Quasi–Newton updates for Preconditioned Nonlinear Conjugate Gradient methods

Giovanni Fasano and Massimo Roma

Contents

1. Introduction

We deal with the large scale unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a twice continuously differentiable function and n is large. We assume that for a given $x_0 \in \mathbb{R}^n$ the level set

$$\Omega_0 = \{x \in \mathbb{R}^n \ \mid \ f(x) \leq f(x_0)\}$$

is compact. The huge number of real world applications which can be modelled as a large scale optimization problem strongly motivates the growing interest for the solution of such problems.

Among the iterative methods for large scale unconstrained optimization, when the Hessian matrix is possibly dense, limited memory quasi–Newton methods are often the methods of choice. As well known (see any textbook, e.g. [11]), they generate a sequence $\{x_k\}$, according to the following scheme

$$x_{k+1} = x_k + \alpha_k p_k, \qquad k = 0, 1, \ldots, \tag{1.2}$$

with

$$p_k = -H_k \nabla f(x_k),$$

where H_k is an approximation of the inverse of the Hessian matrix $\nabla^2 f(x_k)$ and α_k is a steplength. In particular, instead of computing H_k at each iteration k, these methods update H_k in a simple manner, in order to obtain the new approximation H_{k+1} to be used in the next iteration. Moreover, instead of storing full dense $n \times n$ approximations, they only save a few vectors of length n, which allow to represent the approximations implicitly.

Among the quasi–Newton schemes, the L–BFGS method is usually considered one of the most efficient. It is well suited for large scale problems because the amount of storage is limited and controlled by the user. This method is based on the construction of the approximation of the inverse of the Hessian matrix, by exploiting curvature information gained only from the most recent iterations. The inverse of the Hessian matrix is updated at the k-th iteration by the formula

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T \tag{1.3}$$

where

$$\rho_k = \frac{1}{y_k^T s_k}, \quad V_k = I - \rho_k y_k s_k^T,$$

and

$$s_k = x_{k+1} - x_k = \alpha_k p_k, \qquad y_k = \nabla f(x_{k+1}) - \nabla f(x_k). \tag{1.4}$$

Observe that H_k also satisfies relation

$$\begin{aligned} H_k \;=\; & (V_{k-1}^T \cdots V_{k-m}^T) H_k^0 (V_{k-m} \cdots V_{k-1}) \\ & + \rho_{k-m} (V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1} \cdots V_{k-1}) \\ & + \rho_{k-m+1} (V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2} \cdots V_{k-1}) \\ & + \cdots \\ & + \rho_{k-1} s_{k-1} s_{k-1}^T, \end{aligned}$$

where m is the *memory* of the method and H_k^0 is an initial approximation of the inverse of the Hessian matrix.

The well known reasons for the success of the L–BFGS method can be summarized in the following two points: firstly, even when m is small, H_{k+1} is an effective approximation of the inverse of the Hessian matrix, secondly H_{k+1} is the unique (positive definite) matrix which solves the problem

$$\begin{aligned} \min_H \;\; & \|H - H_k\|_F \\ s.t. \;\; & H = H^T \\ & s_k = H y_k, \end{aligned}$$

where $\|\cdot\|_F$ is the Frobenius norm. Namely, H_{k+1} is the positive definite matrix "closest" to the current approximation H_k, satisfying the *secant equation* $s_k = H y_k$. However, L–BFGS method presents some drawbacks, including the slow convergence on ill–conditioned problems, namely when the eigenvalues of

the Hessian matrix are very spread. Moreover, on some applications, the performances of L–BFGS method and the Nonlinear Conjugate Gradient method are comparable.

In this paper we focus on the latter method: the Nonlinear Conjugate Gradient method (NCG). As well known (see any textbook, e.g. [11]) it is a natural extension to general functions of the linear Conjugate Gradient (CG) method for quadratic functions. It generates a sequence $\{x_k\}$ according to scheme (1.2), with

$$p_k = -\nabla f(x_k) + \beta_k p_{k-1},$$

where β_k is a suitable scalar. Different values of β_k give rise to different algorithms (see [8] for a survey). The most common are the Fletcher and Reeves (FR), the Polak and Ribière (PR) and the Hestenes and Stiefel (HS) algorithms.

Although the NCG methods have been widely studied and are often very efficient when solving large scale problems, a key point for increasing their efficiency is the use of a preconditioning strategy, especially when solving difficult ill–conditioned problems. Defining good preconditioners for NCG methods is currently still considered a challenging research topic. On this guideline, this work is devoted to investigate the use of quasi–Newton updates as preconditioners. In particular, we want to propose preconditioners which possibly inherit the effectiveness of the L–BFGS update. Indeed, here we build preconditioners iteratively defined and based on quasi–Newton updates of the inverse of the Hessian matrix. This represents an attempt to improve the efficiency of the NCG method by conveying information collected from a quasi–Newton method, in a Preconditioned Nonlinear Conjugate Gradient method (PNCG). In particular, we study new symmetric low–rank updates of the inverse of the Hessian matrix, in order to iteratively define preconditioners for PNCG.

It is worth to note that there exists a close connection between BFGS and NCG [10], and on the other hand, NCG algorithms can be viewed as memoryless quasi–Newton methods (see e.g., [13], [12], [11]).

The idea of using a quasi–Newton update as a preconditioner within NCG algorithms is not new. In [2], when storage is available, a preconditioner defined by m quasi–Newton updates is used within NCG algorithm. In [1] a scaled memoryless BFGS matrix is used as preconditioner in the framework

of NCG. Moreover, an automatic preconditioning strategy based on a limited memory quasi–Newton update for the linear CG is proposed in [9], within Hessian free Newton methods, and is extended to the solution of a sequence of linear systems.

In this paper, we propose two classes of parameters dependent preconditioners. In particular, in the next section we briefly recall a scheme of a general PNCG method. In Section 3 a new symmetric rank-2 update is introduced and its theoretical properties are studied. Section 4 is devoted to describe a new BFGS–like quasi–Newton update. Finally, in Section 5 the results of a preliminary numerical experience are reported, showing a comparison between one of our proposals and an L–BFGS–based preconditioner for PNCG.

