Received 26 April 2016; | Revised 6 June 2016; | Accepted 6 June 2016

DOI: xxx/xxxx

ARTICLE TYPE

A logarithmic barrier method for linear programming based on a new minorant function

Linda Menniche*[1] | Djamel Benterki[2] | Imene Benchetta[3]

[1]LMAM,Faculty of Sciences, Jijel Mohammed Seddik ben Yahia University, Jijel,18000, Algeria
[2]Laboratory of Fundamental and Numerical Mathematics LMFN,Faculty of Sciences, Setif-1 Ferhat Abbas University, Setif,19000, Algeria
[3]Laboratory of Fundamental and Numerical Mathematics LMFN,Faculty of Sciences, Setif-1 Ferhat Abbas University, Setif,19000, Algeria

Correspondence
*Linda Menniche, LMAM,Faculty of Sciences, Jijel Mohammed Seddik ben Yahia University, Jijel,18000, Algeria Email: djbenterki@univ-setif.dz

Summary

This paper presents a logarithmic barrier method without line search for solving linear programming problem. The descent direction is the classical Newton's one. However, the displacement step is determined by a simple and efficient technique based on the notion of the minorant function approximating the barrier function. Numerical tests show the efficiency of this approach versus classical line search methods.

KEYWORDS:
Linear Programming, Logarithmic Barrier methods. Minorant function.

1 | INTRODUCTION

Linear programs are ubiquitous in many areas of applied science today. The primary reason for this is their flexibility: linear programs frame problems in optimization as a system of linear inequalities. This template is general enough to express many different problems in engineering, operations research, economics, and even more abstract mathematical areas such as combinatorics. The linear programming problem is usually solved through the use of one of two algorithms: either simplex, or an algorithm in the family of interior point methods. Karmarkar's[5] projective method has initiated the fast developing field of interior point methods for linear programming. Since than, many different interior point methods have been proposed which appear to have several similarity if one analysis then more carefully.

A well-known interior point method logarithmic barrier function method, proposed by Frish[4] and further developed by Fiacco and McCormick[3]. In this method the nonnegativity constraints of the linear programming problem are replaced by an additional term in the objective function.

Several algorithms have been proposed to solve the linear programming problem, by the algorithmic barrier methods[2], the two covered basic elements are the direction of displacement that dominates the cost computation at each iteration and the displacement step which plays an important role in the speed of convergence. Many alternatives are proposed to overcome the last difficulty[2,8]. Unfortunately, the computation of displacement step, especially while using line search methods is expensive and even more delicate in semidefinite programming problems[2]. Moreover, A. Leulmi et al.[7] proposed a minorant function to determine easily the displacement step. Unfortunately, this function is limited for linear programming problem with small size and does not work well for those of large size. The numerical tests that we effected prove it.

In this sens, we propose a new approximate function G_{eff} of the barrier function called minorant function to compute the displacement step by a simple and easy manner. This new minorant function G_{eff} is more efficient than that proposed by A. Leulmi

et al.[7], and the line search methods, as shown by the numerical experiments that we carried out.
The paper is organized as follows: In section 2, we describe the linear programming problem to be studied, thereafter we give his perturbed problem with some necessary notations and some theoretical results. In section 3, we discuss the numerical aspects of perturbed problem and we present briefly the barrier logarithmic algorithm. In section 4, we present our new minorant function to determine an effective displacement step. this procedure is used to avoid the line search methods and to accelerate the speed of convergence of the algorithm. In section 5, we present numerical tests on some different examples to illustrate the effectiveness of our new approach in comparison with the minorant function given by A. leulmi et al.[7], and classical line search methods.

