Numer Algor (2006) **42**: 63–73
DOI 10.1007/s11075-006-9023-9

An acceleration of gradient descent algorithm with backtracking for unconstrained optimization

Neculai Andrei

Received: 5 May 2005 / Accepted: 7 January 2006

Abstract In this paper we introduce an acceleration of gradient descent algorithm with backtracking. The idea is to modify the steplength t_k by means of a positive parameter θ_k, in a multiplicative manner, in such a way to improve the behaviour of the classical gradient algorithm. It is shown that the resulting algorithm remains linear convergent, but the reduction in function value is significantly improved.

Keywords acceleration methods · backtracking · gradient descent methods

AMS subject classifications 49M20 · 65K05 · 90C30

1. Introduction

One of the first and very well known method for unconstrained optimization

$$\min f(x) \tag{1}$$

where $f: R^n \to R$ is convex and continuously differentiable, is the gradient descent method, designed by Cauchy early in 1847 [3]. In this method the negative gradient direction is used to find the local minimizers. The algorithm starts with an initial point $x_0 \in \operatorname{dom} f$ and generates a sequence of points according to the following iterative procedure:

$$x_{k+1} = x_k + t_k d_k, \tag{2}$$

where t_k is the stepsize and $d_k = -g_k = -\nabla f(x_k)$.

The method proved to be effective for functions very well conditioned, but for functions poorly conditioned it is excessively slow, thus being of no practical value.

N. Andrei (✉)
Research Institute for Informatics, Center for Advanced Modeling and Optimization,
8-10, Averescu Avenue, Bucharest 1, Romania
e-mail: nandrei@ici.ro

Even for quadratic functions the gradient descent method with exact line search behave increasingly badly when the condition number of the matrix deteriorates. Early attempts to increase the performance of the method have been considered by Humphrey [9], Forsythe and Motzkin [7] and Schinzinger [14].

As we know, at the current point x_k the direction of the negative gradient is the best direction of search for a minimizer of function f, and this is the direction of gradient descent method. However, as soon as we move in this direction, it ceases to be the best and continue to deteriorate until it becomes orthogonal to $-\nabla f(x_k)$ That is, the method begin to take small steps without making significant progress to minimum. This is the major drawback of the gradient descent method, the steps it takes are too long, i.e. there are some other points z_k on the line segment connecting x_k and x_{k+1}, where $-\nabla f(z_k)$ provides a better new search direction than $-\nabla f(x_{k+1})$.

The purpose of this paper is to present an acceleration of the gradient descent method. The idea is to modify the steplength t_k (computed by backtracking) by means of a positive parameter θ_k in a multiplicative manner in such a way to improve the behaviour of the classical gradient algorithm. It is shown that the resulting algorithm is linear convergent, but the reducing in function value is significantly improved. The structure of the paper is as follows. In section two we present the line search with backtracking and prove that the Armijo rule in a backtracking scheme generates steplengths bounded away from zero. Section 3 presents the accelerated gradient descent algorithm. It is shown that the accelerated algorithm with backtracking reduces the function values with a factor that is smaller than the corresponding factor of the classical gradient descent algorithm. Some numerical examples and comparisons are given in Section 4.

2. Line search with backtracking

For implementing the algorithm (2) one of the crucial element is the stepsize computation. Many procedures have been suggested. In the *exact line search* the step t_k is selected as:

$$t_k = \arg\min_{t>0} f(x_k + t d_k). \tag{3}$$

In some special cases (for example quadratic problems) it is possible to compute the step t_k analytically, but in the most cases it is computed to approximately minimize f along the ray $\{x_k + td_k : t \geq 0\}$ or at least to reduce f sufficiently. In practice the most used are the *inexact procedures*. A lot of inexact line search methods have been proposed: Goldstein [8], Armijo [1], Wolfe [16], Powell [13], Dennis and Schnabel [4], Fletcher [6], Potra and Shi [12], Lemaréchal [10], Moré and Thuente [11], and many others. In particular, one of the very simple and efficient line search procedure is the backtracking line search. This procedure considers the following scalars $0 < \alpha < 0.5$, $0 < \beta < 1$ and $s_k = -g_k^T d_k / \|d_k\|^2$ and takes the following steps based on the Armijo's rule:

