NON MONOTONE ALGORITHMS FOR UNCONSTRAINED MINIMIZATION: UPPER BOUNDS ON FUNCTION VALUES

U.M. Garcia-Palomares[1]
[1] *Universidad Simón Bolívar, Dep Procesos y Sistemas, Caracas, Venezuela, garciap@usb.ve*[*]

Abstract Non monotone algorithms allow a possible increase of function values at certain iterations. This paper gives a suitable control on this increase to preserve the convergence properties of its monotone counterpart. A new efficient MultiLineal Search is also proposed for minimization algorithms.

keywords: Non Monotone, Lineal Search, Trust Region.

1. Introduction

This paper is concerned with algorithms for solving the unconstrained minimization problem of finding a *local* minimizer $\bar{x}$ and the *local* minimum value $\bar{f} = f(\bar{x})$ of a scalar function $f(\cdot) \in C^1 : S \subset I\!R^n \to I\!R$. Armijo's inequality (1) has been frequently used by monotone algorithms: given $x_i, d_i \in I\!R^n$, the algorithm must determine a stepsize λ_i so that the new iterate x_{i+1} gives a sufficient decrease in the function value,

$$f(x_{i+1}) = f(x_i + \lambda_i d_i) \leq f(x_i) + 0.01\lambda_i \nabla f(x_i)^T d_i. \tag{1}$$

Under suitable assumptions (**A1-A4** below) a (*sub*)sequence $\{x_i\}_{i \in I}$ fulfilling (1) converges to a point $\bar{x}$ satisfying the first order necessary optimality condition; namely $\nabla f(\bar{x}) = 0$ [14]. Additional conditions, mainly in the choice of $\{d_i\}_1^\infty$, are obviously required to ensure a superlinear rate of convergence. Monotone algorithms force strict decrease of function values, i.e., $f(x_{i+1}) < f(x_i)$. This stringent condition may impair the convergence of the algorithm. Although the asymptotic rate of convergence is preserved, narrow valleys may demand an excessive number of function evaluations, which is normally considered a poor performance index when comparing optimization

[*]Paper written with support of USB and Universidade de Vigo, Spain.

algorithms. Non monotone algorithms (NMAs) climb on the surrounding hills; i.e., $f(x_{i+1}) > f(x_i)$, and may avoid this undesirable behavior. Moreover, it has been shown that NMAs may jump over local minima [20] and become more fitted to global optimization [4].

A well known non monotone line search strategy was proposed by De Gripo et al [8, Section 3]. An iterate $x_{i+1} = x_i + \lambda_i d_i$ is accepted if

$$f(x_{i+1}) = f(x_i + \lambda_i d_i) \leq \max_{0 \leq j \leq q} f(x_{i-j}) + 0.01 \lambda_i \nabla f(x_i)^T d_i. \qquad (2)$$

This strategy has been adopted by many researchers in constrained and unconstrained problems with success. Currently many monotone algorithms have a non monotone counterpart [6]. The reader may consult additional material in [8, 9, 12, 16, 17, 19, 21] and references therein.

This paper adapts a sufficient decrease condition that does not require the computation of derivatives [5, 15]. Therefore, it can be used in derivative-free optimization and gradient-related algorithms. Furthermore, line search is not mandatory and can be replaced by trust region or some other technique. Finally, the maximum $f(\cdot)$ value on the previous q iterations is replaced by an upper bound $\varphi_i \geq f(x_i)$, which essentially has to be decreased a certain number of iterations (assumption **A5** below). An iterate x_{i+1} is accepted if

$$f(x_{i+1}) \leq \varphi_i - \phi(||d_i||), \qquad (3)$$

where $\phi(\cdot) : I\!R_+ \rightarrow I\!R_+$, $\lim_{\tau \downarrow 0} \phi(\tau)/\tau = 0$. Note that a monotone algorithm is recovered when $\varphi_i = f(x_i)$ for all i, and the algorithm will behave as explained before; on the other hand, if the function upper bound φ_i is very loose, the algorithm would trend to stay more often on the hills, which implies extra function evaluations.

