Development of a Procedure for Finding Active Points of Linear Constraints

Said Choufi*
Department of Mathematics, University of Batna, Route de Biskra, Batna, Algeria

Abstract
In this paper, we present an iterative method to determine active point of linear constraints. It is based on two basic operations which are addition and permutation of constraints. This procedure generates a finite sequence of points that basis in a new lemma and a new formula direction, the laspoint of sequence constitutes an active point, and this procedure gives also two matrices. The first one is constituted by the active constraints which are linearly independent and the second one is a matrix whose columns are the basis vectors of the kernel of the first matrix.

Keywords: Active point; Interior point; Kernel of matrix; Optimisation continuous

Introduction
Currently, the domain of optimization is attracting considerable interest from the academic and industrial communities, see, for instances [1-5]. The various existing techniques for solving a given problem and the efficient algorithmic implementations open up many perspectives and diverse applications in different areas [6-8].

There are different methods of optimization exist in the literature, among other, we cite, the simplex [1], the interior point [1,2], the exterior point methods, and evidently with their improved versions [9-13].

In most optimization problems, initialization points are necessary and required in the resolution algorithms for performing numerical implementations [6-8,11,12]. However, the choice of the initialization points is not general, and the values of these points depend strongly on the adopted technique. Furthermore, these points are considered as active or feasible in the applied method.

In this study, we are interested by the optimization problem of the CSLP type (constraint satisfaction linear problems), where the set of constraints are linear and it is defined by determined the active point \( x_{act} \) of a set \( E \) such as:

\[
E = \{x \in \mathbb{IR}^n, \text{subject to } Ax \leq b\} \quad \text{(CSLP)},
\]

where

\( A \) is an \( mxn \) data matrix, not necessarily full rank and \( b \) is a given as vector \( IR^m \).

The problem to solve is the determination of the active points satisfying all the aforementioned constraints.

If the values of the matrix \( A \) and the vector \( b \) components are integer numbers, the above problem is discrete and can be solved by using the ellipse method [4]. However, in the case of optimization continuous, this is not studied in literature.

This past has motivated this investigation in the purpose of giving a theoretical and practical method of resolution of this problem.

The method that we propose is based on the construction of an iterative algorithm, such that, from any initial point (feasible or not) one produces another better point \( x \), then one associates to it two matrices \( A \) and \( Z \). The lines of \( A \) are constituted by the active constraints, which are linearly independent. The columns of matrix \( Z \) are constituted of the kernel of matrix \( A \), and are also linearly independent.

This process allows to generate a sequence of points \( \{x_k\}_{k \geq 0} \) which converge to the point that one search (feasible or active). It is important to mention that our method can be applied without knowing whether this domain of constraints is empty or not.

In the numerical implementation, we used the scientific environment FOTRAN F90 under windows, and the obtained results were very satisfactory.

The rest of this paper is organized as follows. In Sec. 2, we give some definitions and propositions that are used in this article. In Sec. 3, we present the construction of a die kernel. In Sec. 4, we give description of the active method and its implementation in Sec.5, we form algorithm for the active method. At last, we summarize our results in the last section.

We note that:

\( I = \{1, 2, 3, ..., m\} \) the set of constraint indices.

\( x_k \): The iteration point \( k \).

\( I_a = \{i \in I : a_i x_k > b\} \): the set of constraints indices containing the point \( x_k \) externally.

\( I_f = \{i \in I : a_i x_k > b\} \): the set of constraints indices containing the point \( x_k \) internally.

\( \perp \): Orthogonal.

\( nt \): The total number of iterations.

\( x_{act} \): The active point.

\( x_k \): The feasible point.

\( A_k \): The set of active constraints in \( x_k \) point.

\( Z_k \): The matrix formed by the active constraints linearly independent at the point \( x_k \).

*Corresponding author: Said Choufi, Department of Mathematics, University of Batna, Algeria, Tel: +213 33 31 91 34; E-mail: choufi_said@yahoo.fr

Received March 03, 2016; Accepted April 28, 2017; Published May 06, 2017

Citation: Choufi S (2017) Development of a Procedure for Finding Active Points of Linear Constraints. J Appl Comput Math 6: 352. doi: 10.4172/2168-9679.1000352

Copyright: © 2017 Choufi S, et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.
Definitions and Propositions

Definition

The constraint \( a^t x \leq b_i \) is called active in \( x \), if \( a^t x = b_i \) [8].

