American Journal of Operations Research, 2011, 1, 191-202
doi:10.4236/ajor.2011.14022 Published Online December 2011 (http://www.SciRP.org/journal/ajor)

Optimal Adjustment Algorithm for *p* Coordinates and the Starting Point in Interior Point Methods

Carla T. L. S. Ghidini[1], Aurelio R. L. Oliveira[1], Jair Silva[2]
[1]*Institute of Mathematics, Statistics and Scientific of Computation* (*IMECC*), *State University of Campinas* (*UNICAMP*), *São Paulo*, *Brazil*
[2]*Federal University of Mato Grosso do Sul* (*UFMS*), *Mato Grosso do Sul*, *Brazil*
E-mail: {*carla*, *aurelio*}*@ime.unicamp.br*, *jairmt@gmail.com*
Received September 15, 2011; *revised October* 16, 2011; *accepted October* 30, 2011

Abstract

Optimal adjustment algorithm for *p* coordinates is a generalization of the optimal pair adjustment algorithm for linear programming, which in turn is based on von Neumann's algorithm. Its main advantages are simplicity and quick progress in the early iterations. In this work, to accelerate the convergence of the interior point method, few iterations of this generalized algorithm are applied to the Mehrotra's heuristic, which determines the starting point for the interior point method in the PCx software. Computational experiments in a set of linear programming problems have shown that this approach reduces the total number of iterations and the running time for many of them, including large-scale ones.

Keywords: Von Neumann's Algorithm, Mehrotra's Heuristic, Interior Point Methods, Linear Programming

1. Introduction

In 1948, von Neumann proposed to Dantzig an algorithm for finding a feasible solution to a linear program with a convexity constraint recast in the form ([1,2]):

$$Px = 0,\ e^t x = 1,\quad x \geq 0 \qquad (1.1)$$

where $P \in \mathrm{R}^{m\times n}$, $x \in \mathrm{R}^n$, $e \in \mathrm{R}^n$ is the vector of all ones, and the columns of P have norm one, *i.e.*, $\|P_j\| = 1$, for $j = 1, \cdots, n$.

This algorithm, which was later studied by Epelman and Freund [3], has interesting properties such as simplicity and fast initial advance. However, it is not very practical for solving linear problems because its convergence rate is slow.

Gonçalves [4] presented four new algorithms based on von Neumann's algorithm and among them the optimal pair adjustment algorithm has the best performance in practice.

The optimal pair adjustment algorithm inherits the good properties of von Neumann's algorithm. Although it is proved in [5] that this algorithm is faster than von Neumann's algorithm, nevertheless, it is impractical to solve linear programming problems, because its convergence rate is also very slow.

In [6], the idea presented by Gonçalves, Storer and Gondzio [5] to develop the optimal pair adjustment algorithm was generalized for *p* coordinates and then the optimal adjustment algorithm for *p* coordinates arose. The value of *p* is bounded by the number of variables of the problem.

The new algorithm is not suitable to solve linear programming problems until achieving optimality. Thus, the idea is to exploit its simplicity and quick initial progress during the early iterations and to use it together with an interior point method to accelerate convergence.

Knowing that the starting point greatly influences the performance of the interior point method, in this work the optimal adjustment algorithm for *p* coordinates is applied within the Mehrotra's heuristic [7], which determines the starting point for the method interior points in the PCx software, so that the starting points can be obtained even better. This approach is different to the classical warm-starting approach, which uses a known solution for some problem (e.g., relaxation in the column-generation) to define a new starting point for a closely related problem or perturbed problem instance ([8-10]).

The paper is organized as follows. In Section 2, the definition of the problem is given and von Neumann's algorithm is described. In Section 3, the optimal pair adjustment algorithm is recalled. In Section 4, the optimal adjustment algorithm for *p* coordinates is proposed

and some theoretical results are described. Section 5 discusses warm-starting in interior point methods. Numerical experiments are shown in Section 6. In Section 7, the conclusions are drawn and future perspectives are suggested.

2. Von Neumann's Algorithm

Considering the problem of finding a feasible solution of the set of linear constraints (1.1), geometrically, the columns P_j can be viewed as points lying on the m-dimensional hypersphere with unit radius and center at the origin. This problem can be described as assigning non-negative weights x_j to columns P_j in such a way that after being re-scaled its center of gravity is the origin.

First, the algorithm finds the column P_s of P which makes the largest angle with the residue $b^{k-1} = Px^{k-1}$ and then the next residual b^k is given by projecting the origin on the line segment connecting b^{k-1} to P_s. See **Figure 1**.

VON NEUMANN'S ALGORITHM

Given: $x^0 \geq 0$, with $e^t x^0 = 1$. Compute $b^0 = Px^0$.

$k = 1$

Do

{

1. Compute:

$s = \arg\min_{j=1,\cdots,n} \left\{ P_j^t b^{k-1} \right\}$.

$v^{k-1} = P_s^t b^{k-1}$

2. If $v^{k-1} > 0$, then STOP. The problem (1.1) is infeasible.

3. Compute:

$u^{k-1} = \left\| b^{k-1} \right\|$,

$\lambda = \dfrac{1 - v^{k-1}}{\left(u^{k-1}\right)^2 - 2v^{k-1} + 1}$.

4. Update:

$b^k = \lambda b^{k-1} + (1-\lambda) P_s$,

$x^k = \lambda x^{k-1} + (1-\lambda) e_s$.

where e_s is the vector of the canonical basis with 1 in position s.

5. Stopping Criterion:

$E^k = \left\| b^k - b^{k-1} \right\| / \left\| b^k \right\|$

$k = k + 1$.

} while $(E^k > \varepsilon)$},

where ε is a predetermined percentage.

Note that $b^k = Px^k$, for all $k \geq 0$. The initial approximation x^0 is arbitrary, since $e^t x^0 = 1$, then $x_j = 1/n$ for $j = 1, \cdots, n$ was used. For any given iteration k column P_s is the column which makes the largest angle with the vector b^{k-1}. Furthermore, if $v^{k-1} = P_s^t b^{k-1} \neq 0$, then $0 < 1 - v^{k-1} < (u^{k-1})^2 - 2v^{k-1} + 1$ thus, $0 < \lambda < 1$. Moreover, the new residual is smaller than the previous one, *i.e.*, $u^k < u^{k-1}$, as can easily be seen in **Figure 1**, the triangle $0b^{k-1}b^k$ has hypotenuse $u^{k-1} = 0b^{k-1}$ and side $u^k = 0b^k$.

The effort per iteration of von Neumann's algorithm is dominated by matrix-vector multiplication needed when selecting column P_s which is $O(nz(P))$ where $nz(P)$ is the number of the entries of P. This effort can be reduced significantly as the matrix P is sparse. For more details of this algorithm see [11,12].

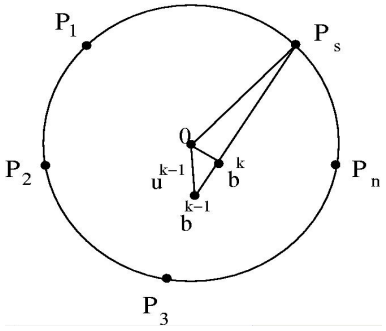

Figure 1. Illustration of von Neumann's algorithm.