Backtracking procedure

Step 1. Consider the descent direction d_k for f at point x_k. Set $t = s_k$.
Step 2. While $f(x_k + td_k) > f(x_k) + \alpha t \nabla f(x_k)^T d_k$, set $t = t\beta$.
Step 3. Set $t_k = t$.

Typically, $\alpha = 0.0001$ and $\beta = 0.8$, meaning that we accept a small portion of the decrease predicted by linear approximation of f at the current point. Observe that, if $d_k = -g_k$, then $s_k = 1$.

Proposition 1. Suppose that d_k is a descent direction and $\nabla f(x)$ satisfies the Lipschitz condition $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ for all x,y in $\{x: f(x) \leq f(x_0)\}$, where L is a positive constant. If the line search satisfies the Armijo condition, then

$$t_k \geq \min\left\{1, \frac{\beta(1-\alpha)}{L} \frac{-g_k^T d_k}{\|d_k\|^2}\right\}. \tag{4}$$

Proof. Set $K_1 = \{k: t_k = s_k\}$ and $K_2 = \{k: t_k < s_k\}$. Then for all $k \in K_1$ we have

$$f_k - f_{k+1} \geq -\alpha s_k g_k^T d_k$$

and for all $k \in K_2$:

$$f_k - f_{k+1} \geq -\alpha t_k g_k^T d_k.$$

By Armijo rule since $t_k/\beta \leq s_k$ for all $k \in K_2$, we have

$$f_k - f\left(x_k + \frac{t_k}{\beta} d_k\right) < -\alpha \frac{t_k}{\beta} g_k^T d_k,$$

for all $k \in K_2$. Now, using the mean value theorem on the left side of this inequality, it follows that there exists $\xi_k \in [0,1]$ such that

$$g\left(x_k + \frac{t_k}{\beta}\xi_k d_k\right)^T d_k > \alpha g_k^T d_k,$$

for all $k \in K_2$.

Having in view the Lipschitz condition, from the above inequality, by Cauchy-Schwartz inequality, we have:

$$\begin{aligned} \frac{t_k}{\beta} L\|d_k\|^2 &\geq \left\|g\left(x_k + \frac{t_k}{\beta}\xi_k d_k\right) - g(x_k)\right\| \|d_k\| \geq \left(g\left(x_k + \frac{t_k}{\beta}\xi_k d_k\right) - g(x_k)\right)^T d_k \\ &\geq \alpha g_k^T d_k - g_k^T d_k = -(1-\alpha) g_k^T d_k, \end{aligned}$$

for all $k \in K_2$.

Rearranging this inequality and combining it with the inequality corresponding for $k \in K_1$ we get the bound (5). (See also [15].) □

3. Accelerated gradient descent algorithm

In this section we present an accelerated gradient descent algorithm for solving unconstrained optimization problem (1). Considering the initial point x_0 we can compute $f_0 = f(x_0)$, $g_0 = \nabla f(x_0)$ and by backtracking procedure (see Section 2) determine t_0. With these, the next iteration is computed as $x_1 = x_0 - t_0 g_0$ where again f_1 and g_1 can be immediately computed. Now, at the iteration $k = 1, 2, \ldots$ we know x_k, f_k and g_k. Using the backtracking procedure the stepsize t_k can be determined with which the following point $z = x_k - t_k g_k$ is computed. By backtracking procedure we get a $t_k \in (0,1]$ such that:

$$f(z) = f(x_k - t_k g_k) \leq f(x_k) - \alpha t_k g_k^T g_k.$$

With these, let us introduce the accelerated gradient descent algorithm by means of the following iterative scheme:

$$x_{k+1} = x_k - \theta_k t_k g_k, \tag{5}$$

where $\theta_k > 0$ is a parameter which follows to be determined in such a manner to improve the behaviour of the gradient descent algorithm.

