# PRECONDITIONED CONJUGATE GRADIENT ALGORITHMS FOR NONCONVEX PROBLEMS WITH BOX CONSTRAINTS

R. Pytlak,[1] and T. Tarnawski,[2]
[1]*Faculty of Cybernetics, Military University of Technology, 00-908 Warsaw, Poland, rpytlak@isi.wat.edu.pl* [2]*Faculty of Cybernetics, Military University of Technology, 00-908 Warsaw, Poland, tarni@isi.wat.edu.pl*

**Abstract** The paper describes a new conjugate gradient algorithm for large scale nonconvex problems with box constraints. In order to speed up the convergence the algorithm employs a scaling matrix which transforms the space of original variables into the space in which Hessian matrices of functionals describing the problems have more clustered eigenvalues. This is done efficiently by applying limited memory BFGS updating matrices. Once the scaling matrix is calculated, the next few iterations of the conjugate gradient algorithms are performed in the transformed space. The box constraints are treated by the projection as previously used in [R. Pytlak, The efficient algorithm for large-scale problems with simple bounds on the variables, SIAM J. on Optimization, Vol. 8, 532-560, 1998]. We believe that the preconditioned conjugate gradient algorithm gives more flexibility in achieving balance between the computing time and the number of function evaluations in comparison to a limited memory BFGS algorithm. The numerical results show that the proposed method is competitive to L-BFGS-B procedure.

**keywords:** bound constrained nonlinear optimization problems, conjugate gradient algorithms, quasi-Newton methods.

## 1. Introduction

In this paper we consider algorithms for the problem:

$$\min_{x \in \mathcal{R}^n} f(x) \tag{1}$$

$$\text{s. t. } l \leq x \leq u, \tag{2}$$

where $l,\ u \in \mathcal{R}^n$.

In [9] (see also [4]) a new family of conjugate gradient algorithms has been introduced whose direction finding subproblem is given by

$$d_k = -\mathbf{Nr}\{g_k, -\beta_k d_{k-1}\}, \tag{3}$$

Please use the following format when citing this chapter:
Author(s) [insert Last name, First-name initial(s)], 2006, in IFIP International Federation for Information Processing, Volume 199, System Modeling and Optimization, eds. Ceragioli F., Dontchev A., Furuta H., Marti K., Pandolfi L., (Boston: Springer), pp. [insert page numbers].

where $\mathbf{Nr}\{a, b\}$ is defined as the point from a line segment spanned by the vectors $a$ and $b$ which has the smallest norm, i.e.,

$$\| \mathbf{Nr}\{a, b\} \| = \min\{\| \lambda a + (1 - \lambda) b \| : 0 \leq \lambda \leq 1\}, \tag{4}$$

$\| \cdot \|$ is the Euclidean norm and $g_k = \nabla f(x_k)$.

Notice that if $\beta_k = 1$ then we have the Wolfe–Lemaréchal algorithm ([7], [13]). In [9] it was shown that the Wolfe–Lemaréchal algorithm is in fact the Fletcher–Reeves algorithm when directional minimization is exact. Moreover, the sequence $\{\beta_k\}$ was constructed in such way that directions generated by (3) are equivalent to those provided by the Polak–Ribiére formula (under the assumption that directional minimization is exact). This sequence

$$\beta_k = \frac{\|g_k\|^2}{|\langle g_k - g_{k-1}, g_k \rangle|} \tag{5}$$

has striking resemblance to the Polak–Ribiére formula.

## 2. General preconditioned conjugate gradient algorithm

The idea behind preconditioned conjugate gradient algorithm is to transform the decision vector by linear transformation $D$ such that after the transformation the nonlinear problem is *easier* to solve. If $\hat{x}$ is transformed $x$:

$$\hat{x} = Dx \tag{6}$$

then our minimization problem will become

$$\min_{\hat{x}} \left[ \hat{f}(\hat{x}) = f(D^{-1}\hat{x}) \right] \tag{7}$$

and for this problem the search direction will be defined as follows

$$\hat{d}_k = -\mathbf{Nr}\{\nabla \hat{f}(\hat{x}_k), -\hat{\beta}_k \hat{d}_{k-1}\} \tag{8}$$

Notice that

$$\nabla \hat{f}(\hat{x}) = D^{-T} \nabla f(\hat{x}) \tag{9}$$

therefore we can write

$$\hat{d}_k = -\mathbf{Nr}\{D^{-T} \nabla f(D^{-1}\hat{x}_k), -\hat{\beta}_k \hat{d}_{k-1}\}. \tag{10}$$