3. Optimal Pair Adjustment Algorithm

In his PhD thesis [4], Gonçalves studied von Neumann's algorithm and introduced four new algorithms based on it. Among them, emphasis is given to the weight-reduction algorithm and the optimal pair adjustment algorithm. The optimal pair adjustment algorithm was the one that performed better in practice.

The weight-reduction algorithm was proposed as an attempt to improve the efficiency of von Neumann's algorithm. It is based on the idea that the residual b^{k-1} can be moved closer to origin 0, increasing the weight x_j for a given column P_j and decreasing the weight x_i for another column P_i.

In particular, it is expected that the new residual b^k is closer to origin 0 than the residual b^{k-1} if the weight in column P_{s+} is increased when P_{s+} has the largest angle with the residual b^{k-1} and the weight in column P_{s-} is decreased when P_{s+} has the smallest angle with the residual b^{k-1}.

This corresponds to making residual b^{k-1} move in the direction $P_{s+} - P_{s-}$. The new residual b^k is the point that minimizes the distance from origin 0 to this line.

Notice that the minimization of this distance is subject to the maximum possible decrease of x_{s-}. Since $x_j \geq 0$ for all j, then x_{s-} can be decreased until it vanishes. **Figure 2** is a geometric illustration of how the algorithm works by iteration.

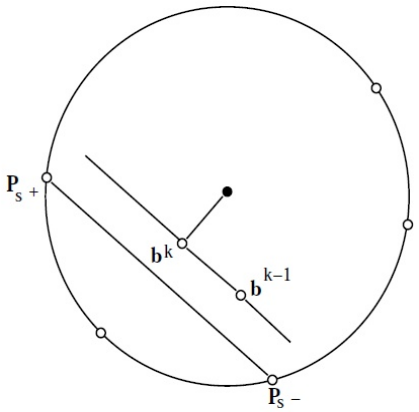

Figure 2. Illustration of weight-reduction algorithm.

3.1. Optimal Pair Adjustment Algorithm

The optimal pair adjustment algorithm is a generalization of the weight-reduction algorithm designed to give the maximum possible freedom to two of the weights x_j (see [5]). In a way, we can say that it prioritizes only two variables by iteration because it finds the optimal value for two coordinates and adjusts the remaining coordinates according to these values.

Similar to the weight-reduction algorithm, the optimal pair adjustment algorithm starts by identifying vectors P_{s+} and P_{s-}, which make the largest and smallest angles with the vector b^{k-1}. Afterwards, values x_{s+}^k, x_{s-}^k, λ are found, where $x_j^k = \lambda x_j^{k-1}$ for all $j \neq s+$ and $j \neq s-$. These values minimize the distance between b^k and the origin while satisfying the convexity and the non-negativity constraints. The solution of this optimization problem is easily computed examining the Karush-Kuhn-Tucher conditions (KKT).

The main difference between the weight-reduction algorithm and the optimal pair adjustment algorithm is that only the weights of P_{s+} and P_{s-} are changed in the first algorithm while in the second algorithm all other weights are also changed.

OPTIMAL PAIR ADJUSTMENT ALGORITHM

Given: $x^0 \geq 0$, with $e^t x^0 = 1$. Compute $b^0 = Px^0$.

$k = 1$

Do

{

1. Compute:

$s^+ = \arg\min_{j=1,\ldots,n} \left\{ P_j^t b^{k-1} \right\}$,

$s^- = \arg\max_{j=1,\ldots,n} \left\{ P_j^t b^{k-1} / x_j > 0 \right\}$,

$v^{k-1} = P_{s+}^t b^{k-1}$

2. If $v^{k-1} > 0$, then STOP. The problem (1.1) is infeasible.

3. Solve the problem:

$$\min \left\| \bar{b} \right\|^2$$

$$\text{s.t.} \quad \lambda_0 \left(1 - x_{s+}^{k-1} - x_{s-}^{k-1}\right) + \lambda_1 + \lambda_2 = 1, \qquad (3.2)$$

$$\lambda_i \geq 0, \text{ for } i = 0,1,2.$$

where

$$\bar{b} = \lambda_0 \left(b^{k-1} - x_{s+}^{k-1} P_{s+} - x_{s-}^{k-1} P_{s-}\right) + \lambda_1 P_{s+} + \lambda_2 P_{s-}.$$

4. Update:

$$b^k = \lambda_0 \left(b^{k-1} - x_{s+}^{k-1} P_{s+} - x_{s-}^{k-1} P_{s-}\right) + \lambda_1 P_{s+} + \lambda_2 P_{s-},$$

$$u^k = \left\| b^k \right\|,$$

$$x^k = \begin{cases} \lambda_0 x_j^{k-1}, & j \neq s^+ \text{ and } j \neq s^- \\ \lambda_1, & j = s^+ \\ \lambda_2, & j = s^- \end{cases}$$

5. Stopping Criterion:

$E^k = \left\| b^k - b^{k-1} \right\| / \left\| b^k \right\|$

$k = k + 1$.

} while $(E^k > \varepsilon)$}.

In iteration k, column P_{s+} is the column which makes the largest angle with the vector b^{k-1} and column P_{s-} is the column which makes the smallest angle with the vector b^{k-1} such that $x_s > 0$.

3.2. Subproblem Solution

In order to solve subproblem (3.2), first the subproblem is rewritten as:

$$\min \left\| \bar{b} \right\|^2$$

$$\text{s.t.} \quad 1 - \lambda_1 - \lambda_2 \geq 0, \qquad (3.3)$$

$$\lambda_i \geq 0, \text{ for } i = 0,1,2.$$

where,

$$\bar{b} = \frac{1 - \lambda_1 - \lambda_2}{\left(1 - x_{s+}^{k-1} - x_{s-}^{k-1}\right)} \left(b^{k-1} - x_{s+}^{k-1} P_{s+} - x_{s-}^{k-1} P_{s-}\right) + \lambda_1 P_{s+} + \lambda_2 P_{s-}.$$

Define:

$g_0(\lambda_1, \lambda_2) = \lambda_1 + \lambda_2 - 1$, $g_1(\lambda_1, \lambda_2) = -\lambda_1$, $g_2(\lambda_1, \lambda_2) = -\lambda_2$.

Denote the objective function by $f(\lambda_1, \lambda_2)$ and let $\left(\bar{\lambda}_1, \bar{\lambda}_2\right)$ be a feasible solution.

Considering the convexity of the objective function and constraints, if $\left(\bar{\lambda}_1, \bar{\lambda}_2\right)$ is an optimal local solution then it is also an optimal global solution. Thus, $\left(\bar{\lambda}_1, \bar{\lambda}_2\right)$ met the KKT constraints given by:

$$\nabla f\left(\lambda_1, \lambda_2\right) + \sum_{i=0}^{2} \mu_i \nabla g_i\left(\lambda_1, \lambda_2\right) = 0.$$

$$\begin{aligned} &g_i(\lambda_1,\lambda_2)\le 0, && \text{for } i=0,1,2.\\ &\mu_i g_i(\lambda_1,\lambda_2)=0, && \text{for } i=0,1,2.\\ &\mu_i \ge 0, && \text{for } i=0,1,2. \end{aligned}$$

where μ_i are Lagrange multipliers.