Next section describes and proves formally the convergence of our non monotone algorithm. It also includes a new approach that we call MultiLine Search (MLS). Section 3 gathers implementation remarks and report preliminary results to compare the monotone version with its non monotone counterpart.

Our notation is standard with minor peculiarities: all vectors are in the Euclidean space $I\!R^n$, unless otherwise stated. $I\!R^*_+$ are vectors in $I\!R^*$ with non negative components; $x^T y$ is the usual inner product $\sum_{k=1}^n x^k y^k$, and $M = xy^T$ is an $n \times n$ matrix with elements $m^{ij} = x^i y^j$. Lower case Greek letters are real values, capital Latin letters I, J, K are subsets of iteration indices. An infinite sequence is denoted as $\{(\cdot)_i\}_1^\infty$, and a subsequence by $\{(\cdot)_i\}_{i \in J}$. The notation $\{(\cdot)_i\}_{i \in I} \leq \alpha$ means that all elements in the subsequence are real numbers not

Table 1. **Non Monotone Algorithm (NMA)**

```
Input: $\mu < 1 \leq \gamma, \epsilon > 0$                constants
       $i = 0$, Choose $x_1, \tau_1$                      Initial values
       DO   $i = i + 1$                                   next iteration
            Update $\varphi_i$                            satisfying A5
            Choose $d_i : 0 < ||d_i|| \leq \tau_i$
            IF $f(x_i + d_i) \leq \varphi_i - \phi(\tau_i)$
                $x_{i+1} = x_i + d_i$                     move
                $0 < \tau_{i+1} \leq \gamma ||d_i||$      $||d||$ may expand
            ELSE
                $x_{i+1} = x_i$                           no move
                $\tau_{i+1} = \mu \tau_i$                 line search, trust region
       LOOP UNTIL $||\nabla f(x_i)|| < \epsilon$
```

bigger than α. Throughout the paper the set $D_i = \{d_{i1}, \ldots, d_{im}\}$ is a set of m unit vectors in $I\!R^n$. The set $\tau D = \{\tau d : d \in D\}$.

2. Non monotone algorithms

Our aim is to propose a non monotone algorithm that generates a converging subsequence $\{x_i\}_{i \in J} \to \bar{x}, \nabla f(\bar{x}) = 0$ under suitable conditions. Table 1 describes the algorithm as close as possible to a gradient related method, including the usual stopping criterium $||\nabla f(x)|| < \epsilon$. This version tries to satisfy (3) for $x_{i+1} = x_i + d_i$. Table 2 describes a more practical version where (3) is *tested* on multiple search directions. Theorem 4 below proves that with a slight modification and some usual extra assumptions on $f(\cdot)$ the non monotone algorithm exhibits a superlinear rate of convergence. We now list the assumptions and prove convergence.

A1: $f(\cdot)$ is bounded below, and $\{x_i\}_1^\infty$ remains in a compact set,

A2: $f(\cdot)$ is Fréchet differentiable, that is, $\nabla f(\cdot) : I\!R^n \to I\!R^n$ is everywhere defined and $f(x + d) = f(x) + \nabla f(x)^T d + o(||d||)$ for all $x, d \in I\!R^n$.

A3: $\exists(\alpha > 0) : \left\{ \frac{\nabla f(x_i)^T d_i}{||\nabla f(x_i)|| \, ||d_i||} \right\}_{i \in I} \leq -\alpha$.

