin{bmatrix} \lambda_\alpha - \lambda \\ x_\alpha - x \end{bmatrix} = - \begin{bmatrix} b + A^T x \\ G(\lambda)x \end{bmatrix}, \tag{7}$$

where $(\lambda, x) \in int(\Omega) \times \mathbb{R}^n_{++}$ is the current point, $(\lambda_\alpha, x_\alpha)$ is the new estimated point. We denote the matrix of the system (7) by $W(\lambda, x)$.

We define a direction $d_\alpha = \lambda_\alpha - \lambda$. So, we have:

$$\begin{bmatrix} 0 & A^T \\ XA & G(\lambda) \end{bmatrix} \begin{bmatrix} d_\alpha \\ x_\alpha \end{bmatrix} = \begin{bmatrix} -b \\ 0 \end{bmatrix}. \tag{8}$$

We perturb on the right-hand side of the system (8) generating a new feasible direction that points to $int(\Omega)$,

$$\begin{bmatrix} 0 & A^T \\ XA & G(\lambda) \end{bmatrix} \begin{bmatrix} d \\ \bar{x} \end{bmatrix} = \begin{bmatrix} -b \\ -\rho x \end{bmatrix}, \tag{9}$$

where $\rho > 0$ is an appropriate positive real number to ensure that d is a descent direction, and $\bar{x}$ is the new estimate of x.

On the other hand, the pair $(d, \bar{x})$ obtained by system (9) can be calculated by (8) and the following system

$$\begin{bmatrix} 0 & A^T \\ XA & G(\lambda) \end{bmatrix} \begin{bmatrix} d_\beta \\ x_\beta \end{bmatrix} = \begin{bmatrix} 0 \\ -x \end{bmatrix}, \tag{10}$$

where,

$$d = d_\alpha + \rho d_\beta, \tag{11}$$

$$\bar{x} = x_\alpha + \rho x_\beta. \tag{12}$$

Once we have computed a descent and feasible direction d, we can determine the next iterative.

A full step in this direction d is generally not allowed, as it would violate (2). To avoid this difficulty, we perform a linear search along the d direction to get the feasibility $g(\lambda + td) < 0$ and an appropriate objective function reduction $f(\lambda + td) < f(\lambda)$ so that the new iteration is $\lambda + td$ for some line search parameter $t \in (0, 1]$, guaranteeing that the sequence of points is $int(\Omega)$.

Evaluating f at point $\lambda + td$,

$$f(\lambda + td) = f(\lambda) + td^T b, \tag{13}$$

if $d^T b < 0$, we obtain

$$f(\lambda + td) < f(\lambda) \text{ for all } t \in \mathbb{R}. \tag{14}$$

In the following section, we show the feasibility of this direction $(d^T b < 0)$.

Evaluating g_i at point $\lambda + td$, for all $i \in I$,

$$g_i(\lambda + td) = g_i(\lambda) + td^T a_i, \tag{15}$$

s We look at the possible values of $d^T a_i$:

– If $d^T a_i < 0$ then $g_i(\lambda + td) < g_i(\lambda)$ for all $t \in \mathbb{R}_{++}$.
– If $d^T a_i = 0$ then $g_i(\lambda + td) = g_i(\lambda)$ for all $t \in \mathbb{R}$.
– If $d^T a_i > 0$, to guarantee feasibility $(g_i(\lambda + td) \leq 0)$, then

$$t \leq \frac{-g_i(\lambda)}{d^T a_i}. \tag{16}$$

The value of t has to be the minimum

$$t = \min_i \left\{ \frac{-g_i(\lambda)}{a_i d} \mid d^T a_i > 0 \right\}. \tag{17}$$

2.1. The statement of the algorithm

The proposed FDIPA-LP is presented as follows.

Algorithm 2.1 (FDIPA-LP)**.** Input parameters: $\xi \in (0, 1)$, $\varphi > 0$, $\mu > 0$, $\gamma > 0$, $x^s > 0$. Initial data: $\lambda \in int(\Omega)$ and $x \in R^n_{++}$.

Step 1: Determination of the descent feasible direction.
(i) Solve the linear system (d_α, x_α):

$$A^T x_\alpha = -b, \tag{18}$$

$$XAd_\alpha + G(\lambda)x_\alpha = 0. \tag{19}$$

If $d_\alpha = 0$, stop.

(ii) Solve the linear system (d_β, x_β):

$$A^T x_\beta = 0, \tag{20}$$

$$XAd_\beta + G(\lambda)x_\beta = -x. \tag{21}$$

(iii) Compute the positive scalar ρ: If $d_\beta^T b > 0$, then it is defined as

$$\rho = \min\{\varphi\|d_\alpha\|^2;\ \ (\xi - 1)d_\alpha^T b / d_\beta^T b\}, \tag{22}$$

otherwise

$$\rho = \varphi\|d_\alpha\|^2. \tag{23}$$

(iv) Compute the search direction d as

$$d = d_\alpha + \rho d_\beta, \tag{24}$$

$$\bar{x} = x_\alpha + \rho x_\beta. \tag{25}$$

Step 2: Line search

(i) If $a_i d > 0;\ i = 1, \ldots, m$ compute

$$\bar{t} = \min\left\{\frac{-g_i(\lambda)}{a_i d}\right\}, \tag{26}$$

otherwise

$$\bar{t} = \infty, \tag{27}$$

set

$$t = \min\{\max\{\gamma\bar{t}, \bar{t} - \|d\|\}, 1\}. \tag{28}$$

Step 3: Update.