This definition leads to the fact that, for any vector \( v \), we can introduce all
\[ I_v = \{ i \in I : a^t v = b_i \} \]  
(3.1)

We then say that the vector \( v \) is a regular point of all eligible
\[ \Delta = \{ v \in IR^n : a^t v \leq b_i, \text{for each } i \in I \} \]  
(or just a regular point of the constraints) if and only if it is a regular point of
\[ D = \{ x \in IR^n : a^t(x) = b_i, i = 1, ..., m \} \]  
(3.2)

Proposition

A direction \( d \) is tangent to \( x \in X \), if and only if there exists a sequence
\( \{ d_k \} \) of unit \( d \), and a sequence \( \{ \mu_k \} \) of positive real zero limit, such that
\( x + \mu_k d_k \in X \) [5].

Remarks

Most of the algorithms fail when they have to solve a problem whose constraints are not qualified in the solution. Therefore it is preferable to change the description of the set of constraints before solving the problem.

Proposition (CS constraints satisfaction)

The constraints of \( D \) domain are qualified at the point \( x \in D \), if the gradients in \( x \) of the active constraints [5].
\[ \forall h_i(x) = 1, ..., p \) \( \forall g_j(x) = 1, ..., q \) are linearly independent, where
\[ h_i(x) = b_i \text{ for } i = 1, ..., p \text{ and } g_j(x) < b_j \text{ for } j = 1, ..., q \text{ such that } p + q = m. \]

Construction of Kernel Matrix

This section contains the most important results for the kernel numerical calculation of any matrix, especially in the case where we have a lot of matrices of large size, and to avoid repetition of the calculations.

The calculation of a matrices and vector kernel obeys certain rules of compliance.

These results are given in the following:

Lemma

Let \( v \) be a vector \( v \in IR^n \), which defines a set of constraints as follows:
\[ \Delta = \{ x \in IR^n : a^t x - a = 0 \} \]  
(4.1)

then \( v, \Delta \) i.e. \( v \) is orthogonal to \( \Delta \).

Proof

Let \( x_1, x_2 \in \Delta \), then \( v^t x_1 - a = 0 \)  
(1)
\[ v^t x_2 - a = 0 \]  
(2)

By substitution (1) of (2) we get:
\[ v^t (x_2 - x_1) = 0 \]
This gives that \( v^t (x_1, x_2) \) is an element of \( \Delta \).

From this lemma, we can construct sets \( \Delta \) and \( \Delta - \) that help us to lead the constraint equations in the following two corollaries:

Corollary

Let \( \{ v \in IR^n : v^t x - a \leq 0 \} \)
Where \( v \) is a vector of \( IR^n \), \( a \in R \). Then \( \Delta = \{ a \in R^n : x^t R^n \} \) such that
\[ a = x - x^t \text{ with } -a < 0, v^t x - a = 0 \]  
(4.2)

Lemma

Let \( v \) be a vector in \( IR^n \), then its kernel is formed by the following basis \( \{ z_1, z_2, z_3, ..., z_{n-1} \} \) where \( v^t z = 0 \), for each i=1,2,...,n-1  
(4.4)

Proof

It suffices to show that the rows of the matrix \( (v z_1, z_2, ... , z_{n-1})' \) are linearly independent. Consider the scalars \( \lambda_1, \lambda_2, ..., \lambda_{n-1}, \lambda \) satisfying
\[ \lambda v z_1 + \lambda_2 z_2 + ... + \lambda_{n-1} z_{n-1} = 0 \]
Multiplying by \( v \), it follows:
\[ \lambda v^t v z_1 + \lambda_2 v^t z_2 + ... + \lambda_{n-1} v^t z_{n-1} = 0 \]  
such as
\[ v^t z = 0, i = 1, 2, ..., n - 1 \]  
and \( v^t v = \sum v_i^2 \) then \( \lambda \) = 0. And (1), we have:
\[ \lambda z_1 + \lambda_2 z_2 + ... + \lambda_{n-1} z_{n-1} = 0 \]
for each i=1,...,n-1, from where the result.