2 | DESCRIPTION OF THE PROBLEM

We consider the following linear programming problem

$$(D)\begin{cases} \min b^t y \\ A^t y \geq c \\ y \in \mathbb{R}^m, \end{cases}$$

where $A \in \mathbb{R}^{m\times n}$ is a matrix of full rank ($\text{rank}(A) = m < n$).
c and b are a vectors of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively.
The dual problem associated to (D) is defined as follows:

$$(P)\begin{cases} \max c^t x \\ Ax = b \\ x \in \mathbb{R}^n \;\; x \geq 0. \end{cases}$$

The problem (D) can be written in the following standard form

$$\begin{cases} \min b^t y \\ A^t y - s = c \\ y \in \mathbb{R}^m \;\; s \in \mathbb{R}^n \; s \geq 0. \end{cases}$$

We denote by:

$Y = \left\{ y \in \mathbb{R}^m : A^t y - c \geq 0 \right\}$, the feasible solutions set of (D),
$\widehat{Y} = \left\{ y \in \mathbb{R}^m : A^t y - c > 0 \right\}$, the strictly feasible solutions set of (D),
$W = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$, the feasible solutions set of (P),
$\widehat{W} = \{x \in \mathbb{R}^n : Ax = b, x > 0\}$, the strictly feasible solutions set of (P),
The scalar product of two vectors u, v in $\mathbb{R}^n$ is defined by

$$< u, v >= u^t v = \sum_{i=1}^{n} u_i v_i.$$

The Euclidean norm of $u \in \mathbb{R}^n$ is $\|u\| =< u, u >^{\frac{1}{2}}$.
Throughout this paper, we assume that the sets $\widehat{Y}$ and $\widehat{W}$ are not empty.
To study (D), we associate it by the perturbed equivalent problem

$$(D_r)\begin{cases} \min f_r(y) \\ y \in \mathbb{R}^m. \end{cases}$$

Where $r > 0$ is the parameter barrier and f_r is the barrier function defined by

$$f_r(y) = \begin{cases} b^t y + nr \ln r - r \sum_{i=1}^{n} \ln < e_i, A^t y - c > & \text{if } A^t y - c > 0, \\ +\infty & \text{otherwise.} \end{cases}$$

Where $(e_1, e_2, \dots, e_n)$ is the canonical base in $\mathbb{R}^n$.

3 | THEORETICAL ASPECTS OF PERTURBED PROBLEM

3.1 | Existence of solution of the problem (D_r)

Firstly, we give the following definition

Definition 1.
Let h be a function defined from $\mathbb{R}^m$ to $\mathbb{R} \cup \{\infty\}$ and $\alpha \geq 0$. Then

1. The set $S_\alpha(h) = \{y \in \mathbb{R}^m,\ h(y) \leq \alpha\}$ is called the α-level set of h.
2. The function h is called inf-compact if the level sets $S_\alpha(h)$ are compact for all $\alpha > 0$.
3. The recession function of h is the function $(h)_\infty : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\}$ defined by
$$(h)_\infty(d) = \lim_{t\to\infty} \frac{f_r(y+td) - f_r(y)}{t}$$
4. The recession cone of h is the 0-level set of the recession function of h, denoted by $S_0((h)_\infty)$.

To prove that (D_r) has an optimal solution, it is sufficient to prove that f_r is inf-compact, which comes back in particular to show that the cone of recession:
$$S_0((f_r)_\infty) = \left\{ d \in \mathbb{R}^n, (f_r)_\infty(d) \leq 0 \right\}$$
amounts to zero, i.e.,
$$(f_r)_\infty(d) \leq 0 \Rightarrow d = 0$$
where $(f_r)_\infty$ is defined by
$$(f_r)_\infty(d) = \lim_{t\to\infty} \frac{f_r(y+td) - f_r(y)}{t} = b^t d.$$
This needs to the following lemma

Lemma 1. [8]

Suppose that $\widehat{Y}$ and $\widehat{W}$ are not empty, if $b^t d \leq 0$ and $A^t d > 0$ then $d = 0$.

3.2 | Uniqueness of the solution of the problem (D_r)

As the function (f_r) is strictly convex, then if an optimal solution of problem (D_r) exists, it is unique.
As consequence, we deduce that (D_r) has a unique optimal solution.

3.3 | Convergence of perturbed problem to (D)

Lemma 2. [8]

For $r > 0$, let y_r an optimal solution of the problem (D_r), then there exists $y \in Y$ an optimal solution of (D) such that: $\lim\limits_{r\to 0} y_r = y$.