If we multiply both sides of (10) by $D^{-1}$ we will get

$$d_k = -\lambda_k D^{-1} D^{-T} \nabla f(x_k) + (1 - \lambda_k) \hat{\beta}_k d_{k-1}. \tag{11}$$

where $0 \leq \lambda_k \leq 1$ and either

$$\hat{\beta}_k = 1 \tag{12}$$

for the Fletcher-Reeves version, or

$$\hat{\beta}_k = \frac{\|\hat{g}_k\|^2}{|\langle \hat{g}_k - \hat{g}_{k-1}, \hat{g}_k\rangle|} = \frac{g_k^T D^{-1} D^{-T} g_k}{|(g_k - g_{k-1})^T D^{-1} D^{-T} g_k|}$$

for the Polak-Ribiere version.

The equation (11) can be stated as

$$d_k = -\lambda_k H \nabla f(x_k) + (1 - \lambda_k)\beta_k d_{k-1}. \tag{13}$$

where $H = D^{-1}D^{-T}$. This suggests that $D$ should be chosen in such a way that $D^T D$ is an approximation to $\nabla^2_{xx} f(\bar{x})$ where $\bar{x}$ is a solution of problem (1).

Moreover, if $D$ is an upper triangular matrix then at each iteration of the algorithm we will have to solve the system of linear equations

$$D^T \hat{g}_k = g_k\,, \qquad D d_k = \hat{d}_k. \tag{14}$$

It is worthwhile to notice that the following holds (see [11]):

$$\langle g_k, d_k\rangle \leq -\|\hat{d}_k\|^2 \text{ and } \langle g_k, d_k\rangle = -\|\hat{d}_k\|^2, \tag{15}$$

if $0 < \lambda_k < 1$.

If box constraints (2) are present in our problem then we can tackle them by using the projection procedure proposed initially in [1] (see also [10]).

In the rest of the paper we consider, for the simplicity of presentation, the problems with simpler constraints $x \geq 0$. We define the set of indices $I_k^+$

$$I_k^+ := \{i \in \overline{1,n} : \ (x_k)_i \leq \varepsilon_k \text{ and } \nabla_{x_i} f(x_k) > 0\}, \tag{16}$$

where $\{\varepsilon_k\}$ is such that $\varepsilon_k > 0$ and

$$\lim_{k \in K} \|x_k - P[x_k - \nabla f(x_k)]\| = 0 \quad \Leftrightarrow \quad \lim_{k \in K} \varepsilon_k = 0. \tag{17}$$

for any subsequence $\{x_k\}_{k \in K}$. Here, by $P[\cdot]$ we denote the projection operator on the set $\{x \in \mathcal{R}^n : \ l \leq z \leq u\}$ ([1]).

The sets $I_k^+$ are used to modify the direction finding subproblem. Instead of solving problem (3) we find a new direction according to the rule

$$d_k = -\mathbf{Nr}\{\nabla f(x_k), -\beta_k d_{k-1}^+\}. \tag{18}$$

Here $d_{k-1}^+$ is defined by

$$(d_{k-1}^+)_i := \left\{ \begin{array}{lll} & (d_{k-1})_i & \text{if } i \notin I_k^+ \\ - & \nabla_{x_i} f(x_k)/\beta_k & \text{if } i \in I_k^+ \end{array} \right. . \tag{19}$$

To complete the description of the main components of our algorithm we have to show how to use scaling matrices in its preconditioned version. Having the set of indices $I_k^+$ we do not scale variables corresponding to them and we apply general scaling to the others. Therefore, we use the scaling matrix of the form

$$D_k = \begin{bmatrix} D & 0 \\ 0 & I_{n_k} \end{bmatrix}.$$

where $n_k = |I_k^+|$.

In order to describe the line search procedure notice that the function $f(P[x_k + \alpha d_k])$ can be interpreted as a composition of two functions: the first one is Lipschitzian and the second one continuously differentiable. If we define

$$x_k(\alpha) = x_k + d_k(\alpha), \quad \text{where } (d_k(\alpha))_i := \begin{cases} \alpha(d_k)_i & \text{if } \alpha \leq \alpha_k^i \\ \alpha_k^i(d_k)_i & \text{if } \alpha > \alpha_k^i \end{cases} \tag{20}$$

and the breakpoints $\{\alpha_k^i\}_1^n$ are calculated as follows

$$\alpha_k^i := -\frac{(x_k)_i}{(d_k)_i}, \quad i = 1, \dots, n \tag{21}$$

(assuming that if $(d_k)_i \geq 0$ then $\alpha_k^i = \infty$), then our directional minimization rule can be stated as follows.