Lemma

Let \( v_1 \) and \( v_2 \) be two vectors in \( IR^n \), the set \( \{ z_1, z_2, ... , z_{n-1} \} \) is a basis of the kernel vector \( v_i \) where the kernel index \( 1, 2, ..., n - 1 \) satisfies
\[ v_{z_{i+1}} \neq \text{Max} v_{z_i} \]
(4.5)

Then the matrix \( (v_1 z_1, v_2 z_2, ..., v_{n-1} z_{n-1}) \) is invertible if and only if \( v_{z_{i+1}} \neq 0 \).

Proof

The same proof of Lemma (4.4), in the rows of the matrix \( (v_1, v_2, z_1, ..., z_{n-1}) \) which are linearly independent vectors.

Corollary

Keeping the same data of Lemma (4.4), but here \( v_{z_{i+1}} \neq 0 \). Then the matrix \( (v_1 z_1, v_2 z_2, ..., z_{n-1}) \) is full rank.

Proof

The vectors \( v_1, v_2 \) are linearly dependent.

Lemma

Let \( v_1 \) and \( v_2 \) be two vectors in \( IR^n \), admitting \( \{ z_1, z_2, ... , z_{n-1} \} \)
as the kernel of a basis vector $v_i\{l_1, l_2, l_3, \ldots, l_i\}$ the index set of {1, 2, 3, ..., n-1} such that $v'_i z_j \neq 0$ for each $j \notin \{l_1, l_2, l_3, \ldots, l_i\}$.

And $v'_i z_j = 0$ for each $j \notin \{l_1, l_2, l_3, \ldots, l_i\}$ (4.6)

Then the set $\{z'_i, z'_{i_1}, z'_{i_2}, \ldots, z'_{i_{k-1}}\} \cup \{z_i, i \notin \{l_1, l_2, l_3, \ldots, l_i\}\}$ form a common basis of the kernel vectors $v_i$ and $v'_i$ and verifying $z'_i = z_i + \beta z_m$ with $\beta_j = -v'_i z_j / v'_i z_m$.

where $v'_i z_m = \max_{j \in \{1, 2, \ldots, n-1\}} v'_i z_j$

Proof

Since $\{z_1, z_2, z_3, \ldots, z_{n-1}\}$ is a basis of the kernel vector $v_i$.

We will show that $\{z'_i, z'_{i_1}, z'_{i_2}, \ldots, z'_{i_{k-1}}, z\} \cup \{z_i, i \notin \{l_1, l_2, l_3, \ldots, l_i\}\}$ (4.7) from a common basis vectors $v_i$ and $v'_i$, knowing $\{z'_i, z'_{i_1}, z'_{i_2}, \ldots, z'_{i_{k-1}}, z\}$ resulting kernel $v_i$ such that $z'_i = z_i + \beta z_m$ and $v'_i z_i = v'_i z_i + \beta z_m = 0, \beta = 0 = 0$.

When $i \in \{l_1, l_2, l_3, \ldots, l_i\}$.

For vector $v_i$, we have: if $i \in \{l_1, l_2, l_3, \ldots, l_i\}$, it comes

$z'_i = z_i + \beta z_m$ and $v'_i z_i = v'_i z_i + \beta z_m = 0, \beta = 0 = 0$.

And if $i \notin \{l_1, l_2, l_3, \ldots, l_i\}$, it comes $z'_i = v'_i z_i$ because $\{i_1, i_2, \ldots, i_k\}$ is the largest subset of {1, 2, ..., n-1} satisfies $z'_i = 0$, when $i \notin \{i_1, i_2, \ldots, i_k\}$.

It remains to show that set (4.7) is linearly independent.