(i) Take

$$\lambda := \lambda + td. \tag{29}$$

(ii) Compute,

$$x_i := \min\{\max\{x_{\alpha i}, \mu\|d_\alpha\|^2\}, x^s\}, \quad \text{for all} \ \ i \in I. \tag{30}$$

(iii) Go to step 1.

The algorithm performs iterations within the feasible region. In step 1(ii) we obtain the value of the deflection ρ that guarantees a descent direction d. In step 2, $\bar{t}$ is the maximum step that guarantees feasibility and with the expression (28) we get a strict feasibility. In step 3(ii), we use the same rule for FDIPA nonlinear programming to update Lagrange multipliers x.

3. Study of convergence

In this section, we will prove, under certain nondegeneracy assumptions, that for any starting point $\lambda^0 \in int(\Omega)$ the sequence generated (λ^k, x_α^k) by the algorithm is convergent to (λ^*, x_α^*) under the Assumptions 1 and 2, where λ^* is a KKT point.

First, let

$$W(\lambda, x) = \begin{bmatrix} 0 & A^T \\ XA & G(\lambda) \end{bmatrix}, \tag{31}$$

where $\lambda \in \Omega$ and $x_i > 0$, for all $i \in I$. The matrix W is non-singular whenever Assumption 2 holds. Proof see Lemma 3.1 of Tits and Zhou [21]. Consequently, $d_\alpha,\ x_\alpha,\ d_\beta\ x_\beta$ obtained in step 1 of the algorithm are well defined.

Note that, if the algorithm ends at iteration $k \in \mathbb{N}$, $d_\alpha^k = 0$ then λ^k solves the equation (19). Now, looking at equation (18), we have $x_\alpha^k = 0$ and consequently $b = 0$. Therefore, when $b \neq 0$, the algorithm generates an infinite sequence of x_α^k. From now on, we study the case $d_\alpha^k \neq 0$ at every iteration.

The iterations of the algorithm FDIPA-LP generate $\lambda^k \in int(\Omega)$ and $X^k > 0$. The following lemma shows that d is a descent direction.

Lemma 3.1. *Let $\xi \in (0,1)$, if $d \neq 0$ and $d_\alpha \neq 0$, then*

$$d^T b \leq \xi d_\alpha^T b < 0. \tag{32}$$

Proof. From the algorithm, multiply (18) by d_α^T

$$d_\alpha^T A^T x_\alpha = -d_\alpha^T b, \tag{33}$$

given $X > 0$, multiplying the equation (19) by $x_\alpha^T X^{-1}$ gives:

$$x_\alpha^T A d_\alpha + x_\alpha^T X^{-1} G(\lambda) x_\alpha = 0. \tag{34}$$

Substituting (33) into (34),

$$d_\alpha^T b = x_\alpha^T X^{-1} G(\lambda) x_\alpha. \tag{35}$$

Clearly, $X^{-1}G(\lambda)$ is a negative definite matrix. Thus, equation (35) for any $x_\alpha \neq 0$ yields

$$d_\alpha^T b < 0. \tag{36}$$

Also, multiplying equation (24) by b, we have that

$$d^T b = d_\alpha^T b + \rho d_\beta^T b, \tag{37}$$

by the algorithm in step 1(iii) to obtain $\rho > 0$ in both cases, moreover

- If $d_\beta^T b \leq 0$, from the equations (37) and (36) we have that

$$d^T b \leq d_\alpha^T b < 0. \tag{38}$$

- If $d_\beta^T b > 0$, from the equation (22) we have that

$$\begin{aligned} \rho &\leq (\xi - 1)\frac{d_\alpha^T b}{d_\beta^T b}, \\ d_\alpha^T b + \rho d_\beta^T b &\leq \xi d_\alpha^T b, \\ d^T b &\leq \xi d_\alpha^T b. \end{aligned} \tag{39}$$

Therefore, from the equations (36), (38) and (39),

$$d^T b \leq \xi d_\alpha^T b < 0. \tag{40}$$

□

As a consequence of the previous lemma, f is monotone decreasing, d_α and d are descent directions of f, yielding

$$f(\lambda + td) = f(\lambda) + td^T b,$$

therefore,

$$f(\lambda + td) < f(\lambda),$$

by construction, $t \in (0,1]$, and Lemma 3.1.

On the other hand, with the choice of ρ in step 1 of the algorithm, we ensure that

$$0 < \rho \leq \varphi \|d_\alpha\|^2. \tag{41}$$

Now we are going to show feasibility of d.