**R1** find the largest positive number $\alpha_k$ from the set $\{\theta^k : k = 0, 1, \dots, \theta \in (0,1)\}$ such that for $\mu \in (0,1)$ we have

$$f(x_k(\alpha_k)) - f(x_k) \leq -\mu \left[ \alpha_k \sum_{i \notin I_k^+} (d_k)_i^2 + \sum_{i \in I_k^+} \frac{\partial f(x_k)}{\partial x^i} \left[ x_k^i - x_k^i(\alpha_k) \right] \right],$$

Notice that in the rule *R1* we employ $d_k$ instead of $\hat{d}_k$ as (15) would imply. Since we assume that $D_k^T D_k$ are uniformly bounded from below and above (in the sense of condition (26)) there exist constants $0 < c_1 < c_2 < +\infty$ such that $c_1 \|d_k\| \leq \|\hat{d}_k\| \leq c_2 \|d_k\|$. Thus the use of $d_k$ on the left on inequality in the rule *R1* is justified (it corresponds to appropriately choosing the coefficient $\mu$). It is worthwhile to observe that our descent direction rule allows for such inaccurate directional minimization search.

Our general algorithm is as follows:

**Algorithm** Parameters: $\mu \in (0,1)$, $\epsilon > 0$, $\{\hat{\beta}_k\}_1^\infty$,
$\{D_k\}_1^\infty$, $D_k \in R^{n \times n}$ nonsingular matrix, $T \in R^{n \times n}$ nonsingular diagonal matrix.
Data: $x_0$

**1)** Set $k = 0$, compute $d_k = -g_k$, go to Step 3).

**2)**Compute: $w_k = x_k - P[x_k - T\nabla f(x_k)]$, $\varepsilon_k = \min(\varepsilon, \|w_k\|)$,

$$\begin{aligned} D_k^T \hat{g}_k &= g_k & (22)\\ \hat{d}_k &= -\mathbf{Nr}\{\hat{g}_k, -\hat{\beta}_k \hat{d}_{k-1}^+\} & (23)\\ D_k d_k &= \hat{d}_k & (24)\end{aligned}$$

If $w_k = 0$ then STOP.

**3)** Find a positive number $\alpha_k$ according to the rule *R1*.

**4)** Substitute $P[x_k + \alpha_k d_k]$ for $x_{k+1}$, increase $k$ by one, go to Step 2.

We can prove the lemma

LEMMA 1 *Assume that $x_k$ is a noncritical point, $D_k^T D_k$ is positive definite and $d_k \neq 0$ is calculated in* Step 2 *of* Algorithm. *Then there exists a positive $\alpha_k$ such that the condition stated in the rule* R1 *is*

$$\lim_{\alpha \to \infty} f(P[x_k + \alpha d_k]) = -\infty. \qquad (25)$$

To investigate the convergence of *Algorithm* we begin by providing a crucial lemma which requires the following assumptions.

ASSUMPTION 1 *There exists $L < \infty$ such that*

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\|$$

*for all $x$, $y$ from a bounded set.*

ASSUMPTION 2 *There exist $d_l$, $d_u$ such that* $0 < d_l < d_u < +\infty$ *and*

$$d_l\|u\|^2 \leq u^T D_k^T D_k u \leq d_u\|u\|^2 \qquad (26)$$

*for all $u \in \mathcal{R}^n$ and $k$.*

LEMMA 2 *Suppose that* Assumptions 1–2 *hold, the direction $d_k$ is determined by* (22)–(24) *and the step–size coefficient $\alpha_k$ is calculated according to the rule* R1. *Then, for any bounded subsequence $\{x_k\}_{k \in K}$ either*

$$\lim_{k \in K} \|x_k - P[x_k + d_k]\| = 0, \qquad (27)$$

*or*

$$\lim_{k \in K} \|x_k - P[x_k - \nabla f(x_k)]\| = 0. \qquad (28)$$

For the convenience of future notations we assume that variables $(x)_i$ have been reordered in such a way that $d_k$ can be partitioned into two vectors $(d_k^1, d_k^2)$ where the first vector $d_k^1$ is represented by

$$d_k^1 := \{(d_k)_i\}_{i \notin I_k^+}.$$

The same convention applies to other vectors.