2. Preconditioned Nonlinear Conjugate Gradient algorithm

In this section we report the scheme of a general *Preconditioned Nonlinear Conjugate Gradient* (PNCG) *algorithm* (see e.g. [12]). In the PNCG scheme M_k denotes the preconditioner at the iteration k.

Preconditioned Nonlinear Conjugate Gradient (PNCG) algorithm

Step 1: Data $x_1 \in \mathbb{R}^n$. Set $p_1 = -M_1 \nabla f(x_1)$ and $k = 1$.

Step 2: Compute the steplength α_k by using a linesearch procedure which guarantees the Wolfe conditions to be satisfied, and set

$$x_{k+1} = x_k + \alpha_k p_k.$$

Step 3: If $\|\nabla f(x_{k+1})\| = 0$ then stop, else compute β_{k+1} and

$$p_{k+1} = -M_{k+1} \nabla f(x_{k+1}) + \beta_{k+1} p_k, \tag{2.1}$$

set $k = k + 1$ and go to *Step 2*.

By setting $M_k = I$ for any k, the popular (*unpreconditioned*) Nonlinear Conjugate Gradient (NCG) method is obtained. The parameter β_{k+1} can be chosen in a variety of ways. For PNCG algorithm the most recurrent choices are the following:

$$\beta_{k+1}^{\mathrm{FR}} = \frac{\nabla f(x_{k+1})^T M_k \nabla f(x_{k+1})}{\nabla f(x_k)^T \nabla f(x_k)}, \tag{2.2}$$

$$\beta_{k+1}^{\mathrm{PR}} = \frac{\left[\nabla f(x_{k+1}) - \nabla f(x_k)\right]^T M_k \nabla f(x_{k+1})}{\nabla f(x_k)^T M_k \nabla f(x_k)}, \tag{2.3}$$

$$\beta_{k+1}^{\mathrm{HS}} = \frac{\left[\nabla f(x_{k+1}) - \nabla f(x_k)\right]^T M_k \nabla f(x_{k+1})}{\left[\nabla f(x_{k+1}) - \nabla f(x_k)\right]^T p_k}. \tag{2.4}$$

We recall that with respect to other gradient methods, a more accurate line-search procedure is required to determine the steplength α_k in a PNCG algorithm. This is due to the presence of the term $\beta_{k+1} p_k$ in (2.1). The latter fact motivates the use of the (strong) Wolfe conditions to compute the steplength α_k, which also guarantee that $s_k^T y_k > 0$ for any k.

As already said, preconditioning is applied for increasing the efficiency of the NCG method. In this regard, we remark a noticeable difference between linear CG and NCG. Whenever the linear CG is applied, the Hessian matrix does not change during the iterations of the algorithm. On the contrary, when NCG is applied to a nonlinear function, the Hessian matrix (possibly indefinite) changes at each iteration.

3. A new Symmetric Rank-2 update

In this section we study a new quasi–Newton updating formula, by considering the properties of a parameter dependent symmetric rank-2 (SR2) update of the inverse of the Hessian matrix. Suppose we generate after k iterations the sequence of iterates $\{x_1, \ldots, x_{k+1}\}$. Then our quasi–Newton update H_{k+1}, which approximates $[\nabla^2 f(x)]^{-1}$, satisfies the secant equation along all previous directions; namely it results

$$H_{k+1} y_j = s_j, \qquad \text{for all} \qquad j \leq k.$$

Observe that the latter appealing property is satisfied by all the updates of the Broyden class, provided that the linesearch adopted is exact (see e.g. [11]). We would like to recover the motivation underlying the latter class of updates, and by using rank-2 updates we would like to define a preconditioner for PNCG.

On this guideline, in order to build an approximate inverse of the Hessian matrix, we consider the update

$$H(\gamma_{k+1},\omega_{k+1}) = H(\gamma_k,\omega_k) + \Delta_k, \qquad \Delta_k \in \mathbb{R}^{n\times n},\ \text{symmetric}, \tag{3.1}$$

where the sequence $\{H(\gamma_k,\omega_k)\}$ depends on the parameters γ_k, ω_k and provides our quasi-Newton updates of $[\nabla^2 f(x)]^{-1}$.

It is first our purpose to propose the new update $H(\gamma_{k+1},\omega_{k+1})$ such that:

(0) $H(\gamma_{k+1},\omega_{k+1})$ is well-defined and nonsingular

(1) $H(\gamma_{k+1},\omega_{k+1})$ can be iteratively updated

(2) $H(\gamma_{k+1},\omega_{k+1})$ collects the information from the iterations $1,2,\ldots,k$ of a NCG method

(3) $H(\gamma_{k+1},\omega_{k+1})$ satisfies the secant equation at iterations $j = 1,2,\ldots,k$

(4) $H(\gamma_{k+1},\omega_{k+1})$ either "tends to preserve" the inertia of the inverse of $\nabla^2 f(x_{k+1})$, in case $f(x)$ is a general quadratic function or, by suitably setting the two parameters, it can be used as a preconditioner for PNCG, i.e. $M_k = H(\gamma_k,\omega_k)$.