4 | THE NUMERICAL ASPECTS OF PERTURBED PROBLEM

4.1 | Newton descent direction

The problem (D_r) can be considered as without constraints so, interior point methods of types logarithmic barrier are conceived for solving this problem type while being based on the optimality conditions which are necessary and sufficient.
As a consequence, (y_r) is an optimal solution of (D_r) when it satisfies the following condition:
$$\nabla f_r(y_r) = 0,$$
therefore, the solution to every iteration of Newton's method is given by $y_{k+1} = y_k + d_k$, where d_k is the solution of the linear system
$$[\nabla^2 f_r(y)]d_k = -\nabla f_r(y).$$

Remark 1.
The strict feasibility of iterate $y_{k+1} = y_k + d_k$ is not always guaranteed. To surmount this difficulty, we introduce a displacement step t_k and we put $y_{k+1} = y_k + t_k d_k$.

4.2 | Prototype algorithm

Begin algorithm

- **Initialization:** start with y_0 a strictly feasible solution of (D), $d_0 \in \mathbb{R}^m$, ε a given precision and $k = 0$.
- **While** $|b^t d_k| > \varepsilon$ **do**
  - Resolve the system $\nabla^2 f_r(y_k) d_k = -\nabla f_r(y_k)$
  - Compute the displacement step t_k
  - Take $y_{k+1} = y_k + t_k d_k$ and $k = k + 1$.
- **End while**
- **End algorithm.**

4.3 | Computation of the displacement step

The most known methods used to compute the displacement step t_k are the line search methods to minimizing the function

$$\theta(t) = \tfrac{1}{r}(f_r(y + td) - f_r(y)), \quad y + td \in \widehat{Y}.$$

Unfortunately, these methods are expensive and in general impossible in certain problem as semidefinite programming problems. For this reason, J.P. Crouzeix et al.[2] and L. Menniche et al.[8] have used the notion of the majorant function that approaches the function $\theta(t)$ and offers a displacement step with a simple technique for linear semidefinite programming and linear programming respectively.

Inspired by this idea, we propose in this work, to approach the function $\theta(t)$ by a simple minorant function giving at each iteration k, a displacement step t_k in an easy way, simple and much less expensive than line search methods.

Remark 2.
At each iteration, it is necessary that the point $y + td$ still in $\widehat{Y}$ to keep function $\theta(t)$ well defined. This in turns requires finding $\hat{t} > 0$ such that $y + td \in \widehat{Y}$ for any $t \in [0, \hat{t}[$.

Lemma 3. [8]
Let $\hat{t} = \sup\left\{t, 1 + tz_i > 0\right\}$ with $z_i = \frac{<e_i, A^t d>}{<e_i, A^t y - c>}$, $\forall i = 1, \dots, n$

$\forall t \in [0, \hat{t}]$, the following function $\theta(t)$ is well defined by

$$\theta(t) = (\sum_{i=1}^{n} z_i - \|z\|^2)t - \sum_{i=1}^{n} \ln(1 + tz_i),$$

with

$$\theta(0) = 0, \theta''(0) = -\theta'(0) = \| z \|^2 .$$

Unfortunately, it does not exist an explicit formula that gives an optimal displacement step $t_{opt} = \arg\min_{t \in [0, \hat{t}[} \theta(t)$, and the resolution of the equation $\theta'(t_{opt}) = 0$ through iterative methods need at each iteration to compute θ and θ'. These computations are too expensive, especially in the problems of large size. These difficulties conduct us to look for other new alternatives.
Now, we look for a minorant function G of the function θ on $[0, \hat{t}[$ which can be used as a lower approximation of θ. Such a lower approximation may be more efficient to manipulate than θ. The function G is chosen to be simple and close enough to θ and to satisfy the following properties

$$G(0) = 0 \text{ and } G''(0) = -G'(0) = \| z \|^2,$$

where G' and G'' denote the first and the second derivative of G respectively.