Problem (3.3) is solved by selecting a feasible solution among all possibilities that meet the KKT conditions. This is done by analyzing the following cases:

a) $\lambda_1 = \lambda_2 = 0$;
b) $\lambda_1 = 0$; $0 < \lambda_2 < 1$;
c) $\lambda_1 = 0$; $\lambda_2 = 1$;
d) $0 < \lambda_1 < 1$; $\lambda_2 = 0$;
e) $0 < \lambda_1 < 1$; $0 < \lambda_2 < 1$; $\lambda_1 + \lambda_2 - 1 = 0$;
f) $\lambda_1 = 1$; $\lambda_2 = 0$;
g) $0 < \lambda_1 < 1$; $0 < \lambda_2 < 1$; $\lambda_1 + \lambda_2 - 1 = 0$;

For each case above, the known values are replaced in the KKT equations and the resulting linear system is solved.

4. Optimal Adjustment Algorithm for *p* Coordinates

The optimal adjustment algorithm for p coordinates is developed generalizing the ideas used in [5] for the optimal pair adjustment algorithm. Instead of only two columns to be used to formulate the problem, any amount of columns can be used and thus, importance will be given to any desired amount of variables.

The strategy to prioritize the variables is free. In this work, we chose to take $p/2$ columns making the largest angle with the vector b^k and the remaining $p/2$ columns that make the smallest angle with the vector b^k. If p is odd then one more column is added to the set of vectors that make the largest angle with the vector b^k, for example.

The structure of the optimal adjustment algorithm for p coordinates is similar to the optimal pair adjustment algorithm. It begins by identifying the s_1 columns that make the largest angle with the vector b^{k-1}, then s_2 columns that make the smallest angle with the vector b^{k-1} are determined, where $s_1 + s_2 = p$ and p is the number of columns that are prioritized. Next, the optimization subproblem is solved and the residual and the current point are updated.

OPTIMAL ADJUSTMENT ALGORITHM FOR p COORDINATES

Given: $x^0 \ge 0$, with $e^t x^0 = 1$. Compute $b^0 = Px^0$.
$k = 1$
Do
{
1. Compute:

$\left\{P_{\eta_1^+},\cdots,P_{\eta_{s_1}^+}\right\}$, which make the largest angle with the vetor b^{k-1}.

$\left\{P_{\eta_1^-},\cdots,P_{\eta_{s_1}^-}\right\}$, which make the smallest angle with the vector b^{k-1} and such that $x_i^{k-1} > 0$, where $s_1 + s_2 = p$.

$v^{k-1} = \text{minimum}_{i=1,\cdots,s_1}\left\{P_{\eta_i^+}^t b^{k-1}\right\}$

2. If $v^{k-1} > 0$, then STOP. The problem (1.1) is infeasible.

3. Solve the problem:

$$\begin{aligned} &\min \left\|\bar{b}\right\|^2\\ &\text{s.t. } \lambda_0\left(1-\sum_{i=1}^{s_1}x_{\eta_i^+}^{k-1}-\sum_{j=1}^{s_2}x_{\eta_j^-}^{k-1}\right)+\sum_{i=1}^{s_1}\lambda_{\eta_i^+}+\sum_{j=1}^{s_2}\lambda_{\eta_j^-}=1,\\ &\lambda_0 \ge 0,\\ &\lambda_{\eta_i^+} \ge 0, \text{ for } i=1,\cdots,s_1,\\ &\lambda_{\eta_j^-} \ge 0, \text{ for } j=1,\cdots,s_2. \end{aligned} \tag{4.4}$$

where

$$\bar{b} = \lambda_0\left(b^{k-1}-\sum_{i=1}^{s_1}x_{\eta_i^+}^{k-1}P_{\eta_i^+}-\sum_{j=1}^{s_2}x_{\eta_j^-}^{k-1}P_{\eta_j^-}\right)+\sum_{i=1}^{s_1}\lambda_{\eta_i^+}P_{\eta_i^+} + \sum_{j=1}^{s_2}\lambda_{\eta_j^-}P_{\eta_j^-}$$

4. Update:

$$b^k = \lambda_0\left(b^{k-1}-\sum_{i=1}^{s_1}x_{\eta_i^+}^{k-1}P_{\eta_i^+}-\sum_{j=1}^{s_2}x_{\eta_j^-}^{k-1}P_{\eta_j^-}\right)+\sum_{i=1}^{s_1}\lambda_{\eta_i^+}P_{\eta_i^+} + \sum_{j=1}^{s_2}\lambda_{\eta_j^-}P_{\eta_j^-},$$

$$u^k = \left\|b^k\right\|,$$

$$x_j^k = \begin{cases} \lambda_0 x_j^{k-1}, & j \notin \left\{\eta_1^+,\cdots,\eta_{s_1}^+,\eta_1^-,\cdots,\eta_{s_2}^-\right\},\\ \lambda_{\eta_i^+}, & j=\eta_i^+;\ i=1,\cdots,s_1,\\ \lambda_{\eta_j^-}, & j=\eta_j^-;\ j=1,\cdots,s_2. \end{cases}$$

5. Stopping Criterion:

$E^k = \left\|b^k - b^{k-1}\right\| / \left\|b^k\right\|$

$k = k + 1$.
} while $(E^k > \varepsilon)$}.

In Step 2 of the algorithm, if $v^{k-1} > 0$ then all columns P_j of matrix P are on the same side of the hyper-plane through the origin and perpendicular to direction b^{k-1}. This means that any convex combination of the columns P_j resulting in the origin can be found. In this case, the

problem is infeasible.

4.1. Subproblem Solution Using Interior Point Methods

For each iteration of the optimal adjustment algorithm for p coordinates, it is necessary to solve subproblem (4.4). This subproblem is solved finding a solution in the positive orthant, subject to a linear equation system of order $(p + 1)$. A way to solve it is to verify all possible feasible solutions as in the pair adjustment algorithm. The drawback that comes naturally in solving the subproblem in this way is that the number of possible cases to be verified grows exponentially with the value of p as shown next.

