Annals of Operations Research 138, 235–257, 2005

An Accelerated Iterative Method with Diagonally Scaled Oblique Projections for Solving Linear Feasibility Problems

N. ECHEBEST
M.T. GUARDARUCCI opti@mate.unlp.edu.ar
Departamento de Matemática, Facultad de Ciencias Exactas, Universidad Nacional de La Plata, Argentina

H.D. SCOLNIK* hugo@dc.uba.ar
Departamento de Computación, Facultad de Ciencias, Exactas y Naturales, Universidad de Buenos Aires, Argentina

M.C. VACCHINO
Departamento de Matemática, Facultad de Ciencias Exactas, Universidad Nacional de La Plata, Argentina

Abstract. The Projected Aggregation Methods (PAM) for solving linear systems of equalities and/or inequalities, generate a new iterate x^{k+1} by projecting the current point x^k onto a separating hyperplane generated by a given linear combination of the original hyperplanes or halfspaces. In Scolnik et al. (2001, 2002a) and Echebest et al. (2004) acceleration schemes for solving systems of linear equations and inequalities respectively were introduced, within a PAM like framework. In this paper we apply those schemes in an algorithm based on oblique projections reflecting the sparsity of the matrix of the linear system to be solved. We present the corresponding theoretical convergence results which are a generalization of those given in Echebest et al. (2004). We also present the numerical results obtained applying the new scheme to two algorithms introduced by García-Palomares and González-Castaño (1998) and also the comparison of its efficiency with that of Censor and Elfving (2002).

Keywords: projected aggregation methods, exact projection, incomplete projections, oblique projections

The class of convex feasibility problems (CFP) consisting in finding an element of a non-empty closed C convex set which is a subset of $\Re^n$, and

$$C = \bigcap\nolimits^{m}_{i=1} C_i,$$

is the intersection of a family of closed convex subsets C_i, $i = 1, 2, \ldots, m$ of the n-dimensional Euclidean space, have many applications in various fields of science and technology, particularly in problems of image reconstruction from projections (Censor, 1988; Herman and Meyer, 1993). Solving systems of linear equalities and/or inequalities is one of them. A common approach to such problems is to use projection algorithms

*Corresponding author.

(Bauschke and Borwein, 1996). That methodology is particularly useful when the projections onto C_i can be easily computed, as for instance in the particular case of hyperplanes or halfspaces. This last observation leads, in more general cases, to generate for each iterate x^k a suitable closed convex set $\mathcal{H}^k$, such that $C \subset \mathcal{H}^k$, which facilitates the computation of the exact projection onto it. The choice of the sets $\mathcal{H}^k$ may have a significant influence on the convergence rate of the algorithms. Furthermore, García-Palomares and González-Castaño (1998) have proposed an incomplete simultaneous projections algorithm (IPA), for obtaining an approximate projection of x^k onto special convex sets $S_i^k, C \subset S_i^k, i = 1, 2, \ldots, q$, arising from the separation of subsets of the violated constraints. They have defined the next iterate by means of the projection of x^k onto a hyperplane H^k strictly separating C, generated by means of a combination of separating hyperplanes H_i^k relative to the incomplete projections onto each set S_i^k. That approach has also been followed in our paper (Echebest, 2004) in which we present an accelerated iterative projection method for solving the problem defined by $Ax \leq b$, $A \in \Re^{m\times n}$, $b \in \Re^m$. That method, whose general scheme is similar to the IPA algorithm, is an extension of the projection methods for solving systems of linear equations given in Scolnik et al. (2001, 2002a). The basic idea introduced is to consider at the current iterate the generated separating hyperplane, deeper than the one given in García-Palomares and González-Castaño (1998), forcing the new iterate to lie on the convex region defined by it.

In this paper we apply that acceleration scheme in an algorithm based on oblique projections and weighting matrices reflecting the sparsity of the linear system. The new acceleration schemes here presented appear in the algorithm ACEOP when exact oblique projections are made, and later on they are applied to a more general method called ACIOP, both for computing incomplete oblique projections onto each block as well as for obtaining the new iterate x^{k+1}.

In Section 1 the acceleration scheme for solving systems of linear inequalities is presented and applied to an algorithm that employs exact oblique projections. Also, the main results leading to an improved rate of convergence are proved. In subsection 2 we apply the acceleration scheme to the ACIOP algorithm, that uses incomplete oblique projections onto blocks of inequalities. In Section 2 the numerical results obtained with the new methods are given, showing the efficiency of the acceleration scheme. The last section summarizes the conclusions.

1. Algorithms and convergence properties

Consider the non-empty convex set C characterized by a system of m linear inequalities

$$C := \left\{x \in \Re^n : a_i^T x \leq b_i,\ i = 1, 2, \ldots, m\right\} \tag{1}$$

The matrix of the system will be denoted by $A \in \Re^{m\times n}$, and $\|x\|$ will be the 2-norm of $x \in \Re^n$. We will assume that each row a_i^T of A is such that $\|a_i\| = 1$. We will use the notation x^* for any feasible solution of $Ax \leq b$. For $i = 1, 2, \dots, m$, we define

$$C_i = \left\{x \in \Re^n : a_i^T x \leq b_i, b_i \in \Re\right\}, \quad \text{and} \quad P_{C_i}(x) = \text{argmin}_{y \in C_i} \|x - y\|.$$

Given a diagonal matrix $G = \text{diag}(g_l)$, with $g_l > 0$ for $l = 1, \dots, n$, let $\|x\|_G^2 = x^T G x$ be the associated norm. The oblique projection (G-projection) of $x \in \Re^n$ onto C_i with respect to G is defined by means of $P_{C_i}^G(x) = \text{argmin}_{y \in C_i} \|x - y\|_G$. Solving the minimization problem leads to

$$P_{C_i}^G(x) = x + \frac{\min\left(0, b_i - a_i^T x\right)}{a_i^T G^{-1} a_i} G^{-1} a_i$$

The general scheme of a parallel projection algorithm for finding an element of C is as follows (García-Palomares and González-Castaño, 1998).

Given x^k, we define the set of violated constraints $J^k := \{j : a_j^T x^k - b_j > -\theta\}$, where θ is zero or a fixed positive constant.This set is splitted into subsets $J_1^k, J_2^k, \dots, J_q^k$, such that $J^k = \bigcup_{i=1}^q J_i^k$, leading to the corresponding subsets of inequalities

$$S_i^k := \left\{x \in \Re^n : a_j^T x \leq b_j, \ j \in J_i^k\right\} \tag{2}$$

In the exact projection methods for each S_i^k the projection of x^k,

$$y_i^k = \text{argmin}_{y \in S_i^k} \|y - x^k\|_G$$

is calculated.

As an alternative, García-Palomares and González-Castaño (1998) proposed an algorithm (IPA) for obtaining approximate projections y_i^k of x^k onto sets S_i^k, $i = 1, 2, \dots, q$, satisfying

$$[x^* \in C] \Rightarrow \left[\left\|y_i^k - x^*\right\|_G < \|x^k - x^*\|_G\right] \tag{3}$$

This condition is in particular satisfied if y_i^k is the exact projection onto S_i^k.

In order to assure convergence to a solution of C, the next iterate x^{k+1} is defined using a combination of the directions $d_i^k = y_i^k - x^k$, $d^k = \sum_{i=1}^q w_i^k d_i^k$, $w_i^k \geq 0$, $\sum_{i=1}^q w_i^k = 1$. Then the next iterate is defined as $x^{k+1} = x^k + \omega_k \lambda_k d^k$, where $\eta \leq \omega_k \leq 2 - \eta$, $0 < \eta \leq 1$, and λ_k depends on the chosen algorithm.

The value defined for λ_k in García-Palomares and González-Castaño (1998), when $\omega_k = 1$, determines that the next iterate coincides with the projection of x^k onto a strictly separating hyperplane of C. In the following, we will describe the adaptation of the algorithms IPA and EPA, given in García-Palomares and González-Castaño (1998), to systems of linear inequalities considering oblique projections, a choice of $\{w_i^k\}_{i=1}^q$, $\eta \leq \omega_k \leq 2 - \eta$, $0 < \eta \leq 1$.

Algorithm 1. Parallel Incomplete projections Algorithm (IPA) (*k-th iteration*).

Given $x^k \notin C$, $0 < \theta < 0.1$, $0 < \eta \leq 1$.

- Define $J^k := \{j : a_j^T x^k - b_j > -\theta\}$, and $q_k = card(J^k)$.
  Define sets $S_1^k, S_2^k, \ldots, S_q^k$ according to (2).
- For $i = 1, 2, \ldots, q$ in parallel
  Compute y_i^k such that it satisfies (3): $\|y_i^k - x^*\|_G < \|x^k - x^*\|_G$
  Define $d_i^k = y_i^k - x^k$
  End For.
- Define $d^k = \sum_{i=1}^q w_i{}^k d_i^k$, $\sum_{i=1}^q w_i{}^k = 1$, $w_i^k \geq 0$. Calculate

$$\lambda_k = \frac{\sum_{i=1}^q w_i^k \|d_i^k\|_G^2}{2\|d^k\|_G^2} \tag{4}$$

- Define $x^{k+1} = x^k + \omega_k \lambda_k d^k$, $\eta \leq \omega_k \leq 2 - \eta$.