4.4 | Minorant functions of θ

Before determining these functions, we need the following theorem.
Let $(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, then its mean $\bar{x}$ and its standard deviation σ_x are respectively defined as

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \bar{x}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2.$$

Theorem 1. [2]
Assume that $x_i > 0$ for $i = \overline{1,n}$, then

$$n\ln(\bar{x} - \sigma_x\sqrt{n-1}) \leq A \leq \sum_{i=1}^{n}\ln(x_i) \leq B \leq n\ln(\bar{x}),$$

with

$$A = (n-1)\ln(\bar{x} + \tfrac{\sigma_x}{\sqrt{n-1}}) + \ln(\bar{x} - \sigma_x\sqrt{n-1}),$$
$$B = \ln(\bar{x} + \sigma_x\sqrt{n-1}) + (n-1)\ln(\bar{x} - \tfrac{\sigma_x}{\sqrt{n-1}}).$$

4.5 | First minorant function

This strategy consists to minimize the minorant approximation G of θ over $[0, \hat{t}[$. To be efficient, this approximation needs to be simple and sufficiently close to θ. In our case, it requires that

$$G(0) = \theta(0) = 0, G''(0) = \theta''(0) = -\theta'(0) = -G'(0) = \| z \|^2 .$$

Firstly, A. Leulmi et al.[7] give the following minorant function G_0 on $[0, \hat{t}[$

$$G_0(t) = \gamma_0 t - \ln(1 + \beta_0 t) - (n-1)\ln(1 + \alpha_0 t).$$

With $\gamma_0 = n\bar{z} - \| z \|^2, \alpha_0 = \bar{z} - \frac{\sigma_z}{\sqrt{n-1}}$ and $\beta_0 = \bar{z} + \sigma_z\sqrt{n-1}$.

The function G_0 is well defined when $t \leq \hat{t}_0$ with $\hat{t}_0 = \begin{cases} \frac{-1}{\alpha_0} & \text{if } \alpha_0 < 0, \\ +\infty & \text{if not.} \end{cases}$

Lemma 4. We have $G_0(t) \leq \theta(t)$, $\forall t \in [0, \hat{t}_0[$.

Proof. Using the previous theorem, we have

$$\sum_{i=1}^{n}\ln(x_i) \leq \ln(\bar{x} + \sigma_x\sqrt{n-1}) + (n-1)\ln(\bar{x} - \frac{\sigma_x}{\sqrt{n-1}}).$$

This implies that

$$\sum_{i=1}^{n}\ln(x_i) + t \| z \|^2 \leq \ln(\bar{x} + \sigma_x\sqrt{n-1}) + (n-1)\ln(\bar{x} - \frac{\sigma_x}{\sqrt{n-1}}) + t \| z \|^2,$$

which gives

$$n\bar{z}(t) - (\sum_{i=1}^{n}\ln(x_i) + \| z \|^2 t) \geq n\bar{z}t - [\ln(\bar{x} + \sigma_x\sqrt{n-1}) + (n-1)\ln(\bar{x} - \tfrac{\sigma_x}{\sqrt{n-1}}) + \| z \|^2 t].$$

Taking $x_i = 1 + tz_i$, for any $i = \overline{1,n}$ and $t \in [0, \hat{t}_0[$, hence $\bar{x} = 1 + t\bar{z}$ and $\sigma_x = t\sigma_z$, we get

$$\gamma_0 t - \ln(1 + \beta_0 t) - (n-1)\ln(1 + \alpha_0 t) \leq (n\bar{z} - \| z \|^2)t - \sum_{i=1}^{n}\ln(1 + tz_i).$$

With

$$\begin{cases} \gamma_0 = n\bar{z} - \| z \|^2, \\ \beta_0 = \bar{z} + \sigma_z\sqrt{n-1}, \\ \alpha_0 = \bar{z} - \frac{\sigma_z}{\sqrt{n-1}}. \end{cases}$$

So, we have shown that

$$G_0(t) \leq \theta(t) \text{ on } [0, \hat{t}_0[.$$

On the other hand, for any $t \in [0, \hat{t}_0[$, we have

$$\theta(0) = G_0(0) = 0, \theta'(0) = -G_0'(0) = \| z \|^2,$$
$$\theta''(0) = G_0''(0) = \| z \|^2 .$$

The function G_0 is strictly convex on $[0, \hat{t}_0[$ and $G_0'(t) < 0$. If $t \to \infty$ and since G_0 minimize θ which is inf-compact, G_0 admits a minimum on $[0, \hat{t}_0[$.