THEOREM 3 *Suppose that* Assumptions 1–2 *are satisfied. Moreover, assume that for any convergent subsequence* $\{x_k\}_{k\in K}$ *whose limit is not a critical point*

*i)* $\{\hat{\beta}_k\}$ *is such that*

$$\liminf_{k\to\infty}\left(\hat{\beta}_k\|d_{k-1}^1\|\right) \geq \nu_1 \liminf_{k\to\infty}\|\nabla^1 f(x_k)\| \tag{29}$$

*where* $\nu_1$ *is some positive constant,*

*ii) there exists a number* $\nu_2$ *such that* $\nu_2\|D_k^{-T}\|_2\|D_{k-1}\|_2 \in (0,1)$ *and*

$$\langle\nabla^1 f(x_k), d_{k-1}^1\rangle \leq \nu_2\|\nabla^1 f(x_k)\|\|d_{k-1}^1\|, \; whenever \; \lambda_k \in (0,1). \tag{30}$$

*Then* $\lim_{k\to\infty} f(x_k) = -\infty$, *or every accumulation point of the sequence* $\{x_k\}_0^\infty$ *generated by* Algorithm *is a critical point.*

Our global convergence result is as follows.

THEOREM 4 *Suppose that* Assumptions 1–2 *are satisfied. Then* Algorithm *generates* $\{x_k\}$ *such that every accumulation point of* $\{x_k\}$ *satisfies necessary optimality conditions for problem (1)–(2) provided that:*

*i)* $\hat{\beta}_k$ *is given by*

$$\hat{\beta}_k = \frac{\|\nabla^1\hat{f}((\hat{x}_k^1,\hat{x}_{k-1}^2))\|^2}{|\langle\nabla^1\hat{f}((\hat{x}_k^1,\hat{x}_{k-1}^2)) - \nabla^1\hat{f}(\hat{x}_{k-1}), \nabla^1\hat{f}((\hat{x}_k^1,\hat{x}_{k-1}^2))\rangle|}, \tag{31}$$

*ii) there exists* $M < \infty$ *such that* $\alpha_k \leq M$, $\forall k$.

## 3. Scaling matrices based on the compact representation of BFGS matrices

In the previous section we showed that for a given nonsingular matrix $H^{-1} = D^TD$ the preconditioned conjugate gradient algorithm is globally convergent. The use of constant scaling matrix is likely to be inefficient since the function $f$ we minimize is nonlinear. Therefore, we are looking at the sequence of matrices $\{H_k\}$ such that each $H_k^{-1}$ is as close as possible to the Hessian $\nabla_{xx}^2 f(x_k)$ and can be easily factorized as $D_k^{-1}D_k^{-T}$ where $D_k$ is a nonsingular matrix. We assume, for the simplicity of presentation, that $n_k \equiv 0$.

In the paper we present the preconditioned conjugate gradient algorithm based on the BFGS updating formula. To this end we recall compact representations of quasi–Newton matrices described in [8]. Suppose that the $k$ vector pairs $\{s_i, y_i\}_{i=0}^{k-1}$ satisfy $s_i^T y_i > 0$ for $i = k - m - 1, \dots, k - 1$. If we assume that $B_0 = \gamma_k I$ and introduce matrices $M_k = [\gamma_k S_k \; Y_k]$,

$$S_k = [s_{k-m-1}, \dots, s_{k-1}], \; Y_k = [y_{k-m-1}, \dots, y_{k-1}] \tag{32}$$

where $s_i = x_{i+1} - x_i$ and $y_i = g_{i+1} - g_i$, then LBFGS approximation to the Hessian matrix is

$$B_k = \gamma_k I - M_k W_k M_k^T \tag{33}$$

and $W_k \in \mathcal{R}^{m \times m}$ is nonsingular ([3]).