A4: $[\{d_i\}_{i \in I} \to 0] \Rightarrow [\{\nabla f(x_i)\}_{i \in I} \to 0]$

A5: Let J be the index set of *successful* iterations. The sequence of reference values $\{\varphi_i\}_1^\infty$

a) *is an upper bound,* $f(x_i) \leq \varphi_i$, and

b) *decreases sufficiently every "q" successful iterations*, i.e.,

  1. $\forall(i \in J)\exists(j \in J, i < j) : \varphi_j \leq \varphi_i - \Phi(||d_i||)$, where $\Phi(\cdot) : \mathbb{R}_+ \to \mathbb{R}_+$, and for any index subset K $[\{\Phi(||d_i||)\}_{i \in K} \to 0] \Rightarrow [\{||d_i||\}_{i \in K} \to 0]$.

  2. Between i and j there are at most "q" successful iterations.

Assumptions **A1-A4** are required by most algorithms that solve smooth problems. **A5** is easy to comply. The sequence $\{\varphi_i\}_1^\infty$ may remain constant except at those iterations where it is forced to decrease. It is easy to show that (2) is a special case. We now prove that the non monotone algorithm is well defined; specifically we have

LEMMA 1 *If* $f(x_j) > \varphi_i - \phi(||d_i||)$ *for all* $i \geq j$, *then* $\nabla f(x_j) = 0$.

Proof: As $||d_{i+1}|| = \mu||d_i||$ we have that $\{||d_i||\}_1^\infty \to 0$. Besides, for all $i \geq j$ we have that $f(x_j + d_i) > \varphi_i - \phi(||d_i||)$; therefore

$$\begin{aligned}\nabla f(x_j)^T d_i = &\ f(x_j + d_i) - f(x_j) - o(||d_i||) \\ > &\ \varphi_i - f(x_j) - \phi(||d_i||) - o(||d_i||) \\ \geq &\ -o(||d_i||) - \phi(||d_i||)\end{aligned}$$

Coupling this inequality with **A3** we assert for $i \in I$ that

$$0 \geq -\alpha||\nabla f(x_j)|| \geq \nabla f(x_j)^T \frac{d_i}{||d_i||} > -\frac{o(||d_i||)}{||d_i||} - \frac{\nu(||d_i||)}{||d_i||}.$$

Since $||d_i|| \to 0$ we deduce that $||\nabla f(x_j)|| = 0$■

LEMMA 2 $\{d_i\}_1^\infty \to 0$.

Proof: If the number of successful iterations is finite then by construction $||d_{i+1}|| = \mu||d_i||$ for all i large enough and the lemma is valid.

If, on the contrary, the number of successful iterations is infinite, let K be the index set where the upper bound actually decreases. For any two consecutive indices $i, j \in K$ we have that $f(x_j) \leq \varphi_j \leq \varphi_i - \Phi(||d_i||)$; hence $\{\Phi(||d_i||)\}_{i \in K} \to 0$, otherwise $\{\varphi_i\}_{i \in K}$ would be unbounded below, which in turn forces $\{f(x_i)\}_{i \in K}$ to be unbounded below contradicting **A1**. By **A4** we deduce that $\{||d_i||\}_{i \in K} \to 0$; but for any $j \notin K$, $||d_j|| \leq \gamma^q||d_i||$, for some $i \in K$. Therefore, we conclude that $\{d_i\}_1^\infty \to 0$■

As a direct consequence of the previous lemma we can state

REMARK 3 *If* $\Phi(||d||) = \phi(||d||) = 0.01\, d^T d$, *the convergence of a descent method that satisfies **A1-A4** is ensured, provided* $||d_{i+1}|| \leq \gamma||d_i||$ *at all successful iterations.*

We now establish the superlinear rate of convergence along the lines given in [14, theorem 4.1.8].

THEOREM 4 (SUPERLINEAR RATE) *Assume that for all i large enough: The Hessian $\nabla^2 f(x_i)$ is uniformly positive definite and $B_i d_i = -\nabla f(x_i)$, where $B_i \in I\!R^{n\times n}$ is a uniformly positive matrix that satisfies the necessary condition for superlinear rate of convergence $||(\nabla^2 f(x_i) - B_i)d_i|| = o(||d_i||)$.*

The proposed Non monotone algorithm exhibits a superlinear rate of convergence if $\lim_{\tau\downarrow 0} \phi(||\tau||)/\tau^2 = 0$.