Observe that the Symmetric Rank-1 (SR1) quasi-Newton update (see Section 6.2 in [11]) satisfies properties (1)-(4) but not the property (0), i.e. it might be possibly not well–defined for a general nonlinear function. The latter result follows from the fact that SR1 update provides only a rank-1 quasi-Newton update, unlike BFGS and DFP. On the other hand, while BFGS and DFP quasi-Newton formulae provide only positive definite updates, the SR1 formula is able to recover the inertia of the Hessian matrix, by generating possibly indefinite updates. Thus, now we want to study an SR2 quasi-Newton update, which satisfies (0)–(4) and where one of the two newest dyads of the

update is provided by information from the NCG method. To this aim, assuming that $H_k = H(\gamma_k, \omega_k)$ is given, we consider the relation (3.1) where we set (see (1.4))

$$\Delta_k = \gamma_k v_k v_k^T + \omega_k \frac{p_k p_k^T}{y_k^T p_k}, \qquad \gamma_k, \omega_k \in \mathbb{R}, \qquad v_k \in \mathbb{R}^n,$$

and p_k is generated at the $k-$th iteration of the (unpreconditioned) NCG method. Thus, we will have the new update

$$H_{k+1} = H_k + \gamma_k v_k v_k^T + \omega_k \frac{p_k p_k^T}{y_k^T p_k}, \qquad \gamma_k, \omega_k \in \mathbb{R}, \qquad v_k \in \mathbb{R}^n, \tag{3.2}$$

and in order to satisfy the secant equation $H_{k+1} y_k = s_k$ the following equality must hold

$$H_k y_k + \gamma_k (v_k^T y_k) v_k + \omega_k \frac{p_k p_k^T}{y_k^T p_k} y_k = s_k,$$

that is

$$\gamma_k (v_k^T y_k) v_k = s_k - H_k y_k - \omega_k p_k. \tag{3.3}$$

Therefore it results

$$v_k = \sigma_k \left(s_k - H_k y_k - \omega_k p_k \right) \tag{3.4}$$

for some scalar $\sigma_k \in \mathbb{R}$. By substituting the expression (3.4) of v_k in (3.3) we have

$$\gamma_k \sigma_k^2 \left[y_k^T (s_k - H_k y_k - \omega_k p_k) \right] (s_k - H_k y_k - \omega_k p_k) = s_k - H_k y_k - \omega_k p_k.$$

Thus, the following relation among the parameters γ_k, σ_k and ω_k must hold

$$\gamma_k \sigma_k^2 = \frac{1}{s_k^T y_k - y_k^T H_k y_k - \omega_k p_k^T y_k}. \tag{3.5}$$

Note that from the arbitrariness of γ_k, without loss of generality, we can set $\sigma_k \in \{-1, 1\}$.

Now, in the next proposition we first consider the case of quadratic functions, and prove that the update (3.2) satisfies the secant equation, along all previous directions.

Proposition 3.1. *Assume that f is the quadratic function $f(x) = \frac{1}{2}x^T Ax + b^T x$, where $A \in \mathbb{R}^{n\times n}$ is symmetric and $b \in \mathbb{R}^n$. Suppose that k steps of the (unpreconditioned) CG are performed, in order to detect the stationary point (if any) of the function f, and that the vectors $p_1, \ldots, p_k$ are generated. Then, the matrix H_{k+1} in (3.2) satisfies the secant equations*

$$H_{k+1}y_j = s_j, \qquad j = 1, \ldots, k, \tag{3.6}$$

provided that the coefficients γ_j, ω_j, $j = 1, \ldots, k$ are computed such that

$$\begin{aligned} \gamma_j &= \frac{1}{s_j^T y_j - y_j^T H_j y_j - \omega_j p_j^T y_j}, \qquad j = 1, \ldots, k, \\ \omega_j &\neq \frac{s_j^T y_j - y_j^T H_j y_j}{p_j^T y_j}, \qquad j = 1, \ldots, k. \end{aligned} \tag{3.7}$$

Proof – The proof proceeds by induction. Equations (3.6) hold for $k = 1$, that is $H_2 y_1 = s_1$, as long as

$$s_1 = \left[H_1 + \gamma_1\sigma_1^2(s_1 - H_1y_1 - \omega_1 p_1)(s_1 - H_1y_1 - \omega_1 p_1)^T + \omega_1 \frac{p_1 p_1^T}{y_1^T p_1}\right] y_1,$$

or equivalently

$$s_1 - H_1y_1 - \omega_1 p_1 = \gamma_1(s_1^T y_1 - y_1^T H_1 y_1 - \omega_1 p_1^T y_1)\left[s_1 - H_1y_1 - \omega_1 p_1\right],$$

which is satisfied selecting γ_1 and ω_1 according with (3.7).
Now, suppose that the relations (3.6) hold for the index $k-1$. To complete the induction we need to prove that the relations (3.6) hold for the index k. Firstly, note that $H_{k+1}y_k = s_k$ holds. In fact

$$s_k = \left[H_k + \gamma_k\sigma_k^2(s_k - H_ky_k - \omega_k p_k)(s_k - H_ky_k - \omega_k p_k)^T + \omega_k \frac{p_k p_k^T}{y_k^T p_k}\right] y_k$$

holds if and only if

$$s_k - H_ky_k - \omega_k p_k = \gamma_k(s_k^T y_k - y_k^T H_k y_k - \omega_k p_k^T y_k)(s_k - H_ky_k - \omega_k p_k),$$

and the latter holds from (3.7) with $j = k$. Now, we have to prove that (3.6) hold for any $j < k$. For $j < k$ we have

$$H_{k+1}y_j = H_ky_j + \gamma_k\sigma_k^2(s_k - H_ky_k - \omega_k p_k)(s_k - H_ky_k - \omega_k p_k)^T y_j + \omega_k \frac{p_k^T y_j}{y_k^T p_k} p_k,$$

where $H_k y_j = s_j$ by the inductive hypothesis. Moreover,

$$(s_k - H_k y_k)^T y_j = s_k^T y_j - y_k^T H_k y_j = s_k^T y_j - y_k^T s_j = s_k^T A s_j - (A s_k)^T s_j = 0,$$

where the third equality holds since $y_j = A s_j$, for any j, for the quadratic function f. Finally,

$$\omega_k p_k^T y_j = \omega_k p_k^T A s_j = \omega_k \alpha_j p_k^T A p_j = 0,$$

which follows from the conjugacy of the directions $\{p_1, \ldots, p_k\}$ generated by the CG. Thus, (3.6) hold for any $j \leq k$ and the induction is complete. □

As an immediate consequence of the previous proposition, we prove now the finite termination property for a quadratic function, i.e. after at most n steps, H_{n+1} is the inverse of the Hessian of the quadratic function.

