REVUE D'ANALYSE NUMÉRIQUE ET DE THÉORIE DE L'APPROXIMATION
Rev. Anal. Numér. Théor. Approx., vol. 41 (2012) no. 2, pp. 169–180
ictp.acad.ro/jnaat

GLOBAL CONVERGENCE OF THE ARMIJO EPSILON STEEPEST DESCENT ALGORITHM

NOUR EDDINE RAHALI*, NACERA DJEGHABA† and RACHID BENZINE‡

Abstract. In this article, we study the unconstrained minimization problem

$$(P) \quad \min\{f(x) : x \in \mathbb{R}^n\}.$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. We introduce a new algorithm which accelerates the convergence of the steepest descent method. We further establish the global convergence of this algorithm in the case of Armijo inexact line search.

MSC 2000. 90C30; 65K05; 49M37.
Keywords. Unconstrained optimization, global convergence, steepest descent algorithm, ε-algorithm, Armijo inexact line search.

1. INTRODUCTION

Consider the following unconstrained minimization problem:

$$\min\{f(x) : x \in \mathbb{R}^n\} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. Numerical methods for problem (1) are iterative. An initial point x_1 should be given, and at the k-th iteration a new iterate point x_{k+1} is to be computed by using the information at the current iterate point x_k and those at the previous points. It is hoped that the sequence $\{x_k\}_{k\in\mathbb{N}}$ generated will converge to the solution of (1).

Most numerical methods for unconstrained optimization can be classified into two groups, namely line search algorithms and trust region algorithms. A line search algorithm chooses or computes a search direction d_k at the k-th iteration, and it sets the next iterate point by

$$x_{k+1} = x_k + \alpha_k d_k$$

*Department of Mathematics, Souk Ahras University, Algeria, e-mail: rahali.noureddine@yahoo.fr.
†Department of Mathematics, Badji Mokhtar University, B.P. 12, Annaba, Algeria, e-mail: djeghaba.nacera@yahoo.fr.
‡Department of Mathematics, Badji Mokhtar University, B.P. 12, Annaba, Algeria, e-mail: rachid.benzine@univ-annaba.org.

where d_k is a descent direction of $f(x)$ at x_k and α_k is a step size. The search direction d_k is generally required to satisfy

$$\nabla f\left(x_k\right)^t d_k < 0,$$

which guarantees that d_k is a descent direction of $f(x)$ at x_k ([5], [28]). In line search methods, if the search direction d_k is given at the k-th iteration, then the next task is to find a step size α_k along the search direction. The ideal line search rule is the exact one that satisfies

$$f(x_k + \alpha_k d_k) = \min_{\alpha>0} f(x_k + \alpha d_k).$$

In fact, the exact step size is difficult or even impossible to obtain in practical computation. Thus many researchers constructed some inexact line search rules, such as Armijo rule, Goldstein rule and Wolfe rule ([4], [26], [34]). In this article, we use the Armijo's line search ([4]), which may be summarized as follows:

Armijo's line search ([4])

Armijo's Rule is driven by two parameters, $0 < c < 1$ and $\beta > 1$, which respectively manage the acceptable step length from being too large or too small. (Typical values are $c = 0.2,\ \beta = 2$). Suppose that $\nabla f\left(x_k\right)^t d_k < 0$. Define the functions $\varphi(\alpha)$ and $\widehat{\varphi}(\alpha)$ as follows:

$$\varphi(\alpha) = f(x_k + \alpha d_k), \quad \alpha \geq 0$$

$$\widehat{\varphi}(\alpha) = \varphi(0) + \alpha c \varphi'(0) = f(x_k) + \alpha c \nabla f\left(x_k\right)^t d_k, \quad \alpha \geq 0,\ 0 < c < 1.$$

A step length $\overline{\alpha}$ is considered to be acceptable, provided that

$$\varphi\left(\overline{\alpha}\right) \leq \widehat{\varphi}(\overline{\alpha}).$$

However, in order to prevent $\overline{\alpha}$ from being too small, Armijo Rule also requires that the following inequality holds

$$\varphi\left(\beta\overline{\alpha}\right) > \widehat{\varphi}(\beta\overline{\alpha}),$$

which yields an acceptable range for $\overline{\alpha}$.