Proof: We assume that the proof is asymptotic: it happens for all i large enough. From lemma 2 we obtain that $\{d_i\}_1^\infty \to 0$ and by the assumptions in the theorem we also obtain that $\{\nabla f(x_i)\}_1^\infty \to 0$. The algorithm exhibits a superlinear rate of convergence if and only if $f(x_i + d_i) \le \varphi_i - \phi(||d_i||)$ [3]. Let $v_i = (\nabla^2 f(x_i) - B_i)d_i$. Note that

$v_i^T d_i = o(d_i^T d_i)$ and $d_i^T \nabla f(x_i) = -d_i^T B_i d_i = v_i^T d_i - d_i^T \nabla^2 f(x_i) d_i$;

therefore

$$\begin{aligned} f(x_i + d_i) &= f(x_i) + d_i^T \nabla f(x_i) + \tfrac{1}{2} d_i^T \nabla^2 f(x_i) d_i + o(d_i^T d_i) \\ &= f(x_i) - \tfrac{1}{2} d_i^T \nabla^2 f(x_i) d_i + v_i^T d_i + o(d_i^T d_i) \\ &= f(x_i) + d_i^T d_i \left(-\frac{1}{2} \frac{d_i^T \nabla^2 f(x_i) d_i}{d_i^T d_i} + \frac{v_i^T d_i + o(d_i^T d_i)}{d_i^T d_i} \right) \end{aligned}$$

Let $\lambda > 0$ be a lower bound of the minimum eigenvalue of $\{\nabla^2 f(x_i)\}$ for all i large enough. When $||d_i||$ is small enough we obtain

$$(|v_i^T d_i| + |o(d_i^T d_i)|)/d_i^T d_i \le \tfrac{1}{4}\lambda;$$

hence, $f(x_i + d_i) \le f(x_i) - \frac{\lambda}{4} d_i^T d_i \le \varphi_i - \phi(||d_i||)$ ∎

2.1 Line Search(LS), MultiLine Search(MLS), Trust Region(TR)

A straightforward implementation of the NMA is by line search (LS); that is: if $f(x_i + d_i) \le \varphi_i - \phi(||d_i||)$, it generates $x_{i+1} = x_i + d_i$ as its monotone counterpart; otherwise, it simply defines $d_{i+1} = \mu d_i$ and proceeds with the next iteration. The Trust Region (TR) approach is a natural extension of LS. It tries to satisfy (3) on a ball of radios τ_i around x_i. This technique has attracted a lot of interest in the optimization community [1, 2, 13]. Essentially TR replaces the *true* function $f(x_i + d_i)$ by a *model* $\tilde{f}(x_i, d)$ and finds

$$d_i = \arg \min_{||d|| \le \tau_i} \tilde{f}(x_i, d). \tag{4}$$

$x_{i+1} = x_i + d_i$ is accepted if (3) holds; otherwise, it is rejected. The τ value is adjusted depending upon the proximity of the model value $\tilde{f}(x_i, d_i)$ to the true value $f(x_i + d_i)$. There are various issues that TR must face, mainly