Lemma 3.2. *There exists $\tau > 0$ such that at any $\Omega_{\mathtt{p}}$ and d computed in Step 1 of the algorithm 1.1, we have $g(\lambda + td) \leq 0$ for all $t \in [0, \tau]$.*

Proof. The step length t is defined in the equation (28) of the algorithm. Since the constraints are linear, to satisfy the line search condition the following inequalities must be true:

$$g_i(\lambda + td) = g_i(\lambda) + ta_i^T d \leq 0 \tag{42}$$

for all $i \in I$. If $a_i^T d \leq 0$, the above inequality is satisfied with any $t > 0$. Otherwise, it follows from (9) that

$$g_i(\lambda)\left(1 - t\frac{\bar{x}_i}{x_i}\right) - t\rho \leq 0, \tag{43}$$

Obviously $\rho t > 0$ and $g(\lambda) < 0$. Thus, the inequality is satisfied if

$$t\left(\frac{\bar{x}_i}{x_i}\right) \leq 1. \tag{44}$$

Now, x is bounded by (30) and, since x_α, x_β and ρ are bounded from above, also $\bar{x}$ is bounded from above. Thus, there exists $\tau > 0$ such that $x_i/\bar{x}_i > \tau$. Therefore, for all $i \in I$ and for all $t \in [0, \tau]$, we have $g_i(\lambda + td) \leq 0$. □

Considering $f(\lambda^0) = \mathtt{p}$, we note that, given a point $\lambda^0 \in int(\Omega)$ and a decreasing function f, we have any sequence $\{\lambda^k\}$ generated by the algorithm is contained in $\Omega_{\mathtt{p}}$. Then, since Assumption 1, $\{\lambda^k\}$ has accumulation point in $\Omega_{\mathtt{p}}$.

Lemma 3.3. *Every accumulation point of the sequence $\{\lambda^k\}$, generated by the algorithm, is a stationary point of the problem. Besides that, (λ^*, x_α^*) is a stationary pair, where x_α^* is the unique multiplier vector associated with λ^*.*

Proof. Consider a sequence $\{\lambda^k\}_{k \in K}$, where $K \subset \mathbb{N}$, converging to λ^*.

By construction $x_i^k \in [0, x^s]$, it follows from (41) that $\{x^k\}$ and $\{\rho^k\}$ are bounded, there exists $K_1 \subset K$, such that

$$\{\lambda^k, x^k, \rho^k\}_{k \in K_1}$$

converge to (λ^*, x^*, ρ^*).

Since d^k depends continuously on λ^k, x^k, and ρ^k, we have that $\{d^k\}_{k \in K_1} \to d^*$, with $d^* = d(\lambda^*, x^*, \rho^*)$.

Since $t^k \in (0, 1]$ for all k, it them follows from Lemma 3.1 that there exists $\hat{t} > 0$ such that

$$f(\lambda^{k+1}) \leq f(\lambda^k) + \hat{t}d^{kT}b, \quad \text{for all} \quad k \in K_1$$

where we assume that $k \in K_1$ and $k + 1$ do not necessarily belong to K_1.

$$f\left(\lambda^{k\prime}\right) < f\left(\lambda^k\right) + \hat{t}d^{kT}b \quad \text{for all} \quad k' \geq k + 1,$$

and taking limits on both sides for $k \to \infty$,

$$f(\lambda^*) \leq f(\lambda^*) + \hat{t}d^{*T}b,$$

following that

$$0 \leq d^{*T}b. \tag{45}$$

Considering now the result of Lemma 3.1 in the limit for $k \to \infty$ we get

$$d^{*T} b = 0. \tag{46}$$

It also follows from Lemma 3.1, yields $d_\alpha^{*T} = 0$.
Taking limits in (18) and (19), we have

$$A^T x_\alpha^* = -b, \tag{47}$$
$$G(\lambda^*) x_\alpha^* = 0. \tag{48}$$

Thus, λ^* is stationary, with multiplier vector x_α^*. From the equation (47) and the Assumption 2 we have the uniqueness of x^*. □

Lemma 3.4. *Suppose $\{\lambda^k\}$ is bounded. Let L be the set of limit points of $\{\lambda^k\}$, then L is connected compact sets with the same active constraints.*

Proof. It is analogous to the proof of Lemmas A.5 and 3.6 in Tits and Zhou [21]. Lemma A.5 proves that L is connected compact set, and Lemma 3.6 proves that L possesses a unique x^* associated with it. □

The Lemma 3.4 appears as a hypothesis in FDIPA for nonlinear programming [14], it helps in the proof of the convergence of the sequence of points generated by the algorithm to a KKT point.
Next, a technical lemma to understand convergence.

