Relaxed Gradient Descent Method with Backtracking for Unconstrained Optimization

Neculai Andrei
Research Institute for Informatics,
Center for Advanced Modeling and Optimization,
8-10, Averescu Avenue, Bucharest 1, Romania
E-mail: nandrei@ici.ro

Abstract. In this work we present the relaxed gradient descent method for solving unconstrained optimization problems. Using a simple multiplicative modification of the steplength by means of a random variable uniformly distributed in (0,1) we get the relaxed gradient descent method. It is shown that for strongly convex functions the relaxed gradient algorithm is linearly convergent. We emphasize that the poor behavior of the classical gradient method is due to the choice of the steplength and not to the choice of the search direction. The paper extends the results obtained by Raydan and Svaiter [15] for quadratic unconstrained optimization problems to strongly convex functions.
Keywords: gradient descent methods, relaxed gradient descent, random optimization, backtracking, quadratric unconstrained optimization
AMS subject classifications: 49M07, 49M10, 65K

1. Introduction

One of the first and very well known method for unconstrained optimization is the gradient descent method, designed by Cauchy early in 1847, in which the negative gradient direction is used to find the local minimizers of a differentiable function. The method proved to be effective for functions very well conditioned, but for functions poorly conditioned the method is excessively slow, thus being of no practical value. Even for quadratic functions the gradient descent method with exact line search behave increasingly badly when the conditioning number of the matrix deteriorates. Early attempts to increase the performances of the method have been considered by Humphrey [9], Forsyte and Motzkin [7] and Schinzinger [16]. Even though the storage requirements for the gradient descent method are minimal ($3n$ locations for an $n-$ dimensional problem), the development of the conjugate gradient and quasi-Newton methods for large-scale unconstrained optimization cast the gradient descent method in a penumbra.

Recently, for the quadratic positive definite case Raydan and Svaiter [15] considered the relaxed gradient method as well as the Cauchy-Barzilai-Borwein methods showing the superiority of the last one against the relaxed gradient descent and BB methods, Particularly, the Cauchy-Barzilai-Borwein method proves to be Q-linearly convergent in a norm defined by the matrix of the problem.

The purpose of this paper is to extend the relaxed gradient method to the convex, well conditioned, functions. It is shown that for strongly convex, well conditioned minimization problems the relaxed gradient descent algorithm is an improvement of the classical gradient descent version. The corresponding algorithm belongs to the same class of linear convergent descent methods. The conclusion is that using only the local information given by the gradient, any procedure for step size computation, of any sophistication, does not change the linear convergence class of algorithms.

The paper is organized as follows. In section 2 we present the backtracking line search. Section 3 is dedicated to present the relaxed gradient descent algorithm and its properties. In section 4 we present some numerical evidence and discussions of this approach.

2. Line Search with Backtracking

In the following let us consider the problem:

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

where $f: R^n \to R$ is convex and twice continuously differentiable. A necessary and sufficient condition for a point x^* to be a local minimum for (1) is

$$\nabla f(x^*) = 0. \tag{2}$$

Usually, the problem is solved by an iterative algorithm which generates a sequence of points $x_0, x_1, \ldots, x_k, \ldots \in dom\, f,$ for which $f(x_k) \to f^*$ as $k \to \infty.$ The algorithms for solving (1) generate a minimizing sequence $x_k,\ k = 0,1,\ldots$ as:

$$x_{k+1} = x_k + t_k d_k, \tag{3}$$

where the scalar t_k is the step size, and the vector d_k is the search direction.

Many procedures for search direction computation have been proposed. One of the first method, and the simplest one, for solving (1) using (3) was the gradient descent method (Cauchy method [3]) where the choice for the search direction at the iteration x_k is the negative gradient, $-\nabla f(x_k).$ Some other known methods dedicated for large-scale problems are based on the conjugate gradient strategy or quasi-Newton selection of the direction.