$\sum_{i=1}^{k} A_i^t z_i + \sum_{i=1}^{n} a_{i,1} z_1 + \sum_{i=1}^{n} a_{i,2} z_2 + \sum_{i=1}^{n} a_{i,3} z_3 + \ldots = 0$ impose

$\sum_{i=1}^{k} A_i^t z_i = 0$

$\sum_{i=1}^{n} a_{i,1} z_1 + \sum_{i=1}^{n} a_{i,2} z_2 + \sum_{i=1}^{n} a_{i,3} z_3 + \ldots = 0$

$\sum_{i=1}^{k} A_i^t z_i + \sum_{i=1}^{n} a_{i,1} z_1 + \sum_{i=1}^{n} a_{i,2} z_2 + \sum_{i=1}^{n} a_{i,3} z_3 + \ldots = 0$

$\Rightarrow \lambda_i = 0 = 0 = 0 \Rightarrow i \in \{i_1, i_2, \ldots, i_k\}$.

Because { $z_1, z_2, z_3, \ldots, z_n$ } are linearly independent, from where the result.

**Description of the Active Method**

We focus in this section on a so-called active point approach.

We construct the iterated $x_{k+1}$ by the formula $x_{k+1} = x_k + \alpha_k d_k$ where $\alpha_k$ is the displacement step in the $d_k$ direction.

The choice of $\alpha_k$ and $d_k$ ensures that $x_{k+1}$ approaches the border of the constraints better than $x_k$ and $\mathcal{A}_{k+1} = (\mathcal{A}_k)^t$ (5.1) such as $\mathcal{A}_k$ is the matrix of active constraints at point $x_{k+1}$.

This process is repeated until the stopping test is satisfied.

**Initialization**

Location of the starting point $x_0$: Let $E$ be a set of constraints in a general form (equalities, inequalities, and mixed) be an arbitrary point and $x_0$ be a point of departure in IR$^n$.

We can distinguish the situation from the point $x_0$ with respect to $E$, in one of the following three cases:

**Case 1:** Point $x_0$ is located within $E$.

**Case 2:** The point $x_0$ is located outside of $E$.

**Case 3:** The point $x_0$ is located in the boundary of $E$.

**Geometric representation at point $x_0$**

**Adding and permutation of Constraints**

**Adding a constraint:** Let $A_k$ be the matrix of active constraints at $x_k$ point of iteration $k$, stitch-forming iteration $k + 1$, we add in the matrix $A_k$ the constraint $a_{k+1}$ resulting from the following two equations:

$a_{k+1} x_{k+1} - b_{k+1} = \max_{i \in \mathcal{I}(x_{k+1})} a_{i} x_{k+1} - b_{i}$ if $x_{k+1}$ is the result of the first or third case cited in sub-section 5.1.1.

Because, When the point $x_{k+1}$ is situated in the interior domain $E(\xi = 1)$, We seek the constraint that nears to this $x_k$ point in fact, if $i \in I_{x_{k+1}}$ it gives all $(a_i x_i - b_i) < 0$. Then we choose the constraint $(a_{k+1} x_{k+1} - b_{k+1})$ that have a negative maximum value $(a_{i} x_{i} - b_{i}) < 0$.

Note that $a_{k+1} x_{k+1} - b_{k+1} = \max_{i \in I_{x_{k+1}}} (a_i x_i - b_i)$

This result is obtained by replacing the $x_0$ point in all constraint of domain $E$ (Figure 1).

Else in other part if $x_{k+1}$ is the result of second case cited in sub-section 5.1.1. i.e.

The point $x_{k+1}$ is cited in exterior of $E$, we seek the constraint which is far to this point $x_{k+1}$ in fact.

If $i \in I_{x_{k+1}}$, it gives all $(a_{i} x_{i} - b_{i}) > 0$. Then we choose the constraint $(a_{i} x_{i} - b_{i}) > 0$ that have a positif maximum value $(a_{i} x_{i} - b_{i}) > 0$.

Note that $a_{i} x_{i} - b_{i} = \max_{i \in I_{x_{k+1}}} (a_i x_i - b_i)$

This result is obtained by replacing the $x_0$ point in all constraint of domain $E$.

**Permutation of constraints:** Let $x_k$ be the point in iteration $k$, in which two matrices are associated $A_k$, $Z_k$, and $i_k$ is the index on constraints that can be added to $A_k$ to obtain $A_{k+1}$, $z_k$ is the column that can be eliminated from the matrix $Z_k - 1$ to reach $Z_k$.