In fact, to solve the subproblem (4.4) first the variable λ_0, as defined in (4.5), is removed from the problem.

$$\lambda_0 = \frac{1 - \sum_{i=1}^{s_1} \lambda_{\eta_i^+} - \sum_{j=1}^{s_2} \lambda_{\eta_j^-}}{1 - \sum_{i=1}^{s_1} x_{\eta_i^+}^{k-1} - \sum_{j=1}^{s_2} x_{\eta_j^-}^{k-1}} \tag{4.5}$$

Thus, the subproblem (4.4) was rewritten as:

$$\begin{aligned} &\min \left\| \bar{b} \right\|^2 \\ &\text{s.t.} \quad 1 - \sum_{i=1}^{s_1} \lambda_{\eta_i^+} - \sum_{j=1}^{s_2} \lambda_{\eta_j^-} \geq 0, \\ &\qquad \lambda_{\eta_i^+} \geq 0, \text{ for } i = 1, \cdots, s_1, \\ &\qquad \lambda_{\eta_j^-} \geq 0, \text{ for } j = 1, \cdots, s_2. \end{aligned} \tag{4.6}$$

where

$$\bar{b} = \lambda_0 \left(b^{k-1} - \sum_{i=1}^{s_1} x_{\eta_i^+}^{k-1} P_{\eta_i^+} - \sum_{j=1}^{s_2} x_{\eta_j^-}^{k-1} P_{\eta_j^-} \right) + \sum_{i=1}^{s_1} \lambda_{\eta_i^+} P_{\eta_i^+} + \sum_{j=1}^{s_2} \lambda_{\eta_j^-} P_{\eta_j^-}$$

Defining:

$$g_0 \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = \sum_{i=1}^{s_1} \lambda_{\eta_i^+} + \sum_{j=1}^{s_2} \lambda_{\eta_j^-} - 1$$

$$g_i \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = -\lambda_{\eta_i^+}, i = 1, \cdots, s_1$$

$$h_j \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = -\lambda_{\eta_j^-}, j = 1, \cdots, s_2$$

and denoting the objective function by $f\left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right)$, the KKT conditions of the subproblem are given by:

$$\begin{cases} \nabla f \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) \\ + \sum_{i=0}^{s_1} \mu_{g_i} \nabla g_i \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) \\ + \sum_{j=0}^{s_2} \mu_{h_j} \nabla h_j \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = 0, \\ \\ \mu_{g_i} g_i \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = 0, \text{ for } i = 0, \cdots, s_1, \\ \mu_{h_j} h_j \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) = 0, \text{ for } j = 1, \cdots, s_2, \\ \mu_{g_i} \geq 0, \text{ for } i = 0, \cdots, s_1, \\ \mu_{h_j} \geq 0, \text{ for } j = 1, \cdots, s_2, \\ g_i \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) \leq 0, \text{ for } i = 0, \cdots, s_1, \\ h_j \left(\lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s_1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s_2}^-} \right) \leq 0, \text{ for } j = 1, \cdots, s_2. \end{cases} \tag{4.7}$$

Then, the subproblem (4.6) is solved by selecting a feasible solution among all the possibilities that satisfy the KKT conditions (4.7). For this, all possible cases of possible values for the variables $\lambda_{\eta_i^+}$, $i = 1, \cdots, s_1$, and $\lambda_{\eta_j^-}$, $j = 1, \cdots, s_2$, must be analyzed. The cases to be considered are:

- Case: $\lambda_{\eta_i^+} = 0$, $i = 1, \cdots, s_1$ and $\lambda_{\eta_j^-} = 0$, $j = 1, \cdots, s_2$,
- Case: one $\lambda_k \neq 0$, with $k \notin \left\{ \eta_1^+, \cdots, \eta_{s_1}^+, \eta_1^-, \cdots, \eta_{s_2}^- \right\}$ and others equal to zero, which gives a combination of C_1^p cases, but again we have to consider the case where $\lambda_k = 1$ and $\lambda_k \neq 1$. Thus, we have $2C_1^p$ possibilities.
- Case: $\lambda_i \neq 0$ and $\lambda_j \neq 0$, and $i, j \in \left\{ \eta_1^+, \cdots, \eta_{s_1}^+, \eta_1^-, \cdots, \eta_{s_2}^- \right\}$ and others equal to zero, which gives a combination of C_2^p cases, but again we have to consider the case where $\lambda_i + \lambda_j = 1$ and $\lambda_i + \lambda_j \neq 1$. Thus, we have $2C_2^p$ possibilities.

The total number of cases when p coordinates are modified is:

$$1 + 2C_1^p + 2C_2^p + 2C_3^p + \cdots + 2C_p^p = 2^{(p+1)} - 1$$

Therefore, this strategy is inefficient even for values of p which are not too large.

In order to deal with such a difficulty, the subproblem (4.4) is approached differently and solved using interior point methods. This is done as follows:

First, the subproblem is rewritten in matrix form:

$$\begin{aligned} &\min \frac{1}{2}\|W\lambda\|^2 \\ &\text{s.t.} \quad a^t\lambda = 1, \\ &\qquad \lambda \geq 0, \end{aligned} \tag{4.8}$$

where

$$\begin{aligned} &W = \left[\, \overline{w} P_{\eta_1^+} \cdots P_{\eta_{s1}^+} P_{\eta_1^-} \cdots P_{\eta_{s2}^-} \,\right], \\ &\overline{w} = b^{k-1} - \sum_{i=1}^{s_1} x_{\eta_i^+}^{k-1} P_{\eta_i^+} - \sum_{j=1}^{s_2} x_{\eta_j^-}^{k-1} P_{\eta_j^-}, \\ &\lambda^t = \left(\lambda_0, \lambda_{\eta_1^+}, \cdots, \lambda_{\eta_{s1}^+}, \lambda_{\eta_1^-}, \cdots, \lambda_{\eta_{s2}^-} \right), \\ &a = \left(a_1 1 \cdots 1\right)^t, a_1 = 1 - \sum_{i=1}^{s_1} x_{\eta_i^+}^{k-1} - \sum_{j=1}^{s_2} x_{\eta_j^-}^{k-1} \end{aligned} \tag{4.9}$$

The KKT equations of problem (4.8) are given by:

$$W^tW\lambda + a\gamma - \mu = 0,\ \mu^t\lambda = 0,\ a^t\lambda - 1 = 0,\ \lambda \geq 0,\ \mu \geq 0. \tag{4.10}$$

where, γ and μ are Lagrange multipliers of equality and inequality constraints respectively and $(p + 1) \times (p + 1)$ is the dimension of the matrix W^tW.

Now the interior point method is applied to those equations.

The linear system arising at each iteration of the interior point method applied to (4.10) are:

$$\begin{bmatrix} W^tW & a & -I \\ U & 0 & \Lambda \\ a^t & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{d}\lambda \\ \mathrm{d}\gamma \\ \mathrm{d}\mu \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \tag{4.11}$$

where

$$U = diag(\mu),\ \ \Lambda = diag(\lambda),$$

$$r_1 = \mu - a\gamma - W^tW\lambda,\ \ r_2 = -\mu^t\lambda,\ \ r_3 = 1 - a^t\lambda.$$

By performing some algebraic manipulations, the directions $\mathrm{d}\mu$, $\mathrm{d}\lambda$ and $\mathrm{d}\gamma$ are given by:

$$\begin{aligned} &\mathrm{d}\mu = \Lambda^{-1}r_2 - \Lambda^{-1}U\mathrm{d}\lambda, \\ &\mathrm{d}\lambda = \left(W^tW + \Lambda^{-1}U\right)^{-1} r_4 - \left(W^tW + \Lambda^{-1}U\right)^{-1} a\mathrm{d}\gamma, \\ &a^t\left(W^tW + \Lambda^{-1}U\right)^{-1} a\mathrm{d}\gamma = a^t\left(W^tW + \Lambda^{-1}U\right)^{-1} r_4 - r_3, \end{aligned} \tag{4.12}$$

where $r_4 = r_1 + \Lambda^{-1}r_2$.