Remark 1. When $\omega_k = 1$, the iterate x^{k+1} is the oblique projection of x^k onto the separating hyperplane

$$H^k := \left\{ x : (Gd^k)^T (x - x^k) = \sum_{i=1}^q w_i^k \frac{\|d_i^k\|_G^2}{2} \right\} \tag{5}$$

that is a combination of the separating hyperplanes

$$H_i^k := \left\{ x : \left(Gd_i^k\right)^T (x - x^k) = \frac{\|d_i^k\|_G^2}{2} \right\} \tag{6}$$

When the y_i^k are the exact projections onto each intermediate set S_i^k, the algorithm is denoted EPA (Exact Projections Algorithm) in García-Palomares and González-Castaño (1998). The modified value of λ_k, with oblique projections,

$$\lambda_k = \frac{\sum_{i=1}^q w_i^k \|d_i^k\|_G^2}{\|d^k\|_G^2}. \tag{7}$$

corresponds to the definition of x^{k+1} as the oblique projection onto the separating hyperplane; that is a combination of those arising from the exact projections onto each S_i^k.

Something characteristic of the PAM methods for linear systems, like those in García-Palomares (1993) and Scolnik et al. (2002a), is the definition of $x^{k+1} = x^k + \lambda_k d^k$, with λ_k satisfying $\lambda_k = \text{argmin}_\lambda \|x^k + \lambda d^k - x^*\|_G^2$.

Remark 2. If x^* is a solution to the system $Ax \leq b$, the ideal value of λ_k satisfying

$$\min_\lambda \|x^k + \lambda d^k - x^*\|_G^2$$

requires to compute the solution of the problem

$$\min_{\lambda} \|x^k - x^*\|_G^2 - 2\lambda (Gd^k)^T (x^* - x^k) + \lambda^2 \|d^k\|_G^2 \tag{8}$$

by means of

$$\lambda_k = \frac{(Gd^k)^T (x^* - x^k)}{\|d^k\|_G^2}. \tag{9}$$

In general, the value of such an optimal λ cannot be obtained by a practical formula. The formulas appearing in (4) and (7), introduced in García-Palomares and González-Castaño (1998) correspond to the related problem (8), avoid this difficulty.

1.1. Algorithm ACEOP: Exact projections

Given $Ax \le b$, we consider as in Censor et al. (2001a) a matrix $G = \text{diag}(1/s_j)$, such that each s_j denotes the number of nonzero elements of column j, for $j = 1, \dots, n$.

We consider a particular case of the EPA algorithm, fixing $\omega_k = 1$, $\theta = 0$, $card(J_i^k) = 1$, for all $i = 1, 2, \dots, q_k$, where $q_k = card(J^k)$. Now, each set S_i^k (2) corresponds to a violated constraint in x^k.

The exact G-projection of x^k onto each halfspace S_i^k, $i = 1, 2, \dots, q_k$, is easily calculated by $y_i^k = P^G_{C_{j(i)}}(x^k)$, if $j(i)$ is the original index of the corresponding inequality of the system (1).

From hereafter we denote by r_j^k the difference $b_j - a_j^T x^k$, and $\beta_j = a_j^T G^{-1} a_j$.

Remark 3. In particular, when $y_i^k = P^G_{C_{j(i)}}(x^k)$ we get $d_i^k = y_i^k - x^k = r^k_{j(i)} G^{-1} a_{j(i)} / \beta_{j(i)}$, being $\beta_{j(i)} \ge \min_{l=1,\dots,n} s_l \ge 1$, and $r^k_{j(i)} = b_{j(i)} - a^T_{j(i)} x^k < 0, i = 1, 2, \dots, q_k$.

It is useful to point out that if x^* is a solution to $Ax \le b$, $(Gd_i^k)^T (x^* - x^k) = (Gd_i^k)^T (x^* - y_i^k + y_i^k - x^k) \ge \|d_i^k\|_G^2$, considering $(Gd_i^k)^T (x^* - y_i^k) \ge 0$ as a consequence of the convexity of S_i^k, the definition of d_i^k and $1 \le \beta_{j(i)} \le m$.

From the assumptions made in this subsection, we can get the following results

Lemma 1. Given x^k, $J^k = \{j : b_j - a_j^T x^k < 0\}$, $q_k = card(J^k)$. If $d^k = \sum_{i=1}^{q_k} w_i^k d_i^k$, $d_i^k = y_i^k - x^k$, $\sum_{i=1}^{q_k} w_i^k = 1$, $w_i^k = 1/q_k$, then

(i) For each $i = 1, \dots, q_k$, $d_i^k = r^k_{j(i)} G^{-1} a_{j(i)} / \beta_{j(i)}$, denoting $j(i)$ the original index of the system of inequalities (1) corresponding to i.
(ii) For each $i = 1, \dots, q_k$, $r^k_{j(i)} = b_{j(i)} - a^T_{j(i)} x^k \ge a^T_{j(i)} x^* - a^T_{j(i)} x^k$, if x^* is a solution to $Ax \le b$.
(iii) $(Gd^k)^T (x^* - x^k) = \sum_{i=1}^{q_k} w_i^k (Gd_i^k)^T (x^* - x^k) = \sum_{i=1}^{q_k} w_i^k r^k_{j(i)} a^T_{j(i)} (x^* - x^k) / \beta_{j(i)}$,
(iv) $(Gd^k)^T (x^* - x^k) \ge \sum_{i=1}^{q_k} w_i^k (r^k_{j(i)})^2 / \beta_{j(i)} > 0$,

(v) If $x^{k+1} = x^k + \lambda d^k$, $\lambda > 0$ then

$$\|x^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\lambda (Gd^k)^T (x^* - x^k) + \lambda^2 \|d^k\|_G^2 \leq \tag{10}$$

$$\|x^k - x^*\|_G^2 - 2\lambda \sum_{i=1}^{q_k} w_i^k \big(r_{j(i)}^k\big)^2 / \beta_{j(i)} + (\lambda)^2 \|d^k\|_G^2 \tag{11}$$

(vi) If λ_k is the argmin of (11), then

$$\lambda_k = \frac{\sum_{i=1}^{q_k} w_i^k \big(r_{j(i)}^k\big)^2 / \beta_{j(i)}}{\|d^k\|_G^2} \tag{12}$$

Furthermore

$$\|x^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\lambda_k (Gd^k)^T (x^* - x^k) + (\lambda_k)^2 \|d^k\|_G^2, \quad \text{satisfies} \tag{13}$$

$$\|x^{k+1} - x^*\|_G^2 \leq \|x^k - x^*\|_G^2 - \alpha_k, \quad \text{where} \tag{14}$$

$$\alpha_k = \lambda_k^2 \|d^k\|_G^2 = \frac{\big(\sum_{i=1}^{q_k} w_i^k \big(r_{j(i)}^k\big)^2 / \beta_{j(i)}\big)^2}{\|d^k\|_G^2} \tag{15}$$

(vii) $x^{k+1} = x^k + \lambda_k d^k$, with λ_k given by (12) is the projection of x^k onto the hyperplane

$$\left\{ x : (Gd^k)^T (x - x^k) = \sum_{i=1}^{q_k} \frac{w_i^k \big(r_{j(i)}^k\big)^2}{\beta_{j(i)}} \right\} \tag{16}$$

(viii) Moreover, (16) is the separating hyperplane of x^k with respect to C.

Proof. (i)–(iii) follow immediately from the definitions and the stated hypothesis. In order to prove (iv) we just have to take into account that for all $i = 1, 2, \ldots, q_k$, $r_{j(i)}^k < 0$, together with Remark 3 and (i)–(iii). By simple comparison (v) follows from (iv). The first part of (vi) follows directly by finding the argmin of problem (11).

The remaining results follow just replacing argmin in (11). For proving (vii) it is enough to replace x^{k+1} in (16). For proving (viii) consider the inequality given in (iv). That inequality shows that (16) is a separating hyperplane of C when $\sum_{i=1}^{q_k} \frac{w_i^k (r_{j(i)}^k)^2}{\beta_{j(i)}} \neq 0$. $\square$

Lemma 2. Given x^k, if $d^k = \sum_{j \in J^k} w_j^k d_j^k$, where $d_j^k = P_{C_j}^G(x^k) - x^k$ for all $j \in J^k$, and $x^{k+1} = x^k + \lambda_k d^k$, $\lambda_k = \frac{\sum_{j \in J^k} w_j^k \|d_j^k\|_G^2}{\|d^k\|_G^2}$, then the sequence $\{x^k\}$ satisfies

$$\|x^{k+1} - x^*\|_G^2 \leq \|x^k - x^*\|_G^2 - \alpha_k, \quad \text{where} \tag{17}$$

$$\alpha_k = \lambda_k^2 \|d^k\|_G^2 = \left(\sum_{j \in J^k} w_j^k \|d_j^k\|_G^2 \right)^2 \bigg/ \|d^k\|_G^2. \tag{18}$$

Proof. It follows from (vi) of the previous Lemma. □

The following results are needed for justifying the acceleration scheme that will be applied to the EPA algorithm with unitary blocks and exact G-projections.