Frequently, Armijo Rule is adopted in the following manner. A fixed step length parameter $\overline{\alpha}$ is chosen. If $\varphi\left(\overline{\alpha}\right) \leq \widehat{\varphi}(\overline{\alpha})$, then either $\overline{\alpha}$ is itself selected as the step size, or $\overline{\alpha}$ is sequentially-doubled (assuming $\beta = 2$) to find the largest integer $t \geq 0$ for which

$$\varphi\left(2^t\overline{\alpha}\right) \leq \widehat{\varphi}(2^t\overline{\alpha})$$

On the other hand, if $\varphi\left(\overline{\alpha}\right) > \widehat{\varphi}(\overline{\alpha})$, then $\overline{\alpha}$ is sequentially halved to find the smallest integer $t \geq 1$ for which

$$\varphi\left(\tfrac{\overline{\alpha}}{2^t}\right) \leq \widehat{\varphi}(\tfrac{\overline{\alpha}}{2^t}).$$

The steepest descent method is one of the simplest and the most fundamental minimization methods for unconstrained optimization. Since it uses the negative gradient as its descent direction, it is also called the gradient method.

For many problems, the steepest descent method is very slow. Although the method usually works well in the early steps, as a stationary point is approached, it descends very slowly with zigzagging phenomena. There are some ways to overcome these difficulties of zigzagging by deflecting the gradient. Rather then moving along $d = -\nabla f(x)$, we can move along $d = -D\nabla f(x)$ ([8], [9], [10], [12], [13], [14], [15], [17], [21], [22], [23], [24], [25], [31], [32], [33]) or along $d = -\nabla f(x) + g$ ([18], [19], [20], [21], [27], [29], [30]), where D is an appropriate matrix, and g is an appropriate vector.

In [16] Benzine and Djeghaba provided another solution to this problem by accelerating the convergence of the gradient method.

They achieved this goal by designing a new algorithm, named the epsilon steepest descent algorithm, in which the Wynn epsilon algorithm ([1], [6], [35], [36]) and exact line searches played a prominent role.

In this work we accelerate the convergence of the gradient method by using the Florent Cordellier epsilon algorithm ([11]).

We study the global convergence of the new algorithm, named the Armijo epsilon steepest descent algorithm, by using Armijo inexact line searches ([4])

2. THE EPSILON ALGORITHM

The Epsilon Algorithm is due to P. Wynn ([35], [36]).
Given a sequence $\{x_k\}_{k\in\mathbb{N}}$, $x_k \in \mathbb{R}^n$. The coordinates of x_k will be noted as follows:

$$x_k = (x_k^1, x_k^2, ..., x_k^i, ..., x_k^n) \in \mathbb{R}^n$$

For $i \in \{1, 2, ..., n\}$, the Epsilon Algorithm calculates quantities with two indices $\varepsilon_j^{k,i} (j, k = 0, 1, ...)$ as follows:

$$\begin{aligned} &\varepsilon_{-1}^{k,i} = 0 \qquad \varepsilon_0^{k,i} = x_k^i \qquad k = 0, 1, ... \\ &\varepsilon_{j+1}^{k,i} = \varepsilon_{j-1}^{k+1,i} + \frac{1}{\varepsilon_j^{k+1,i} - \varepsilon_j^{k,i}} \qquad j, k = 0, 1, ... \end{aligned} \tag{2}$$

For $i \in \{1, 2, ..., n\}$, these numbers can be placed in an array as follows:

This array is called the ε-array. In this array the lower index denotes a column while the upper index denotes a diagonal.

For $i \in \{1, 2, ..., n\}$, the Epsilon algorithm relates the numbers located at the four vertices of a rhombus:

$$\begin{array}{ccc} & \varepsilon_j^{k,i} & \\ \varepsilon_{j-1}^{k+1,i} & & \varepsilon_{j+1}^{k,i} \\ & \varepsilon_j^{k+1,i} & \end{array}$$

To calculate the quantities $\varepsilon_{j+1}^{k,i}$, we need to know the numbers $\varepsilon_{j-1}^{k+1,i}$, $\varepsilon_j^{k+1,i}$ and $\varepsilon_j^{k,i}$.