If $\hat{t}_0 < +\infty$ so $G_0(t) \to \infty$ if $t \to \hat{t}_0$. Consequently, G_0 admits a unique minimum on $[0, \hat{t}_0[$. This minimum is obtained at $\bar{t}_0$ such that $G_0'(\bar{t}_0) = 0$.

This leads us to solve the second order following equation

$$t^2 - 2b_0 t + c_0 t = 0.$$

With $b_0 = \frac{1}{2}(\frac{n}{\gamma_0} - \frac{1}{\beta_0} - \frac{1}{\alpha_0})$ and $c_0 = -\frac{\|z\|^2}{\alpha_0\beta_0\gamma_0}$

Let's take one root of the two roots $\bar{t}_0 = b_0 \pm \sqrt{b_0^2 - c_0}$ that belong to $[0, \hat{t}_0[$.

4.5.1 | Second minorant function

We can also think of better and simpler functions than the function G_0. We consider our new function G_{eff} defined by

$$G_{eff}(t) = t\frac{\| z \|^2}{\beta_0} - p\ln(1 + t\frac{\| z \|^2}{\beta_0}), \quad \forall t \geq 0, \quad 0 < p < 1.$$

Lemma 5.
G_{eff} is strictly convex for all $t \geq 0$, and we have

$$-\infty \leq G_{eff}(t) \leq \theta(t).$$

Proof.
One has $\theta(t) = n\bar{z}t - t \| z \|^2 - \sum_{i=1}^{n} \ln(1 + tz_i)$, we put

$$k(t) = \theta(t) - G_{eff}(t)$$

Then

$$k(0) = k'(0) = 0$$

and we have for all $t \geq 0$

$$k''(t) = \sum_{i=1}^{n} \frac{z_i^2}{(1 + tz_i)^2} - \frac{z_i^2}{(1 + t\frac{\|z\|^2}{\beta_0})^2} \geq 0.$$

because

$$|z_i| \leq \| z \| \text{ and } \beta_0 \leq \| z \| .$$

Which gives $k(t) \geq 0$, $\forall t \geq 0$.

So,

$$\theta(t) \geq G_{eff}(t), \forall t \geq 0.$$

■

We deduce that the function θ_{eff} reaches its minimum in one unique point $t_{eff} = \frac{\beta_0^2}{\|z\|^2}$.

5 | NUMERICAL EXPERIMENTS

In this section, we present a comparative tests on different numerical examples of linear programming problem taken from the literature[6]. The precision ε is taken between 10^{-2} and 10^{-6}. In the following tables:

Iter: indicate the number of iterations that have executed,

Time: indicate the time measured in seconds,
div: means that the algorithm has not converged,
ls: indicate the classical line search of Armijo-Goldstein-Price type.

5.1 Example with fixed size

Example 1:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

$b = (2, 2)^t$, $c = (1, 1, -1, -1)^t$.
The initial strictly feasible point is $y_0 = (1.5, 1.5)^t$.
The optimal solution is $y^* = (1, 1)^t$.

Example 2:

$$A = \begin{pmatrix} -2 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & -1 & -1 & -1 & -1 & -1 \end{pmatrix},$$

$b = (0, 0, -1)^t$, $c = (-3, 1, -1, 0, 0, 0)^t$.
The initial strictly feasible point is $y_0 = (-1, -1, -2)^t$.
The optimal solution is $y^* = (-0.5, -0.0713, -0.5)^t$.