Now, we have:

$$f(x_k - t_k g_k) = f(x_k) - t_k g_k^T g_k + \frac{1}{2} t_k^2 g_k^T \nabla^2 f(x_k) g_k + o\left(\|t_k g_k\|^2\right).$$

On the other hand, for $\theta > 0$ we have:

$$f(x_k - \theta t_k g_k) = f(x_k) - \theta t_k g_k^T g_k + \frac{1}{2}\theta^2 t_k^2 g_k^T \nabla^2 f(x_k) g_k + o\left(\|\theta t_k g_k\|^2\right).$$

We can write:

$$f(x_k - \theta t_k g_k) = f(x_k - t_k g_k) + \Psi_k(\theta), \tag{6}$$

where

$$\begin{aligned} \Psi_k(\theta) = (1-\theta) t_k g_k^T g_k - \frac{1}{2}\left(1-\theta^2\right) t_k^2 g_k^T \nabla^2 f(x_k) g_k \\ +\theta^2 t_k o\left(t_k \|g_k\|^2\right) - t_k o\left(t_k \|g_k\|^2\right). \end{aligned} \tag{7}$$

Let us denote:

$$\begin{aligned} a_k &= \quad t_k g_k^T g_k \geq 0, \\ b_k &= t_k^2 g_k^T \nabla^2 f(x_k) g_k, \\ \varepsilon &= \quad o\left(t_k \|g_k\|^2\right). \end{aligned}$$

Observe that for convex functions $b_k \geq 0$.

Therefore

$$\Psi_k(\theta) = (1-\theta) a_k - \frac{1}{2}\left(1-\theta^2\right) b_k + \theta^2 t_k \varepsilon - t_k \varepsilon. \tag{8}$$

Now, we see that $\Psi_k{}'(\theta) = (b_k + 2t_k\varepsilon)\theta - a_k$ and $\Psi_k{}'(\theta_m) = 0$ where

$$\theta_m = \frac{a_k}{b_k + 2t_k\varepsilon}.$$

Observe that $\Psi_k'(0) = -a_k < 0$. Therefore, $\Psi_k(\theta)$ is a convex quadratic function with minimum value in point θ_m and

$$\Psi_k(\theta_m) = -\frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2(b_k + 2t_k\varepsilon)} \leq 0.$$

Considering $\theta = \theta_m$ in (6), and $b_k \geq 0$ we see that for every k

$$f(x_k - \theta_m t_k g_k) = f(x_k - t_k g_k) - \frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2(b_k + 2t_k\varepsilon)} \leq f(x_k - t_k g_k),$$

which is a possible improvement on the values of function f, (when $a_k - (b_k + 2t_k\varepsilon) \neq 0$).

Therefore, using this simple multiplicative modification of the stepsize t_k as $\theta_k t_k$ where $\theta_k = \theta_m = a_k/(b_k + 2t_k\varepsilon)$ we get:

$$
\begin{aligned}
f(x_{k+1}) &= f(x_k - \theta_k t_k g_k) \le f(x_k) - \alpha t_k g_k^T g_k - \frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2\,(b_k + 2t_k\varepsilon)} \\
&= f(x_k) - \left[\alpha a_k + \frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2(b_k + 2t_k\varepsilon)}\right] \le f(x_k),
\end{aligned} \tag{9}
$$

since

$$
\alpha a_k + \frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2(b_k + 2t_k\varepsilon)} \ge 0.
$$

Observe that, if $a_k < b_k$ then

$$
\frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2(b_k + 2t_k\varepsilon)} > \frac{(a_k - b_k)^2}{2b_k}
$$

and from (9) we get:

$$
\begin{aligned}
f(x_{k+1}) &\le f(x_k) - \left[\alpha a_k + \frac{(a_k - (b_k + 2t_k\varepsilon))^2}{2\,(b_k + 2t_k\varepsilon)}\right] \\
&< f(x_k) - \left[\alpha a_k + \frac{(a_k - b_k)^2}{2\,(b_k)}\right] \le f(x_k).
\end{aligned}
$$

Therefore, in this case neglecting the contribution of ε we still get an improvement of the function values.