$\varepsilon_{-1}^{0,i}=0$							
	$\varepsilon_0^{0,i}=x_0^i$						
$\varepsilon_{-1}^{1,i}=0$		$\varepsilon_1^{0,i}$					
	$\varepsilon_0^{1,i}=x_1^i$		$\varepsilon_2^{0,i}$				
$\varepsilon_{-1}^{2,i}=0$		$\varepsilon_1^{1,i}$		$\varepsilon_3^{0,i}$			
	$\varepsilon_0^{2,i}=x_2^i$		$\varepsilon_2^{1,i}$		$\varepsilon_4^{0,i}$		
$\varepsilon_{-1}^{3,i}=0$		$\varepsilon_1^{2,i}$		$\varepsilon_3^{1,i}$		$\varepsilon_5^{0,i}$	
	$\varepsilon_0^{3,i}=x_3^i$		$\varepsilon_2^{2,i}$		$\varepsilon_4^{1,i}$		$\varepsilon_6^{0,i}$
$\varepsilon_{-1}^{4,i}=0$		$\varepsilon_1^{3,i}$		$\varepsilon_3^{2,i}$		$\varepsilon_5^{1,i}$	
	$\varepsilon_0^{4,i}=x_4^i$		$\varepsilon_2^{3,i}$	$\varepsilon_4^{2,i}$		$\varepsilon_6^{1,i}$	
$\varepsilon_{-1}^{5,i}=0$		$\varepsilon_1^{4,i}$		$\varepsilon_3^{3,i}$		$\varepsilon_5^{2,i}$	
	$\varepsilon_0^{5,i}=x_5^i$		$\varepsilon_2^{4,i}$		$\varepsilon_4^{3,i}$		$\varepsilon_6^{2,i}$
$\varepsilon_{-1}^{6,i}=0$		$\varepsilon_1^{5,i}$		$\varepsilon_3^{4,i}$		$\varepsilon_5^{3,i}$	
	$\varepsilon_0^{6,i}=x_6^i$		$\varepsilon_2^{5,i}$		$\varepsilon_4^{4,i}$		$\varepsilon_6^{3,i}$
$\varepsilon_{-1}^{7,i}=0$		$\varepsilon_1^{6,i}$		$\varepsilon_3^{5,i}$		$\varepsilon_5^{4,i}$	

Table 1. Epsilon Algorithm

3. THE ARMIJO EPSILON STEEPEST DESCENT ALGORITHM

To construct our algorithm, we use the column $\varepsilon_2^{k,i}$ $(i=1,2,...,n)$. Given a sequence $\left\{x_k^i\right\}_{k\in\mathbb{N}}$ $(i=1,2,...,n)$, F. Cordellier ([11]) proposed another formula to calculate the epsilon algorithm of order 2. The quantities $\varepsilon_2^{k,i}$can be calculated as follows

$$\varepsilon_2^{k,i}=x_{k+1}^i+\left[\frac{1}{x_{k+2}^i-x_{k+1}^i}-\frac{1}{x_{k+1}^i-x_k^i}\right]^{-1},\quad (i=1,2,...,n) \tag{3}$$

To calculate $\varepsilon_2^{k,i}$, we use the elements x_k^i, x_{k+1}^i and x_{k+2}^i $(i=1,2,...,n)$. Numerical calculations ([11]) showed that the epsilon algorithm of order 2 with the Cordellier formula (3) is more stable than Wynn epsilon algorithm (2).

We are now in measure to introduce our new algorithm: the Armijo epsilon steepest descent algorithm.

The Armijo epsilon steepest descent algorithm

Initialization step: Choose an initial point $x_0\in\mathbb{R}^n$ an initial point. The coordinates of x_0 will be noted as follows:

$$x_0=(x_0^1,x_0^2,...,x_0^i,...,x_0^n)\in\mathbb{R}^n$$

Let $k=0$ and go to Main step.

Main Step: Starting with the vector x_k,

$$x_k=(x_k^1,x_k^2,...,x_k^i,...,x_k^n).$$

If $\|\nabla f(x_k)\| = 0$, stop. Otherwise, let should be $r_k = x_k$ and compute the vectors s_k and t_k by using twice the steepest descent algorithm, with Armijo inexact line search

$$s_k = r_k - \lambda_k \nabla f(r_k),$$

and

$$t_k = s_k - \beta_k \nabla f(s_k),$$

λ_k and β_k are positive scalars obtained by the Armijo inexact line search.

If

$$s_k^i - r_k^i \neq 0, \quad t_k^i - s_k^i \neq 0 \quad \text{and} \quad \frac{1}{t_k^i - s_k^i} - \frac{1}{s_k^i - r_k^i} \neq 0, \quad i = 1, ...n$$

Let

$$\varepsilon_2^{k,i} = s_k^i + \left[\frac{1}{t_k^i - s_k^i} - \frac{1}{s_k^i - r_k^i}\right]^{-1}, \qquad i = 1, ...n,$$

and

$$\varepsilon_2^k = \left(\varepsilon_2^{k,1}, ..., \varepsilon_2^{k,i}, ...\varepsilon_2^{k,n}\right).$$