Defining $sl1 = (W^tW + \Lambda^{-1}U)^{-1}\, a$ and $sl2 = (W^tW + \Lambda^{-1}U)^{-1}r_4$, to compute the directions the following linear systems must be solved:

$$\left(W^tW + \Lambda^{-1}U\right)sl1 = a,\ \ \left(W^tW + \Lambda^{-1}U\right)sl2 = r_4 \tag{4.13}$$

Note that $(W^tW + \Lambda^{-1}U)$ is a positive definite matrix of order $p + 1$ and both systems can be solved using the same factorization.

4.2. Theoretical Properties of the Optimal Adjustment Algorithm for *p* Coordinates

Theorems 1 and 2, described below and proved in [6], ensure that the optimal adjustment algorithm for p coordinates converge in the worst case with the same convergence rate of von Neumann's algorithm and that, from the theoretical point of view, for larger values of p the algorithm is more robust and will perform better.

Theorem 1 *The decrease in* $\left\|b^k\right\|$ *obtained by an iteration of the optimal adjustment algorithm for p coordinates*, *with* $1 \leq p \leq n$, *where n is the column number of matrix P*, *in the worst case*, *is equal to the decrease obtained by an iteration of von Neumann's algorithm.*

Theorem 2 *The decrease in* $\left\|b^k\right\|$ *obtained by an iteration of the optimal adjustment algorithm for* p_2 *coordinates*, *in the worst case is equal to that obtained by an iteration of the optimal adjustment algorithm for* p_1 *coordinates with* $p_1 \leq p_2 \leq n$, *where n is the P column number.*

On the other hand, it is not advisable to choose a very large value of p since there is a cost of building and updating matrix W in (4.13). Such a cost is negligible for small values of p. However, it becomes noticeable for larger values.

5. Mehrotra's Heuristic and Starting Point in Interior Point Methods

It is well known that the starting point can influence the performance of interior point methods. On the other hand, as the optimal adjustment algorithm for p coordinates has a small computational cost per iteration, a natural idea is to use this algorithm to find a good starting point for interior point methods or improve the current one. In this work, few iterations of the optimal adjustment algorithm for p coordinates are performed to improve the starting point introduced by Mehrotra's heuristic [7]. This heuristic consists of the following steps:

1) Use least squares to compute the points:

$$\tilde{y} = \left(AA^t\right)^{-1} Ac,\ \ \tilde{z} = c - A^t\tilde{y},\ \ \tilde{x} = A^t\left(AA^t\right)^{-1} b$$

where x, y and z are the primal variables, dual variables and dual slacks respectively.

2) Find values δx and δz such that $\tilde{x}+\delta_x$ and $\tilde{z}+\delta_z$ are non-negative:

$$\delta_x = \max\left(-1.5\min\{\tilde{x}_i\}, 0\right),\ \ \delta_z = \max\left(-1.5\min\{\tilde{z}_i\}, 0\right).$$

3) Determine $\tilde{\delta}_x$ and $\tilde{\delta}_z$ such that the points x^0 and

z^0 are centralized:

$$\tilde{\delta}_x = \delta_x + \frac{\left(\tilde{x} + \delta_x e\right)^t \left(\tilde{z} + \delta_z e\right)}{2\sum_{i=1}^{n}\left(\tilde{z}_i + \delta_z\right)},$$

$$\tilde{\delta}_z = \delta_z + \frac{\left(\tilde{x} + \delta_x e\right)^t \left(\tilde{z} + \delta_z e\right)}{2\sum_{i=1}^{n}\left(\tilde{x}_i + \delta_x\right)}.$$

4) Compute the starting points as follows:

$$y^0 = \tilde{y}, \quad z^0 = \tilde{z} + \tilde{\delta}_z e, \quad x^0 = \tilde{x} + \tilde{\delta}_x e.$$

Remark 5.1 *Any linear programming problem can be reduced to problem* (1.1)*. For details of this transformation see* [4].

Usually, to improve the performance of interior point methods, modifications are introduced after Step 4 of the algorithm described above, unlike our proposal, which consists of performing a few iterations of the optimal adjustment algorithm for *p* coordinates before centralizing, *i.e.*, before Step 2. One explanation for this is that if the algorithm is then used after to centralize the points, these are improved. However, the centrality, which is important for the interior point methods, may be lost.

We have adapted the optimal adjustment algorithm for *p* coordinates in the PCx code [13] aiming to improve the starting point given by Mehrotra [7].

The starting point for the optimal adjustment algorithm for *p* coordinates is determined by solving the least squares in Step 1.

6. Computational Experiments

In the following experiments, all testing was performed on an Intel Core 2 Duo T7250, 2 GB RAM, 2 GHz and 250 GB hd and Ubuntu 32-bit operating system. We used 76 test problems to compare the performance of the original PCx and the modified PCx (PCxMod), which uses the optimal adjustment algorithm for *p* coordinates as a warm-starting approach. Most tested problems have free access such as NETLIB problems, QAPLIB problems and KENNINGTON problems ([14,15]). Other problems, which are not available publicly, were obtained from Gonçalves [4]. The test problems are presented in **Table 1**.

Table 1. Test problems.

Problem	Rows	Columns	Problem	Rows	Columns	Problem	Rows	Columns
	NETLIB			**Miscellaneous**			**Gonçalves**	
80bau3b	2140	11066	**nug06**	372	486	**bl**	5729	12462
agg2	514	750	**nug07**	602	931	**bl2**	5729	12462
agg3	514	750	**nug08**	912	1632	**co5**	4849	10787
czprob	671	2779	**cre-a**	2994	6692	**co9**	9090	19997
degen3	1503	2604	**cre-b**	5336	36382	**cq9**	8004	19317
dfl001	5984	12143	**cre-c**	2375	5412	**ex01**	235	1556
etamacro	334	669	**cre-d**	4102	28601	**ex02**	227	1548
fffff800	322	826	**ken-11**	10085	16740	**ex05**	832	7805
fit2d	25	10524	**ken-13**	22534	36561	**ex06**	825	7797
fit2p	3000	13525	**ken-18**	78862	128434	**ex09**	1821	18184
maros-r7	2152	7440	**osa-07**	1081	25030	**fort45**	1152	1582
modszk1	665	1599	**osa-14**	2300	54760	**fort47**	1152	1582
perold	593	1389	**osa-30**	4313	104337	**fort48**	1152	1582
pilot	1368	4543	**osa-60**	10243	243209	**fort53**	1156	1586
pilot87	1971	6373	**els19**	4350	13186	**fort56**	1156	1586
pilotwe	701	2814	**chr25a**	8149	15325	**fort58**	1156	1586
scfxm3	915	1704	**chr22b**	5587	10417	**fort59**	1156	1586
seba	448	901	**scr15**	2234	6210	**fort60**	1156	1586
ship08l	470	3121	**scr20**	5079	15980	**fort61**	1156	1586
stocfor3	15362	22228	**rou20**	7359	37640	**ge**	9150	14990
wood1p	171	1718	**pds-02**	2609	7339	**nl**	6665	14680
			pds-20	32287	106080	**x1**	1050	1480
			pds-30	47968	156042	**x2**	1050	1480
			pds-40	64276	214385			
			pds-50	80339	272513			
			pds-60	96514	332862			
			pds-70	111896	386238			
			pds-80	126120	430800			
			pds-90	139752	471538			
			pds-100	152300	498530			