Now, in order to establish the algorithm we must determine a way for b_k computation. For this, at point $z = x_k - t_k g_k$ we have:

$$
f(z) = f(x_k - t_k g_k) = f(x_k) - t_k g_k^T g_k + \frac{1}{2} t_k^2 g_k^T \nabla^2 f(\tilde{x}_k) g_k,
$$

where $\tilde{x}_k$ is a point on the line segment connecting x_k and z. On the other hand, at point $x_k = z + t_k g_k$ we have:

$$
f(x_k) = f(z + t_k g_k) = f(z) + t_k g_z^T g_k + \frac{1}{2} t_k^2 g_k^T \nabla^2 f(\bar{x}_k) g_k,
$$

where $g_z = \nabla f(z)$ and $\bar{x}_k$ is a point on the line segment connecting x_k and z.

Having in view the local character of searching and that the distance between x_k and z is enough small, we can consider $\tilde{x}_k = \bar{x}_k = x_k$. With these, adding the above equalities we get:

$$
b_k = t_k^2 g_k^T \nabla^2 f(x_k) g_k = -t_k y_k^T g_k, \tag{10}
$$

where $y_k = g_z - g_k$.

As we said above, since along the iterations of gradient descent algorithm $g_k \to 0$ we can neglect the contribution of ε and consider in our algorithm $\theta_k = \theta_m = a_k/b_k$ With this we can present our algorithm.

Accelerated Gradient Descent Algorithm (AGD)

Step 1. Consider a starting point $x_0 \in \operatorname{dom} f$ and compute: $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $k = 0$.

Step 2. Using the line search with backtracking from Section 2 determine the step length t_k.
Step 3. Compute: $z = x_k - t_k g_k$, $g_z = \nabla f(z)$ and $y_k = g_z - g_k$.
Step 4. Compute: $a_k = t_k g_k{}^T g_k$, $b_k = -t_k y_k^T g_k$ and $\theta_k = a_k / b_k$.
Step 5. Update the variables: $x_{k+1} = x_k - \theta_k t_k g_k$ and compute f_{k+1} and g_{k+1}.
Step 6. Test a criterion for stopping the iterations. If the test is satisfied, then stop; otherwise consider $k = k + 1$ and go to step 2. □

The gradient descent (GD) algorithm can be immediately particularized from AGD by skipping steps 3 and 4 and considering $\theta_k = 1$ in step 5 where variables are updated.

Observe that if $a_k > b_k$, then $\theta_k > 1$. In this case $\theta_k t_k > t_k$ and it is possible that $\theta_k t_k \leq 1$ or $\theta_k t_k > 1$, i.e. it is possible that the steplength $\theta_k t_k$ to be greater than 1. On the other hand, if $a_k \leq b_k$, then $\theta_k \leq 1$. In this case $\theta_k t_k \leq t_k \leq 1$ i.e. the steplength $\theta_k t_k$ is reduced. Therefore, if $a_k \neq b_k$ then $\theta_k \neq 1$ and the steplength t_k computed by backtracking will be modified, by its increasing or reducing through factor θ_k, thus avoiding the algorithm to take orthogonal steps along the iterations.

Neglecting ε, from (8), we see that if $a_k \leq b_k/2$ then $\Psi_k(0) = a_k - b_k/2 \leq 0$ and $\theta_k < 1$. For any $\theta \in [0,1]$, $\Psi_k(\theta) \leq 0$. As a consequence for any $\theta \in (0,1)$, $f(x_k - \theta t_k g_k) < f(x_k)$. In this case, for any $\theta \in [0,1]$, $\theta_k t_k \leq t_k$. However, in our algorithm we selected $\theta_k = \theta_m$ as the point achieving the minimum value of $\Psi_k(\theta)$.