If $f\left(\varepsilon_2^k\right) < f(t_k)$, let $x_k = \varepsilon_2^k$. Replace k by $k+1$ and go to main step.
If $f\left(\varepsilon_2^k\right) \geq f(t_k)$ or if

$$s_k^{i_0} - r_k^{i_0} = 0 \quad \text{or} \quad t_k^{i_0} - s_k^{i_0} = 0 \quad \text{or} \quad \frac{1}{t_k^{i_0} - s_k^{i_0}} - \frac{1}{s_k^{i_0} - r_k^{i_0}} = 0, \ i_0 \in \{1, ..., n\}.$$

Let $x_k = t_k$. Replace k by $k+1$ and go to Main step.

Remark 1. According to the Algorithm, the vectors s_k and t_k are obtained by using twice the steepest descent method, with Armijo inexact line search. Then we have

$$f(s_k) < f(r_k) = f(x_k)$$

and

$$f(t_k) < f(s_k)$$

Now, by considering the Algorithm, if the calculation of ε_2^k is possible, two cases are possible:
a) $f\left(\varepsilon_2^k\right) < f\left(t_k\right)$. Then we have

$$f\left(x_{k+1}\right) = f\left(\varepsilon_2^k\right) < f\left(x_k\right)$$

b) $f\left(\varepsilon_2^k\right) \geq f\left(t_k\right)$ or if the calculation of ε_2^k is not possible. In this case and according to the algorithm we have

$$f\left(x_{k+1}\right) = f(t_k) < f\left(x_k\right)$$

In conclusion the Armijo epsilon steepest descent Algorithm guarantees

$$f\left(x_{k+1}\right) < f\left(x_k\right), \quad k = 0, 1, 2, ... \tag{4}$$

□

4. GLOBAL CONVERGENCE OF THE ARMIJO EPSILON STEEPEST DESCENT ALGORITHM

The foregoing preparatory results enable us to establish the following theorem

THEOREM 2. *For the unconstrained minimization problem* (1), *we let* x_0 *be a starting point of the Armijo epsilon steepest descent Algorithm, and assume that the following assumptions hold*

a) *The function* f *is continuously differentiable in a neighborhood* $\mathcal{L}$ *of the level set* $\delta(x_0) = \{x \in \mathbb{R}^n : \ f(x) \leq f(x_0)\}$.

b) *The gradient of* f *is Lipschitzian in* $\mathcal{L}$, *i.e. there exists* $K > 0$, *such that*

$$\|\nabla f(x) - \nabla f(y)\| \leq K \|x - y\|, \qquad \forall (x, y) \in \mathcal{L} \times \mathcal{L}$$

Then, the sequence $\{x_k\}_{k\in\mathbb{N}}$ *generated by the Armijo epsilon steepest descent algorithm must satisfy one of the properties:* $\nabla f(x_{k_0}) = 0$ *for some* $k_0 \in \mathbb{N}$, *or* $\|\nabla f(x_k)\| \underset{k\to\infty}{\longrightarrow} 0$.

Proof. Suppose that an infinite sequence $\{x_k\}_{k\in\mathbb{N}}$ is generated by the Armijo epsilon steepest descent Algorithm. In the main step of the Algorithm, the vectors s_k and t_k are obtained by using twice the steepest descent method, with Armijo inexact line search. The vectors s_k and t_k are the successors of x_k, and used to calculate x_{k+1} (see the main step of the algorithm). Note that

$$s_k = x_k - \lambda_k \nabla f(x_k),$$

$\lambda_k = \frac{\overline{\alpha}}{2^t}$ ($\overline{\alpha} > 0$ is a constant defined in the Armijo inexact line search) verifying the following Armijo criterion

$$\varphi\left(\tfrac{\overline{\alpha}}{2^t}\right) \leq \widehat{\varphi}\left(\tfrac{\overline{\alpha}}{2^t}\right) \tag{5}$$

with

$$\varphi(\tfrac{\overline{\alpha}}{2^t}) = f\left(x_k - \tfrac{\overline{\alpha}}{2^t} \nabla f\left(x_k\right)\right), \tag{6}$$

and

$$\widehat{\varphi}\left(\tfrac{\overline{\alpha}}{2^t}\right) = f(x_k) - \tfrac{\overline{\alpha}}{2^t} c \nabla f\left(x_k\right)^t \nabla f\left(x_k\right), \tag{7}$$