Corollary 3.1. *Assume that f is the quadratic function $f(x) = \frac{1}{2} x^T A x + b^T x$, where $A \in \mathbb{R}^{n \times n}$ is symmetric and $b \in \mathbb{R}^n$. Suppose that n steps of the (unpreconditioned) CG are performed, in order to detect the stationary point of the function f, and that the vectors $p_1, \ldots, p_n$ are generated. If (3.7) holds, we have $H_{n+1} = A^{-1}$.*

Proof – By applying Proposition 3.1, we have that (3.6) hold for $k = n$, i.e.

$$H_{n+1} y_j = s_j, \qquad j = 1, \ldots, n.$$

Since f is quadratic then $y_j = A s_j$, for any j, i.e.

$$H_{n+1} A s_j = s_j, \qquad j = 1, \ldots, n.$$

Now, since $s_j = \alpha_j p_j$, $j = 1, \ldots, n$, the conjugacy of the vectors $\{p_1, \ldots, p_n\}$ implies that $H_{n+1} = A^{-1}$. □

We highlight that, whenever $k = n$, Corollary 3.1 justifies the first part of the statement (4) on page 8. Moreover, later on in the paper we show that for $k < n$, the update matrix in (3.2) can be suitably modified to provide a preconditioner.

After analyzing the case of $f(x)$ quadratic, we turn now to the general case of a nonlinear twice continuously differentiable function. In particular, since we are interested in using the matrix H_{k+1} in (3.2) as a preconditioner, we need to investigate if there exists a suitable setting of the parameters such that H_{k+1} is positive definite, provided that (3.7) are satisfied. In the next proposition we prove that if the parameter ω_k is below a threshold value, then the matrix H_{k+1} is *almost always* positive definite.

Proposition 3.2. *Let f be a nonlinear twice continuously differentiable function. Suppose that the (unpreconditioned) NCG method is used to minimize the function f. Suppose that* (3.7) *is satisfied and*

$$0 \le \omega_k < \frac{s_k^T y_k - y_k^T H_k y_k}{p_k^T y_k}, \tag{3.8}$$

with

$$y_k^T s_k + y_k^T H_k y_k \le 0 \qquad or \qquad y_k^T s_k - y_k^T H_k y_k \ge 0, \tag{3.9}$$

where $s_j = \alpha_j p_j$. Then the matrix H_{k+1} in (3.2) *is positive definite.*

PROOF – By substituting (3.4) in (3.2), recalling that $\sigma_k^2 = 1$ we obtain

$$\begin{aligned} H_{k+1} = \quad & \gamma_k \left[(\alpha_k - \omega_k)^2 p_k p_k^T + (\alpha_k - \omega_k) \left((H_k y_k) p_k^T + p_k (H_k y_k)^T \right) \right. \\ & \left. + (H_k y_k) (H_k y_k)^T \right] + \omega_k \frac{p_k p_k^T}{y_k^T p_k}. \end{aligned}$$

Hence H_{k+1} can be rewritten in the form

$$\begin{pmatrix} p_k & \vdots & H_k y_k \end{pmatrix} \begin{pmatrix} \gamma_k(\alpha_k - \omega_k)^2 + \dfrac{\omega_k}{y_k^T p_k} & & \gamma_k(\alpha_k - \omega_k) \\ & & \\ \gamma_k(\alpha_k - \omega_k) & & \gamma_k \end{pmatrix} \begin{pmatrix} p_k^T \\ \\ \dots \\ \\ (H_k y_k)^T \end{pmatrix}.$$

Therefore H_{k+1} is positive definite if and only if the following inequalities hold:

$$\begin{aligned} \gamma_k(\alpha_k - \omega_k)^2 + \frac{\omega_k}{y_k^T p_k} &> 0 \\ \gamma_k \left(\gamma_k(\alpha_k - \omega_k)^2 + \frac{\omega_k}{y_k^T p_k} \right) - \gamma^2(\alpha_k - \omega_k)^2 &> 0. \end{aligned} \tag{3.10}$$

Using the expression of γ_k in (3.5) and recalling that $y_k^T s_k > 0$ (as a consequence of the Wolfe conditions), (3.10) are equivalent to

$$\begin{aligned} \frac{(\alpha_k - \omega_k)^2 y_k^T p_k}{(\alpha_k - \omega_k) p_k^T y_k - y_k^T H_k y_k} + \omega_k &> 0 \\ \frac{\omega_k}{(\alpha_k - \omega_k) p_k^T y_k - y_k^T H_k y_k} &> 0. \end{aligned}$$

After some computation we obtain that there exist values of the parameter ω_k for which the latter inequalities admit solutions, with only one exception. In fact, they are satisfied for any value of ω_k such that

$$0 \leq \omega_k < \frac{\alpha_k p_k^T y_k - y_k^T H_k y_k}{p_k^T y_k}$$

but they do not admit solution in case

$$\alpha_k y_k^T p_k + y_k^T H_k y_k > 0 \qquad \text{and} \qquad \alpha_k y_k^T p_k - y_k^T H_k y_k < 0,$$

i.e. when (3.9) does not hold. □

From Proposition 3.1 and Corollary 3.1, we could use the matrix H_{k+1} as an approximate inverse of $\nabla^2 f(x)$. However, Proposition 3.2 evidences that conditions (3.7) and (3.8) do not suffice to ensure H_{k+1} positive definite. In fact, whenever (3.9) occurs, additional safeguard is needed since H_{k+1} is possibly indefinite. Thus, the definition of H_{k+1} should be possibly modified in order to obtain positive definite updates.

4. A preconditioner using a BFGS–like low–rank quasi-Newton update

In this section we partially address the final remark of Section 3. Indeed, we introduce a new class of preconditioners which are still iteratively constructed by using information from the NCG iterations and, as in the case of BFGS updates, they are always positive definite. On this purpose, the price we pay with respect to (3.2), is that the secant equation is satisfied only at the current iterate, and not necessarily along all the previous iterates.