Proposition 2. Suppose that f is a strongly convex function on the level set $S = \{x: f(x) \leq f(x_0)\}$. Then the sequence x_k generated by AGD converges linearly to x^*.

Proof. From (9) we have that $f(x_{k+1}) \leq f(x_k)$ for all k. Since f is bounded below, it follows that

$$\lim_{k\to\infty} (f(x_k) - f(x_{k+1})) = 0.$$

Since f is strongly convex there are positive constants m and M such that $mI \leq \nabla^2 f(x) \leq MI$ on the level set S. Suppose that $x_k - tg_k \in S$ and $x_k - \theta_m tg_k \in S$, for $0 < t \leq 1$.

Supposing that $a_k < b_k$ we have

$$f(x_k - \theta_m t g_k) \leq f(x_k - tg_k) - \frac{(a_k - b_k)^2}{2b_k}.$$

But, from strong convexity we have the following quadratic upper bound on $f(x_k - tg_k)$:

$$f(x_k - tg_k) \leq f(x_k) - t\|g_k\|_2^2 + \frac{Mt^2}{2}\|g_k\|_2^2.$$

Observe that for $0 \leq t \leq 1/M$, $-t + Mt^2/2 \leq -t/2$ which follows from the convexity of $-t + Mt^2/2$ Using this result we get:

$$\begin{aligned} f(x_k - tg_k) &\leq f(x_k) - t\|g_k\|_2^2 + \frac{Mt^2}{2}\|g_k\|_2^2 \\ &\leq f(x_k) - \frac{t}{2}\|g_k\|_2^2 \leq f(x_k) - \alpha t\|g_k\|_2^2, \end{aligned}$$

since $\alpha \leq 1/2$.

The backtracking terminates either with $t = 1$ or with a value $t \geq \beta/M$. This provides a lower bound on the decrease in the function f. For $t = 1$ we have:

$$f(x_k - tg_k) \leq f(x_k) - \alpha \|g_k\|_2^2$$

and for $t \geq \beta/M$

$$f(x_k - tg_k) \leq f(x_k) - \frac{\alpha\beta}{M} \|g_k\|_2^2.$$

Therefore, for $0 \leq t \leq 1/M$ we always have

$$f(x_k - tg_k) \leq f(x_k) - \min\left\{\alpha, \frac{\alpha\beta}{M}\right\} \|g_k\|_2^2. \tag{11}$$

On the other hand

$$\frac{(a_k - b_k)^2}{2b_k} \geq \frac{\left(t\|g_k\|_2^2 - t^2 M\|g_k\|_2^2\right)^2}{2t^2 M\|g_k\|_2^2} = \frac{(1 - tM)^2}{2M}\|g_k\|_2^2.$$

Now, as above, for $t = 1$

$$\frac{(a_k - b_k)^2}{2b_k} \geq \frac{(1 - M)^2}{2M}\|g_k\|_2^2.$$

For $t \geq \beta/M$

$$\frac{(a_k - b_k)^2}{2b_k} \geq \frac{(1 - \beta)^2}{2M}\|g_k\|_2^2.$$

Therefore,

$$\frac{(a_k - b_k)^2}{2b_k} \geq \min\left\{\frac{(1 - M)^2}{2M}, \frac{(1 - \beta)^2}{2M}\right\}\|g_k\|_2^2. \tag{12}$$

Considering (11) together with (12) we get:

$$f(x_k - \theta_m t g_k) \leq f(x_k) - \min\left\{\alpha, \frac{\alpha\beta}{M}\right\}\|g_k\|_2^2 - \min\left\{\frac{(1 - M)^2}{2M}, \frac{(1 - \beta)^2}{2M}\right\}\|g_k\|_2^2. \tag{13}$$

Therefore

$$f(x_k) - f(x_{k+1}) \geq \left[\min\left\{\alpha, \frac{\alpha\beta}{M}\right\} + \min\left\{\frac{(1 - M)^2}{2M}, \frac{(1 - \beta)^2}{2M}\right\}\right]\|g_k\|_2^2.$$

But, $f(x_k) - f(x_{k+1}) \to 0$ and as a consequence g_k goes to zero, i.e. x_k converges to x^*. Having in view that $f(x_k)$ is a nonincreasing sequence, it follows that $f(x_k)$ converges to $f(x^*)$.