Permute the constraint of index $i_k$ by another constraint of index $i_0$ that result of equality: $a_{i_0} x_{i_0} - b_{i_0} = \max_{i \in \mathcal{I}(x_{i_0})} (a_i x_i - b_i)$

If and only if where the algorithm is moved from iteration $k$ to iteration $k + 1$ we meet the condition $a_{i_0} Z_k - Z_k'$

Where $n_k$ is the number of columns of matrix $Z_k$.

and { $z_{i_0}, i = 1, \ldots, k$ } is a set of columns eliminated on the matrix $Z_k$.

**Remarks:** (1) The rows of the matrix $A_k$, are linearly independent, they are also active at the point $x_{k+1}$.

(2) From the kernel of the matrix $A_k$, we can easily determine the $Z_k$ matrix whose columns form a basis of the kernel of $A_k$.

**Direction of displacement**

We consider the matrix $A_k$ composed of active constraints linearly independent at the point $x_k$ and the columns of the matrix $Z_k$ form a basis of the kernel of $A_k$. 
the constraint that may be added to the matrix $A_k$.

To determine the direction $d_k$, we distinguish two alternatives:

If $k=0$, we pose

$$I_k \xi = \nabla - \ldots$$

Where, $i_k$ is the index of the constraint to added to $A_k$.

$\xi$ is indicative of the position $x_k$, and

if $x_k$ is the result of the second case. (§5.1.1)

$\xi=1$ if $x_k$ is the result of another case. (§5.1.1)

If $k \neq 0$, here, we find also two other alternative:

If $a'_{i_k} Z_k \neq 0_{a_k}$, the direction $d_k$ is resulted by solution of the following linear system:

$$\begin{bmatrix} A_k \\ a'_{i_k} \\ Z_k \end{bmatrix} \begin{bmatrix} x_k \\ b_k \end{bmatrix} \xi = \begin{bmatrix} 0 \\ -\xi \\ 0 \end{bmatrix}$$

Where $Z_k=\{Z_i，z_i\}$, $\xi = \begin{cases} -1 if x_k \in \text{case 2}(§4-1-1) \\ 1 if x_k \in E \end{cases}$

and $z_k$ is the column that can be eliminated from the matrix $Z_{k-1}$ to obtain $Z_k$.

If $a'_{i_k} Z_k = 0_{a_k}$, the direction $d_k$ is resulted by the solve of the following linear system:

$$\begin{bmatrix} A_k \\ a'_{i_k} \\ Z_k \end{bmatrix} \begin{bmatrix} x_k \\ b_k \end{bmatrix} \xi = \begin{bmatrix} 0 \\ \xi \\ 0 \end{bmatrix}$$

Step of displacement

Let $x_k$ is the point of iteration $k$, and $d_k$ the direction of displacement at the point $x_k$.

After finding the associated constraint of iteration $(k+1)$ which is active at the point $x_{k+1}$, then $\alpha_{i_k} x_{k+1} = b_{i_k}$. From the determination of the direction $d_k$, it comes that $\alpha_{i_k} d_k = \xi$. Which gives $\alpha_{i_k} x_{k+1} - b_{i_k} = -\xi / \alpha_{i_k}$ as $\xi > 0$ and $\xi = 1$.

We conclude that $\alpha_k = Abs(\alpha_{i_k}, x_{k+1})$. So, the step in the direction of displacement $dk$ denoted by $\alpha_k$ is given by the following expression

$$\alpha_k = Abs(b_k - d_k, x_k)$$

Where $ik$ is the index of constraint to added in the matrix $A_k$.

Remarks: The active constraint at $x_k$ is also active at the point $x_{k+1}$.

Theorem of convergence

Let $(x_k)_{k \in \mathbb{N}}$ be an iterative sequence defined by $x_k = x_{k-1} + \alpha_k, d_{k-1}$, where $\alpha_{k-1}$ is the displacement step along the direction $d_{k-1}$, and $x_k$ is a finite starting point of IR$^n$.

Then the sequence $(x_k)_{k \in \mathbb{N}}$ formed by a set of the directions
(d_k)_{k=0}^{\infty} for the two cases (§5-1-1) is convergent after a finite number of iterations.

**Proof**

It sufficiently to show that the set of direction (d_k)_{k=0}^{\infty} is linearly independent.