6.1. Choice of *p*

For the optimal adjustment algorithm for p coordinates to work properly an appropriate choice of parameter p is essential. Thus, several computational experiments were done to determine a heuristic that works well in any linear programming problem. With results obtained in the tests, it became clear that the value of p must be chosen depending on the size of the problem. Recalling that m is the number of rows and n is the number of columns of the linear problem constraint matrix, the heuristic that showed better results is as follows:

0	$<$	$(m+n)$	$\leq$	10,000	$\rightarrow$	$p=4$;
10,000	$<$	$(m+n)$	$\leq$	20,000	$\rightarrow$	$p=8$;
20,000	$<$	$(m+n)$	$\leq$	400,000	$\rightarrow$	$p=20$;
400,000	$<$	$(m+n)$	$\leq$	600,000	$\rightarrow$	$p=40$;
600,000	$<$	$(m+n)$			$\rightarrow$	$p=80$.

6.2. Stopping Criterion

The number of iterations of the optimal adjustment algorithm for p coordinates to be performed is an important parameter to be determined, since it directly influences the performance of the PCx. This algorithm achieves better results when the solution is determined with good accuracy. However, in some cases, the number of iterations needed for convergence of the algorithm can be very large, making it impractical to use. In these cases, a maximum number of iterations should be adopted. Therefore, the stopping criterion for the optimal adjustment algorithm for p coordinates used is the following: maximum number of iterations (100) or the relative error of the residual norm smaller than a given tolerance (10^{-4}) (the one that occurs first).

6.3. Results

The performance of the PCx and PCxMod was compared with respect to the total number of iterations and the run time. The results are presented in **Table 2**.

The total number of interior point iterations (Column Iterations) and the total running time in seconds (Column Time) of two versions of the PCx are showed in **Table 2**. The PCxMod is the version that uses the optimal adjustment algorithm for p coordinates in Mehrotra's heuristic and PCx is the one that does not adopt it. Column p shows the value of p used by the optimal adjustment algorithm. Column ItAux gives the number of iterations performed by the optimal adjustment algorithm for p coordinates.

According to the results, the PCxMod takes less time to obtain the optimal solution in 55.3% of the tested problems and the PCx in 34.2% of the problems. Moreover, the total number of iterations was lower in 40.8% of the problems for the PCxMod and 1.3% (only one problem) for the PCx. In this case, the run time of the PCxMod was lower because a better starting point was used.

In almost all problems with larger running time, the new approach performed better than the traditional one. This reveals a welcome feature of the proposed approach, since those are the most important problems to be solved.

Although the total number of iterations did not decrease in some problems, a better starting point was determined as the optimal adjustment algorithm for p coordinates was used after Step 1 of the Mehrotra's heuristic and, consequently, the running time was reduced.

An interesting result is the fact that two problems (co9 and nug08) were solved only by PCxMod.

In another experiment, the same set of test problems were solved using only the PCxMod. First, p = 2 (2-coord) was considered, which represents the optimal pair adjustment algorithm and after the value of p according to the stopping criterion previously described was determined. The results obtained are shown in **Table 3**.

The results showed that the PCxMod takes less iterations with values of p greater than 2 in 40.8% of the tests problems. For p = 2 this number is approximately 5.3%. The total running time was reduced by about 60.5% of the problems for p-coordinates and only 22.4% were solved in a lower time as p = 2 was considered.

Again the problem co9 obtained status optimal only for p = 8.

These tests confirmed that the performance of the algorithm is improved by increasing the value of p. That happens because the residual b^k has a greater reduction from one iteration to another and also because the given point by the algorithm as a solution is different for each value of p. The obtained points for $p > 2$ achieved better performance in most cases.

It should be mentioned that the required total time to obtain a solution for the optimal adjustment algorithm for p coordinates is not significant in relation to the resolution total time of the problems. This time is almost null in many of the tested problems. Considering the pds-80 problem, the time spent by the optimal adjustment algorithm was 24 seconds, which represents approximately 0.07% of the total running time.

7. Conclusions and Future Work

In this work, the optimal adjustment algorithm for p co-

Table 2. PCx × PCxMod.

	Dimension				Iterations		Time	
Problem	**Rows**	**Columns**	**p**	**itAux**	**PCx**	**PCxMod**	**PCx**	**PCxMod**
80bau3b	2140	11066	8	2	36	36	**0.4**	0.44
agg2	514	750	4	2	21	21	0.1	**0.07**
agg3	514	750	4	2	19	19	0.12	**0.05**
czprob	671	2779	4	2	26	**25**	0.05	0.05
degen3	1503	2604	4	10	15	15	**0.77**	0.8
dfl001	5984	12143	8	2	45	**41**	77.27	**69.27**
etamacro	334	669	4	10	26	**25**	0.04	0.04
fffff800	322	826	4	2	29	**28**	0.07	0.07
fit2d	25	10524	4	2	22	22	**0.52**	0.55
fit2p	3000	13525	8	2	20	20	**0.3**	0.36
maros-r7	2152	7440	8	2	16	16	1.94	1.94
modszk1	665	1599	4	4	20	20	0.04	0.04
perold	593	1389	4	2	32	32	0.15	**0.12**
pilot	1368	4543	4	2	34	**30**	2.06	**1.93**
pilot87	1971	6373	4	2	25	25	5.19	**5.15**
pilotwe	701	2814	4	6	45	**44**	0.2	**0.16**
scfxm3	915	1704	4	2	19	19	0.08	**0.06**
seba	448	901	4	2	13	13	0.25	**0.24**
ship08l	470	3121	4	2	14	14	0.06	**0.04**
stocfor3	15362	22228	10	2	30	30	**1.16**	1.2
wood1p	171	1718	4	10	22	22	**0.26**	0.3
cre-a	2994	6692	8	2	23	23	0.25	**0.23**
cre-b	5336	36382	8	2	35	35	**3.21**	3.34
cre-c	2375	5412	8	2	25	25	**0.19**	0.21
cre-d	4102	28601	8	2	35	35	**2.73**	2.79
ken-11	10085	16740	8	2	20	20	**0.62**	0.72
ken-13	22534	36561	10	2	23	23	**1.89**	2.24
ken-18	78862	128434	20	2	26	26	**15.96**	18.14
osa-07	1081	25030	4	2	22	22	**0.56**	0.6
osa-14	2300	54760	8	2	25	25	**1.65**	1.74
osa-30	4313	104337	8	4	24	**21**	**3.86**	4.8
osa-60	10243	243209	8	5	31	**24**	**15.38**	17.06
bl	5729	12462	8	2	32	32	**1.48**	1.54
bl2	5729	12462	8	7	37	**35**	**1.69**	1.71
co5	4849	10787	8	2	47	47	1.72	1.72
co9	9090	19997	8	20	*	**46**	*	**6.14**
cq9	8004	19317	8	2	46	**44**	4.67	**4.59**
ex01	235	1556	4	2	25	25	0.12	**0.1**
ex02	227	1548	4	3	31	**30**	0.11	0.11
ex05	832	7805	4	4	32	**31**	0.68	0.68
ex06	825	7797	4	9	63	**58**	1.17	**1.16**
ex09	1821	18184	4	2	37	**36**	**1.61**	1.63
fort45	1152	1582	4	12	17	**15**	0.09	**0.08**
fort47	1152	1582	4	2	16	16	0.11	**0.07**
fort48	1152	1582	4	10	15	**12**	0.1	**0.07**
fort53	1156	1586	4	3	17	**16**	0.11	**0.08**
fort56	1156	1586	4	3	17	17	0.11	**0.09**
fort58	1156	1586	4	3	17	17	0.13	**0.09**
fort59	1156	1586	4	3	16	16	0.12	**0.08**
fort60	1156	1586	4	19	17	**16**	0.13	**0.11**
fort61	1156	1586	4	19	18	**16**	0.12	**0.11**
ge	9150	14990	8	4	43	**37**	2.17	**2.1**
nl	6665	14680	8	2	33	33	2.62	**2.61**
x1	1050	1480	4	2	10	10	0.08	**0.04**
x2	1050	1480	4	10	20	**16**	**0.07**	0.08
els19	4350	13186	8	10	20	20	145.35	**144.32**
chr25a	8149	15325	8	10	23	**21**	40.65	**37.08**
chr22b	5587	10417	8	10	21	21	**15.58**	15.79
scr15	2234	6210	8	10	16	16	**22.48**	22.69
scr20	5079	15980	8	2	**14**	15	281.85	**266.86**