From(13) we see that

$$f(x_{k+1}) \leq f(x_k) - \left[\min\left\{\alpha, \frac{\alpha\beta}{M}\right\} + \min\left\{\frac{(1 - M)^2}{2M}, \frac{(1 - \beta)^2}{2M}\right\}\right]\|g_k\|_2^2.$$

Combining this with

$$\|g_k\|_2^2 \geq 2m\left(f(x_k) - f^*\right)$$

and subtract f^* from both sides of the above inequality we conclude:

$$f(x_{k+1}) - f^* \leq c\big(f(x_k) - f^*\big), \tag{14}$$

where

$$c = 1 - \min\left\{2m\alpha, \frac{2m\alpha\beta}{M}\right\} - \min\left\{\frac{(1-M)^2 m}{M}, \frac{(1-\beta)^2 m}{M}\right\} < 1. \tag{15}$$

Therefore, $f(x_k)$ converges to f^* at least as fast as a geometric series with a factor that depends on the backtracking parameters and the condition number bound M/m Therefore, the convergence is at least linear. □

At every iteration k, selecting $\theta_k = \theta_m$ in (5), the AGD algorithm reduces the function values according to (14) where c is given by (15). Since GD algorithm achieves (14) with

$$c = 1 - \min\left\{2m\alpha, \frac{2m\alpha\beta}{M}\right\} < 1, \tag{16}$$

it follows that, if $\theta_m \neq 1$, the AGD algorithm is an improvement, i.e. an acceleration, of GD.

4. Numerical results and comparisons

In this section we report some numerical results obtained with a Fortran implementation of the above gradient descent algorithms. All codes are written in Fortran and compiled with f77 (default compiler settings) on a Workstation Intel Pentium 4, 1.8 GHz. We selected 34 large-scale unconstrained optimization test problems (some from CUTE library [2]) in extended or generalized form. For each test function we have considered ten numerical experiments with number of variables $n = 100, 200, \ldots, 1{,}000$.

In the following we present the numerical performance of AGD and GD codes, in which we stopped the iterations as soon as one of the following criteria is satisfied:

$$\left\|\nabla f(x_{k+1})\right\|_\infty \leq \varepsilon_g \quad \text{or} \quad t_k\left|g_k^T d_k\right| \leq \varepsilon_f \left|f(x_{k+1})\right|,$$

with $\varepsilon_g = 10^{-6}$ and $\varepsilon_f = 10^{-20}$. In all numerical experiments the backtracking procedures uses $\alpha = 0.0001$ and $\beta = 0.8$.

Table 1 presents the global characteristics, corresponding to these 340 test problems, referring to the total number of iterations, total number of function evaluations and total cpu time for these algorithms.

Table 1 Global characteristics of AGD and GD, 340 problems.

Global characteristics	AGD	GD
Total number of iterations	657,915	1,598,990
Total number of function evaluations	17,432,902	40,979,772
Total cpu time (s)	4,547.80	10,043.26

Table 2 Performance of AGD and GD algorithms, 340 problems.

Performance criterion	Number of problems
AGD achieved minimum number of iterations in	320
GD achieved minimum number of iterations in	36
AGD and GD achieved the same number of iterations in	16
AGD achieved minimum number of function evaluations in	322
GD achieved minimum number of function evaluations in	18
AGD achieved minimum cpu time in	325
GD achieved minimum cpu time in	28
AGD and GD achieved the same cpu time in	13

Table 2 shows the number of problems, out of 340, for which AGD and GD achieved the minimum number of iterations, minimum number of function evaluations and the minimum cpu time, respectively.