By recurrence we can write that the scalar product

\[ \sum_{i=1}^{n} \lambda_i d_i = 0 \] such that \( \lambda_i \in IR \) and \( d_i \) the set of direction.

First we consider \( \lambda_1 d_1 + \lambda_2 d_2 = 0 \), we have \( d_1', d_2' \neq 0 \) and \( d_i' = 0 \) then \( \lambda_1 \xi + \lambda_2 \xi = 0 \) and \( \lambda_1 = \lambda_2 = 0 \)

we replace in (*) we obtain \( \lambda_2 d_2 = 0 \)

we as know \( d_2 \) is a non-null direction, then it result \( \lambda_2 = 0 \)

now we have \( \lambda_1 = \lambda_2 = 0 \)

we suppose that are true for all step \( m \), and we proof it for step \( m+1 \).

\( \lambda_1 d_1 + \lambda_2 d_2 + \ldots + \lambda_m d_m = 0 \)

we have \( a_i' d_i = \xi \neq 0 \) and \( a_i' d_i' = 0 \) then \( \lambda_1 \xi + \lambda_2 \xi = 0 \)

we replace in (*) we obtain \( \lambda_2 d_2 = 0 \)

we as know \( d_2 \) is a non-null direction, then it result \( \lambda_2 = 0 \)

now we have \( \lambda_1 = \lambda_2 = 0 \)

Finally the set of direction \( (d_k)_{k=0}^{\infty} \) is linearly independent.

**Algorithm for the Active Method**

**Data:** The matrix \( A \), the vector \( b \) and the departure \( x_0 \).

**Output:** The point to find is active exact point.

1- Choose an arbitrary starting point, \( x_i \) in \( IR^a \), set \( k=0 \).

2- As long as stopping criterion is defined.

a) Computation of a searched direction, calculate \( d_0 \).

b) Determine the step \( a_i \), and the new point \( x_{i+1} = x_i + a_i d_i \) and add the active constraint \( a_{i+1} \) in \( x_{i+1} \).

c) Test: if \( a_i' Z_a = 0 \), we call the permute procedure.

d) Construct the active matrix
\[
A_{i+1} = \begin{bmatrix} A_i \\ d_a \end{bmatrix}
\]

The same way, we calculate the basis of \( KerA_{i+1} \).

e) \( K = K + 1 \) and return to a).

**Remarks:**

i) This method determines the active points of a problem \( E \), without any constraints condition i.e., it does not require to make the linearly independent constraints.

ii) This method can be applied to any set \( E \), defined by linear constraints, and even if it is empty.

**Numerical Tests**

From a practical point of view, our method has remarkable advantages.

This will be shown by numerical application of this method in different cases that may exist: the number of constraints, the number of variables, and the size of the matrix to be taken.

The obtained results are listed in the following tables:

**Case 1: Standard form (Small size, Large size)**

a - Let \( m=11 \) and \( n=5 \)

\[
\begin{bmatrix} 1 & 1 & 1 & -1 & 0 \\ -1 & -1 & -1 & 1 & 0 \\ 2 & 1 & 3 & 0 & -1 \\ -2 & -1 & -3 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 
\end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 8 \\ -8 \\ 20 \\ -20 \\ 5 
\end{bmatrix}
\]

\[ A = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 
\end{bmatrix}, \quad \begin{bmatrix} b \\ -5 \\ 0 \\ 0 \\ 0 
\end{bmatrix}
\]

Table 1 shows the Standard form of small size and large size \[9\].

b - Let \( m=26 \) and \( n=12 \)

\[
\begin{array}{ccccccc}
X_{11} + X_{12} + X_{13} + X_{14} &=& 50 \\
X_{21} + X_{22} + X_{23} &=& 30 \\
X_{31} + X_{32} + X_{33} + X_{34} &=& 70 \\
X_{11} + X_{12} &=& 30 & x_i & 0 \quad i = 1, 2, 3 & and & j = 1, \ldots, 4 \\
X_{12} + X_{13} &=& 60 & X_{11} + X_{12} + X_{13} &=& 20 & X_{14} + X_{24} + X_{34} &=& 40 \\
\end{array}
\]

Table 2 shows the inequality form in Large size.