Lemma 3. Given x^k, $J_k = \{j : b_j - a_j^T x^k < 0\}$, d^k, λ_k and x^{k+1} as defined in Lemma 2, then

(i) $(x^* - x^{k+1})^T G d^k \geq 0$.
(ii) $(x^* - x^{k+1})^T G d^k \geq \sum_{j \in J_k} w_j^k r_j^k r_j^{k+1}/\beta_j$.
(iii) $r_j^{k+1} = r_j^k - \lambda_k a_j^T d^k,\ j \in J^k$.
(iv) $\sum_{j \in J_k} w_j^k r_j^k r_j^{k+1}/\beta_j = 0$.

Proof. Using the definition of x^{k+1} and λ_k in $(x^* - x^{k+1})^T G d^k$, (i) follows.

Due to the fact that $(x^* - x^{k+1})^T G d^k = \sum_{j \in J_k} w_j^k r_j^k a_j^T (x^* - x^{k+1})/\beta_{j(i)}$, and considering $a_j^T x^* \leq b_j$, we get $(x^* - x^{k+1})^T G d^k \geq \sum_{j \in J_k} w_j^k r_j^k (b_j - a_j^T x^{k+1})/\beta_{j(i)} = \sum_{j \in J_k} w_j^k r_j^k r_j^{k+1}/\beta_{j(i)}$.

To prove (iv) we substitute (iii) in $\sum_{j \in J_k} w_j^k r_j^k r_j^{k+1}/\beta_{j(i)}$. Then, replacing the expression of λ_k in $\sum_{j \in J_k} w_j^k (r_j^k)^2/\beta_j - \lambda_k (\sum_{j \in J_k} w_j^k r_j^k a_j{}^T d^k/\beta_j)$, we get $\sum_{j \in J_k} w_j^k r_j^k r_j^{k+1}/\beta_j = 0$. □

Lemma 4. Given x^k, consider J^k, d^k, λ_k and x^{k+1} as defined in Lemma 2.

If $J^{k+1} := \{j : a_j^T x^{k+1} > b_j\}$, $J_1^{k+1} := \{j : j \in J^{k+1}, a_j^T x^k \leq b_j\}$, and $J_2^{k+1} := J^{k+1} \bigcap J^k$, then

(i) For all $j \in J_1^{k+1}$, $a_j^T d^k > 0$.
(ii) If $t_1^{k+1} := \sum_{j \in J_1^{k+1}} w_j^{k+1} r_j^{k+1} G^{-1} a_j/\beta_j$, then $(t_1^{k+1})^T G d^k < 0$.
(iii) If $t_2^{k+1} := \sum_{j \in J_2^{k+1}} w_j^{k+1} r_j^{k+1} G^{-1} a_j/\beta_j$, the sign of $(t_2^{k+1})^T G d^k$ tends to be negative when the negative residuals of the constraints increase in absolute value. On the other hand, it tends to be positive when $|r_j^{k+1}| < |r_j^k|$.

Proof. To prove (i) we consider $j \in J_1^{k+1}$, $r_j^{k+1} = r_j^k - \lambda_k a_j^T d^k < 0$, and $r_j^k \geq 0$. Then, it follows that $a_j^T d^k > 0$. As a consequence of (i) $t_1^{k+1} := \sum_{j \in J_1^{k+1}} w_j^{k+1} r_j^{k+1} G^{-1} a_j/\beta_j$, satisfies $(t_1^{k+1})^T G d^k < 0$. For all $j \in J_2^{k+1}$, we get $r_j^{k+1} < 0$ and $r_j^k < 0$. In that case, the sign of $r_j^{k+1} - r_j^k = -\lambda_k a_j^T d^k$, depends on the sign of $a_j^T d^k$. Thus, if $a_j^T d^k < 0$ then $|r_j^{k+1}| < |r_j^k|$, while if $a_j^T d^k > 0$ the opposite holds. Therefore, since $t_2^{k+1} = \sum_{j \in J_2^{k+1}} w_j^{k+1} r_j^{k+1} G^{-1} a_j/\beta_j$, the sign of $(t_2^{k+1})^T G d^k$ can be either negative or positive depending on the absolute values of the negative residuals. □

From Lemma 4 we can infer the direction d^{k+1}, that combines the exact oblique projections to the violated constraints at x^{k+1}, may satisfy $(d^{k+1})^T G d^k < 0$. Such property has been observed in different numerical experiences in almost all iterations. Considering (ii) and (iii) from the previous Lemma, we see that such a situation is possible due to the zigzagging appearing when non-violated constraints in a given iteration are violated in the next, and the residuals of those constraints that remain violated ($|r_j^{k+1}| \simeq |r_j^k|$) do not decrease in a sensible way.

Moreover, if $(d^{k+1})^T G d^k < 0$, then the next iterate along the direction d^{k+1} will lie outside of the convex region defined by the separating hyperplane (16) which contains the current point x^{k+1}. This observation led us to define a new algorithm such that the defined direction takes into account that property.

Lemma 5. Given x^{k+1}, if $\sigma = (d^{k+1})^T G v < 0$, and $v = d^k$, then the direction $\tilde{d}^{k+1} := P^G_{v^\perp} d^{k+1}$ satisfies

(i) $\tilde{d}^{k+1} = d^{k+1} - \frac{(d^{k+1})^T G d^k)}{\|d^k\|_G^2} d^k$.
(ii) $(G\tilde{d}^{k+1})^T(x^* - x^{k+1}) \geq (Gd^{k+1})^T(x^* - x^{k+1}) > 0$.
(iii) $\|P^G_{v^\perp} d^{k+1}\|_G^2 < \|d^{k+1}\|_G^2$.
(iv) $\|P^G_{v^\perp} d^{k+1}\|_G \neq 0$

Proof. From the definition of $\tilde{d}^{k+1}$, (i) follows. Multiplying (i) by $x^* - x^{k+1}$, and considering the result of Lemma 3 (i), we get $(G\tilde{d}^{k+1})^T(x^* - x^{k+1}) \geq (Gd^{k+1})^T(x^* - x^{k+1})$. Then, since $(Gd^{k+1})^T(x^* - x^{k+1}) \geq \sum_{i=1}^{q_{k+1}} w_i^{k+1} (r_{j(i)}^{k+1})^2 / \beta_{j(i)} > 0$, we obtain (ii). The results (iii) and (iv) follow directly from (i) and (ii) respectively. □

Lemma 6. Given x^k, $v = x^k - x^{k-1}$, consider J^k, d^k as defined in Lemma 2.

If $(d^k)^T G v < 0$, and $\tilde{x}^{k+1} = x^k + \tilde{\lambda}_k \tilde{d}^k$, with $\tilde{d}^k$ defined as in Lemma 5, and

$$\tilde{\lambda}_k = \frac{\sum_{i=1}^{q_k} w_i^k \left(r_{j(i)}^k\right)^2 / \beta_{j(i)}}{\|\tilde{d}^k\|_G^2}, \tag{19}$$

then

$$\|\tilde{x}^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\tilde{\lambda}_k (G\tilde{d}^k)^T(x^* - x^k) + \tilde{\lambda}_k^2 \|\tilde{d}^k\|_G^2$$

satisfies

$$\|\tilde{x}^{k+1} - x^*\|_G^2 < \|x^{k+1} - x^*\|_G^2 \tag{20}$$

where $x^{k+1} = x^k + \lambda_k d^k$, and λ_k defined in (12).

Proof. To derive the inequality (20) it is enough to consider the definitions of $\tilde{x}^{k+1}$, x^{k+1} and their distances to x^*, $\|\tilde{x}^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\tilde{\lambda}_k (G\tilde{d}^k)^T(x^* - x^k) + \tilde{\lambda}_k^2 \|\tilde{d}^k\|_G^2$, where $\tilde{\lambda}_k = \frac{\sum_{i=1}^{q_k} w_i^k (r_i^k)^2/\beta_{j(i)}}{\|\tilde{d}^k\|_G^2}$, $\|x^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\lambda_k G(d^k)^T(x^* - x^k) + \lambda_k^2 \|d^k\|_G^2$, with $\lambda_k = \frac{\sum_{i=1}^{q_k} w_i^k (r_i^k)^2/\beta_{j(i)}}{\|d^k\|_G^2}$.