rou20	7359	37640	8	2	13	13	1524.12	**1459.71**
nug06	372	486	4	4	21	**10**	0.19	**0.08**
nug07	602	931	4	2	9	9	0.26	**0.23**
nug08	912	1632	4	3	*	**7**	*	**0.76**
pds-02	2609	7339	8	2	24	24	**0.25**	0.28
pds-06	9156	28472	8	10	29	29	**8.18**	8.49
pds-10	15648	48780	10	2	33	33	42.51	**39.14**
pds-20	32287	106080	20	2	43	**39**	407.83	**392.01**
pds-30	47968	156042	20	9	43	**42**	1369.47	**1318.48**
pds-40	64276	214385	20	2	50	50	**4506.78**	4644.72
pds-50	80339	272513	20	2	52	**51**	8406.11	**8301.49**
pds-60	96514	332862	20	10	52	**50**	14293.89	**13687.98**
pds-70	111896	386238	20	2	55	55	24542.93	**23943.35**
pds-80	126120	430800	20	10	54	**51**	33795.18	**32035.45**
pds-90	139752	471538	20	2	52	**51**	37007.94	**36638.86**
pds-100	152300	498530	40	2	57	57	**47793.96**	48147.86

*: means that the method failed.

Table 3. 2-coordinates × *p*-coordinates.

	Dimension			**ItAux**		**Iterations**		**Time**	
Problem	**Rows**	**Columns**	**p**	**2-coord**	**p-coord**	**2-coord**	**p-coord**	**2-coord**	**p-coord**
80bau3b	2140	11066	8	2	2	36	36	**0.43**	0.44
agg2	514	750	4	2	2	21	21	0.07	0.07
agg3	514	750	4	2	2	19	19	0.06	**0.05**
czprob	671	2779	4	2	2	25	25	0.06	**0.05**
degen3	1503	2604	4	2	10	16	**15**	0.82	**0.8**
dfl001	5984	12143	8	2	2	42	**41**	72.53	**69.27**
etamacro	334	669	4	2	10	26	**25**	0.04	0.04
fffff800	322	826	4	6	2	28	28	0.08	**0.07**
fit2d	25	10524	4	2	4	22	22	0.55	0.55
fit2p	3000	13525	8	2	2	20	20	**0.33**	0.36
maros-r7	2152	7440	8	2	2	**12**	16	2.31	**1.94**
modszk1	665	1599	4	4	4	20	20	0.05	**0.04**
perold	593	1389	4	2	2	32	32	0.12	0.12
pilot	1368	4543	4	2	2	34	**30**	2.09	**1.93**
pilot87	1971	6373	4	2	2	25	25	5.23	**5.15**
pilotwe	701	2814	4	2	6	45	**44**	0.18	**0.16**
scfxm3	915	1704	4	2	2	19	19	0.06	0.06
seba	448	901	4	2	2	13	13	0.28	**0.24**
ship08l	470	3121	4	2	2	14	14	**0.03**	0.04
stocfor3	15362	22228	10	2	2	30	30	1.2	1.2
wood1p	171	1718	4	2	10	22	22	**0.27**	0.3
cre-a	2994	6692	8	2	2	23	23	**0.22**	0.23
cre-b	5336	36382	8	2	2	35	35	3.36	**3.34**
cre-c	2375	5412	8	2	2	25	25	0.22	**0.21**
cre-d	4102	28601	8	2	2	35	35	2.81	**2.79**
ken-11	10085	16740	8	6	2	20	20	0.73	**0.72**
ken-13	22534	36561	10	2	2	23	23	**2.03**	2.24
ken-18	78862	128434	20	2	2	29	**26**	**15.63**	18.14
osa-07	1081	25030	4	2	2	22	22	0.6	0.6
osa-14	2300	54760	8	2	2	25	25	1.74	1.74
osa-30	4313	104337	8	2	4	21	21	**4.32**	4.8
osa-60	10243	243209	8	2	5	31	**24**	**15.87**	17.06
bl	5729	12462	8	2	2	32	32	**1.5**	1.54
bl2	5729	12462	8	2	7	37	**35**	**1.69**	1.71
co5	4849	10787	8	2	2	47	47	1.72	1.72
co9	9090	19997	8	*	20	*	**46**	*	**6.14**
cq9	8004	19317	8	2	2	45	**44**	4.62	**4.59**
ex01	235	1556	4	2	2	25	25	0.11	**0.1**
ex02	227	1548	4	2	3	31	**30**	0.16	**0.11**
ex05	832	7805	4	2	4	32	**31**	0.68	0.68
ex06	825	7797	4	2	9	64	**58**	1.22	**1.16**