At the same time, for step size selection many algorithms have been considered. In the *exact line search* the step t_k is selected as:

$$t_k = \arg\min_{t>0} f(x_k + t d_k). \tag{4}$$

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 $\{x_k + t d_k : t \geq 0\},$ or at least to reduce f enough. In practice the most used are the *inexact procedures*. A lot of inexact line search methods have been proposed: Goldstein [8], Armijo [1], Wolfe [17], Powell [14], Denis and Schnabel [4], Fletcher [6], Potra and Shi [13], Lemaréchal [10], Moré and Thuente [12], 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 $f(x_k + t d_k) > f(x_k) + \alpha t \nabla f(x_k)^T d_k,$ set $t = t\beta$.

Step 3. Set $t_k = t.$

Typically, $\alpha = 0.0001$ and $\beta = 0.8,$ meaning that we accept a small decrease in f of the prediction based on the linear extrapolation. If $d_k = -g_k,$ then $s_k = 1.$

Proposition 1. *Suppose that d_k is a descent direction and $\nabla f(x_k)$ satisfies the Lipschitz condition $\|\nabla f(x) - \nabla f(x_k)\| \leq L\|x - x_k\|$ for all x on the line segment connecting x_k and x_{k+1}, where L is a constant. If the line search satisfies the Armijo condition, then*

$$t_k \geq \frac{\beta(\alpha-1)}{L}\frac{g_k^T d_k}{\|d_k\|^2}. \tag{5}$$

Proof. Observe that for all k for which $t_k < s_k$ it follows that $t_k / \beta \leq s_k$. Therefore

$$f(x_k + t_k d_k / \beta) - f(x_k) > \alpha(t_k / \beta) g_k^T d_k.$$

Using the mean value theorem on the left side of the above inequality, the Cauchy-Schwartz inequality and the Lipschitz condition, there exists $\xi_k \in [0,1]$ such that

$$\alpha g_k^T d_k < g(x_k + t_k \xi_k d_k / \beta)^T d_k = [g(x_k + t_k \xi_k d_k / \beta) - g(x_k)]^T d_k + g_k^T d_k$$
$$\leq \|g(x_k + t_k \xi_k d_k / \beta) - g(x_k)\| \|d_k\| + g_k^T d_k \leq L t_k \|d_k\|^2 / \beta + g_k^T d_k.$$

Rearranging this inequality gives (5). ■

3. Relaxed Gradient Descent Method

As we said, the goal of this paper is to show that the linear behavior of the gradient descent method for well-conditioned functions could be improved by a subrelaxation of the step size. The idea is to modify the gradient descent method by introducing a relaxation of the following form:

$$x_{k+1} = x_k + \theta_k t_k d_k, \tag{6}$$

where θ_k is the relaxation parameter, a random variable uniformly distributed between 0 and 1. With this, the Relaxed Gradient Descent algorithm can be presented as:

Relaxed Gradient Descent Algorithm (RGD)

Step 1. Consider a starting point $x_0 \in dom\, f$. Set $k = 0$.

Step 2. Compute the search direction: $d_k = -\nabla f(x_k)$.

Step 3. Line search. Choose the steplength t_k via exact or backtracking line search procedures.

Step 4. Update the variables: Select $\theta_k \in (0,1)$ and update the variables: $x_{k+1} = x_k + \theta_k t_k d_k$.

Step 5. Test a criterion for stopping the iterations. If the test is satisfied, then stop; otherwise consider $k = k+1$ and continue with step 2.

Clearly, if $\theta_k = 1$, for all k, we get the classical Gradient Descent (GD) method. For quadratic positive definite problems an overrelaxation have been considered by Raydan and Svaiter [15]. They proved that the poor behavior of the steepest descent methods is due to the optimal Cauchy choice of step size and not to the choice of the search direction. Indeed for the quadratic problem

$$min\, f(x) = \frac{1}{2} x^T Q x - b^T x, \tag{7}$$

where $Q \in R^{n \times n}$ is symmetric and positive definite, the classical Cauchy method can be written as: $x_{k+1} = x_k - t_k g_k$, where $g_k = \nabla f(x_k)$ and

$$t_k = \frac{g_k^T g_k}{g_k^T Q g_k}. \tag{8}$$

Taking θ_k as relaxation parameters between 0 and 2, Raydan and Svaiter get the relaxed gradient descent algorithm:

$$x_{k+1} = x_k - \theta_K t_k g_k, \tag{9}$$

where again t_k is given in (8). The following theorem shows the convergence of relaxed Cauchy algorithm (9) for quadratic symmetric and positive definite functions:

Theorem 1. *(Raydan and Svaiter [15]) If the sequence θ_k has an accumulation point $\bar{\theta} \in (0,2)$, then x_k generated by the relaxed Cauchy method (9), where t_k is given in (8), converges to x^*.*

Now we extend this idea of relaxation to general convex functions. But, before considering the theoretical aspects for the general convex functions let us take an example.