The difference $\|x^{k+1} - x^*\|_G^2 - \|\tilde{x}^{k+1} - x^*\|_G^2$ coincides with

$$\left[-2\lambda_k (Gd^k)^T(x^* - x^k) + {\lambda_k}^2 \|d^k\|_G^2\right] - \left[-2\tilde{\lambda}_k (G\tilde{d}^k)^T(x^* - x^k) + \tilde{\lambda}_k^2 \|\tilde{d}^k\|_G^2\right].$$

Reordering this expression we get

$$\left[2\frac{\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}}{\|\tilde{d}^k\|_G^2}(G\tilde{d}^k)^T(x^* - x^k) - 2\frac{\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}}{\|d^k\|_G^2}(Gd^k)^T(x^* - x^k)\right]$$
$$-\left[\frac{\left(\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}\right)^2}{\|\tilde{d}^k\|_G^2} - \frac{\left(\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}\right)^2}{\|d^k\|_G^2}\right]$$

The first bracket

$$\left[2\frac{\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}}{\|\tilde{d}^k\|_G^2}(G\tilde{d}^k)^T(x^* - x^k) - 2\frac{\sum_{i=1}^{q_k} w_i^k \left(r_i^k\right)^2/\beta_{j(i)}}{\|d^k\|_G^2}(Gd^k)^T(x^* - x^k)\right] \geq$$

$c_1[\frac{1}{\|\tilde{d}^k\|_G^2} - \frac{1}{\|d^k\|_G^2}]$, where $c_1 = 2(Gd^k)^T(x^* - x^k)\sum_{i=1}^{q_k} w_i^k (r_i^k)^2/\beta_{j(i)}$.

The second bracket coincides with $c_2[\frac{1}{\|\tilde{d}^k\|_G^2} - \frac{1}{\|d^k\|_G^2}]$, where $c_2 = (\sum_{i=1}^{q_k} w_i^k (r_i^k)^2/\beta_{j(i)})^2$.

Hence, since $c_1 \geq 2c_2$, $c_2 > 0$ and $\frac{1}{\|\tilde{d}^k\|_G^2} - \frac{1}{\|d^k\|_G^2} > 0$, we get (20). □

Now, we have the necessary results for presenting the new algorithm. Due to the hypotheses of Lemma 6, the iterate x^{k+1} is defined along the direction $\tilde{d}^k = P_{v^\perp}^G(\sum_{i=1}^{q_k} w_i^k d_i^k)$ where v is the direction $G\tilde{d}^{k-1}$ from the previous iteration.

Given x^k, let us consider J^k, $card(J^k) = q_k$. We will denote by Q_k^G the projector onto the G-orthogonal subspace to $v = x^k - x^{k-1}$. In particular, $Q_0^G = I_n$, where I_n is the identity matrix. The following scheme describes the iterative step ($k > 0$) of the new algorithm, in a version not yet adapted to parallel processing.

Algorithm 2. *ACEOP* (*k-th iteration*) : *Given* x^k, J_k, Q_k^G, $v = \tilde{d}^{k-1}$.

```
• For $i = 1, \ldots, q_k$ in parallel
  Compute $y_i^k = P_{C_{j(i)}}^G(x^k)$.
  Define $d_i^k = y_i^k - x^k$.
  End For.
• Define $d^k = \sum_{i=1}^{q_k} w_i^k d_i^k$,
• Compute $\sigma = v^T G d^k$
  If $\sigma < 0$, define $\tilde{d}^k = Q_k^G(d^k)$, else $\tilde{d}^k = d^k$.
• Compute $x^{k+1} = x^k + \tilde{\lambda}_k \tilde{d}^k$, $\tilde{\lambda}_k$ defined by (19).
```

This algorithm can be easily adapted to parallel multiprocessing.

The following Lemma proves that algorithm ACEOP is well defined.

Lemma 7. Given x^k, $x^k \neq x^*$, the new direction $\tilde{d}^k$ and $\tilde{\lambda}_k$ as in (19) are well defined.

Proof. It is of interest to consider the case $v^T G d^k < 0$, $v = x^k - x^{k-1}$. Taking into account (ii) and (iii) of Lemma 5 it follows immediately that $\tilde{d}^k \neq 0$ and therefore, $\tilde{\lambda}_k$ is well defined. □

Lemma 8. The sequence $\{x^k\}$ generated by ACEOP satisfies $\|x^{k+1} - x^*\|_G^2 \leq \|x^k - x^*\|_G^2 - \tilde{\alpha}_k$, where

$$\tilde{\alpha}_k = (\tilde{\lambda}_k)^2 \|\tilde{d}^k\|^2 = \frac{\left(\sum_{i=1}^{q_k} w_i^k \|d_i^k\|_G^2\right)^2}{\|\tilde{d}^k\|_G^2}, \tag{21}$$

satisfying $\tilde{\alpha}_k \geq \alpha_k$, α_k given in (18).

Proof. The result follows using the expressions of $\tilde{\lambda}_k$, the Lemmas 2 and 6. □

1.1.1. Convergence

The convergence of the sequence generated by the algorithm ACEOP to a solution x^* is proved applying the theory developed by Gubin et al. (1967).

Denote by $d(x, C_i) = \|P_{C_i}^G(x) - x\|_G$ the distance between a point $x \in \Re^n$ and a closed convex set $C_i \subseteq \Re^n$, and define $\Phi(x) = \max_{i=1,\ldots,m}\{d(x, C_i)\}$.

Definition 1. A sequence $\{x^k\}$ is called Fejér-monotone with respect to the set C, if for any $x^* \in C$, and for all $k \geq 0$, $\|x^{k+1} - x^*\|_G \leq \|x^k - x^*\|_G$.

It is easy to check that every Fejér-monotone sequence is bounded. The fundamental theorem of Gubin et al. (1967), is:

Theorem 1. Let $C_i \subset \Re^n$, be closed convex sets, for all $i = 1, \ldots, m$, $C = \bigcap_{i=1,\ldots,m} C_i$, $C \neq \emptyset$. If the sequence $\{x^k\}$ satisfies the properties:

(i) $\{x^k\}$ is Fejér-monotone with respect to C, and

(ii) $\lim_{k\to\infty} \Phi(x^k) = 0$,

then $\{x^k\}$ converges to x^*, $x^* \in C$.

Proof. It follows from Lemmas 5 and 6 of Gubin et al. (1967). □

Lemma 9. Any sequence $\{x^k\}$ generated by ACEOP Algorithm, satisfies (i) and (ii) of Theorem 1, rovided $x^k \notin C$, for all $k \geq 0$.

Proof. The proof of (i) follows immediately from Lemma 8. Moreover, taking into account that the sequence $\{\|x^k - x^*\|_G\}$ converges since it is bounded and monotonically decreasing, $\tilde{\alpha}_k$ tend to zero when k tend to ∞. Furthermore, considering the results of Lemma 6 we know $\tilde{\alpha}_k \geq \alpha_k = \frac{(\sum_{i=1}^{q_k} w_i^k \|d_i^k\|_G^2)^2}{\|d^k\|_G^2}$. Since d^k is a convex combination of $P_{C_i}^G(x^k) - x^k$, then $\|d^k\|_G^2 \leq \sum_{i=1}^{q_k} w_i^k \|d_i^k\|_G^2$ (Iusem and De Pierro 1986). Hence, since $\tilde{\alpha}_k$ tend to zero, and $\tilde{\alpha}_k \geq \sum_{i=1}^{q_k} w_i^k \|d_i^k\|_G^2$, we get $lim_{k\to\infty}\Phi(x^k) = 0$. □

1.2. *Accelerated incomplete projections algorithm (ACIOP)*

In this subsection we apply the acceleration scheme to the ACIOP algorithm, by using incomplete oblique projections onto blocks of inequalities as in the IPA method described in García-Palomares and González-Castaño (1998). To compute the approximate oblique projection onto each S_i^k, we use the ACEOP algorithm with a stopping condition such that it finds a y_i^k satisfying the equivalent condition to (3) (one iteration guarantees this).

As a consequence of the procedure to find y_i^k, the next result justifies that $\tilde{H}_i^k$ is a deeper separating hyperplane the one given in García-Palomares and González-Castaño (1998).

Denoting by z^j the intermediate iterates of ACEOP until obtaining y_i^k, and using $z^0 = x^k$, we get

Lemma 10. Given y_i^k, $x^* \in C$. If $y_i^k - x^k = \sum_{j=1}^{j_i}(z^j - z^{j-1})$, then

(i) $\gamma_i = \sum_{j=1}^{j_i} \|z^j - z^{j-1}\|_G^2 > 0$,
(ii) $\|y_i^k - x^*\|_G^2 \leq \|x^* - x^k\|_G^2 - \sum_{j=1}^{j_i} \|z^j - z^{j-1}\|_G^2$
(iii) $(y_i^k - x^k)^T G(x^* - x^k) \geq (\|y_i^k - x^k\|_G^2 + \gamma_i)/2$, with $\gamma_i > 0$

Proof. The first assertion follows immediately because if x^k is infeasible, at least one iteration of ACEOP is performed and, therefore, we obtain the result given in (17).