The performance of these algorithms have been evaluated using the profiles of Dolan and Moré [5]. That is, for each algorithm, we plot the fraction of problems for which the algorithm is within a factor of the best number of iterations, the best number of function evaluations, or the best cpu time, respectively. The left side of these Figures gives the percentage of the test problems, out of 340, for which an algorithm is more performant; the right side gives the percentage of the test problems that were successfully solved by each of the algorithms. Mainly, the right side represents a measure of an algorithm's robustness.

In figures 1, 2 and 3 we display the performance profiles of Dolan and Moré, corresponding to AGD and GD, refering to the number of iterations, number of functions evaluations and cpu time, respectively.

Observe that AGD outperforms GD in the vast majority of problems, and the differences are substantial. From table 1 we see that refering to cpu time AGD is about twice fastest that GD. We explain this difference in behaviour of these algorithms by recalling that as the stationary point is approached, GD method takes small, nearly orthogonal steps. This poor convergence of the GD algorithm

Figure 1 Performance based on number of iterations, 340 problems.

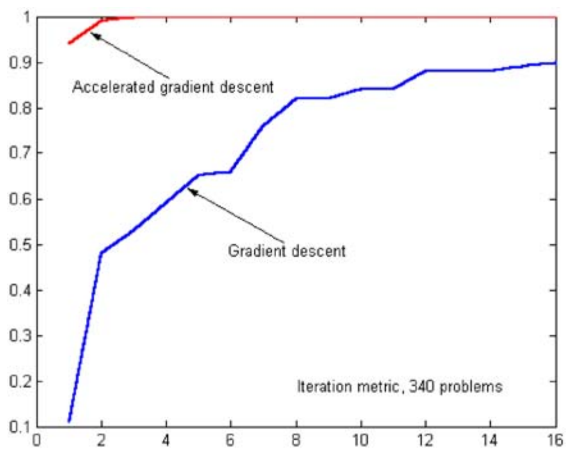
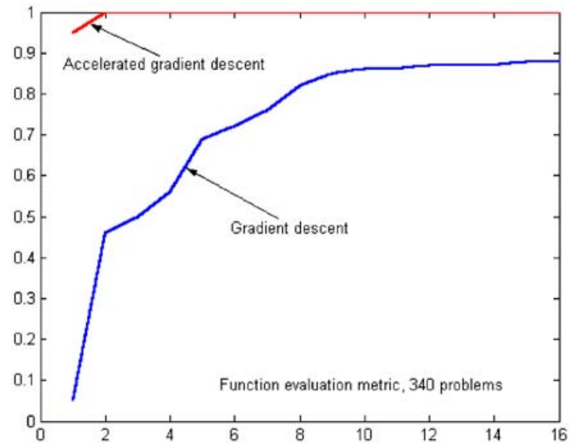


Figure 2 Performance based on number of function evaluations, 340 problems.

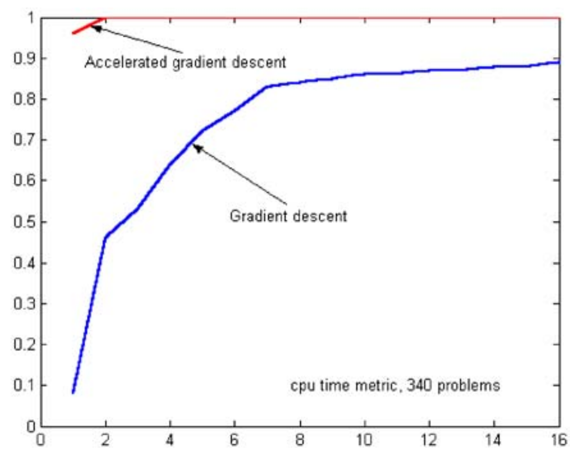


at the later iterations can be explained by considering the following expression of function f:

$$f(x_k - t_k g_k) = f(x_k) - t_k \|g_k\|^2 + \frac{1}{2} t_k^2 \gamma_k \|g_k\|^2, \tag{17}$$

where $\gamma_k I \cong \nabla^2 f(z)$ is a scalar approximation of the Hessian at the point z which belongs to the line segment connecting x_k and x_{k+1}. Observe that if x_k is close to a stationary point with zero gradient, and f is continuously differentiable, then $\|g_k\|^2$ will be small. Therefore, $t_k\|g_k\|^2$ in (17) is of a small order of magnitude, its contribution to reduce the function values being almost insignificant. Since the gradient descent method uses only the linear approximation of f to find the search direction, ignoring completely the second order term $(t_k^2/2)\gamma_k\|g_k\|^2$ we expect that the direction generated will not be very effective, if the ignored term contributes significantly to the description of f, even for relatively small values of t_k. In AGD this is compensated by modifying the steplength in order to destroy the orthogonality of

Figure 3 Performance based on cpu time metric, 340 problems.



the successive search directions giving thus the possibility for a substantial progress towards minimum.

5. Conclusions

In this paper we have introduced an acceleration of the gradient descent algorithm by means of a simple multiplicative modification of the steplength given by a backtracking procedure in the classical gradient descent algorithm. The accelerated algorithm is linear convergent with a factor which is smaller than the factor corresponding to the classical gradient descent algorithm.

References

[1] Armijo, L.: Minimization of functions having Lipschitz continuous first partial derivatives. Pac. J. Math. **6**, 1–3 (1966)

[2] Bongartz, I., Conn, A.R., Gould, N.I.M., Toint, P.L.: CUTE: Constrained and unconstrained testing environments. ACM Trans. Math. Softw. **21**, 123–160 (1995)

[3] Cauchy, A.: Méthodes générales pour la résolution des systémes déquations simultanées. C. R. Acad. Sci., Paris **25**, 536–538 (1848)

[4] Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewoods Cliffs, New Jersey (1983)

[5] Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)

[6] Fletcher, R.: Practical Methods of Optimization. Wiley, New York (1987)

[7] Forsythe, G.E., Motzkin, T.S.: Asymptotic properties of the optimum gradient method. Bull. Am. Soc. **57**, 183 (1951)

[8] Goldstein, A.A.: On steepest descent. SIAM J. Control **3**, 147–151 (1965)

[9] Humphrey, W.E.: A general minimising routine – minfun. In: Lavi, A., Vogl, T.P. (eds.) Recent Advances in Optimisation Techniques. Wiley, New York (1966)

[10] Lemaréchal, C.: A view of line search. In: Auslander, A., Oettli, W., Stoer, J. (eds.) Optimization and Optimal Control, pp. 59–78. Springer, Berlin (1981)

[11] Moré, J.J., Thuente, D.J.: On line search algorithms with guaranteed sufficient decrease. Mathematics and Computer Science Division Preprint MCS-P153-0590, Argonne National Laboratory, Argonne (1990)

[12] Potra, F.A., Shi, Y.: Efficient line search algorithm for unconstrained optimization. J. Optim. Theory Appl. **85**, 677–704 (1995)

[13] Powell, M.J.D.: Some global convergence properties of a variable-metric algorithm for minimization without exact line searches. SIAM-AMS Proc., Philadelphia **9**, 53–72 (1976)

[14] Schinzinger, R.: Optimization in electromagnetic system design. In: Lavi, A., Vogl, T.P. (eds.) Recent Advances in Optimisation Techniques. Wiley, New York (1966)

[15] Zhen-Jun, S.: Convergence of line search methods for unconstrained optimization. Appl. Math. Comput. **157**, 393–405 (2004)

[16] Wolfe, P.: Convergence conditions for ascent methods. SIAM Rev. **11**, 226–235 (1968)