Lemma 3.5. *The direction d computed by the Algorithm 1.1 satisfies*

$$d^T a_i < 0, \quad \text{for each} \quad \bar{x}_i < 0. \tag{49}$$

Proof. Rewriting the equations (19) and (21):

$$\begin{aligned} XA^T d_\alpha + G(\lambda) x_\alpha &= 0, \\ XA^T d_\beta + G(\lambda) x_\beta &= -x, \end{aligned}$$

and for all $i \in I$ yield

$$a_i^T d_\alpha = -\frac{x_{\alpha i}}{x_i} g_i(\lambda), \tag{50}$$
$$a_i^T d_\beta = -\frac{x_{\beta i}}{x_i} g_i(\lambda) - 1. \tag{51}$$

multiplying equation (24) by a_i^T

$$a_i^T d = a_i^T d_\alpha + \rho a_i^T d_\beta. \tag{52}$$

Substituting (50) and (51) into (52) gives

$$\begin{aligned} a_i^T d &= -\frac{x_{\alpha i}}{x_i} g_i(\lambda) - \rho \frac{x_{\beta i}}{x_i} g_i(\lambda) - \rho \\ &= -\frac{g_i(\lambda)}{x_i}(x_{\alpha i} + \rho x_{\beta i}) - \rho \\ &= -\frac{g_i(\lambda)}{x_i} \bar{x}_i - \rho, \end{aligned} \tag{53}$$

in the equation (53), if $\bar{x}_i < 0$ then

$$a_i^T d < 0. \tag{54}$$

□

The constraint functions associated with estimated negative multipliers are decreasing functions ($g_i(\lambda+td) < g_i(\lambda)$). This ensures that stationary points that are not KKT points will be avoided.

Finally, Theorem 3.6 proves that the presented algorithm converges to the solution of the problem.

Theorem 3.6. *Any accumulation point λ^* of any sequence generated by the algorithm $\{\lambda^k\}$ is a Karush–Kuhn–Tucker point of the problem.*

Proof. As λ^* is a stationary point of the problem (1), it is only necessary to prove that the Lagrange multipliers x^*_α are nonnegative.

We have that $d^*_\alpha = 0$ and $d^* = 0$ at a stationary point. Then, it follows from (19) that

$$G(\lambda^*)x^*_\alpha = 0,$$

where $x^*_{\alpha i} = 0$ in the case when $g_i(\lambda^*) < 0$.

Consider now a constraint $g_j(\lambda^*) = 0$, yielding the following consequences. As the method is strictly feasible,

$$g_j\left(\lambda^k\right) < 0 \text{ for all } k \in \mathbb{N}.$$

Then, we can define a sequence $\{\lambda^k\}_{k\in K}$, $K \subset \mathbb{N}$, converging to λ^* such that

$$g_j\left(\lambda^k\right) > g_j\left(\lambda^{k-1}\right) \text{ for any } k \in K.$$

Note that $k-1$ may not belong to K and from of Lemma 3.1, we get $\bar{x}^{k-1}_j \geq 0$.

Now, we proceed by contradiction and assume that $x^*_{\alpha j} < 0$. It follows from (41) that $\rho^* = 0$ and from (12)

$$x^* = x^*_\alpha + \rho^* x^*_\beta,$$

we have $\bar{x}^*_j < 0$. Then, since Lemma 3.4 holds, there exists an open ball $B(\gamma_j) \equiv \{\lambda \,|\, \|\lambda - \lambda^*\| < \gamma_j\}$, such that $\bar{x}_j < 0$ for any $\lambda \in \{int(\Omega_{\mathtt{p}}) \cap B(\gamma_j)\}$ and any x and ρ generated by the algorithm.

Consider now the sequence $\{\lambda^{k-1}\}_{k\in K}$. As $\bar{x}^{k-1}_j \geq 0$, for any $k \in K$, we have that $\lambda^{k-1} \notin B(\gamma_j)$. Thus, $\{\lambda^{k-1}\}_{k\in K}$ is in the compact set $\{\Omega_{\mathtt{p}} - B(\gamma_j)\}$, therefore this set processes an accumulation point. In consequence of Lemma 3.3, this point is a stationary point of problem (1).

Let $\bar{\lambda}$ be one of these points and $\{\lambda^{k-1}\}_{k\in\bar{K}}$, $\bar{K} \subset K$, a sequence convergent to $\bar{\lambda}$ therefore we have that $\|d^{k-1}\| \to 0$. Then, for I large enough, $\|d^{k-1}\| < \gamma_j$ for any $k > I$, $k \in \bar{K}$ and, in consequence of the line search,

$$\begin{aligned} \left\|\lambda^k - \lambda^{k-1}\right\| &= \left|t^{k-1}\right|\left\|d^{k-1}\right\| \\ \left\|\lambda^* - \bar{\lambda}\right\| &< \gamma_j. \end{aligned}$$

This result is in contradiction with the fact that $\bar{\lambda} \notin B(\gamma_j)$ proving the theorem. □

4. Implementation of interior points feasible directions for linear programming

First, we will present the split preconditioned conjugate gradient method, necessary for solving the internal linear systems.

4.1. Split preconditioned conjugate gradient method (SPCG)

The SPCG method is an algorithm for the numerical solution of systems of linear equations, such as those whose matrices are symmetric and positive-definite.

The algorithm employs the use of preconditioned matrix L, when working with poorly conditioned matrices, to guarantee the convergence of the method [23]

The algorithm to solve ($Qx = B$), with the incomplete Cholesky decomposition L where $M = LL^T$ is a symmetric positive definite.