Example 1. Let us illustrate the behavior of the relaxed gradient descent method in comparison with the classical gradient descent on the following function:

$$f(x)=\sum_{i=1}^{n} i x_i^2+\frac{1}{100}\left(\sum_{i=1}^{n} x_i\right)^2.$$

Considering $x_0=[0.5,0.5,\ldots,0.5]$, $\alpha=0.0001$ and $\beta=0.8$ in the backtracking procedure, as well as the following criteria for stopping the iterations

$$\left\|\nabla f(x_k)\right\|_{\infty}\le\varepsilon_g \quad \text{or} \quad t_k\left|g_k^T d_k\right|\le\varepsilon_f\left|f(x_{k+1})\right|, \tag{10}$$

with $\varepsilon_g=10^{-6}$ and $\varepsilon_f=10^{-20}$, for $n=100$, the evolution of $\left|f(x_k)-f^*\right|$ given by the gradient descent method and the relaxed gradient descent method, respectively, are presented in Figure 1.

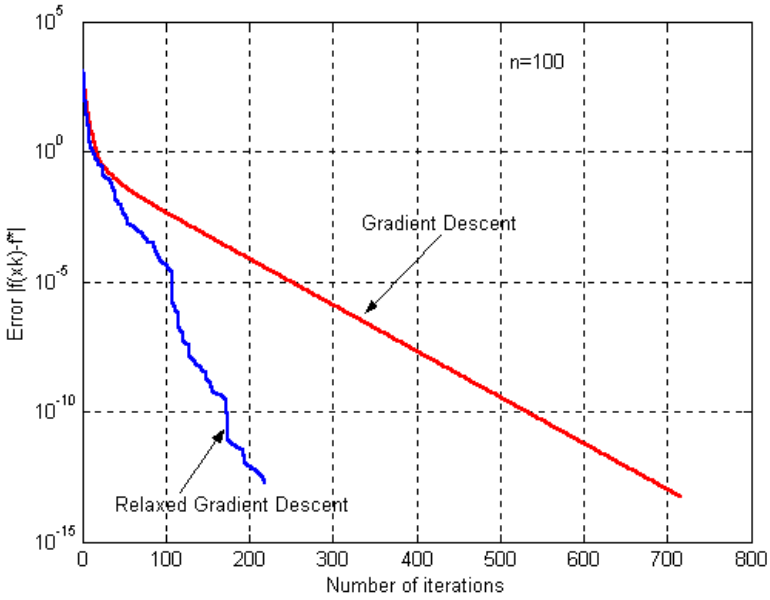

Figure 1. Gradient Descent versus Relaxed Gradient Descent.

Table 1 shows the number of iterations corresponding to these algorithms, as well as the average steplength, for different values of n, using the above criteria for stopping the iterations.

Table 1. Number of iterations and the average steplength.

	GD		RGD	
n	#iter	average step	#iter	average step
100	717	0.009979	219	0.03157
200	1431	0.005011	330	0.01621
300	2146	0.003358	344	0.01643
400	2866	0.002506	488	0.01219
500	3575	0.002006	452	0.01044
600	4290	0.001681	486	0.01121
700	5064	0.001442	599	0.008647
800	5742	0.001249	565	0.008685
900	6448	0.001123	718	0.007778
1000	7160	0.001000	501	0.009060

Observe that the relaxed version of gradient descent method clearly outperforms the classical gradient method. Both methods exhibits a linear convergence to the minimizer of the function, and they converge monotonically to the minimizer, but RGD algorithm takes a significant smaller number of iterations. In this case the total number of iterations for Gradient Descent Algorithm is 39439, and 4702 for Relaxed Gradient. Notice that, subject to the number of iterations, RGD is about 8 times more performant than GD. Similar results have been obtained using different random

number generators in interval (0,1). The parameters α and β from backtracking linear search has a noticeable but not a dramatic influence on the number of iteration. Numerical experiments with different values for α and β leads to the same behaviour of the algorithm. We also observe that the average stepsize in both GD and RGD algorithms are bounded away from zero, as we proved in Proposition 1. However, the average stepsize in RGD algorithm is significantly greater than in GD version. This explains its efficiency.