We draw our inspiration from [4], where a new preconditioner for Newton–Krylov methods is described. In particular, in [4] the set of directions generated by the Krylov subspace method is used to provide an approximate inverse preconditioner, for the solution of Newton's systems. On this guideline, observe that if $f(x) = \frac{1}{2}x^T Ax + b^T x$, where A is positive definite and $b \in \mathbb{R}^n$, then it is well known (see e.g. [6]) that the CG method may generate n conjugate directions $\{p_j\}$ such that

$$A^{-1} = \sum_{j=1}^{n} \frac{p_j p_j^T}{p_j^T A p_j}. \tag{4.1}$$

Now, in order to introduce a class of preconditioners for the NCG, in case of a general twice continuosly differentiable function f, suppose we have performed k iterations of the (unpreconditioned) NCG, so that the directions $p_1, \ldots, p_k$ are generated. Let us consider the matrix M_{k+1} defined by

$$M_{k+1} = \tau_k C_k + \gamma_k v_k v_k^T + \omega_k \sum_{j=k-m}^{k} \frac{p_j p_j^T}{p_j^T \nabla^2 f(x_j) p_j}, \tag{4.2}$$

where $0 \leq m \leq k$, $\gamma_k, \omega_k \geq 0$, $\tau_k > 0$, $C_k \in \mathbb{R}^{n \times n}$ is symmetric positive definite and $v_k \in \mathbb{R}^n$. In order to use M_{k+1} as a preconditioner and to update its expression iteratively, we set $\tau_k = 1$, $C_k = H(\tau_k, \gamma_k, \omega_k)$ (with $H(\tau_0, \gamma_0, \omega_0)$ given) and rewrite (4.2) in the form

$$H(\tau_{k+1}, \gamma_{k+1}, \omega_{k+1}) = H(\tau_k, \gamma_k, \omega_k) + \gamma_k v_k v_k^T + \omega_k \sum_{j=k-m}^{k} \frac{p_j p_j^T}{p_j^T \nabla^2 f(x_j) p_j}. \tag{4.3}$$

$H(\tau_{k+1}, \gamma_{k+1}, \omega_{k+1})$ may be treated as a symmetric quasi–Newton update. However, for simplicity, in the sequel we prefer to use the more general form given by (4.2).

Observe that in the expression of M_{k+1}, $v_k v_k^T$ represents a rank-1 update and from (4.1) the dyads $p_j p_j^T / p_j^T \nabla^2 f(x_j) p_j$ are aimed to build an approximate inverse. The integer m can be viewed as a "limited memory" parameter, similarly to the L–BFGS method. Moreover, we can set the vector v_k and the parameters $\tau_k, \gamma_k, \omega_k$ such that the class of preconditioners M_k satisfies, for

any k, the secant equation

$$M_{k+1} y_k = s_k. \tag{4.4}$$

Indeed, from (4.4) we have

$$\tau_k C_k y_k + \gamma_k (v_k^T y_k) v_k + \omega_k \sum_{j=k-m}^{k} \frac{p_j^T y_k}{p_j^T \nabla^2 f(x_j) p_j} p_j = s_k;$$

hence, assuming $\gamma_k (v_k^T y_k) \neq 0$,

$$v_k = \sigma_k \left[s_k - \tau_k C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{p_j^T y_k}{p_j^T \nabla^2 f(x_j) p_j} p_j \right], \tag{4.5}$$

for some $\sigma_k \in \mathbb{R}$. Using (4.5) in (4.4) we have

$$\gamma_k \sigma_k^2 \left[s_k^T y_k - \tau_k y_k^T C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{(p_j^T y_k)^2}{p_j^T \nabla^2 f(x_j) p_j} \right] \cdot \left[s_k - \tau_k C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{p_j^T y_k}{p_j^T \nabla^2 f(x_j) p_j} p_j \right] =$$
$$s_k - \tau_k C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{p_j^T y_k}{p_j^T \nabla^2 f(x_j) p_j} p_j.$$

Thus, the following relation among the parameters $\gamma_k, \sigma_k, \tau_k$ and ω_k has to be satisfied

$$\gamma_k \sigma_k^2 = \frac{1}{-\tau_k y_k^T C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{(p_j^T y_k)^2}{p_j^T \nabla^2 f(x_j) p_j} + s_k^T y_k} \tag{4.6}$$

and without loss of generality we can set $\sigma_k \in \{+1, -1\}$. Then, observe that unlike the update proposed in the previous section (namely (3.2)), the matrix M_{k+1} in (4.4) satisfies the secant equation only at the k-th iteration (even for quadratic functions), and possibly not along all the previous iterations, as proved in Proposition 3.1 for the update (3.2). As regards the positive

definiteness of M_{k+1}, the Wolfe conditions used in the linesearch procedure for computing the steplength α_k ensure that $s_k^T y_k > 0$, so that for $\tau_k > 0$ and $\omega_k \geq 0$ sufficiently small in (4.6) the matrix M_{k+1} is positive definite. Indeed, suppose that $\omega_k \to 0$, then $M_{k+1} \approx \tau_k C_k + \gamma_k v_k v_k^T$. Now, since $\tau_k > 0$ and C_k is positive definite, by (4.6) for τ_k sufficiently small we have $\gamma_k > 0$, i.e. we definitely have that M_{k+1} is positive definite.

Finally, observe that the different choices for the parameters τ_k and ω_k in (4.6) provide a different scaling of the matrices C_k and $\displaystyle\sum_{j=k-m}^{k} \frac{p_j p_j^T}{p_j^T \nabla^2 f(x_j) p_j}$, in the preconditioners.