ex09	1821	18184	4	2	2	37	**36**	1.67	**1.63**
fort45	1152	1582	4	2	12	17	**15**	0.12	**0.08**
fort47	1152	1582	4	2	2	16	16	0.12	**0.07**
fort48	1152	1582	4	2	10	15	**12**	**0.06**	0.07
fort53	1156	1586	4	4	3	16	16	0.08	0.08
fort56	1156	1586	4	2	3	17	17	0.1	**0.09**
fort58	1156	1586	4	2	3	17	17	0.1	**0.09**
fort59	1156	1586	4	2	3	16	16	0.13	**0.08**
fort60	1156	1586	4	2	19	17	**16**	0.14	**0.11**
fort61	1156	1586	4	2	19	18	**16**	0.12	**0.11**
ge	9150	14990	8	2	4	43	**37**	2.2	**2.1**
nl	6665	14680	8	2	2	33	33	2.62	**2.61**
x1	1050	1480	4	2	2	10	10	0.07	**0.04**
x2	1050	1480	4	2	10	20	**16**	0.12	**0.08**
els19	4350	13186	8	2	10	21	**20**	152.69	**144.32**
chr25a	8149	15325	8	2	10	22	**21**	39.89	**37.08**
chr22b	5587	10417	8	2	10	21	21	**15.53**	15.79
scr15	2234	6210	8	2	10	16	16	22.48	**22.69**
scr20	5079	15980	8	2	2	16	**15**	283.56	**266.86**
rou20	7359	37640	8	2	2	13	13	1466.29	**1459.71**
nug06	372	486	4	2	4	**8**	10	0.11	**0.08**
nug07	602	931	4	2	2	**8**	9	0.26	**0.23**
nug08	912	1632	4	2	3	7	7	**0.66**	0.76
pds-02	2609	7339	8	2	2	24	24	**0.26**	0.28
pds-06	9156	28472	8	2	10	30	**29**	**8.44**	8.49
pds-10	15648	48780	10	2	2	**32**	33	41.98	**39.14**
pds-20	32287	106080	20	2	2	42	**39**	421.59	**392.01**
pds-30	47968	156042	20	2	9	45	**42**	1413.25	**1318.48**
pds-40	64276	214385	20	2	2	51	**50**	4795.93	**4644.72**
pds-50	80339	272513	20	2	2	52	**51**	8609.67	**8301.49**
pds-60	96514	332862	20	2	10	53	**50**	14876.38	**13687.98**
pds-70	111896	386238	20	2	2	56	**55**	25584.8	**23943.35**
pds-80	126120	430800	20	2	10	51	51	32484.67	**32035.45**
pds-90	139752	471538	20	2	2	54	**51**	38492.48	**36638.86**
pds-100	152300	498530	40	2	2	57	57	48772.58	**48147.86**

*: means that the method failed.

ordinates was used in conjunction with the interior point method to determine good starting points, enabling the method to perform better and consequently converge faster. The main advantages of this algorithm are simplicity and quick initial progress.

Numerical experiments on a set of problems showed that by using this approach it is possible to reduce the total number of iterations and the running time for many problems, mainly large-scale ones. In addition, two problems were solved to optimality only by this approach. That is, using the algorithm for p coordinates to improve the starting point computed by PCx leads to a more robust implementation.

By incorporating the optimal adjustment algorithm for p coordinates on the code PCx, more specifically Mehrotra's heuristic, and performing a few iterations, a more robust implementation was obtained.

The computational experiments on a set of linear programming problems showed that this approach can reduce the total number of iterations and total run time for several problems, especially the larger problems and those with higher running time. Moreover, the optimal solution of some unsolved problems was found.

As future work, sophisticated heuristics for the choice of the number of coordinates p will be developed as after several experiments, values for p that reduce the number of iterations by about 90% of the test problems were found. Also other stopping criterion for the optimal adjustment algorithm for p coordinates will be investigated, since the number of iterations performed by such an algorithm directly influences the quality of the starting point determined. New forms of choosing the p columns will be developed.

8. Acknowledgements

This research was sponsored by the Brazilian Agencies CAPES and CNPq.

9. References

[1] G. B. Dantzig, "Converting A Converging Algorithm into a Polynomially Bounded Algorithm," Technical Report, Stanford University, SOL 91-5, 1991.

[2] G. B. Dantzig, "An $\in$-Precise Feasible Solution to a Linear Program with a Convexity Constraint in $1/\in^2$ Iterations Independent of Problem Size," Technical Report, Stanford University, SOL 92-5, 1992.

[3] M. Epelman and R. M. Freund, "Condition Number Complexity of an Elementary Algorithm for Computing a Reliable Solution of a Conic Linear System," *Mathematical Programming*, Vol. 88, No. 3, 2000, pp. 451-485.

[4] J. P. M. Gonçalves, "A Family of Linear Programming Algorithms Based on the Von Neumann Algorithm," Ph.D. Thesis, Lehigh University, Bethlehem, 2004.

[5] J. P. M. Gonçalves, R. H. Storer and J. Gondzio, "A Family of Linear Programming Algorithms Based on an Algorithm by Von Neumann," *Optimization Methods and Software*, Vol. 24, No. 3, 2009, pp. 461-478. doi:10.1080/10556780902797236

[6] J. Silva, "Uma Família de Algoritmos para Programação Linear Baseada no Algoritmo de Von Neumann," Ph.D. Thesis, IMECC-UNICAMP, Campinas, 2009 (in Portuguese).

[7] S. Mehrotra, "On the Implementation of a Primal-Dual Interior Point Method," *SIAM Journal on Optimization*, Vol. 2, No. 4, 1992, pp. 575-601.

[8] H. Y. Benson and D. F. Shanno, "An Exact Primal Dual Penalty Method Approach to Warm Starting Interior-Point Methods for Linear Programming," *Computational Optimization and Applications*, Vol. 38, No. 3, 2007, pp. 371-399. doi:10.1007/s10589-007-9048-6

[9] E. John and E. A. Yildirim, "Implementation of Warm-Start Strategies in Interior-Point Methods for Linear Programming in Fixed Dimension," *Computational Optimization and Applications*, Vol. 41, No. 2, 2008, pp. 151-183. doi:10.1007/s10589-007-9096-y

[10] A. Engau, M. F. Anjos and A. Vannelli, "A Primal-Dual Slack Approach to Warm Starting Interior-Point Methods for Linear Programming," *Operations Research and Cyber-Infrastructure*, Vol. 47, 2009, pp. 195-217. doi:10.1007/978-0-387-88843-9_10

[11] G. B. Dantzig and M. N. Thapa, "Linear Programming 2: Theory and Extensions," Springer-Velag, New York, 1997.

[12] D. Bertsimas and J. N. Tsitsiklis, "Introduction to Linear Optimization," Athena Scientific, Belmont, 1997.

[13] J. Czyzyk, S. Mehrotra, M. Wagner and S. J. Wright, "PCx an Interior Point Code for Linear Programming," *Optimization Methods & Software*, Vol. 11, No.1-4, 1999, pp. 397-430. doi:10.1080/10556789908805757

[14] NETLIB Collection LP Test Sets, "NETLIB LP Repository". http://www.netlib.org/lp/data

[15] M. L. Models, "Hungarian Academy of Sciences OR Lab". http://www.sztaki.hu/meszaros/puplic-ftp/lptestset/mist