$c \in]0, 1[$, $\overline{\alpha} > 0$, $t \in \mathbb{N}$, such that the inequality (5) is satisfied. Taking into account the relations (5), (6) and (7), we obtain

$$\begin{aligned} f\left(s_k\right) &= f\left[x_k - \tfrac{\overline{\alpha}}{2^t} \nabla f\left(x_k\right)\right] \\ &\leq f(x_k) - c \tfrac{\overline{\alpha}}{2^t} \left\|\nabla f\left(x_k\right)\right\|^2. \end{aligned} \tag{8}$$

(8) implies

$$f\left(s_k\right) - f(x_k) \leq -c \tfrac{\overline{\alpha}}{2^t} \left\|\nabla f\left(x_k\right)\right\|^2. \tag{9}$$

On the other hand, by using the mean value theorem, we have

$$f\left(s_k\right) - f(x_k) = \left(s_k - x_k\right)^T \nabla f\left(\widetilde{x}\right); \quad \widetilde{x} = \lambda s_k + (1 - \lambda) x_k, \quad \lambda \in]0, 1[. \tag{10}$$

In as much as

$$s_k = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k = \frac{\overline{\alpha}}{2^t}, \tag{11}$$

then

$$\begin{aligned} f(s_k) - f(x_k) &= -\alpha_k \nabla f(x_k)^t . \nabla f(\widetilde{x}) \\ &= -\alpha_k \nabla f(x_k)^t [\nabla f(x_k) - \nabla f(x_k) + \nabla f(\widetilde{x})] \\ &= -\alpha_k \|\nabla f(x_k)\|^2 + \alpha_k \nabla f(x_k)^t [\nabla f(x_k) - \nabla f(\widetilde{x})]. \end{aligned} \tag{12}$$

$\widetilde{x} = \lambda s_k + (1-\lambda) x_k$, $\lambda \in]0,1[$. Noting that the sequence $\{f(x_k)\}_{k\in\mathbb{N}}$, is decreasing, then

$$x_k \in \delta(x_0), \quad k = 0, 1, ...$$

Now, by using the Cauchy Schwarz inequality and the fact that the gradient of f is Lipschitzian of constant K, we have

$$\begin{aligned} f(s_k) - f(x_k) &\leq -\alpha_k \|\nabla f(x_k)\|^2 + \alpha_k \|\nabla f(x_k)\| \|\nabla f(x_k) - \nabla f(\widetilde{x})\| \\ &\leq -\alpha_k \|\nabla f(x_k)\|^2 + \alpha_k \|\nabla f(x_k)\| K \|x_k - \widetilde{x}\| \end{aligned} \tag{13}$$

Noting that $\widetilde{x} = \lambda s_k + (1-\lambda) x_k, \quad \lambda \in]0,1[$, then

$$\|x_k - \widetilde{x}\| = \|(1-\lambda)(x_k - x_s)\| \leq |(1-\lambda)| \|(x_k - x_s)\| < \|(x_k - x_s)\|. \tag{14}$$

(13) and (14) imply

$$f(s_k) - f(x_k) \leq -\alpha_k \|\nabla f(x_k)\|^2 + \alpha_k \|\nabla f(x_k)\| K \|x_s - x_k\|$$

Remark that $x_s - x_k = x_k - \alpha_k \nabla f(x_k) - x_k = -\alpha_k \nabla f(x_k)$, $\alpha_k = \frac{\overline{\alpha}}{2^t}$, then

$$f(s_k) - f(x_k) \leq -\alpha_k \|\nabla f(x_k)\|^2 + \alpha_k \|\nabla f(x_k)\| K \alpha_k \|\nabla f(x_k)\|. \tag{15}$$

Hence, we obtain

$$f(s_k) - f(x_k) \leq -\frac{\overline{\alpha}}{2^t} \|\nabla f(x_k)\|^2 \left[1 - K \frac{\overline{\alpha}}{2^t}\right]. \tag{16}$$

Now choose $t > 0$, the smallest integer such that the following relation is true

$$2^t \geq \frac{K\overline{\alpha}}{1-c} \tag{17}$$

(17) implies

$$1 - K \frac{\overline{\alpha}}{2^t} \geq c. \tag{18}$$

Hence

$$1 - K \frac{\overline{\alpha}}{2^{t-1}} < c. \tag{19}$$

(18) implies

$$-\frac{\overline{\alpha}}{2^t} \|\nabla f(x_k)\|^2 \left[1 - K \frac{\overline{\alpha}}{2^t}\right] \leq -\frac{\overline{\alpha}}{2^t} c \|\nabla f(x_k)\|^2, \quad c \in]0,1[, \tag{20}$$

and (19) gives

$$-K \frac{\overline{\alpha}}{2^{t-1}} < c - 1 \Rightarrow K \frac{\overline{\alpha}}{2^{t-1}} > 1 - c.$$