Table 2. **Multiple LineSearch NonMonotone Algorithm (MLSNMA)**

```
PSEUDOCODE                                                   REMARKS
$\tau = 2; \epsilon = 10^{-6}\sqrt{n}, x \in I\!R^n$                      Remark 7
$f_x = f(x), \varphi = \max(f_z/2, 2f_z) + 10$
success= 0
DO  Generate $d$                                              Remark 8
    IF $||d|| > 500\sqrt{n}\min(0.01, \tau)$
        Contract $d : ||d|| = 500\sqrt{n}\min(0.01, \tau)$
    ELSEIF $(||d|| < \min(10^{-4}, \tau)\,\tau)$                     Remark 9
        Expand $d : ||d|| = \min(10^{-4}, \tau)\,\tau$
    $\tau = ||d||$                                              Keep $||d||$
    Generate $D = \{d_1, \ldots, d_n\}$                               Remark 10
    $k = 0$; done= FALSE
    WHILE (NOT DONE) AND $(k \leq n)$
        LINESEARCH $(d_k)$                                    Remark 11
        $k = k + 1$                                           Next direction
    IF (NOT DONE)                                             $x$ is blocked
        $\tau = 0.2\,\tau$
UNTIL $(\tau < \epsilon)$                                         Remark 12
```

- To define an appropriate model, and
- to solve subproblem 4

Current research offers several options that greatly affect the TR performance. See [1, 11, 18] and references therein to be aware of the difficulties encountered in TR methods. We propose here another technique, which is, computationally, between LS and TR. Instead of solving subproblem (4), we carry out a multiline search (MLS). Specifically, given the iterate $x_i \in I\!R^n, \varphi_i \geq f(x_i)$, a set of m unit directions $D_i = \{d_{i1}, \ldots, d_{im}\}$ and a parameter $\tau_i > 0$, we declare that x_i is *blocked* if

$$\forall (d \in \tau_i D_i) : f(x_i + d) > \varphi_i - \phi(\tau_i), \tag{5}$$

where $\phi(\cdot) : I\!R_+ \longrightarrow I\!R_+$ and $\lim_{\tau \downarrow 0} \phi(\tau)/\tau = 0$. To try to unblock x_i the algorithm imposes a reduction on the norm of the next search directions; namely $d_{i+1} \in \mu\tau_i D_{i+1}, \mu < 1$. The iteration will be considered successful if $x_{i+1} = x_i + d$ satisfies (3) for some $d \in \tau_i D_i$. It is obvious that under assumption **A6** below the algorithm ensures convergence.

A6: $D_i = \{d_{i1}, \ldots, d_{im}\}$ is a *finite* set of m unit directions and $\exists d \in D_i$ that satisfies **A3**.

This assumption cannot be verified on some practical problems, where no derivative information is at hand; however the following theorem is very useful when D_i positively spans $I\!R^n$, that is,

$\forall(x \in I\!R^n)\exists(\alpha_1 \geq 0, \ldots, \alpha_m \geq 0) : x = \alpha_1 d_{i1} + \cdots + \alpha_m d_{im}$.

THEOREM 5 *If $\{D_i\}_1^\infty \rightarrow D = \{d_1, \ldots, d_m\}$ positively span $I\!R^n$, and $f(\cdot)$ is strictly differentiable at limit points of $\{x_i\}_1^\infty$, then*

1 $m \geq (n+1)$.

2 $\forall(x \in I\!R^n)\exists(d \in D_i) : x^T d < 0$.

3 $$\left[\lim_{\substack{x_i \rightarrow \bar{x}, \tau_i \downarrow 0 \\ d_{ik} \rightarrow d_k, k=1,\ldots,m}} \frac{f(x_i + \tau_i d_{ik}) - f(x_i)}{\tau_i} \geq 0\right] \Rightarrow \nabla f(\bar{x}) = 0.$$

Proof: These are known facts. The proof can be found in [4, 5]■

Based on the previous theorem, we propose the following MLS strategy: Generate d_i, let $\tau_i = ||d_i||$, and generate a set D_i of unit directions that satisfies **A6**, with $(d_i/\tau_i) \in D_i$. If (3) holds for $x_{i+1} = x_i + d$, for some $d \in \tau_i D_i$ the iteration has been successful and we proceed with the next iteration; otherwise, we declare that x_i is blocked and go to the next iteration forcing $||d_{i+1}|| = \mu\tau_i$. We now outline the convergence proof of the algorithm described in table 2 and remark 11, which contains this MLS strategy.