Algorithm 4.1 (SPCG)**.** Data: $Q \in R^{n\times n}$ symmetric positive definite, $B \in R^n$ and $x_0 \in R^n$.

Step 1: Calculate

$$\begin{aligned} r_0 &= B - Qx_0, \\ \hat{r}_0 &= L^{-1} r_0, \\ p_0 &= L^{-T}\hat{r}_0. \end{aligned}$$

Step 2: Compute for $j = 0, 1, \ldots,$ until converge, do

$$\begin{aligned} \alpha_j &= (\hat{r}_j, \hat{r}_j)/(Qp_j, p_j), \\ x_{j+1} &= x_j + \alpha_j p_j, \\ \beta_j &= (\hat{r}_{j+1}, \hat{r}_{j+1})/(\hat{r}_j, \hat{r}_j), \\ p_{j+1} &= L^{-T}\hat{r}_{j+1} + \beta_j p_j, \end{aligned}$$

end.

Convergence criterion used here was $\frac{\|\hat{r}_j\|}{\hat{r}_0} < \epsilon$. Also we compute L, an incomplete Cholesky preconditioner, using Matlab's ICHOL function.

4.2. Solving internal linear systems

The internal linear systems of FDIPA-LP can be solved by various numerical methods, which depend on the form and characteristics that they obtain when manipulated.

From the equation (18) and (19) we get:

$$-A^T G^{-1}(\lambda) X A d_\alpha = -b, \tag{55}$$

$$x_\alpha = -G^{-1}(\lambda) X A d_\alpha. \tag{56}$$

From the system (20) and (21) we get,

$$-A^T G^{-1}(\lambda) X A d_\beta = A^T G^{-1}(\lambda) X, \tag{57}$$

$$x_\beta = -G^{-1}(\lambda)(x + XAd_\beta). \tag{58}$$

The matrix $Q = -A^T G^{-1}(\lambda) XA$ is guaranteed symmetric and positive definite for all $x \in int(\Omega_{\mathrm{p}})$. Therefore, SPCG method can be used to solve (55) and (57).

We observe, in the development of the FDIPA-LP algorithm, that the iterations generated by SPCG are candidates to be a feasible descent direction. Therefore, we stop the SPCG method at an early iteration (SPCG-T). It which represents an attractive and economic alternative for large-scale problems.

In Table 1, we will compare the number of iterations of SPCG (Iter SPCG) *vs* the number of SPCG with truncated iteration (Iter SPCG-T), needed to solve the internal FDIPA-LP systems for tested problems from the NETLIB [24] collection. The first column contains the name of the problem.

Noting that in both cases, we got the same number of FDIPA iterations to reach the target value. Then we see that with SPCG-T the number of iterations to solve the internal systems is reduced since we stop the process at the first iteration, which means that fewer numerical operations are performed to arrive at the target value, reducing the computational cost.

TABLE 1. Comparison of SPCG and SPCG-T iterations.

Problem	Iter SPCG	Iter SPCG-T
Adlittle	92	46
Afiro	38	20
Agg	259	92
Agg2	141	54
Agg3	164	86

TABLE 2. Number of variables and constraints before presolve *vs* after presolve.

	Problem		Problem with presolve	
Name	Variables	Constraints	Variables	Constraints
Adlittle	138	56	137	55
Afiro	51	27	51	27
Agg	615	488	478	390
Agg2	758	516	755	514
Agg3	758	516	755	514
Blend	114	74	111	71
Degen2	757	444	757	444
Degen3	2604	1503	2604	1503
Israel	316	174	316	174
Qap8	1632	912	1632	912
Qap12	8856	3192	8856	3192
Sc205	317	205	315	203
Sc50A	78	50	77	49
Sc50B	78	50	76	48
Scagr7	185	129	183	127
Scagr25	671	471	669	469
Scorpion	466	388	412	346
Sctap1	660	300	644	284
Sctap2	2500	1090	2443	1033
Sctap3	3340	1480	3268	1408
Share1b	253	117	248	112
Stocfor1	165	117	161	113

4.3. Presolve

The algorithm begins by trying to simplify the problem by eliminating redundancies and simplifying the restrictions. With this, we managed to reduce the size of the problem and the execution time, the presolve methods used are in [22].

In Table 2 we present some computational results clearly advocating for the use of an involved presolve analysis. The columns variables and constraints show the number of rows and columns in A, respectively. The following columns variables and constraints show the same numbers, but after presolve. The results collected in Table 2 also show that there exist almost irreductible problems, for example, Afiro, Degen2, Degen3, Israel, Qap8 and Qap12.

We also use stepping techniques since they improve the computational properties of LP problem. Scaling is used before the application of FDIPA-LP algorithm to reduce the condition number of the constraint matrix,

TABLE 3. Types of scaling techniques.