\[
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} = \begin{bmatrix} A \\ A_2 \\ A_3 \\ A_5 \\ A_6 
\end{bmatrix}
\]

\[ A = \begin{bmatrix} 8 & -8 \\ -20 & 20 \\ -1 & -20 \\ 5 & 5 \\ 0 & 0 
\end{bmatrix}, \quad \begin{bmatrix} b \\ -5 \\ 0 \\ 0 \\ 0 
\end{bmatrix}
\]

Table 2 shows the inequality form in Large size.
Case 2: Inequality form (Large size)

Let \( m=39 \) and \( n=10 \)

\[
A = \begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4 \\
A_5
\end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4 \\
b_5
\end{bmatrix}
\]

\[
A_1 = \begin{bmatrix}
9.119 & 61.555 \\
-9.119 & -61.555 \\
0 & -61.555 \\
0 & -3.475 \\
3.475 & 0
\end{bmatrix}
\]

\[
A_2 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}
\]

\[
A_3 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
A_4 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
A_5 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
b = \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4 \\
b_5
\end{bmatrix} = \begin{bmatrix}
50 \\
100 \\
100 \\
100 \\
100
\end{bmatrix}
\]

\[
A_1 b = \begin{bmatrix}
-3.475 \\
-7.407 \\
9.119 \\
-3.475 \\
-7.407
\end{bmatrix}
\]

\[
A_2 b = \begin{bmatrix}
50 \\
50 \\
100 \\
100 \\
0
\end{bmatrix}
\]

\[
A_3 b = \begin{bmatrix}
100 \\
100 \\
100 \\
100 \\
100
\end{bmatrix}
\]

\[
A_4 b = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

\[
A_5 b = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

Table 3 shows the mixed form in large size.

Case 3 Mixed form (Large size)

\[ a \cdot \text{Let} \ m=23 \ \text{and} \ n=7 \]

\[
A = \begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4 \\
A_5
\end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4 \\
b_5
\end{bmatrix}
\]

\[
A_1 = \begin{bmatrix}
9.119 & 61.555 \\
-9.119 & -61.555 \\
0 & -61.555 \\
0 & -3.475 \\
3.475 & 0
\end{bmatrix}
\]

\[
A_2 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}
\]

\[
A_3 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
A_4 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
A_5 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

\[
b = \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4 \\
b_5
\end{bmatrix} = \begin{bmatrix}
50 \\
100 \\
100 \\
100 \\
100
\end{bmatrix}
\]

\[
A_1 b = \begin{bmatrix}
-3.475 \\
-7.407 \\
9.119 \\
-3.475 \\
-7.407
\end{bmatrix}
\]

\[
A_2 b = \begin{bmatrix}
50 \\
50 \\
100 \\
100 \\
0
\end{bmatrix}
\]

\[
A_3 b = \begin{bmatrix}
100 \\
100 \\
100 \\
100 \\
100
\end{bmatrix}
\]

\[
A_4 b = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

\[
A_5 b = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]
Table 4: Standard form (large size).

| x0  | r0  | a1  | x1  | a2  | x2  | a3  | x3  | a4  | n  | r  | xsol |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|----|------|
| 8   | 2536.12 | 127,015 | 2,271 | 12.2 | 62,967 | 65,346 | 3.8 | -12.2 | 1  | -1600 | Is not found |

Table 5: Mixed form (large size).

| x0  | x1  | r0  | xact | xact | n  | A1 | A2 | z  | r  |
|-----|-----|-----|------|------|----|----|----|----|----|
| -8  | -8  | -30 | -8   | -9.18 | 4  | A1 | A2 | -1.62×10^4 |
| -8  | -8  | -10 | -8   | -9.36 | 4  | A1 | A2 | -9.41 |
| -8  | -10 | -10 | -2.87 | -7.88 | 4  | A1 | A2 | -13.27 |
| -8  | -10 | -10 | -9.36 | -6.17 | 4  | A1 | A2 | -6.25 |
| -8  | -10 | -10 | -9.36 | -6.17 | 4  | A1 | A2 | -6.81 |
| -8  | -10 | -10 | -9.36 | -6.17 | 4  | A1 | A2 | -10.82 |

Table 6: Where, the remaining matrices are given by: Zx=observe in the above table.