This numerical experiment reveals the serious limitation of the steplength choice by backtracking when searching is along the negative gradient. This reveals a lack of robustnees of the gradient descent algorithm with backtracking at steplength perturbations.

The convergence analysis of RGD algorithm for general strongly convex function is given by the following theorems.

Theorem 2. *Suppose that f is a strongly convex function with $\nabla^2 f(x) \le MI$, where M is a positive constant. If the sequence θ_k has an accumulation point $\bar{\theta} \in (0,1)$, then for strongly convex functions the sequence x_k generated by the RGD algorithm converges linearly to x^*.*

Proof. Let us consider $\Phi_k(\theta) = f(x_k - \theta t_k g_k)$, where $g_k = \nabla f(x_k)$. We can write:

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

Since the function f is strongly convex it follows that $\Phi_k(\theta)$ is a convex function and $\Phi_k(0) = f(x_k)$. From the strong convexity of f it follows that:

$$f(x_k - \theta t_k g_k) \le f(x_k) - \left(\theta - \frac{Mt_k}{2}\theta^2\right) t_k \|g_k\|_2^2 .$$

But $\theta - \frac{Mt_k}{2}\theta^2$ is a concave and nonnegative function on $(0, 2/Mt_k)$ and has the maximum value equal with $1/2Mt_k$ at point $1/Mt_k$. Therefore $f(x_{k+1}) \le f(x_k)$ for all k. Since f is bounded below, it follows that

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

Now observe that $\Phi_k(\theta)$ is a convex function with the minimum value at the point

$$\theta_m = \frac{g_k^T g_k}{t_k (g_k^T \nabla^2 f(x_k) g_k)} > 0.$$

On the other hand, $\Phi_k(0) = f(x_k)$ and $\Phi_k(\theta_u) = f(x_k)$, where $\theta_u = 2\theta_m$. But,

$$\Phi_k(\theta_m) = f(x_k) - \frac{(g_k^T g_k)^2}{2(g_k^T \nabla^2 f(x_k) g_k)} < f(x_k).$$

Therefore, for $\theta \in [0, 2\theta_m]$, $\Phi_k(\theta) \le \Phi_k(0)$.

There exists some $\gamma \in (0, \theta_m)$ with $\gamma < 1$, such that $\gamma \le \bar{\theta} \le 2\theta_m - \gamma$. Therefore, there exists a subsequence θ_{k_j} contained in the interval $[\gamma, 2\theta_m - \gamma]$. Using again the convexity of $\Phi_k(\theta)$ we get that

$$\Phi_k(0) - \Phi_k(\gamma\theta_m) \ge -\gamma[\Phi_k(\theta_m) - \Phi_k(0)].$$

But,

$$\Phi_k(\theta_m) - \Phi_k(0) = -\frac{(g_k^T g_k)^2}{2(g_k^T \nabla^2 f(x_k) g_k)}.$$

Hence

$$\Phi_k(0) - \Phi_k(\gamma\theta_m) \geq \frac{\gamma}{2}\frac{(g_k^T g_k)^2}{(g_k^T \nabla^2 f(x_k) g_k)} \geq \frac{\gamma}{2M}\|g_k\|_2^2.$$

Therefore,

$$f(x_{k_j}) - f(x_{k_j+1}) \geq \frac{\gamma}{2M}\|g_k\|_2^2.$$

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

In the following let us assume that f is strongly convex and the sublevel set $S = \{x \in dom f : f(x) \leq f(x_0)\}$ is closed. Strong convexity of f on S involves that there exists positive constants m and M such that $mI \leq \nabla^2 f(x) \leq MI$, for all $x \in S$. A consequence of strong convexity of f on S is that we can bound $f(x^*)$ as:

$$f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \leq f^* \leq f(x) - \frac{1}{2M}\|\nabla f(x)\|_2^2. \tag{11}$$

In these circumstances the following theorem can be proved.