In order to transform the matrix $B_k$ to the form $D_k^T D_k$ we do the QR factorization of the matrix $M_k^T$:

$$M_k^T = Q_k R_k \tag{34}$$

where $Q_k$ is $n \times n$ orthogonal matrix and $R_k$ the $n \times m$ matrix which has zero elements except the elements constituting the upper $m \times m$ submatrix. Taking into account that $Q_k^T Q_k = I$ we can write (33) as

$$B_k = Q_k^T \left[\gamma_k I - R_k^T W_k R_k\right] Q_k. \tag{35}$$

Notice that the matrix $R_k^T W_k R_k$ has zero elements except those lying in the upper left $m \times m$ submatrix. We denote this submatrix by $T_k$ and we can easily show that it is a positive definite matrix. If we compute the Cholesky decomposition of the matrix $\gamma_k I_k - T_k$, $\gamma_k I_k - T_k = C_k^T C_k$ then eventually we come to the relation

$$B_k = Q_k^T F_k^T F_k Q_k \tag{36}$$

with

$$F_k = \begin{bmatrix} C_k & 0 \\ 0 & \sqrt{\gamma_k} I_{n-k} \end{bmatrix}. \tag{37}$$

The desired decomposition of the matrix $B_k$ is thus given by

$$B_k = D_k^T D_k, \; D_k = F_k Q_k \tag{38}$$

where the matrix $D_k$ is nonsingular provided that $s_i^T y_i > 0$ for $i = k - m - 1$, $\dots, k - 1$. Notice that the matrix $Q_k$ does not have to be stored since it can be easily evaluated from the Householder vectors which have been used in the QR factorization.

Recall the relation (14) which now can be written as

$$Q_k^T F_k^T \hat{g}_k = g_k, \qquad F_k Q_k d_k = \hat{d}_k. \tag{39}$$

## 4. Scaling matrices - the reduced Hessian approach

The approach is based on the limited memory reduced Hessian method proposed by Gill and Leonard ([5],[6], see also [12])

Suppose that $\mathcal{G}_k = \text{span}\,\{g_0, g_1, \ldots, g_k\}$ and let $\mathcal{G}_k^\perp$ denote the orthogonal complement of $\mathcal{G}_k$ in $\mathcal{R}^n$. If $B_k \in \mathcal{R}^{n \times r_k}$ have columns that define the basis of $\mathcal{G}_k$ and

$$H_k = Q_k T_k$$

is the QR decomposition of $B_k$ then

$$Q_k^T B_k Q_k = \begin{pmatrix} Z_k^T B_k Z_k & 0 \\ 0 & \sigma I_{n-r_k} \end{pmatrix} \tag{40}$$

where $Q_k = (Z_k\, W_k)$ and $\text{range}(B_k) = \text{range}(Z_k)$. (40) follows from the theorem which was stated, among others, in [6]:

THEOREM 5 *Suppose that the BFGS method is applied to a general nonlinear function. If $B_0 = \sigma I_n$ and*

$$B_k d_k = -g_k,$$

*then $d_k \in \mathcal{G}_k$ for all $k$. Furthermore, if $z \in \mathcal{G}_k$ and $w_k \in \mathcal{G}_k^\perp$, then $B_k z \in \mathcal{G}_k$ and $B_k w = \sigma w$.*

From (40) we have

$$B_k = D_k^T D_k = Q_k \begin{pmatrix} Z_k^T B_k Z_k & 0 \\ 0 & \sigma I_{n-r_k} \end{pmatrix} Q_k^T$$

Therefore, it follows that we can take as $D_k$:

$$D_k = \begin{pmatrix} R_k & 0 \\ 0 & \sqrt{\sigma} I_{n-r_k} \end{pmatrix} Q_k^T = G_k Q_k^T$$

At every iteration we have to solve equations

$$Q_k G_k^T \hat{g}_k\,, \qquad = g_k G_k Q_k^T d_k = \hat{d}_k.$$

Solving these equations requires multiplication of vectors in $\mathcal{R}^n$ by the orthogonal matrix $Q_k$ (or $Q_k^T$), and this can be achieved by the sequence of $m$ multiplications of the Householder matrices $H_k^i$, $i = 1, \ldots, m$ such that $Q_k = H_k^1 H_k^2 \cdots H_k^m$. The cost of these multiplications is proportional to $n$. Furthermore, we have to solve the set on $n$ linear equations with the upper triangular matrix $G_k$, or its transpose.

## 5. Numerical experiments

In order to verify the effectiveness of our algorithm we have tested it on problems from the CUTE collection ([2]). We tried it on problems with various dimension although its application is recommended for solving large scale problems.

*Algorithm* has been implemented in C on Intel PC under Linux operating system. We compared our algorithm with the L-BFGS-B code which is the benchmark procedure for problems with box constraints for which evaluating the Hessian matrix is too expensive. L-BFGS-B code was used with the parameter $m = 5$ and we applied $m = 5$ and we recalculated matrices $D_k$ every five iterations in *Algorithm*. The stopping criterion was $\|\nabla f(x)\|/\max(1, \|x\|) \leq 10^{-7}$. We used the scaling matrices as described in Section 3.