THEOREM 6 *Let $f(\cdot)$ be strictly differentiable. Under assumptions **A1,A2, A4,A5,A6** the algorithm shown in table 2 generates a subsequence $\{x_i\}_{i\in I}$ that converges to a point $\bar{x}$ satisfying a necessary optimality condition.*

Proof: If the number of blocked points is infinite we use theorem 5, or lemma 1; otherwise, we use lemma 2■

3. Implementation and numerical results

This section shows up a number of remarks that complement the description of the algorithm given in table 2.

REMARK 7 *The starting point x, the stopping value ϵ, τ, φ, and the number of iterations q where φ is constant may be input parameters.*

REMARK 8 *d may be randomly generated when no derivative information is available. Depending upon the amount of information at hand d could even be the Newton direction.*

REMARK 9 *This safeguard prevents a premature stop due for instance to singularities when* $d = -B\nabla f(x)$.

REMARK 10 *The choice of* D *seems to have a tremendous impact on the performance of derivative free optimization algorithms [5]. When derivative information is available we suggest the orthogonal directions* $d_k = -\mathbf{sign}(u^k)(e_k - 2u^k u)$, *where* $u = -\nabla f(x_i)/||\nabla f(x_i)||, e_k$ *is the* $k-th$ *column of the identity matrix, and* $\mathbf{sign}(\alpha) = 1$ *if* $\alpha \geq 0$, $\mathbf{sign}(\alpha) = 0$ *otherwise. Note that* $d_k^T u = |u^k| \geq 0, k = 1, \ldots, n$; *hence we assert that* $\exists d \in D : d^T u \geq 1/\sqrt{n}$, *because otherwise*

$$1 = \sum_{k=1}^{n} |u^k|^2 = \sum_{k=1}^{n} (d_k^T u)^2 < \sum_{k=1}^{n} (1/n) = 1,$$

a contradiction.

REMARK 11 *This procedure assumes* $||d|| = 1$ *and evaluates* $f(x + \tau d)$. *It returns* TRUE *if the iteration is successful. It also updates* x *and* φ. *We use the updating on* φ *suggested above.*

```
LINESEARCH (d)
    $z = x + \tau d; f_z = f(z)$
    done= FALSE
    IF ($f_z \leq \varphi - \tau \min(10^{-4}, \tau^2)$)
        $x = z; f_x = f_z$            Accept $z$
        success= success+ 1
        IF (success= q)
            $\varphi = f_x$; success = 0    Update $\varphi$
        done= TRUE
end of linesearch
```

REMARK 12 *We have chosen this termination criterium because it is also valid for derivative free optimization.*

We carried out preliminary numerical tests with functions from the Moré, Garbow and Hillstrom collection. The MatLab code was taken from [10] and run on a Pentium 4 desk computer. We used the quasi Newton direction $d = -\mathbf{sign}(\nabla f(x)^T B \nabla f(x)) B \nabla f(x)$, and B was updated with the symmetric formula $B = B + (s - By)(s - By)^T/(s - By)^T y$, where $s = x_{i+1} - x_i, y = \nabla f(x_{i+1}) - \nabla f(x_i)$. Table 3 shows the number of function evaluations needed for functions SINGX, ROSENX, which have an adjustable number of variables. For $q \in \{1, 5, 10, 20\}$ it was observed that the algorithm's performance generally improves for $q > 1$. It was also observed in tests not reported here that LS was superior to MLS on the steepest descent method. These results are by no way conclusive, and a more complete numerical test must be carried out in future research.

Table 3. **# of Function evaluations**

	singx				rosenx			
	φ constant (q)				φ constant (q)			
variables	**1**	**5**	**10**	**20**	**1**	**5**	**10**	**20**
8	321	283	159	154	533	362	343	343
12	647	583	366	232	988	1006	990	793
24	930	930	844	844	3949	2772	2838	2463

References

[1] B. Addis, S. Leiffer. A trust region algorithm for global optimization. Preprint ANL/MCS-P1190-0804, Argonne National Laboratory, Il, USA, 2004.