Theorem 3. *For strongly convex functions the Relaxed Gradient Descent algorithm with backtracking is linear convergent and*

$$f(x_k) - f^* \leq \left(\prod_{i=0}^{k-1} c_i\right)(f(x_0) - f^*),$$

where

$$c_i = 1 - min\{2m\alpha\theta_i, 2m\alpha\beta\theta_i / M\} < 1.$$

Proof. Consider $0 < \theta < 1$, then

$$f(x_k - \theta t_k g_k) \leq f(x_k) - \left(\theta - \frac{Mt_k}{2}\theta^2\right)t_k\|g_k\|_2^2.$$

Notice that $\theta - \frac{Mt_k}{2}\theta^2$ is a concave function and for all $0 \leq \theta \leq 1/Mt_k$, $\theta - \frac{Mt_k}{2}\theta^2 \geq \theta/2$. Hence

$$f(x_k - \theta t_k g_k) \leq f(x_k) - \frac{\theta}{2}t_k\|g_k\|_2^2 \leq f(x_k) - \alpha\theta t_k\|g_k\|_2^2,$$

since $\alpha \leq 1/2$. Therefore the backtracking line search procedure terminates either with $t_k = 1$ or with a value $t_k \geq \beta / M$. With this, at step k we can get a lower bound on the decrease of the function. In the first case we have

$$f(x_{k+1}) \leq f(x_k) - \alpha\theta_k\|g_k\|_2^2,$$

and in the second one

$$f(x_{k+1}) \leq f(x_k) - \alpha\theta_k\frac{\beta}{M}\|g_k\|_2^2.$$

Therefore, we have

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

Hence

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

But, from (11) it follows that $\|g_k\|_2^2 \ge 2m(f(x_k) - f^*)$. Therefore, combining this with the above inequality we get

$$f(x_{k+1}) - f^* \le \left(1 - min\left\{2m\alpha\theta_k, 2m\alpha\theta_k \frac{\beta}{M}\right\}\right)(f(x_k) - f^*).$$

Denoting: $c_k = 1 - min\left\{2m\alpha\theta_k, 2m\alpha\theta_k \frac{\beta}{M}\right\}$, it follows that for every $k = 0,1,\ldots$

$$f(x_{k+1}) - f^* \le c_k\left(f(x_k) - f^*\right),$$

which prove the second part of the theorem.

Since $c_k < 1$, the sequence $f(x_k)$ converges to f^* like a geometric series with an exponent that partialy depends on the condition number bound M/m, the backtracking parameters α and β, the sequence θ_k of random uniform distributed numbers in $(0,1)$ interval and on the initial suboptimality. Therefore, the RGD algorithm is linear convergent. ■

4. Numerical Results

In this section we report some numerical results obtained with a Fortran implementation of the above gradient descent algorithms for a number of 360 unconstrained optimization problems. The full description of these experiments is documented at the web page

http://www.ici.ro/camo/neculai/ansoft.htm.

All codes are written in Fortran and compiled with f77 (default compiler settings) on a Pentium 1.5 Ghz. We selected 36 large-scale unconstrained optimization test problems (5 from CUTE library [2]) in extended or generalized form. For each test function we have considered 10 numerical experiments with number of variables $n = 100,200,\ldots,1000$.

In the following we present the numerical performance of RGD and GD codes, in which the computations are terminated as soon as the stopping criteria in (10) are satisfied. In all numerical experiments the backtracking procedure considers $\alpha = 0.0001$ and $\beta = 0.8$.

The main indicator of performance is the relative number of iterations (iter), or the relative number of function/gradient evaluations (fg) or the relative cpu time (time), of methods A and B, introduced by Morales [11], and measured by:

$$r_{AB}^i = -\log_2\left(\frac{ind_A^i}{ind_B^i}\right)$$

where ind_A^i represents the number of iterations, or the number of function/gradient evaluations, or the cpu time, corresponding to algorithm A, for solving the $i-$th problem, respectively. ind_B^i has a similar interpretation for algorithm B. The sign of r_{AB}^i indicate the winer: all the cases in which algorithm A wins correspond to a positive r_{AB}^i. The number of times by which the winer outperforms the loser is given by $2^{|r_{AB}^i|}$. In this interpretation we can refer this number as the *outperforming factor*. In this numerical experiment A stands for RGD, and B for GD. Observe that this is exactly the Dolan and Moré's [5] performance profile for $\tau = 1$.