According to Lemma 2 each intermediate z^j satisfies $\|z^j - x^*\|_G^2 \leq \|x^* - z^{j-1}\|_G^2 - \|z^j - z^{j-1}\|_G^2$, hence obtaining (ii). Moreover, since $\|y_i^k - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2(y_i^k - x^k)^T G(x^* - x^k) + \|y_i^k - x^k\|_G^2$ we derive (iii) using the inequality (ii). □

This implies that the hyperplane $\tilde{H}_i^k = \{x : (y_i^k - x^k)^T G(x - x^k) = (\|y_i^k - x^k\|_G^2 + \gamma_i)/2\}$ is deeper than the equivalent H_i^k given in (6) and introduced in García-Palomares and González-Castaño (1998). Likewise, the hyperplane generated from the convex combination of the previous ones

$$\tilde{H}^k = \left\{x : (d^k)^T G(x - x^k) = \sum_{i=1}^{q} w_i^k \left(\|d_i^k\|_G^2 + \gamma_i\right)/2\right\} \tag{22}$$

where $d_i^k = y_i^k - x^k, d^k = \sum_{i=1}^q w_i^k d_i^k$, shares the same property when compared to (5).

In principle, as in the IPA algorithm, the new iterate will be obtained projecting x^k onto the deeper separating hyperplane $\tilde{H}^k$. Taking into account x^k is on the separating hyperplane $\tilde{H}^{k-1}$, when the new direction d^k satisfies $(d^k)^T Gv < 0$ and leads to a point exterior to the halfspace limited by $\tilde{H}^{k-1}$ containing C, the direction will be modified by projecting it onto the "correct" region. Such a modification is identical to the one proposed in the ACEOP algorithm when dealing with the same situation. Therefore, if v is the direction at step $x^k - x^{k-1}$, and d^k satisfies $(d^k)^T Gv < 0$, we define

$\tilde{d}^k := P_{v^\perp}^G d^k$, and $\tilde{x}^{k+1} = x^k + \tilde{\lambda}_k \tilde{d}^k$, being

$$\tilde{\lambda}_k = \frac{\sum_{i=1}^q w_i^k(\|d_i^k\|_G^2 + \gamma_i)}{2\|\tilde{d}^k\|_G^2} \tag{23}$$

the argmin of the problem

$$\|x^k - x^*\|^2 - 2\lambda \sum_{i=1}^q w_i^k\big(\|d_i^k\|_G^2 + \gamma_i\big)/2 + (\lambda)^2\|\tilde{d}^k\|_G^2 \tag{24}$$

that is an upper bound of the one given in (8), now using the new direction $\tilde{d}^k$.

We describe in the following the iterative step of the incomplete oblique projections algorithm ACIOP.

Given x^k, denoting by Q_k^G the projector onto the G-orthogonal subspace to the one defined by the previous direction $v = x^k - x^{k-1}$, and defining $Q_0^G = I_n$.

Algorithm 3. ACIOP (k-th iteration). Given $x^k \notin C, v = \tilde{d}^{k-1}, 0 < \theta < 0.1$, $0 < \eta \leq 1$.

```
• Define $J_k = \{j : a_j^T x^k - b_j \geq -\theta\}$, and $S_i^k, i = 1, 2, \ldots, q$.
• For $i = 1, 2, \ldots, q$ in parallel
    Compute $y_i^k$ using Algorithm ACEOP, and compute $\gamma_i$.
    Define ${d_i}^k = {y_i}^k - x^k$.
  End For;
• Define $w_i^k \geq 0$, such that $\sum_{i=1}^q w_i^k = 1$.
  Define $d^k = \sum_{i=1}^q w_i^k d_i^k$, and compute $\sigma = (d^k)^T Gv$,
  Define $\tilde{d}^k = Q_k^G(d^k)$ if $\sigma < 0$, else $\tilde{d}^k = d^k$.
  Compute $\tilde{\lambda}_k$ given in (23).
• Define $x^{k+1} = x^k + \omega_k \tilde{\lambda}_k \tilde{d}^k$, with $\eta \leq \omega_k \leq 2 - \eta$.
```

We are now going to prove the algorithm is well defined and later on the convergence results.

Lemma 11. Given x^k, $x^k \neq x^*$, $x^* \in C$, the direction $\tilde{d}^k$ of ACIOP is well defined and satisfies

(i) $(\tilde{d}^k)^T G(x^* - x^k) \geq (d^k)^T G(x^* - x^k) > 0$, being $d^k = \sum_{i=1}^q w_i^k d_i^k$.
(ii) $\|\tilde{d}^k\|_G < \|d^k\|_G$, if $d^k = \sum_{i=1}^q w_i^k d_i^k$ satisfies $v^T G d^k < 0$, where $v = \tilde{d}^{k-1}$, $k \geq 1$.

Proof. For $k = 0$, $\tilde{d}^0$ coincides with the direction given by the IPA algorithm. Assuming the direction $\tilde{d}^{k-1}$, $k > 0$, is well defined, it is interesting to analyze the case when $\sigma = (d^k)^T G \tilde{d}^{k-1} < 0$. According to the definition, $\tilde{d}^k$ satisfies $\tilde{d}^k = d^k - \sigma \frac{v}{\|v\|_G^2}$. Thus, multiplying both sides by $G(x^* - x^k)$, and using $(\tilde{d}^{k-1})^T G(x^* - x^k) \geq 0$ due to the definition of x^k, we get $(\tilde{d}^k)^T G(x^* - x^k) = (d^k)^T G(x^* - x^k) - \frac{v^T G d^k}{(\|v\|)_G^2}(v^T G(x^* - x^k)) \geq (d^k)^T G(x^* - x^k) > 0$. From this inequality we derive (i) and also that $\|\tilde{d}^k\|_G > 0$. Therefore, the direction is well defined in the special case when its definition modifies the one of the IPA algorithm. On the other hand, and for the same special case, the following holds $\|\tilde{d}^k\|_G{}^2 = \|d^k\|_G{}^2 - (\sigma^2)/\|v\|_G^2$. Thus, $\tilde{d}^k$ satisfies (ii). □

The differences between ACIOP and IPA are the following:

(1) we calculate y_i^k explicitly (algorithm ACEOP);
(2) x^{k+1} is the oblique projection of x^k onto a deeper separating hyperplane;
(3) we preserve the fact that the new iterate does not fall outside of the region defined by the last separating hyperplane.

In order to theoretically analyze the comparative behaviour of the sequences generated by the algorithms ACIOP and IPA we proceed in two stages. First, we compare the step given by ACIOP with the one obtained by an algorithm using the same approximate projections y_i^k, but for defining x^{k+1} it uses directly d^k without projections. Second, we compare such a sequence with that obtained by the original IPA algorithm with oblique projections.

For comparison purposes we will denote by $\tilde{x}^{k+1}$ the ACIOP iterate and by $\underline{x}^{k+1}$ the one given by IPA.

Lemma 12. Given x^k, y_i^k, $i = 1, 2, \ldots, q$, d^k, $\tilde{d}^k$ and $\tilde{\lambda}_k$ defined in ACIOP.
If $\sigma = (d^k)^T G \tilde{d}^{k-1} < 0$, and $\tilde{x}^{k+1} = x^k + \tilde{\lambda}_k \tilde{d}^k$, then

(i) $\|\tilde{x}^{k+1} - x^*\|_G < \|x^{k+1} - x^*\|_G$ if $x^{k+1} = x^k + \lambda_k d^k$, and $\lambda_k = \frac{\sum_{i=1}^q w_i^k(\|d_i^k\|_G^2 + \gamma_i)}{2\|d^k\|_G^2}$.
Furthermore,
(ii) If x^{k+1} is defined as in (i), then $\|x^{k+1} - x^*\|_G < \|\underline{x}^{k+1} - x^*\|_G$ when $\underline{x}^{k+1}$ is obtained using the original λ_k of García-Palomares and González-Castaño (1998), explicitly stated in (4), $\underline{\lambda}_k := \frac{\sum_{i=1}^q w_i^k(\|d_i^k\|_G^2)}{2\|d^k\|_G^2}$

Proof. To prove the inequality (i) let us consider, $\|\tilde{x}^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\tilde{\lambda}_k(\tilde{d}^k)^T G(x^* - x^k) + \tilde{\lambda}_k^2\|\tilde{d}^k\|_G^2$, replacing $\tilde{\lambda}_k$ by the expression given in (23). Analogously,

replacing the expression of λ_k in $\|x^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\lambda_k(d^k)^T G(x^* - x^k) + \lambda_k^2\|d^k\|_G^2$, the difference
$\|x^{k+1} - x^*\|_G^2 - \|\tilde{x}^{k+1} - x^*\|_G^2$ coincides with

$$\left[-2\lambda_k(d^k)^T G(x^* - x^k) + \lambda_k^2\|d^k\|_G^2\right] - \left[-2\tilde{\lambda}_k(\tilde{d}^k)^T G(x^* - x^k) + \tilde{\lambda}_k^2\|\tilde{d}^k\|_G^2\right].$$

Reordering as in Lemma 6 we observe that the involved expressions are similar to those appearing there, except by their numerators λ_k and $\tilde{\lambda}_k$, but they have no influence on the comparison we are interested in. Hence, repeating the steps of Lemma 6 we prove (i).