Number	Scaling techniques
1	Arithmetic mean
2	Buchet for the case $p = 1$
3	Buchet for the case $p = 2$
4	Buchet for the case $p = \infty$
5	Entropy
6	Equilibration

improve the performance number of the algorithms, reduce the number of iterations required to solve LP, and simplify the setting of tolerances.

In [22] present eleven scaling techniques used before the execution of an LP algorithm where a computational study is carried out, comparing the execution time of the scaling techniques, and investigate the impact of scaling before the application of LP algorithms. We will use 6 types of scaling techniques presented in Table 2.

For more details of its mathematical formulation, illu strative numerical example and implementation in MATLAB of each type of scaling technique see [22].

4.4. Starting point

In the algorithm, we have assumed that we start from a starting point λ^0 feasible and interior. This point will be found by solving the following problem:

$$\begin{aligned} &\underset{\lambda,z}{\text{minimize}} \quad z \\ &\text{subject to} \quad A\lambda - c \leq 1^T z, \end{aligned} \tag{59}$$

we obtain a feasible point when $z < 0$.

4.5. Stopping criteria

We must establish criteria to determine when the current point λ^k is close enough to λ^*.

Our criteria happens when the relative improvement in the objective function is small,

$$\frac{\left|b^T\lambda^k - b^T\lambda^{k+1}\right|}{1 + |b^T\lambda^k|} \leq \epsilon$$

and $\|d_\alpha\| \leq \epsilon$.

4.6. Numerical results

In this section, we report experimental results from NETLIB [24] collection. This library brings together LP problems from different areas.

Every primal linear programming problem has another associated problem called dual, where at the optimal point each has the same objective value. Duality theory relates both problems [12, 25]. The dual problem fits well to be solved with FDIPA-LP proposed in this article.

Our implementation was done in MATLAB environment. All computational experiments were performed on a microcomputer with an Intel CORE i5 processor, 8 GB of RAM and 2.40 GHz of frequency running on the Windows 10 platform.

The first step in solving a test problem was to perform a presolve described in Section 4.3, then apply a kind of scaling technique from Table 3. Next, FDIPA-LP needs as input a point λ^0 within the feasible region,

TABLE 4. Results for test instances.

Problem	Objective value	Objective value (dual) FDIPA	Iter FDIPA	Iter SPCG-T	Iter x^0	Scaling techniques
Adlittle	2.2549496316E+05	2.2549496316E+05	20	40	5	3
Afiro	−4.6475314286E+02	−4.6475314286E+02	11	22	1	6
Agg	−3.5991767287E+07	−3.5991767287E+07	39	78	4	2
Agg2	−2.0239252356E+07	−2.0239252356E+07	30	60	2	1
Agg3	1.0312115935E+07	1.0312115935E+07	37	54	1	1
Bandm	−1.5862801845E+02	−1.5862801845E+02	31	62	5	6
Beaconfd	3.3592485807E+04	3.3592485807E+04	24	48	2	6
Blend	−3.0812149846E+01	−3.0812149846E+01	15	30	5	2
Degen2	−1.4351780000E+03	−1.4351780000E+03	19	38	2	1
Degen3	−9.8729400000E+02	−9.8729400004E+02	27	54	7	1
Fffff800	5.5567961165E+05	5.5567956482E+05	79	158	22	2
Israel	−8.9664482186E+05	−8.9664482186E+05	23	46	6	6
Qap8	2.0350000000E+02	2.0349999989E+02	18	36	3	4
Qap12	5.2289435056E+02	5.2289435056E+02	34	68	2	1
Sc105	−5.2202061212E+01	−5.2202061212E+01	16	32	4	1
Sc205	−5.2202061212E+01	−5.2202061212E+01	18	36	9	1
Sc50A	−6.4575077059E+01	−6.4575077059E+01	13	26	3	1
Sc50B	−7.0000000000E+01	−70.000000000E+01	11	22	2	1
Scagr7	−2.3313892548E+06	−2.3313898243E+06	21	42	7	1
Scagr25	−1.4753433061E+07	−1.4753433061E+07	27	54	10	1
Scorpion	1.8781248227E+03	1.8781248227E+03	21	42	4	1
Scsd1	8.6666666743E+00	8.6666666743E+00	11	22	2	1
Scsd6	5.0500000078E+01	5.0500000075E+01	18	36	2	1
Scsd8	9.0499999993E+02	9.0499999993E+02	16	32	2	1
Sctap1	1.4122500000E+03	1.4122500000E+03	17	34	6	1
Sctap2	1.7248071429E+03	1.7248071429E+03	15	30	4	6

but the NETLIB library does not provide workable points for your problems. Therefore, a feasible solution is sought $\lambda^0 \in int(\Omega)$. To strictly feasible initial point, FDIPA-LP is started to solve the auxiliary problem (59), the optimization process is interrupted when the objective function reaches a negative value z^0.

In the Tables 4 and 5, presents 52 linear programming problems solved with FDIPA-LP. The seven columns give, respectively, the name of the problem, the primal objective value, the dual objective value with FDIPA, the number of iterations required for convergence, the sum of the number of SPCG Truncate iterations needed to solve two systems, the number of iterations required to calculate the starting point and type of scaling technique.