Hence

$$-\frac{\overline{\alpha}}{2^t} c < -\frac{c(1-c)}{2K}. \tag{21}$$

Note that the choice of $t > 0$ satisfying (17) implies that the inequality (9) holds true. Therefore and taking into account the relations (16), (18), (19), (20), (21), we obtain

$$f(s_k) - f(x_k) \leq -\frac{c(1-c)}{2K} \|\nabla f(x_k)\|^2. \tag{22}$$

Denote by $G = -\frac{c(1-c)}{2K}$. It is clear that $G < 0$ and (22) gives

$$f(s_k) - f(x_k) \leq G \|\nabla f(x_k)\|^2. \tag{23}$$

Consider now t_k the successor of s_k, $t_k = s_k - \beta_k \nabla f(s_k)$. By doing the same with t_k, we obtain

$$f(t_k) - f(s_k) \leq G \|\nabla f(s_k)\|^2. \tag{24}$$

We will prove now that

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0.$$

To this end, consider

$$f(x_{k+1}) - f(x_k) = f\left(\varepsilon_2^k\right) - f(t_k) + f(t_k) - f(s_k) + f(s_k) - f(x_k)$$

Note that

$$f\left(\varepsilon_2^k\right) - f(t_k) < 0 \quad (see\ (4)).$$

then

$$f(x_{k+1}) - f(x_k) < f(t_k) - f(s_k) + f(s_k) - f(x_k)$$

The relations (23) and (24) imply

$$f(x_{k+1}) - f(x_k) < G\left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2\right). \tag{25}$$

Noting that $\{f(x_k)\}_{k\in\mathbb{N}}$ is a monotone decreasing sequence and so has a limit (otherwise $\inf f(x) = -\infty$). Hence, the relation (25) implies

$$\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 < \frac{1}{G}\left[f(x_{k+1}) - f(x_k)\right]. \tag{26}$$

Taking $\overline{\lim}$ as $k \to \infty$, we get

$$\overline{\lim_{k\to\infty}}\left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2\right) \leq 0 \tag{27}$$

On the other hand we have

$$\underline{\lim_{k\to\infty}}\left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2\right) \leq \overline{\lim_{k\to\infty}}\left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2\right) \tag{28}$$

and

$$0 \leq \underline{\lim_{k\to\infty}}\left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2\right) \tag{29}$$

The inequalities (27), (28) and (29) imply

$$0 \leq \underline{\lim}_{k\to\infty} \left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \right) \leq \overline{\lim}_{k\to\infty} \left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \right) \leq 0 \tag{30}$$

which implies

$$\underline{\lim}_{k\to\infty} \left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \right) = \overline{\lim}_{k\to\infty} \left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \right) = \lim_{k\to\infty} \left(\|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \right) = 0$$

Notice that we have

$$0 \leq \|\nabla f(x_k)\|^2 \leq \|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \tag{32}$$

and

$$0 \leq \|\nabla f(s_k)\|^2 \leq \|\nabla f(x_k)\|^2 + \|\nabla f(s_k)\|^2 \tag{33}$$

Finally, the inequalities (31), (32) and (33) imply

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = \lim_{k\to\infty} \|\nabla f(s_k)\| = 0.$$

□

5. NUMERICAL RESULTS AND COMPARISONS

In this section we report some numerical results obtained with an implementation of the Armijo Epsilon Steepest Descent algorithm. For our numerical tests, we used test functions and Fortran programs from ([2],[7]). Considering the same criteria as in ([3]), the code is written in Fortran and compiled with f90 on a Workstation Intel Pentium 4 with 2 GHz. We selected a number of 52 unconstrained optimization test functions in generalized or extended form [34] (some from CUTE library [7]). For each test function we have taken twenty (20) numerical experiments with the number of variables increasing as $n = 2, 10, 30, 50, 70, 100,\ 300, 500, 700, 900, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000$. The algorithm implements the Armijo line search conditions ([1]), and the same stopping criterion $\|\nabla f(x_k)\| < 10^{-6}$. In all the algorithms we considered in this numerical study the maximum number of iterations is limited to 100000.

The comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal value found by ALG1 and ALG2, for problem $i = 1, ..., 962$, respectively. We say that, in the particular problem i, the performance of ALG1 was better than the performance of ALG2 if:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3}$$

and the number of iterations, or the number of function-gradient evaluations, or the CPU time of ALG1 was less than the number of iterations, or

the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively.