In order to get (ii) we consider as in (i)
$\|x^{k+1}-x^*\|_G^2 = \|x^k-x^*\|_G^2-2\lambda_k(d^k)^T G(x^*-x^k)+\lambda_k^2\|d^k\|_G^2$, replacing the expression corresponding to λ_k. Also $\|\underline{x}^{k+1} - x^*\|_G^2 = \|x^k - x^*\|_G^2 - 2\underline{\lambda}_k(d^k)^T G(x^* - x^k) + \|d^k\|_G^2$, and taking into account that the value of $\underline{\lambda}_k$ is the one given in (4). It is easy to see the difference $\|\underline{x}^{k+1} - x^*\|_G^2 - \|x^{k+1} - x^*\|_G^2$, is equal to $[2(\lambda_k - \underline{\lambda}_k)(d^k)^T G(x^* - x^k)] - [(\lambda_k^2 - \underline{\lambda}_k^2)\|d^k\|_G^2]$. Therefore, using $(d^k)^T G(x^* - x^k) \geq \lambda_k\|d^k\|_G^2$, $\lambda_k - \underline{\lambda}_k > 0$, (ii) follows. □

Lemma 13. If $\tilde{d}^k$ is defined as in the ACIOP algorithm, $x^{k+1} = x^k + w_k\tilde{\lambda}_k\tilde{d}^k$, with $\tilde{\lambda}_k$ given in (23), $\eta \leq w_k \leq 2 - \eta$ then the generated sequence $\{x^k\}$ satisfies

$$\|x^{k+1} - x^*\|_G^2 \leq \|x^k - x^*\|_G^2 - \tilde{\alpha}_k, \tag{25}$$

where

$$\tilde{\alpha}_k = w_k(2 - w_k)\left(\sum_{i=1}^{q} w_i^k\left(\|d_i^k\|_G^2 + \gamma_i\right)\right)^2 \Big/ \left(4\|\tilde{d}^k\|_G^2\right) \tag{26}$$

where d_i^k and γ_i are those from ACIOP.

Proof. The result follows from the definition of x^{k+1}, $\tilde{d}^k$ and that of $\tilde{\lambda}_k$ given in (23) is the solution of (24). □

1.2.1. Convergence
The convergence of the sequence generated by the ACIOP algorithm is a consequence of the comparisons made in Lemma 12 with the IPA algorithm, together with Theorem 1 in García-Palomares and González-Castaño (1998) and Lemma 13.

2. Numerical experiences

The first purpose of the numerical experiences is to illustrate the behaviour of the acceleration of ACEOP (Algorithm 2), using exact oblique projections onto the violated constraints, in comparison to the EPA method in García-Palomares and González-Castaño

(1998). For that purpose a version of the EPA method was implemented, called EOPA, using $card(J_i^k) = 1$, $w_i^k = 1/q_k$ if the number of violated inequalities in x^k, whose indexes are stored in $J^k = \{j : a_j^T x^k - b_j > 0\}$, is q_k. We briefly describe this version as follows:

EOPA: *Given* x^k, J^k, $q_k = card(J^k)$ (*as in ACEOP*).
Define $d^k = \sum_{i=1}^{q_k} w_i^k (P_{C_i}^G(x^k) - x^k)$ *where* $w_i^k = \frac{1}{q_k}$.
The new iterate is $x^{k+1} = x^k + \lambda_k d^k$, *with* λ_k *as in* (12).

The second purpose is to compare the results of ACIOP (Algorithm 3), that uses incomplete projections onto blocks of violated inequalities, with those obtained with a version of the IPA algorithm in García-Palomares and González-Castaño (1998) called IOPA. In both algorithms the computation of the approximate projections y_i^k is similarly done by means of the procedure of ACEOP, accepting an approximation if the condition described below is satisfied. Hence, in this version of IPA the convex combination $d^k = \sum_{i=1}^{q} w_i{}^k d_i{}^k$ where $d_i{}^k = y_i^k - x^k$ for $i = 1, 2, \ldots, q$, is similar to the one used in ACIOP.
Now, we will describe briefly the algorithms being compared, using an experimental code.

IOPA (*Incomplete Oblique Projections Algorithm*): *Given an iterate* x^k, J^k, *and* S_i^k, y_i^k, for $i = 1, 2, \ldots, q$.
Define $d^k = \sum_{i=1}^{q} w_i{}^k d_i{}^k$, *where* $d_i^k = y_i^k - x^k$, *for* $i = 1, 2, \ldots, q$.
$x^{k+1} = x^k + \lambda_k d^k$, *where* λ_k *is adapted from the original in [8], as given in* (4).
ACIOP (*Accelerated Incomplete Projections Algorithm*): *Given an iterate* x^k, $\tilde{d}^{k-1}$, Q_k^G, J^k, S_i^k, y_i^k, $i = 1, 2, \ldots, q$, *we define the direction* d^k *as in* IOPA.
Define $\tilde{d}^k = Q_k^G(d^k)$ *if* $(d^k)^T G \tilde{d}^{k-1} < 0$, *otherwise* $\tilde{d}^k = d^k$.
Define $x^{k+1} = x^k + \tilde{\lambda}_k \tilde{d}^k$, *where* $\tilde{\lambda}_k$ *is the one defined in* (23).

In the implementations of ACIOP and IOPA we consider:

(i) *A constraint is violated* at x^k if $a_j^T x^k - b_j \geq -(5 * 10^{-5})$. The set J^k of violated constraints is in principle splitted into $q = 4$ blocks of equal cardinality, adding if necessary the remaining inequalities to the fourth block.

(ii) *Incomplete projection onto each block.* We compute it by means of the ACEOP algorithm:
Given $x^k \in \Re^n$, and a block S_i^k, from $z^0 = x^k$ we compute $z^1, \ldots, z^l$ until one of the following conditions is satisfied:

(1) $r(z^l) < 10^{-2} * r(z^0)$, where $r(z^l) = \max_{j \in S_i^k}(0, a_j^T(z^l) - b_j)$, or

(2) $\|z^l - z^{l-1}\|_G < 10^{-4}\|z^1 - z^0\|_G$, or

(3) the maximum number of allowed iterations (15) has been reached.

Also we make the comparison with BICAV (Block Iterative CAV) as described in Censor et al. (2001b). Such algorithm splits A into blocks, denoting each block A_t, for $t = 1, \ldots, q$, and I_t the set of original indices in A. For each block A_t, they consider s_j^t the number of non-zero elements in column j, defining for each block a matrix $G_t = diag(g_j)$, where $g_j = 1/s_j{}^t$, if $s_j^t > 0$, otherwise $g_j = 0$. The BICAV algorithm is sequentially iterative by blocks. Let $t(k)$ be a control sequence at the k- iteration which indicates the block considered in the k-th iteration, for all $k \geq 0$.

BICAV (*Block iterative CAV Algorithm*): *Given* x^k, $J_t^k = \{i : i \in I_{t(k)}, a_i x^k - b_i > 0\}$ *the set of indices of inequalities of* $t(k)$*-block which are violated by* x^k.
Define $x^{k+1} = x^k + \lambda_k \sum_{\{i \in J_t^k\}} \mu_i^{t(k)} r_i^k a_i$, *being* $\mu_i^{t(k)} = \frac{1}{\sum_{j=1}^n s_j^{t(k)} (a_{i_j})^2}$, *and* λ_k *is a relaxation parameter.*

In particular, if $q = 1$, it coincides with the fully simultaneous version CAV, and the matrix G_t is the same than the one considered in the algorithms ACEOP and ACIOP. In our experiences we have used $q = 1$ (BICAV(1)) and $q = 4$ (BICAV(4)) for comparing with ACEOP and ACIOP respectively. We have also tested different values of the parameter λ_k (0.9, 1.0, 1.9) within the classic range (0, 2). In the following tables we report the best obtained values of BICAV with the different values of λ_k.

Test problems

We have run different problems of the type $Ax \leq b$, where the matrix $A \in \Re^{m \times n}$ has been chosen to reflect a variety of condition numbers and/or sparsity patterns. For that purpose we used the Zlatev routine from SPARSKIT2/Library (Saad, 1990).

Another set of problems has been obtained by generating randomly different sparsity patterns according to predefined densities. More precisely, the indices of nonnull entries were generated randomly, as well as the corresponding matrix values. Approximately $density * m * n$ entries of the matrix will be nonnull. After generating the matrix, the code computes the independent term b in such a way that $Ax \leq b$ is compatible. The initial approximation used was a vector x^0 whose components were zero.

Numerical results

The problems were run on a PC Pentium III, 800 MHz, with 256 Mb Ram and 128 Mb Swap using FORTRAN 90.
The stopping criteria were:

If $Rm_k \leq 10^{-6} * \max\{1, Rm_0\}$, where $Rm_k = \max_{i=1,2,\ldots,m}(0, a_i^T x^k - b_i)$, or if the number of iterations reaches the maximum allowed $ITMAX$, with $ITMAX = 5000$.