Now we note that the quantities $p_j^T \nabla^2 f(x_j) p_j$, $j = 1, \ldots, k$, in the expression (4.2) of M_{k+1} are in general unavailable. By considering that the Hessian matrix is not constant at the points in the closed segment $[x_j, x_{j+1}]$, then we can use the Mean Value Theorem to estimate the average curvature of f along the direction p_j, that is

$$\int_0^1 s_j^T \nabla^2 f[x_j + \beta(x_{j+1} - x_j)] s_j \; d\beta = s_j^T y_j$$

and recalling that $s_j = \alpha_j p_j$, we can estimate the quantity $p_j^T \nabla^2 f(x_j) p_j$, in the expression of M_{k+1}, by

$$p_j^T \nabla^2 f(x_j) p_j \approx \int_0^1 p_j^T \nabla^2 f[x_j + \beta(x_{j+1} - x_j)] p_j \; d\beta = \frac{s_j^T y_j}{\alpha_j^2} = \frac{p_j^T y_j}{\alpha_j}.$$

Observe that by the Wolfe conditions used in the linesearch procedure, the latter quantity satisfies the condition

$$\frac{p_j^T y_j}{\alpha_j} > 0.$$

Moreover, in case f is the quadratic function $f(x) = \frac{1}{2} x^T A x + b^T x$ then

$$\int_0^1 p_j^T \nabla^2 f[x_j + \beta(x_{j+1} - x_j)] p_j \; d\beta = p_j^T A p_j, \tag{4.7}$$

i.e. the left hand side of (4.7) may be regarded as a generalization (to the general nonlinear case) of the quantity $p_j^T \nabla^2 f(x_j) p_j$.

As regards the matrix C_k in (4.2), an obvious choice could be for any k

$$C_k = \varepsilon_k I, \qquad \varepsilon_k \in \mathbb{R}.$$

Furthermore, ε_k may be computed as the least squares solution of the equation $(\varepsilon I)y_k - s_k = 0$, i.e. ε_k solves

$$\min_{\varepsilon} \|(\varepsilon I)y_k - s_k\|^2 .$$

Hence,

$$\varepsilon_k = \frac{s_k^T y_k}{\|y_k\|^2}$$

so that since $s_k^T y_k > 0$ by the Wolfe conditions, the matrix

$$C_k = \frac{s_k^T y_k}{\|y_k\|^2} I$$

is positive definite.

For the sake of clarity we report here the resulting expression of our class of preconditioners (4.2):

$$M_{k+1} = \tau_k \frac{s_k^T y_k}{\|y_k\|^2} I + \gamma_k v_k v_k^T + \omega_k \sum_{j=k-m}^{k} \frac{s_j s_j^T}{y_j^T s_j}, \tag{4.8}$$

where

$$v_k = \sigma_k \left[s_k - \tau_k \frac{s_k^T y_k}{\|y_k\|^2} y_k - \omega_k \sum_{j=k-m}^{k} \frac{s_j^T y_k}{y_j^T s_j} s_j \right], \qquad \sigma_k \in \{-1, 1\},$$

and

$$\gamma_k \sigma_k^2 = \frac{1}{(1-\tau_k) s_k^T y_k - \omega_k \sum_{j=k-m}^{k} \frac{(s_j^T y_k)^2}{y_j^T s_j}}.$$

We conclude this section by highlighting that, interestingly enough, similarly to (4.3) we can construct a class of preconditioners based on DFP-like quasi-Newton updates. Indeed, we can iteratively build matrices

$$B(\tau_{k+1}, \gamma_{k+1}, \omega_{k+1})$$

approximating $\nabla^2 f(x)$ and not its inverse. Then, by the Sherman-Morrison-Woodbury formula applied to $B(\tau_{k+1}, \gamma_{k+1}, \omega_{k+1})$ we can compute a class of preconditioners.

5. Preliminary numerical experiences

In order to investigate the reliability of the classes of preconditioners we have introduced, we preliminarily performed a numerical testing for the use of the preconditioners defined in (4.8). This choice is motivated by the fact that for this class of preconditioners we can easily guarantee the positive definitiveness, whereas in case of the class of preconditioners given by (3.2) an alternative strategy must be proposed to guarantee the positive definiteness.

Therefore, we embedded the preconditioners (4.8) within the standard CG+ code [5]. We used the same linesearch and the same stopping criterion used by default in CG+ code. Thus we refer to [5] for a complete description of all the details. We tested both the Fletcher and Reeves (FR) and the Polak and Ribiere (PR) versions of the PNCG method at page 6.

As regards the test problems, we selected all the large scale unconstrained test problems in the CUTEr collection [7]. The dimension of the test problems is between $n = 1000$ and $n = 10000$ (we considered 110 resulting problems). The parameters of the preconditioner (4.8) have been chosen as follows: $m = 4$, $\sigma_k = 1$ and

$$\tau_k = \omega_k = \frac{\frac{1}{2} s_k^T y_k}{y_k^T C_k y_k + \sum_{j=k-m}^{k} \frac{(p_j^T y_k)^2}{p_j^T \nabla^2 f(x_j) p_j}}$$

for all k (this choice ensures that the denominator of (4.6) is equal to $\frac{1}{2} s_k^T y_k > 0$). As preliminary investigation, we considered the results in terms of the number of iterations and the number of function evaluations. We compared the results obtained by (4.8), the unpreconditioned case, and the case where M_k coincides with the L–BFGS update H_{k+1} in (1.3). This comparison is reported by using performance profiles [3]. For a fair comparison, we have excluded in each profile all the test problems where the three algorithms converge to different stationary points.