4.7. Computational results FDIPA LP *vs* FDIPA without perturbation

Herskovits [14] stated that original FDIPA solves LP problems, using only system (8). In other words, it was not necessary to carry out the perturbation (ρ) to the complementarity condition.

Then Tits and Zhou [21], develop unperturbed FDIPA (where direction is d_α) for LP, showing the global convergence and the rate of convergence. Already in Celis [26] some computational results are observed.

In theory, both algorithms work. let's compare the efficiency of both. Table 6 shows the obtained computational results. In the test, the following parameter values were assumed $\xi = 0.7,\ \varphi = 1,\ \mu = 10^{-3},\ \gamma = 0.9995,\ x^s = 10^{15}$.

The first column of Table 6 contains the name of the problem. The next two columns show the number of iterations that FDIPA-LP performs to obtain the solution, with its respective gap. Columns 4 and 5 contain the iteration number of the FDIPA solving the first unperturbed system with its respective gap.

Table 5. Results for test instances.

Problem	Objective value	Objective value(dual) FDIPA	Iter FDIPA	Iter CG-T	Iter x^0	Scaling techniques
Sctap3	1.4240000000E+03	1.4240000000E+03	19	38	4	1
Share1b	−7.6589318579E+04	−7.6589318579E+04	33	66	9	1
Share2b	−4.1573224074E+02	−4.1573224074E+02	21	42	2	6
Ship04l	1.7933245380E+06	1.7933245380E+06	38	76	5	4
Ship04s	−1.7987147004E+06	1.7987147004E+06	24	48	5	6
Stocfor1	−4.1131976219E+04	−4.1131976220E+04	14	28	17	6
Stocfor2	−3.9024408538E+04	−3.9024408538E+04	32	64	39	4
Bore3d	1.3730803942E+03	1.3730803942E+03	21	42	22	6
Ganges	−1.0958636356E+05	−1.0958638221E+05	20	40	7	8
Gfrd-pnc	6.9022359995E+06	6.9022359957e+06	25	50	13	5
Grow7	−4.7787811815E+07	−4.7787811815E+07	21	42	1	8
Grow22	−1.6083433648E+08	−1.6083433648E+08	25	50	2	6
Grow15	−1.0687094129E+08	−1.0687094129E+08	25	50	4	4
Capri	2.6900129138E+03	2.6900129127E+03	51	102	21	6
Etamacro	−7.5571521774E+02	−7.5571541174E+02	55	110	8	8
Fit1d	−9.1463780924E+03	−9.1463780924E+03	21	42	7	6
Fit1p	9.1463780924E+03	9.1463780920E+03	45	90	8	2
Kb2	−1.7499001299E+03	−1.7499001299E+03	20	40	2	6
Gfrd-pnc	6.9022359995E+06	6.9022359957E+06	25	50	13	1
Standgub	1.2576995000E+03	1.2576995000E+03	20	40	9	1
Standmps	1.4060175000E+03	1.4060175000E+03	45	90	9	6
Standata	1.2576995000E+03	1.2576994988E+03	21	42	9	1
Czprob	2.1851966989E+06	2.1851966989E+06	118	236	4	8
D6cube	3.1549166667E+02	3.1549166404E+02	28	56	2	0
Pilot4	−2.5811392641E+03	−2.5811393097E+03	70	140	9	8
Pilotnov	−4.4972761882E+03	−4.4972772003E+03	46	92	36	4

Table 6. Numerical tests iterations FDIPA.

Problem	Iter d	Gap d	Iter d_α	Gap d_α
Adlittle	23	0,00E+00	45	1,93E+01
Afiro	13	0,00E+00	88	2,27E− 04
Agg2	31	0,00E+00	58	1,50E − 02
Agg3	36	0,00E+00	721	4,71E+00
Blend	21	0,00E+00	63	8,06E − 06
Degen2	19	0,00E+00	18	4,83E − 01
Israel	32	8,29E − 03	34	1,59E − 02
Sc205	30	0,00E+00	235	9,35E − 05
Sc50A	13	0,00E+00	40	1,34E − 05
Sc50B	11	6,30E+02	42	1,50E − 05
Scagr7	22	5,78E − 01	111	6,10E − 01
Scagr25	32	9,00E − 03	55	9,00E − 03
Scorpion	34	3,00E − 07	82	2,72E − 02

5. Conclusion

In this paper we proposed a feasible direction algorithm for linear programming FDIPA-LP, inspired from iteration due to Herskovits. The method start by performing an iteration of Newton's method to KKT conditions, generating a system, then perturbs the complementarity conditions. The linear system solved at each iteration is identical to that of the primal-dual logarithmic barrier method [27,28]. However, FDIPA-LP is not a penalty or barrier method, the perturbs strategy of the new algorithm is drastically different.

Under assumptions a theoretical study proving convergence to a KKT point was presented. Also, important aspects for efficient computational implementations were considered, such as use conjugate gradient method for system internal solved, by stopping in an early stage to find the direction, also the use of presolver and some kind of scaling, which presents an advantage to reduce the computational cost.