In Figure 2 we display the values of r_{AB}^i, for this set of problems, where ind_A^i and ind_B^i represent the number of iterations corresponding to algorithm RGD and GD respectively, and the problems have been placed in decreasing order with respect to their values of $|r_{AB}^i|$.

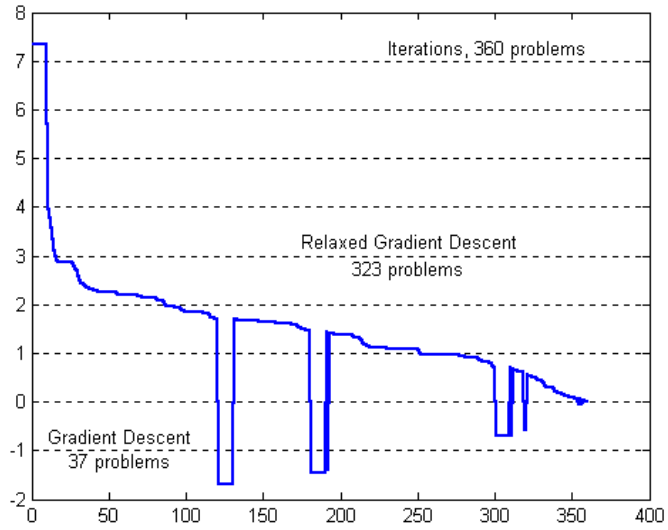

Fig. 2. Relative performance (iter) of RGD and GD.

In Figure 3 we show the values of r_{AB}^{i} corresponding to the number of function/gradient evaluations of the algorithms RGD and GD, respectively.

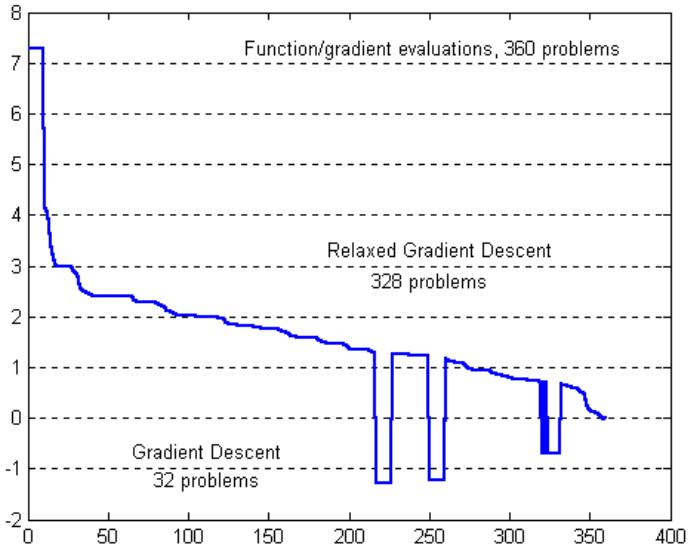

Fig. 3. Relative performance (fg) of RGD and GD.

Figure 4 shows the values of r_{AB}^{i} corresponding to cpu time of the algorithms RGD and GD, respectively.

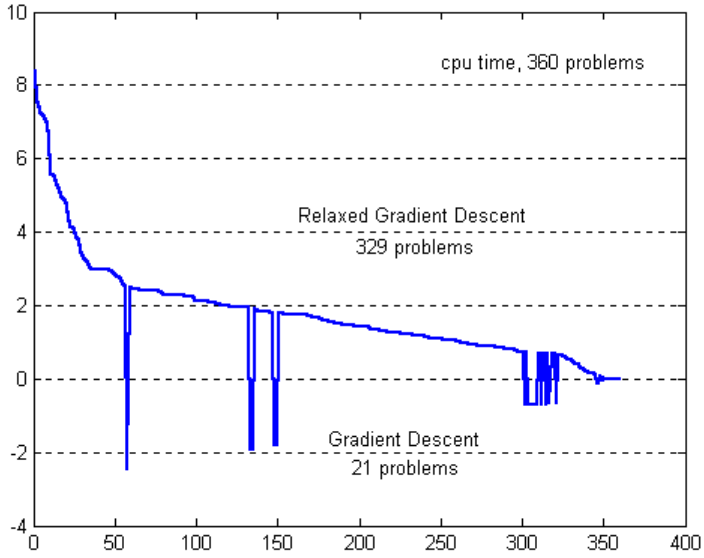

Fig. 4. Relative performance (time) of RGD and GD.