In the set of numerical experiments we compare Armijo Epsilon Steepest Descent algorithm versus Steepest descent algorithm. Figure 1 shows the Dolan and Moré CPU performance profile of Armijo Epsilon Steepest Descent algorithm versus Steepest descent algorithm.

In a performance profile plot, the top curve corresponds to the method that solved the most problems in a time that was within a factor τ of the best time. The percentage of the test problems for which a method is the fastest is given on the left axis of the plot. The right side of the plot gives the percentage of the test problems that were successfully solved by these algorithms, respectively. Mainly, the right side is a measure of the robustness of an algorithm. When comparing Armijo Epsilon Steepest Descent algorithm with Steepest descent algorithm subject to CPU time metric we see that Armijo Epsilon Steepest Descent algorithm is top performer. The Armijo Epsilon Steepest Descent algorithm is more successful than the Epsilon Steepest Descent algorithm.

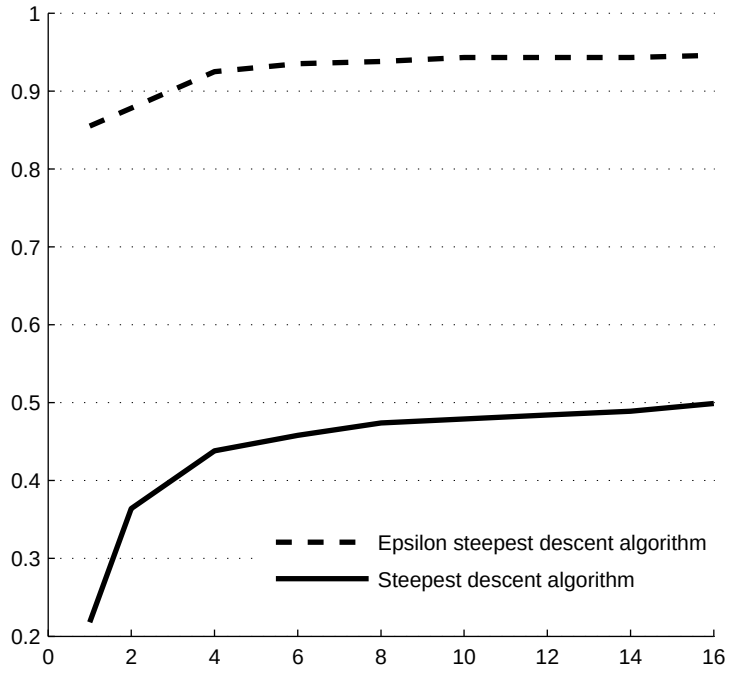

Fig. 1. The Dolan and Moré CPU performance profile of Armijo Epsilon Steepest Descent algorithm *versus* Steepest descent algorithm.

ACKNOWLEDGEMENT. The authors are very grateful to Professor Neculai Andrei for his help and his valuable comments and suggestions on the early version of the paper.