In particular, as regards the FR version, in Figure 1 we report the comparison among the three algorithms in terms of number of iterations. Figure 2

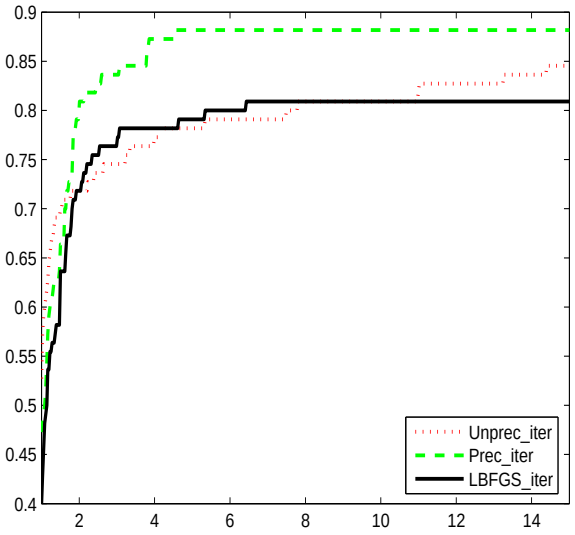

Figure 1: Comparison of the FR algorithms in terms of number of iterations

reports the same plot with a different scale. In Figures 3 and 4 the comparison among the three algorithms is reported in terms of number of function evaluations. These profiles show that using the FR algorithm, the preconditioner (4.8) tends to be preferable, both in terms of number of iterations and number of function evaluations.

Now we turn to the PR version of the PNCG algorithm and, in Figure 5 we report the comparison among (4.8), the unpreconditioned algorithm and the L–BFGS based preconditioner in terms of number of iterations. Figure 6 reports the same plot with a different scale.

In Figures 7 and 8 the comparison among the three algorithms is reported in terms of number of function evaluations.

From the observation of these plots it is easy to ascertain that the situation

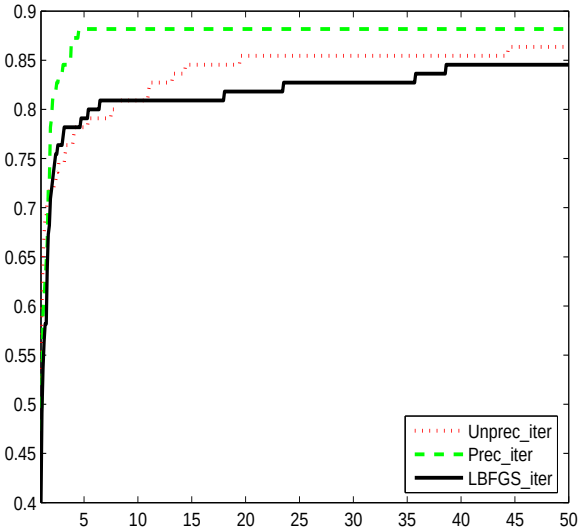

Figure 2: Comparison of the FR algorithms in terms of number of iterations (expanded)

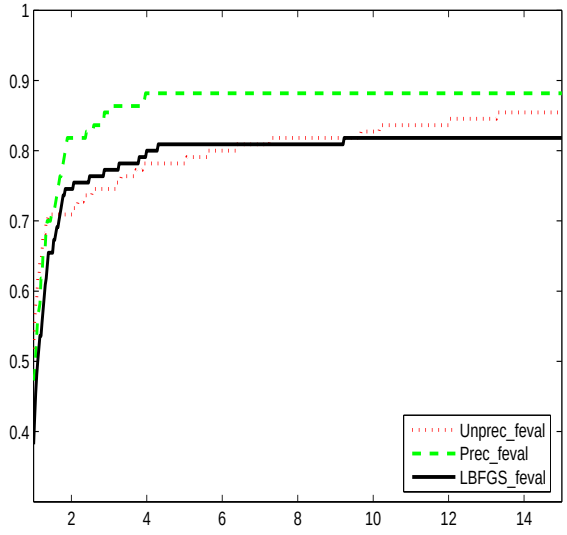

Figure 3: Comparison of the FR algorithms in terms of number of function evaluations

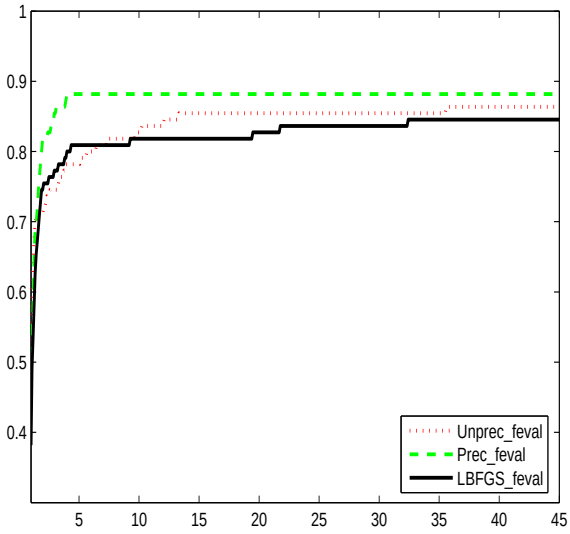

Figure 4: Comparison of the FR algorithms in terms of number of function evaluations (expanded)

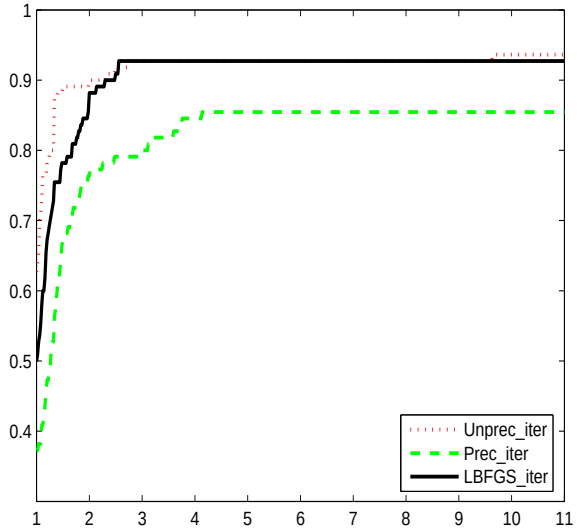

Figure 5: Comparison of the PR algorithms in terms of number of iterations

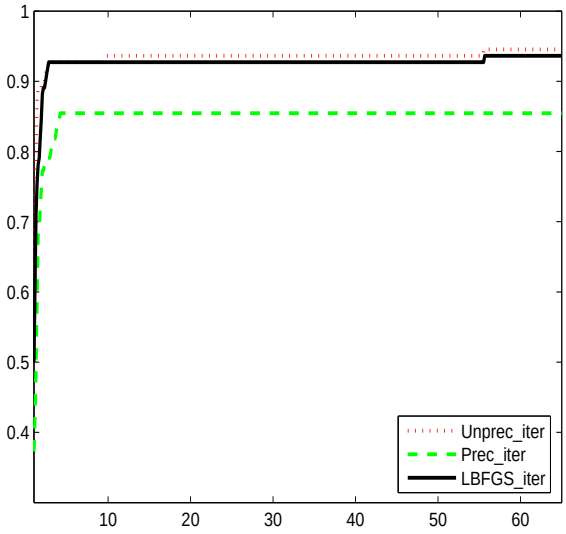

Figure 6: Comparison of the PR algorithms in terms of number of iterations (expanded)