This algorithm was tested on 52 test problems, guarantee the good development of the algorithms and it should be noted that the fact that all tests are solved with the same parameters shows that the algorithm is robust, strong and efficient compared with alternative implementations.

Acknowledgements. The author gratefully acknowledges the many helpful suggestions of Grigori Chapiro during the preparation of the paper. This work was supported by CNPq and COPPETEC.

References

[1] G.B. Dantzig, Maximization of a Linear Function of Variables Subject to Linear Inequalities. New York (1951).

[2] G.B. Dantzig, Linear Programming and Extensions. Princeton University (1963).

[3] J.V. Neumann, On a Maximization Problem. Manuscript. Institute for Advanced Studies, Princeton University, Princeton, NJ (1947).

[4] A. Hoffman, M. Mannos, D. Sokolowsky and N. Wiegmann, Computational experience in solving linear programs. *J. Soc. Ind. Appl. Math.* **1** (1953) 17–33.

[5] N. Karmarkar, A new polynomial-time algorithm for linear programming, in *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, ACM (1984) 302–311.

[6] I. Dikin, Iterative solution of problems of linear and quadratic programming, in *Soviet Mathematics Doklady.* Vol. 8 (1967) 674–675.

[7] E.R. Barnes, A variation on Karmarkar's algorithm for solving linear programming problems, *Math. Program.* **36** (1986) 174–182.

[8] T. Cavalier and A. Soyster, Some Computational Experience and a Modification of the Karmarkar Algorithm. Pennsylvania State Univ., College of Engineering, Department of Industrial and Management Systems Engineering (1985).

[9] C.C. Gonzaga, An algorithm for solving linear programming problems in $o(n^3l)$ operations, in *Progress in Mathematical Programming*, Springer (1989) 1–28.

[10] C.C. Gonzaga, Path-following methods for linear programming. *SIAM Rev.* **34** (1992) 167–224.

[11] Y. Ye, An $o(n^3l)$ potential reduction algorithm for linear programming. *Math. Program.*, **50** (1991) 239–258.

[12] S.J. Wright, Primal-dual Interior-Point Methods. Vol. 54, SIAM (1997).

[13] J.V. Robert, Linear Programming: Foundations and Extensions. Kluwer Academic, Boston (2001).

[14] J. Herskovits, Feasible direction interior-point technique for nonlinear optimization. *J. Optim. Theory Appl.* **99** (1998) 121–146.

[15] A.E. Gutierrez, S.R. Mazorche, J. Herskovits and G. Chapiro, An interior point algorithm for mixed complementarity nonlinear problems. *J. Optim. Theory Appl.* **175** (2017) 432–449.

[16] J.R. Roche, J. Herskovits, E. Bazán and A. Zúñiga, A feasible direction algorithm for general nonlinear semidefinite programming. *Struct. Multidiscip. Optim.* **55** (2017) 1261–1279.

[17] M. Aroztegui, J. Herskovits, J.R. Roche and E. Bazá, A feasible direction interior point algorithm for nonlinear semidefinite programming. *Struct. Multidiscip. Optim.* **50** (2014) 1019–1035.

[18] J. Herskovits, W.P. Freire, M. Tanaka Fo and A. Canelas, A feasible directions method for nonsmooth convex optimization. *Struct. Multidiscip. Optim.* **44** (2011) 363–377.

[19] J. Herskovits, A. Leontiev, G. Dias and G. Santos, Contact shape optimization: a bilevel programming approach. *Struct. Multidiscip. Optim.* **20** (2000) 214–221.

[20] R.H. Byrd, J.C. Gilbert and J. Nocedal, A trust region method based on interior point techniques for nonlinear programming. *Math. Program.* **89** (2000) 149–185.

[21] A.L. Tits and J.L. Zhou, A Simple, Quadratically Convergent Interior Point Algorithm for Linear Programming and Convex Quadratic Programming. Springer US, Boston, MA (1994) 411–427.

[22] N. Ploskas and N. Samaras, Linear Programming using MATLAB®. Vol. 127, Springer (2017).

[23] Y. Saad, Iterative Methods for Sparse Linear Systems. Vol. 82, SIAM (2003).
[24] D.M. Gay, Electronic mail distribution of linear programming test problems. *Mathematical Programming Society COAL Newsletter* **13** (1985) 10–12.
[25] N. Maculan and M.H.C. Fampa, Otimização Linear. EdUnB, Brasília (2006).
[26] A.M.V. Celis, *Algoritmo de ponto interior para programação linear baseado no fdipa*. Master's thesis, Universidade Federal do Rio de Janeiro (2018).
[27] N. Megiddo. Pathways to the optimal set in linear programming, in *Progress in Mathematical Programming*, Springer (1989) 131–158.
[28] M. Kojima, S. Mizuno and A. Yoshise, A Primal-Dual Interior-Point Method for Linear Programming, Progress in Mathematical Programming: Interior-point and Related Methods, Edited by N. Megiddo, Springer Verlag, New York, New York, (1989) 29–47.