Example 3:

$$A = \begin{pmatrix} -1 & 0 & 4 & -3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -5 & -3 & -1 & 0 & 1 & -3 & 0 & -1 & 0 & 0 & 0 & 0 \\ -4 & -5 & 3 & -3 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 & -1 & 5 & 0 & 0 & 0 & -1 & 0 & 0 \\ -2 & -1 & -1 & -1 & -2 & -2 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & 3 & -2 & 1 & -4 & -5 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix},$$

$b = (-1, -4, -4, -5, -7, -5)^t$, $c = (4, 5, 1, 3, -5, 8, 0, 0, 0, 0, 0, 0)^t$.
The initial strictly feasible point is $y_0 = (2, -4, -1, -1, -1, -1)^t$.
The optimal solution is $y^* = (-0.5, -1.5, 0, 0, -1.5, 0)^t$.

Example 4:

$$A = \begin{pmatrix} 1 & -1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 2 & 2 & 1 & 0 & 0 & 1 \end{pmatrix}$$

$b = (6, 2, 4)^t$, $c = (4, -2, -2, 0, 0, 0)^t$.
The initial strictly feasible point is $y_0 = (0.5, 1, 1)^t$.
The optimal solution is $y^* = (0, 0, 0)^t$.
The following table summarizes the results obtained for different sizes.

	$\bar{t}_{eff}$		$\bar{t}_0$		ls	
	Iter	Time(s)	Iter	Time(s)	Iter	Time(s)
Example 1	17	0.0149	19	0.0207	45	0.1961
Example 2	18	0.0198	19	0.0152	50	0.1236
Example 3	22	0.0380	div	div	46	0.2592
Example 4	13	0.0170	11	0.0178	45	0.2221

5.2 | Example with variable size

Example 5:

$$(DL)\begin{cases} \min \sum_{i=1}^{m} 2y_i \\ y_i - 1 \geq 0, i = 1, \ldots, m, n = 2m. \end{cases}$$

the initial strictly feasible point is $y_0 = (1.5, 1.5, \ldots, 1.5)^t \in \mathbb{R}^m$.
The optimal solution is $y^* = (1, 1, \ldots, 1)^t \in \mathbb{R}^m$.
The following table summarizes the results obtained for different sizes.

	$\bar{t}_{eff}$		ls	
size(m, n)	Iter	Time(s)	Iter	Time(s)
$(100, 200)$	32	5.6187	77	13.8407
$(200, 400)$	35	23.2758	78	33.1803
$(300, 600)$	37	66.2671	79	137.942
$(400, 800)$	38	138.543	79	242.434
$(500, 1000)$	39	294.9521	79	605.497
$(600, 1200)$	40	552.5750	79	1304.16
$(1000, 2000)$	42	2564.8	81	4850.9

Example 6:

$$A(i,j) = \begin{cases} 1 \text{ if } i = j \text{ or } j = m + i, \\ 0 \text{ if not.} \end{cases}$$

$$b(i) = 4; i = 1, \ldots, m.$$

$$c(i) = \begin{cases} -1 & i = 1, \ldots, m \\ 0 & i = m + 1, \ldots, n \end{cases}, n = 2m.$$

The initial strictly feasible point is $y^0 = (4, 4, \ldots, 4)^t \in \mathbb{R}^m$
The optimal solution is $y^* = (0, 0, \ldots, 0)^t \in \mathbb{R}^m$.
The following table summarizes the results obtained for different sizes.

	$\bar{t}_{eff}$		ls	
Size(m, n)	Iter	Time(s)	Iter	Time(s)
$(100, 200)$	35	6.1255	50	8.9519
$(200, 400)$	38	25.6466	51	32.1801
(300.600)	40	71.6350	52	89.4249
$(400, 800)$	41	128.7171	52	161.5772
$(500, 1000)$	42	320.9764	52	400.5419
$(600, 1200)$	43	643.0831	56	816.1027
$(1000, 2000)$	45	2754.8	58	3542.4

Example 7:

$$A(i,j) = \begin{cases} \frac{1}{i+j} & \text{if } \quad i, j = 1, \ldots, m, \\ 1 & \text{if } \quad j = m + i, \\ 0 & \text{if not.} \end{cases}$$

$$b(i) = \sum_{j=1}^{m} \frac{1}{j+i}, i = 1, \ldots, m.$$

$$c(i) = \begin{cases} b(i) + \frac{1}{i+1} & i = 1, \ldots, m \\ 0 & i = m + 1, \ldots, n \end{cases}, n = 2m.$$

The initial strictly feasible point is $y^0 = (1, 1, \ldots, 1)^t \in \mathbb{R}^m$.