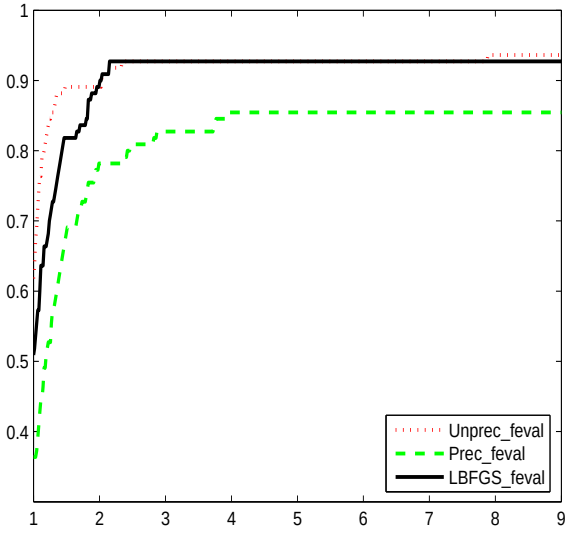

Figure 7: Comparison of the PR algorithms in terms of number of function evaluations

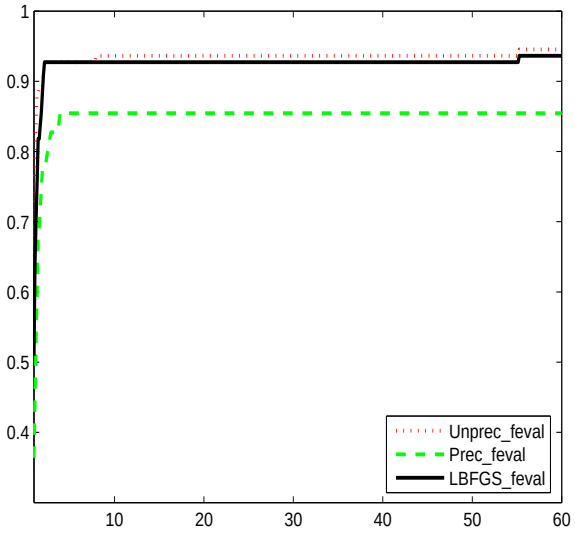

Figure 8: Comparison of the PR algorithms in terms of number of function evaluations (expanded)

is reversed with respect to the FR version of the algorithms.

On the overall, even if these preliminary results do not allow us to draw final conclusions, they show that the preconditioning strategies proposed may be reliable and in same cases they are beneficial. In particular, we observe that our proposals are cheaper than the L–BFGS based preconditioner. However, observing the case of PR setting, since in (4.2) we convey only informations from the current iterate, we guess that a more sophisticated choice of the matrix $\tau_k C_k$ is definitely needed, in order to preserve efficiency.

6. Conclusions and future works

In this paper we propose two new classes of quasi–Newton update, aiming at using the update matrix as preconditioner within NCG method. In the first proposal the satisfaction of the secant equations at each previous iteration is ensured (in the quadratic case), but we can not ensure, in general, that the resulting update is positive definite. In the latter cases, an alternative strategy is needed.

In the second proposal the satisfaction of the secant equation only at the current iteration is ensured but the resulting update is guaranteed to be positive definite. We numerically tested the latter approach both with the unconstrained case and L–BFGS based preconditioning approach. The results obtained, thought preliminary, showed that it may be promising in some cases, even if non–carefully selected settings of the parameters are chosen.

Acknowledgements

The authors wish to thank Marco D'Apuzzo, who inspired these afternotes, for his cheerful attitude to life, which greatly highlighted and completed his professionalism.

References

[1] N. Andrei. Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization. *Optimization Methods and Software*, 22:561–571, 2007.

[2] B. Buckley and A. Lenir. QN–like variable storage conjugate gradients. *Mathematical Programming*, 27:155–175, 1983.

[3] E. D. Dolan and J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[4] G. Fasano and M. Roma. Preconditioning Newton–Krylov methods in nonconvex large scale optimization. Submitted to *Computational Optimization and Applications*.

[5] J.C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, 2:21–42, 1992.

[6] G.H. Golub and C.F. Van Loan. *Matrix Computations*. The John Hopkins Press, Baltimore, 1996. Third edition.

[7] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr (and sifdec), a constrained and unconstrained testing environment, revised. *ACM Transaction on Mathematical Software*, 29:373–394, 2003.

[8] W. Hager and H. Zhang. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, 2:35–58, 2006.

[9] J.L. Morales and J. Nocedal. Automatic preconditioning by limited memory quasi–Newton updating. *SIAM Journal on Optimization*, 10:1079–1096, 2000.

[10] L. Nazareth. A relationship between the BFGS and conjugate gradient algorithms and its implications for new algorithms. *SIAM Journal on Numerical Analysis*, 16:794–800, 1979.

[11] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006. Second edition.

[12] R. Pytlak. *Conjugate Gradient Algorithms in Nonconvex Optimization*. Springer, Berlin, 2009.

[13] D.F. Shanno. Conjugate gradient methods with inexact searches. *Mathematics of Operations Research*, 3:244–256, 1978.