[2] A.R. Conn, N.I.M. Gould, P.L. Toint. *Trust region methods.* MPS-SIAM Series on Optimization, Philadelphia, ISBN 0-89871-460-5, 2000.

[3] J.E. Dennis, J.J. Moré. A characterization of superlinear convergence and its application to quasi-Newton methods. *Mathematics of Computation* 28:549-560, 1974.

[4] U.M. García-Palomares, F.J. González-Castaño, J.C. Burguillo-Rial. A combined global & local search (CGLS) approach to global optimization. *Journal of Global Optimization* To appear, 2006.

[5] U.M. García-Palomares, J.F. Rodríguez. New sequential and parallel derivative-free algorithms for unconstrained optimization. *SIAM Journal on Optimization* 13:79-96, 2002.

[6] N.I.M. Gould, D. Orban, Ph.L. Toint. GALAHAD a library of thread-safe Fortran 90 packages for large scale nonlinear optimization. *Transactions of the ACM on Mathematical Software* 29-4:353-372, 2003.

[7] N.I.M. Gould, C. Sainvitu, Ph.L. Toint. A filter-trust-region method for unconstrained minimization. Report 04/03, Rutherford Appleton Laboratory, England, 2004.

[8] L. Grippo, F. Lampariello, S. Lucidi. A nonmonotone line search technique for Newton's method. *SIAM Journal Numerical Analysis* 23-4:707-716, 1986.

[9] L. Grippo, M. Sciandrone. Nonmonotone globalization techniques for the Barzilai-Borwein gradient method. *Computational Optimization and Applications* 23:143-169, 2002.

[10] C. Gurwitz, L. Klein, M. Lamba. A MATLAB library of test functions for unconstrained optimization. Report 11/94, Brooklin College, USA, 1994.

[11] W.W. Hager. Minimizing a quadratic over a sphere. *SIAM Journal on Optimization* 12:188-208, 2001.

[12] J. Han, J. Sun, W. Sun. Global convergence of non-monotone descent methods for unconstrained optimization problems. *Journal of Computational and Applied Mathematics* 146-1:89-98, 2002.

[13] P.D. Hough, J.C. Meza. A class of trust region methods for parallel optimization. *SIAM Journal on Optimization* 13-1:264-282, 2002.

[14] C.T. Kelley. Iterative methods for optimization. *SIAM Frontiers in Applied Mathematics* ISBN 0-89871-433-8, 1999.

[15] S. Lucidi, M. Sciandrone. On the global convergence of derivative free methods for unconstrained optimization. *SIAM Journal on Optimization* 13-1:119-142, 2002.

[16] V.P. Plagianakos, G.D. Magoulas, M.N. Vrahatis. Deterministic nonmonotone strategies for effective training of multilayer perceptrons. *IEEE Transactions on Neural Networks* 13-6:1268-1284, 2002.

[17] M. Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal on Optimization* 7:26-33, 1997.

[18] M. Rojas, S. Santos, D. Sorensen. LSTRS: Matlab software for large-scale trust-region subproblems and regularization. Technical Report 2003-4, Department of mathematics, Wake Forest University, NC, USA, 2003.

[19] P.L. Toint. An assesment of non-monotone linesearch techniques for unconstrained optimization. *SIAM Journal on Scientific and Statistical Computing* 8-3:416-435, 1996.

[20] P.L. Toint. Non-monotone trust region algorithms for nonlinear optimization subject to convex constraints. *Mathematical Programming* 77:69-94, 1997.

[21] J.L. Zhang, X.S. Zhang. A modified SQP method with nonmonotone linesearch technique. *Journal of Global Optimization* 21:201-218, 2001.