Table 1
Zlatev's matrices, $m = 12000$, $n = 10000$.

index = 20	G	Meth	Iter	Rm_s	CPU
$Zla(2^2)$	I_n	ACEOP	163	8.9d-7	2.7
		EOPA	858	9.2d-7	19.2
	diag$(1/s_j)$	ACEOP	122	8.8d-7	2.1
		EOPA	1186	9.8d-7	23.2
		BICAV(1)	5000	6.0d-6	127.2
$Zla(2^4)$	I_n	ACEOP	231	9.9d-7	3.5
		EOPA	888	9.7d-7	19.9
	diag$(1/s_j)$	ACEOP	181	8.3d-7	3.1
		EOPA	1196	9.9d-7	26.5
		BICAV(1)	5000	8.9d-6	127.3
$Zla(2^8)$	I_n	ACEOP	182	7.3d-7	2.9
		EOPA	882	9.6d-7	19.8
	diag$(1/s_j)$	ACEOP	235	9.3d-7	3.8
		EOPA	1153	9.9d-7	26.1
		BICAV(1)	5000	1.2d-5	127.5
$Zla(2^{12})$	I_n	ACEOP	344	8.5d-7	5.1
		EOPA	896	9.7d-7	20.0
	diag$(1/s_j)$	ACEOP	308	8.0d-7	4.8
		EOPA	1146	9.3d-7	25.6
		BICAV(1)	5000	1.2d-5	127.3

The obtained results are presented in the following Tables using the notation:

- **Iter:** Number of performed iterations.
- **$\mathbf{Rm_s}$:** $\max_{i=1,2,\ldots,m}(0, a_i^T x^s - b_i)$, x^s being the iterate satisfying the stopping criteria.
- **CPU:** time measured in seconds.

The starting point was $x^0 = 0$.
In Tables 1 and 2 we present the results obtained with the algorithms ACEOP, EOPA and BICAV(1) for the problems of SPARSKIT2 Library derived from Zlatev's matrices. These matrices have been generated with different condition depend on the parameter α. The dimensions were $m = 12000$, $n = 10000$, and the problems were run with $\alpha = 2^i$, $i = 2, 4, 8, 12$, denoting each problem by Zla(α) according to the sort of conditioning with which the matrix was generated. In Table 1 results obtained using $index = 20$, a parameter that indicates the average number of non-zero elements per row are given, while in Table 2 the values correspond to $index = 100$.
These results show the effect of the acceleration scheme of the algorithm ACEOP in regard to the algorithm EOPA, because the number of required iterations and the CPU time are drastically reduced in all problems.
Table 3 shows a comparison of the ACEOP algorithm with the EOPA version of EPA and BICAV(1) using a set of dense randomly generated problems. Results are given

Table 2
Zlatev's matrices, $m = 12000$, $n = 10000$.

index = 100	G	Meth	Iter	Rm_s	CPU
$Zla(2^2)$	I_n	ACEOP	1064	9.9d-7	75.4
		EOPA	5000	1.3d-2	493.9
	diag$(1/s_j)$	ACEOP	912	9.6d-7	68.5
		EOPA	5000	2.4d-2	492.8
		BICAV(1)	5000	5.9d-2	554.2
$Zla(2^4)$	I_n	ACEOP	1130	7.8d-7	71.8
		EOPA	5000	9.4d-3	496.8
	diag$(1/s_j)$	ACEOP	1204	9.7d-7	90.2
		EOPA	5000	1.2d-2	491.9
		BICAV(1)	5000	6.0d-2	553.9
$Zla(2^8)$	I_n	ACEOP	2070	9.9d-7	121.4
		EOPA	5000	1.1d-2	493.5
	diag$(1/s_j)$	ACEOP	1118	3.2d-7	80.6
		EOPA	5000	8.1d-3	490.4
		BICAV(1)	5000	6.2d-2	559.6
$Zla(2^{12})$	I_n	ACEOP	1094	7.8d-7	67.4
		EOPA	5000	6.6d-3	500.8
	diag$(1/s_j)$	ACEOP	1571	8.4d-7	103.9
		EOPA	5000	7.4d-3	491.5
		BICAV(1)	5000	6.4d-2	554.4

Table 3
Random matrices, density = 1. Average timing (seconds).

G	Meth	$m = 800$ $n = 200$	$m = 400$ $n = 100$	$m = 200$ $n = 50$	$m = 100$ $n = 25$
I_n	ACEOP	0.39 (37)	0.11 (36)	0.01 (42)	0 (46)
	EOPA	0.77 (73)	0.27 (71)	0.05 (95)	0.01 (127)
diag$(1/s_j)$	ACEOP	0.38 (37)	0.05 (35)	0.01 (44)	0 (48)
	EOPA	0.71 (73)	0.17 (74)	0.03 (103)	0.02 (135)
	BICAV (1)	53.2 (5000)	11.1 (5000)	0.94 (5000)	0.27 (5000)

indicating the row and column dimensions m, n of the randomly generated problems with density 1. The reported CPU time is the average of solving each problem five times. Between parenthesis the required number of iterations for satisfying the stopping criteria as in Tables 1–2 are given. These results also show the effectiveness of ACEOP

In Tables 4–8 we report the results obtained with the ACIOP algorithm, using incomplete projections onto the violated constraints blocks, IOPA (the implemented version of IPA) and BICAV(4) as described at the beginning of this section. The key difference between them is the definition of x^{k+1}. Moreover, while in BICAV(4) a matrix G_t is used for each block in such a way of reflecting the sparsity pattern, in other algorithms the matrix G corresponds to the sparsity of A.

Table 4
Zlatev's matrices, $m = 12000$, $n = 10000$.

index = 20	G	Meth	Iter	Rm_s	CPU
$Zla(2^2)$	I_n	ACIOP	54	9.4d-7	8.2
		IOPA	115	9.3d-7	21.3
	diag$(1/s_j)$	ACIOP	54	1.6d-7	8.4
		IOPA	115	9.0d-7	24.3
	G_t	BICAV(4)	3190	9.9d-7	78.8
$Zla(2^4)$	I_n	ACIOP	77	8.0d-7	9.1
		IOPA	71	4.3d-7	17.6
	diag$(1/s_j)$	ACIOP	81	5.4d-7	10.3
		IOPA	96	8.4d-7	22.7
	G_t	BICAV(4)	3192	9.9d-7	78.9
$Zla(2^8)$	I_n	ACIOP	55	6.3d-7	7.8
		IOPA	898	9.9d-7	126.6
	diag$(1/s_j)$	ACIOP	72	4.5d-7	10.0
		IOPA	197	7.6d-7	29.8
	G_t	BICAV(4)	3187	9.9d-7	78.8
$Zla(2^{12})$	I_n	ACIOP	67	5.1d-7	9.0
		IOPA	92	8.9d-7	19.7
	diag$(1/s_j)$	ACIOP	65	8.0d-7	9.8
		IOPA	94	6.5d-7	23.0
	G_t	BICAV(4)	3187	9.9d-7	78.7

Table 5
Zlatev's matrices, $m = 12000$, $n = 10000$.

index = 60	G	Meth	Iter	Rm_s	CPU
$Zla(2^2)$	I_n	ACIOP	82	7.5d-7	22.0
		IOPA	239	8.7d-7	149.6
	diag$(1/s_j)$	ACIOP	59	2.5d-7	22.3
		IOPA	234	8.4d-7	169.5
	G_t	BICAV(4)	5000	1.8d-2	333.9
$Zla(2^4)$	I_n	ACIOP	109	8.4d-7	25.3
		IOPA	263	8.5d-7	151.1
	diag$(1/s_j)$	ACIOP	97	9.7d-7	26.5
		IOPA	291	7.1d-7	182.5
	G_t	BICAV(4)	5000	1.8d-2	320.7
$Zla(2^8)$	I_n	ACIOP	147	2.6d-7	28.2
		IOPA	241	9.1d-7	144.8
	diag$(1/s_j)$	ACIOP	108	2.6d-7	27.0
		IOPA	204	9.5d-7	152.8
	G_t	BICAV(4)	5000	1.6d-2	322.1
$Zla(2^{12})$	I_n	ACIOP	71	5.8d-7	21.6
		IOPA	247	8.2d-7	145.8
	diag$(1/s_j)$	ACIOP	87	6.9d-7	25.8
		IOPA	202	7.7d-7	143.4
	G_t	BICAV(4)	5000	1.5d-2	320.4

Table 6
Zlatev's matrices, $m = 12000$, $n = 10000$.