Observe that RGD outperforms GD in the vast majority of problems, and the differences are substantial: the outperforming factor is greater than 15 for cpu time, or 10 for function/gradient evaluations. 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 at the later iterations can be explained by considering the following expression of function f :

$$f(x_k - t_k g_k) = f(x_k) - t_k \|g_k\|^2 + \frac{1}{2} t_k^2 \gamma_k \|g_k\|^2, \tag{12}$$

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 (12) 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 RGD this is compensated by modifying the steplength in order to distroy the orthogonality of the successive searching directions giving thus the possibility for a substantial progress towards minimum.

5. Conclusion

We have presented the relaxed gradient descent algorithm and a numerical study of this algorithm in comparison with the classical gradient descent algorithm. The RGD and GD algorithms mainly differ in their strategy for steplength selection. A simple modification of the steplength by means of a random variable, uniformly distributed in (0,1), multiplying the steplength, represents an improvement of the classical gradient descent algorithm. For strongly convex function we proved its linear convergence. We proved that at each iteration the factor of reducing the suboptimality for RGD algorithm is greater than that corresponding to GD. This explains the superiority of RGD against GD algorithm.

The conclusion is that using only the local information given by the gradient of the minimizing function, any procedure for steplength computation, does not change the linear convergence property of the gradient descent algorithms.

References

[1] Armijo L. (1966) Minimization of functions having Lipschitz continuous first partial derivatives. Pacific Journal of Mathematics, **6** 1-3.

[2] Bongartz I, Conn AR, Gould NIM, Toint PL (1995) CUTE: constrained and unconstrained testing environments. ACM Trans. Math. Software. **21** 123-160.

[3] Cauchy A (1847) Méthodes générales pour la résolution des systémes déquations simultanées, C.R. Acad. Sci. Par., **25** 536-538.

[4] Dennis JE, Schnabel RB (1983) Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Englewoods Cliffs, N.J.,Prentice-Hall.

[5] Dolan ED, Moré JJ (2002) Benchmarking optimization software with performance profiles. Math. Programming, **91**, 201-213.

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

[7] Forsythe GE, Motzkin TS (1951) Asymptotic properties of the optimum gradient method. Bull. American Society, **57** 183.

[8] Goldstein AA (1965) On steepest descent. SIAM Journal on Control, **3** 147-151.

[9] Humphrey WE (1966) A general minimising routine - minfun. in A. Lavi, and T.P. Vogl, (Eds.), Recent Advances in Optimisation Techniques, John Wiley.

[10] Lemaréchal C (1981) A view of line search. in A. Auslander, W. Oettli and J. Stoer (Eds.) Optimization and Optimal Control, Springer Verlag, 59-78.

[11] Morales JL (2002) A numerical study of limited memory BFGS methods. Applied Mathematics Letters, **15** 481-487.

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

[13] Potra FA, Shi Y (1995) Efficient line search algorithm for unconstrained optimization. Journal of Optimization Theory and Applications, **85** 677-704

[14] Powell MJD (1976) Some global convergence properties of a variable-metric algorithm for minimization without exact line searches. SIAM-AMS Proceedings, Philadelphia, **9** 53-72

[15] Raydan M, Svaiter BF (2002) Relaxed Steepest Descent and Cauchy-Barzilai-Borwein Method, Computational Optimization and Applications, **21** 155-167

[16] Schinzinger R (1966) Optimization in electromagnetic system design. in A. Lavi, and T.P. Vogl (Eds.), Recent Advances in Optimisation Techniques, John Wiley

[17] Wolfe P (1968) Convergence conditions for ascent methods. SIAM Review, **11** 226-235

March 2, 2005