| x0  | x1  | r0  | xact | xact | n  | A1 | A2 | z  | r  |
|-----|-----|-----|------|------|----|----|----|----|----|
| 1   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 1   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0   | 1   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | -0.15 | 0   | -1.1 | -0.15 | 1.35 | -1.5 | 1.35 | -0.3 | -0.4 |
| 0   | 0   | 0   | 1    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | -0.25 | 0   | -1.5 | -0.25 | -0.75 | 0.5  | 2.25 | -0.5 | 0  |
| 0   | 0   | 0   | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| 0   | 0.05 | 0   | 0.7  | 0.05 | -0.45 | 1.5  | -0.45 | 0.1  | 0.8 |
| 0   | 0.65 | 0   | 2.1  | 0.65 | 0.149 | 0.5  | -2.85 | 0  |

Discussion

The numerical results obtained in the previous examples, show the efficiency of the proposed a new method to solve any initialization problem of optimization with linear constraints.
From a numerical point of view, it is difficult to take the best starting point in IR^n, which helps us to obtain easily the active point that we search.

In a numerical application, it is competent to verify whether the domain of optimization is empty or not. This problem is very easy to solve by our method.

We easily can know the state of the domain. This has been illustrated in the numerical test-case 3 (Table 4).

All the above results show the efficiency of this method in the problem of optimization, where the domains are consecutive, and with small size.

For large size, the problem is substantially the same; one has only to do a large amount of calculations.

So, the discussion is similar to that of domains of small sizes.

These results are showed in examples of three cases.

**Conclusion**

After a long scientific research, we have not found anything on the method that discusses to solve this type of continues problem optimization, and then we have suggested this method with a new formula direction d_k.

In this work, we studied theoretically and algorithmically an active method, which determines the extremes of a set defined by linear constraints. This set is in the form of equalities, inequalities or both of them. These m constraints are linear and function of n variables, our results can be given in the following points:

- Starting from any initial point, it generates points belonging to the set E.
- It is possible to construct from the m constraints two matrices, where the lines of the first are linearly independent and active, and the columns of the second form a basis of the kernel of the first matrix.
- Our method can be applied to matrices of large sizes.
- The active point is determined in at most n+ np iterations.
- The other advantage is the simplification of the computation, because each used constraint appears at most only once.
- Our method can be used in other algorithms of resolution of optimization problems to simplify their initializations and to improve their results.

**References**

1. Adler L, Karmarkar N, Resende MGC, Veiga G (1989) An implementation of the Karmarkar algorithm for linear programming. Math Program 44: 297-335.
2. Gill PE, Murray W, Saunders MA, Wright MH (1991) Inertia controlling method of the quadratic programming. SIAM Rev Soc Ind Appl Math 33: 1-36.
3. Mansour MS, Kadri EH, Kenai S, Ghrici M, Bennaceur R (2011) Influence of calcined kaolin on mortar properties. Construction and Building Materials 25: 2275-2282.
4. Altman A, Gondzio J (1998) Symmetric indefinite systems in interior point methods for linear and quadratic optimization. Optimization methods and Software, pp 275-302.
5. Gerard SG (2001) Introduction aux méthodes de point intérieurs.
6. Gondzio J, Sarkissian R (2003) Interior point solver for structured linear programs. Math Program 96: 561-584.
7. Adil MB, Zhang J (2003) Comparative analysis of the cutting angle and simulated annealing methods in global optimization. Journal of Mathematical Programming Operation Research 52: 363-378.
8. Chinchuluun A, Pardalos PM, Enkhbat R (2005) Global minimisation Algorithms for concave quadratic programming problems. Journal of Mathematical Programming Operation Research 54: 627-639.
9. Farouk A (2006) Programmation linéaire.
10. Azevedo AT, Oliveira LRA, Soares S (2008) Interior point method for long-term generation scheduling of large-scale hydrothermal systems. Ann Oper Res 169: 55.
11. Morales JL, Nocedal J, Wu Y (2012) A sequential quadratic algorithm with an additional equality constraint phase.IMA Journal of Numerical analysis 32: 553-579.
12. Dussquill JP (2011) Programmation non linéaire.
13. Culici J C (2012) Introduction à l’optimisation, pp 384.