The optimal solution is $y^* = (0, 0, \dots, 0)^t \in \mathbb{R}^m$.
The following table summarizes the results obtained for different sizes.

	$\bar{t}_{eff}$		ls	
Size(m, n)	Iter	Time(s)	Iter	Time(s)
$(100, 200)$	32	6.1143	47	8.5130
$(200, 400)$	34	20.6473	48	32.9780
$(300, 600)$	37	71.6678	48	88.2086
$(400, 800)$	38	142.7643	49	154.4763
$(500, 1000)$	39	301.4535	54	413.2245
$(600, 1200)$	44	557.5541	53	735.1828
$(1000, 2000)$	42	2567.3	53	3239.6

Comments
These numerical tests, show that the new minorant function, gives an optimal solution in reasonable time with small number of iterations in comparison with the minorant function of A. Leulmi et al.[7], and classical line search of Armijo-Goldstein-Price. Note that if the dimension of the problem becomes important, the convergence of the algorithm based on the first minorant function[7] is no guaranteed. For this raison, we havn't presented in tables of examples 5, 6 and 7 the optimal solution $\bar{t}_0$ of G_0, because the algorithm diverges in this case.

6 | CONCLUSION

In this work, we have presented a logarithmic barrier method for solving the linear programming problem. Since the perturbed is strictly convex, the KKT condition are necessary and sufficient. For this, we use Newton's method that allows us to compute a good descent direction and determine a new iterate, better than the current iterate.
To compute the displacement step, we have proposed in this work a new approach based on minorant functions. This allows us to compute the effective displacement step by a simple and easy manner.
The numerical tests above show that the new minorant function that we have developed in this paper leads to a significant reduction in computational time and number of iterations and an improvement in the quality of the results in comparison with the classical line search methods.
The minorant function technique is a very reliable alternative that will be confirmed as an effective technique for determining displacement steps in logarithmic barrier methods for linear programming. This technique will be also extended favorably for other classes of optimization problems.

References

1. J.P. Crouzeix, A. Seeger, New bounds for the extreme values of a finite sample of real numbers, *Journal of Mathematical Analysis and Applications*, **197** (2008) 411–426.

2. J.P. Crouzeix, B. Merikhi, A logarithm barrier method for semidefinite programming, *RAIRO-Operations Research*, **42** (2008) 123–139.

3. A.V. Fiacco, G.P. McCormick, Nonlinear programming: Sequential unconstrained minimization techniques, *Wiley Reprinted as volume 4 of SIAM Classics in Applied Mathematics Series*, (1990).

4. R.A.K. Frish, The logarithmic potential method of convex programming, *Technical report, University Institute of Economics, Olso, Noway* (1955).

5. N. Karmarkar, A new polynomial-time algorithm in linear programming, *Combinatorica* **4** (1984) 373–395.

6. A. Keraghel, D. Benterki, Sur les performances de l'algorithme de Karmarkar pour la programmation linéaire, *Revue Roumaine des sciences techniques mécaniques appliquées*, **46** (2001) 87–96.

7. A. Leulmi, S. Leulmi, Logarithmic barrier method via minorant function for linear programming, *Journal of Siberian Federal University. Mathematics and Physic*, **2(2)** (2019) 91–201.

8. L. Menniche, D. Benterki, A logarithmic barrier approach for linear programming, *Journal of computational and Applied Mathematics* **312** (2017) 267–275.

How to cite this article: Williams K., B. Hoskins, R. Lee, G. Masato, and T. Woollings (2016), A regime analysis of Atlantic winter jet variability applied to evaluate HadGEM3-GC2, *Q.J.R. Meteorol. Soc., 2017;00:1–6*.