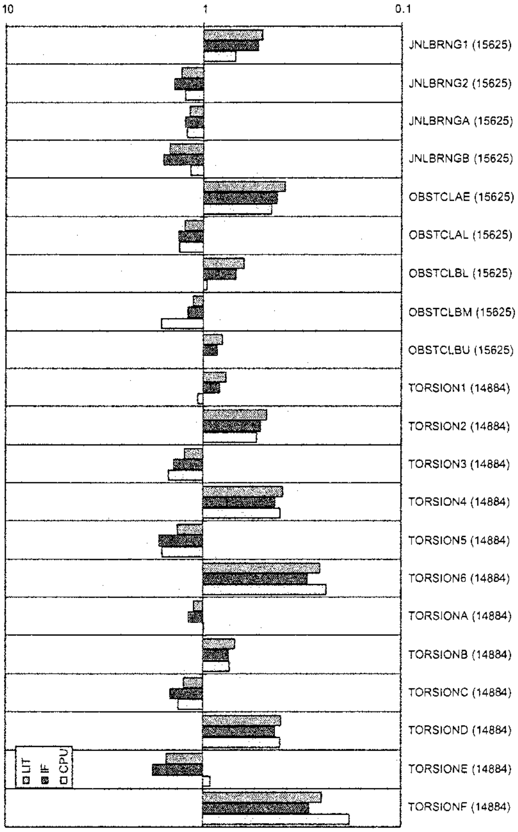

*Figure 1.* Performance comparison of *Algorithm* against the L-BFGS-B code.

The performance comparison of *Algorithm* is given in Figure 1. where we compare it with the code L-BFGS-B presented in [14]. For each problem the bars represent the ratio of the number of iterations (LIT), number of function evaluations (IF) and computing time (CPU) needed by the *Algorithm* divided by those from the executions of the L-BFGS-B code. Therefore values above one testify in favor of the L-BFGS-B and below one – in favor of our algorithm.

## References

[1] D.P. Bertsekas, Projected Newton methods for optimization problems with simple constraints. *SIAM J. Control and Optimiz.* 20:221-245. 1982.

[2] I. Bongartz, A.R. Conn, N.I.M. Gould, Ph.L. Toint, CUTE: Constrained and Unconstrained Testing Environment. Research Report RC 18860, IBM T.J. Watson Research Center, Yorktown Heights, NY, USA, 1994.

[3] R. Byrd, J. Nocedal, R. B. Schnabel, Representations of quasi–Newton matrices and their use in limited memory methods. Technical Report NAM-03, 1996.

[4] Y. Dai, Y. Yuan, Global convergence of the method of shortest residuals. *Numerische Mathematik* 83:581-598, 1999.

[5] P.E. Gill, M.W. Leonard, Reduced–Hessian methods for unconstrained optimization. *SIAM J. Optimiz.* 12:209-237, 2001.

[6] P.E. Gill, M.W. Leonard, Limited–memory reduced hessian methods for large–scale unconstrained optimization. *SIAM J. Optimiz.* 14:380-401, 2003.

[7] C. Lemaréchal, An Extension of Davidon methods to nondifferentiable Problem. In *Mathematical Programming Study 3.* North-Holland, Amsterdam, 1975.

[8] J. Nocedal, S.J. Wright, *Numerical optimization.* Springer–Verlag, New York, 1999.

[9] R. Pytlak, On the convergence of conjugate gradient algorithms. *IMA Journal ofNumerical Analysis.* 14:443-460, 1994.

[10] R. Pytlak, An efficient algorithm for large-scale nonlinear programming problems with simple bounds on the variables. *SIAM J. on Optimiz.* 8:532-560, 1998.

[11] R. Pytlak, T. Tarnawski, The preconditione conjugate gradient algorithm for nonconvex problems. Research Report, Military University of Technology, Faculty of Cybernetics, N.-1, 2005.

[12] D. Siegel, Modifying the BFGS update by a new column scaling technique. *Mathematical Programming.* 66:45-78, 1994.

[13] P. Wolfe, A Method of Conjugate Subgradients for Minimizing Nondifferentiable Functions. In *Mathematical Programming Study 3.* North-Holland, Amsterdam, 1975.

[14] C. Zhu, R.H. Byrd, P. Lu, J. Nocedal, Algorithm 778: L-BFGS-B, FORTRAN subroutines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software.* 23:550-560, 1997.