$index = 100$	G	Meth	Iter	Rm_s	CPU
$Zla(2^2)$	I_n	ACIOP	85	6.0d-7	41.6
		IOPA	537	7.2d-7	565.8
	$\text{diag}(1/s_j)$	ACIOP	109	9.6d-7	51.2
		IOPA	508	7.1d-7	510.4
	G_t	BICAV(4)	5000	5.4d-2	514.2
$Zla(2^4)$	I_n	ACIOP	149	5.0d-7	50.5
		IOPA	637	8.4d-7	630.7
	$\text{diag}(1/s_j)$	ACIOP	121	9.9d-7	52.6
		IOPA	574	9.4d-7	551.7
	G_t	BICAV(4)	5000	5.5d-2	515.0
$Zla(2^8)$	I_n	ACIOP	165	7.4d-7	52.6
		IOPA	568	5.5d-7	516.7
	$\text{diag}(1/s_j)$	ACIOP	256	7.4d-7	71.0
		IOPA	616	9.5d-7	583.1
	G_t	BICAV(4)	5000	5.5d-2	515.1
$Zla(2^{12})$	I_n	ACIOP	124	7.8d-7	47.1
		IOPA	551	9.6d-7	545.8
	$\text{diag}(1/s_j)$	ACIOP	199	4.0d-7	65.7
		IOPA	514	3.6d-7	535.3
	G_t	BICAV(4)	5000	5.7d-2	514.5

Table 7
Random matrix, density = 1.

m/n	G	Meth	Iter	Rm_s	CPU
4000/2000	I_n	ACIOP	243	1.8d-6	258.1
		IOPA	5000	3.0d-4	7180.5
	$\text{diag}(1/s_j)$	ACIOP	246	1.8d-6	261.0
		IOPA	5000	5.0d-4	7758.6
	G_t	BICAV(4)	5000	9.9d-2	2960.5

We present in Tables 4–6 the results corresponding to the Zlatev matrices considering different sparsity patterns defined by $index$ in each case.

The results of Tables 4–6 show the efectiveness of the direction $\tilde{d}^k$ and the parameter $\tilde{\lambda}_k$ used in the ACIOP algorithm. In particular, such a performance is more evident in Table 6, corresponding to more dense problems. The results corresponding to ACIOP do not differ much if oblique or orthogonal projections are used.

It is also worthwhile to point out that Table 4 shows that the number of iterations can be similar in ACIOP and IOPA, while the CPU time is not. This means that the cardinality of the blocks corresponding to the violated constraints in IOPA is greater than those in ACIOP. In other words, ACIOP generates better iterates as predicted by

Table 8
Random matrices, density = 0.1.

m/n	G	Meth	Iter	Rm_s	CPU
7500/2500	I_n	ACIOP	118	2.1d-6	75.3
		IOPA	198	2.1 d-6	121.0
	diag$(1/s_j)$	ACIOP	118	2.3d-6	76.7
		IOPA	191	2.2 d-6	111.6
	G_t	BICAV(4)	5000	3.3d-3	627.7
9500/2000	I_n	ACIOP	35	1.6d-6	26.3
		IOPA	53	1.9d-6	40.3
	diag$(1/s_j)$	ACIOP	35	1.7d-6	25.2
		IOPA	49	1.7d-6	40.0
	G_t	BICAV(4)	4406	1.9d-6	596.5
10000/1900	I_n	ACIOP	28	1.4d-6	22.3
		IOPA	41	1.7 d-6	34.3
	diag$(1/s_j)$	ACIOP	28	1.6d-6	21.6
		IOPA	41	1.4 d-6	35.7
	G_t	BICAV(4)	3634	1.9d-6	496.8

the theoretical results. Moreover, the version of IPA that we called IOPA uses the same procedure ACEOP as in ACIOP for finding the approximations y_i^k to the blocks. Thus, the observed differences in the numerical results clearly arise from the different definitions of x^k. It is worthwhile to point out that for those problems both algorithms are more efficient than BICAV(4). We believe that this is a consequence of the definition of x^{k+1}, which in ACIOP and IOPA is the projection of x^k onto a separating hyperplane.

We include in the next Table 7 the results obtained with ACIOP, IOPA and BICAV(4) for the randomly generated problems with dimension $m = 4000$, $n = 2000$, and density 1. The reported CPU time corresponds to the average of five runs of each problem. Those results exhibit the same tendency observed in the other problems.

Finally, we run the same problems of Table 7 using a density of 0.1 and different dimensions as shown in Table 8 using the ACIOP, IOPA and BICAV(4) algorithms.

In all Tables we observe the efficiency of the new algorithms that include the projection onto the half-space defined by the separating hyperplane (22), and forcing the new iterate to lie on the convex region defined by it.

3. Conclusions

In algorithm ACEOP we introduced the basic idea of forcing a new iterate to belong to the convex region defined by the computed separating hyperplane. Both, the theoretical results and the numerical experiences, showed the advantages of this approach.

The acceleration scheme applied in the ACEOP algorithm is the basis for extending its applicability to other class of algorithms suitable for parallel processing. Among them is the IPA algorithm (García-Palomares and González-Castaño, 1998). In particular, we used the same approach in the more general ACIOP algorithm based on the original IPA, both for computing incomplete projections onto each block, as well as for obtaining the new iterate x^{k+1}. As a consequence of the procedure to find the incomplete oblique projection y_i^k onto each block using ACEOP, a hyperplane $\tilde{H}_i^k$ is obtained, deeper than the one given in García-Palomares and González-Castaño (1998). Therefore, the new iterate x^{k+1} is the projection of x^k onto a deeper separating hyperplane (22). The direction $\tilde{d}^k$ defined by this algorithm preserves the fact that the new iterate does not fall outside of the region defined by the last separating hyperplane. The numerical results show a very competitive behaviour when dealing with blocks of inequalities by means of approximate oblique projections such as in the ACIOP algorithm, as predicted by the theory. In a forthcoming paper we will present a generalization of these results consisting in minimizing a proximity function for problems which can be inconsistent.

Acknowledgment

Work supported by the universities of Buenos Aires and La Plata (Project 11/X346), Argentina.

References

Bauschke, H.H. and J.M. Borwein. (1996). "On Projection Algorithms for Solving Convex Feasibility Problems." *SIAM Rev.* 38, 367–426.

Censor, Y. (1988). "Parallel Application of Block-Iterative Methods in Medical Imaging and Radiation Therapy." *Math. Programming* 42, 307–325.

Censor, Y., D. Gordon, and R. Gordon. (2001a). "Component Averaging: An Efficient Iterative Parallel Algorithm for Large and Sparse Unstructured Problems." *Parallel Computing* 27, 777–808.

Censor, Y., D. Gordon, and R. Gordon. (2001b). "BICAV: An Inherently Parallel Algorithm for Sparse Systems with Pixel-Dependent Weighting." *IEEE Trans. on Medical Imaging* 20, 1050–1060.

Censor, Y. and T. Elfving. (2002). "Block-Iterative Algorithms with Diagonally Scaled Oblique Projections for the Linear Feasibility Problem." *SIAM Journal on Matrix Analysis and Applications* 24, 40–58.

Echebest, N., M.T. Guardarucci, H.D. Scolnik, and M.C. Vacchino. (2004). "An Acceleration Scheme for Solving Convex Feasibility Problems Using Incomplete Projection Algorithms." *Numerical Algorithms* 35, 335–350.

García-Palomares, U.M. (1993). "Parallel Projected Aggregation Methods for Solving the Convex Feasibility Problem." *SIAM J. Optim.* 3, 882–900.

García-Palomares, U.M. and F.J. González-Castaño. (1998). "Incomplete Projection Algorithms for Solving the Convex Feasibility Problem." *Numerical Algorithms* 18, 177–193.

Gubin, L.G., B.T. Polyak, and E.V. Raik. (1967). "The Method of Projections for Finding the Common Point of Convex Sets." *USSR Comput. Math. and Math.Phys.* 7, 1–24.

Iusem, A.N. and A. De Pierro. (1986). "Convergence Results for an Accelerated Nonlinear Cimmino Algorithm." *Numer. Math.* 49, 367–378.

Herman, G.T. and L.B. Meyer. (1993). "Algebraic Reconstruction Techniques can be Made Computationally Efficient." *IEEE Trans. Medical Imaging* 12, 600–609.

Saad, Y. (l990). “SPARSKIT: A Basic Tool Kit for Sparse Matrix Computations.” Technical Report 90-20, Research Institute for Avanced Computer Science. NASA Ames Research Center, Moffet Field, CA.

Scolnik, H., N. Echebest, M.T. Guardarucci, and M.C. Vacchino. (2001). “New Optimized and Accelerated PAM Methods for Solving Large Non-symmetric Linear Systems: Theory and Practice.” In D. Butnariu, Y. Censor, and S. Reich (eds.), *Inherently Parallel Algorithms in Feasibility and Optimization and their Applications*, Studies in Computational Mathematics. Amsterdam: Elsevier Science, Volume 8, pp. 457–470.

Scolnik, H., N. Echebest, M.T. Guardarucci, and M.C. Vacchino. (2002a). “A Class of Optimized Row Projection Methods for Solving Large Non-Symmetric Linear Systems.” *Applied Numerical Mathematics* 41, 499–513.

Scolnik, H., N. Echebest, M.T. Guardarucci, and M.C. Vacchino. (2002b). “Acceleration Scheme for Parallel Projected Aggregation Methods for Solving Large Linear Systems.” *Annals of Operations Research* 117, 95–115.