REFERENCES

[1] A.C. AITKEN, *On Bernoulli's numerical solution of algebraic equations*, Proc. Roy. Soc. Edinburgh, **46** (1926), pp. 289–305.
[2] N. ANDREI, *An unconstrained optimization test functions collection*, Advanced Modeling and Optimization, **10** (2008), pp. 147–161.
[3] N. ANDREI, *Another conjugate gradient algorithm for unconstrained optimization*, Annals of Academy of Romanian Scientists, Series on Science and Technology of Information, **1** (2008) no. 1, pp. 7–20.
[4] L. ARMIJO, *Minimization of functions having Lipschitz continuous first partial derivatives*, Pacific J. Mathematics, **16** (1966) 1, pp. 1–3.
[5] M.S. BAZARAA, H.D. SHERALI and C.M SHETTY, *Nonlinear Programing*, John Wiley & Sons, New York, 1993.
[6] C. BREZINSKI, *Acceleration de la convergence en analyse numérique*, Lecture Notes in Mathematics, **584** (1977), Springer Verlag.
[7] I. BONGARTZ, A.R. CONN, N.I.M. GOULD and P.L. TOINT, *CUTE: constrained and unconstrained testing environments*, ACM Trans. Math. Software, **21** (1995), pp.123–160.
[8] C.G. BROYDEN, *Quasi-Newton Methods and their application to Function Minimization*, Mathematics of Coputation, **21** (1967), pp. 368–381.
[9] C.G. Broyden, *The convergence of a class of double rank minimization algorithms 2. The new algorithm*, J. Institute of Mathematics and its applications, **6** (1970), pp. 222–231.
[10] C.G. BROYDEN, J.E. JR. DENNIS and J .J. MORÉ, *On the local and superlinear convergence of quasi-Newton methods*, J .Inst. Math. Appl., **12** (1973), pp. 223–246.
[11] F. CORDELLIER, *Transformations de suites scalaires et vectorielles*, Thèse de doctorat d'état soutenue à l'université de Lille I, 1981.
[12] W.C. DAVIDON, *Variable Metric Method for Minimization*, AEC research Development, Report ANL-5990, 1959.
[13] J.E.JR. DENNIS and J.J. MORÉ, *A characterization of superlinear convergence and its application to quasi-Newton methods*, Math. Comp., **28** (1974), pp. 549–560.
[14] J.E. DENNIS and J.J. MORÉ, *Quasi-Newton methods, motivation and theory*, SIAM. Rev., **19** (1977), pp. 46–89.
[15] L.C.W. DIXON, *Variable metric algorithms: necessary and sufficient conditions for identical behavior on nonquadratic functions*, J. Opt. Theory Appl., **10** (1972), pp. 34–40.
[16] N. DJEGHABA and R. BENZINE, *Accélération de la convergence de la méthode de la plus forte pente*, Demonstratio Mathematica., **39** (2006) No. 1, pp. 169–181.
[17] R. FLETCHER, *A new approch to Variable Metric Algorithms*, Computer Journal, **13** (1970), pp. 317–322.
[18] R. FLETCHER, *Practical methods of Optimization*, Second Edition, John Wiley & Sons, Chichester, 1987.
[19] R. FLETCHER, *An overview of unconstrained optimization*, Algorithms for Continuous Optimization: the State of Art, E. Spedicato, ed., Kluwer Academic Publishers, 1994.
[20] R. FLETCHER, and M. REEVES, *Function minimization by conjugate gradients*, Computer J., **7** (1964), pp. 149–154.
[21] R. FLETCHER, and M.M. POWELL, *A rapidly Convergent Descent Method for Minimization*, Computer Journal, **6** (1963), pp. 163–168.

[22] G.E. Forsythe, *On the asymptotic directions of the s-dimentional optimum gradient method*, Numerische Mathematik, **11**, pp. 57–76.
[23] P.E. Gill and W. Marray, *Quasi-Newton Methods for unconstrained optimization*, J.Inst. Maths applics, **9** (1972), pp. 91–108.
[24] A. Griewank, *The global convergence of partitioned BFGS on problems with convex decompositions and Lipschitz gradients*, Math. Prog., **50** (1991), pp. 141–175.
[25] D. Goldfarb, *A Family of Variable Metric Methods Derived by Variational Means*, Mathematics of Computation, **24** (1970), pp. 23–26.
[26] A.A. Goldstein and J.F. Price, *An effective Algorithm for Minimization*, Numerische Mathematik, **10** (1967), pp. 184–189.
[27] M.R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nation-Bureau Stand., **49** (1952) No. 6, pp. 409–436.
[28] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, Second edition, 2006.
[29] E. Polak and G. Ribiere, *Note sur la convergence de méthodes de directions conjuguées*, Revue Française Informatique et Recherche opérationnelle, **16** (1969), pp. 35–43.
[30] B.T. Polyak, *The Method of Conjugate Gradient in Extremum Problems*, USSR Computational Mathematics and Mathematical Physics, (English Translation), **9** (1969), pp. 94–112.
[31] M.J.D, Powell, *On the convergence of the variable metric algorithms*, J. Inst. Math. Appl., **7** (1971), pp. 21–36.
[32] M.J.D. Powell, *Some global convergence properties of variable metric algorithms for minimization without exact line searches*, Nonlinear Programming, SIAM-AMS Proceedings, **IX** (1976), R.W. Cottle and C.E. Lemke eds., SIAM.
[33] D.F. Shanno, *Conditionning of quasi-Newton Methods for function minimization*, Mathematics of Computation, **24** (1970), pp. 641–656.
[34] P. Wolfe, *Convergence conditions for ascent méthods*, Siam Review, **11** (1969), pp. 226–235.
[35] P. Wynn, *On a device for computing the $e_m(S_n)$ transformation*, M.T.A.C., **10** (1956), pp. 91–96.
[36] P. Wynn, *Upon systems of recursions which obtain among quotients of the Padé table*, Numer. Math., **8** (1966), pp. 264–269.

Received by the editors: January 23, 2012.