Numer. Math. (2003) 94: 479–500
Digital Object Identifier (DOI)10.1007/s00211-002-0410-4

Numerische Mathematik

Matrix algebras in Quasi-Newton methods for unconstrained minimization

Carmine Di Fiore[1], **Stefano Fanelli**[1], **Filomena Lepore**[1], **Paolo Zellini**[1,2]

[1] Dipartimento di Matematica, Università di Roma "Tor Vergata", Via della Ricerca Scientifica, 00133 Roma, Italy
[2] e-mail: zellini@mat.uniroma2.it

Received December 30, 2001 / Revised version received December 20, 2001 /
Published online November 27, 2002 –

Summary. In this paper a new class of quasi-Newton methods, named $\mathcal{L}QN$, is introduced in order to solve unconstrained minimization problems. The novel approach, which generalizes classical $BFGS$ methods, is based on a Hessian updating formula involving an algebra $\mathcal{L}$ of matrices simultaneously diagonalized by a fast unitary transform. The complexity per step of $\mathcal{L}QN$ methods is $O(n \log n)$, thereby improving considerably $BFGS$ computational efficiency. Moreover, since $\mathcal{L}QN$'s iterative scheme utilizes single-indexed arrays, only $O(n)$ memory allocations are required. Global convergence properties are investigated. In particular a global convergence result is obtained under suitable assumptions on f. Numerical experiences [7] confirm that $\mathcal{L}QN$ methods are particularly recommended for large scale problems.

Mathematics Subject Classification (1991): 65K10

1 Introduction

In this paper some new algorithms for the unconstrained minimization of an arbitrary function $f : \mathbb{R}^n \to \mathbb{R}$ are investigated. The novel methods are based on a generalized $BFGS$-type iterative scheme and are particularly efficient when n is large.

In the class of variable metric algorithms, which originated in the work of Davidon [14], the method of Broyden, Fletcher, Goldfarb and Shanno ($BFGS$) is considered to be one of the most efficient [33]. Under suitable

Correspondence to: P. Zellini

conditions on f, the $BFGS$ method is globally convergent [38] and has a local superlinear rate of convergence [11], [15], [38]. Moreover, $BFGS$ is competitive even with a modified Newton-Raphson-type method, since each iteration can be performed with only $O(n^2)$ flops and no Hessian evaluation is required. In fact, the matrix B_{k+1}, replacing the Hessian $\nabla^2 f(\mathbf{x}_{k+1})$ in the $BFGS$ Quasi-Newton (QN) formula, is a rank-2 perturbation of the previous positive definite (pd) Hessian approximation B_k, defined in terms of the two current difference vectors $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$, satisfying the following iterative equation

$$B_{k+1}\mathbf{s}_k = \mathbf{y}_k. \tag{1.1}$$

A variant of $BFGS$ method, named $M{-}BFGS$ (see [45], [46]) has been recently introduced by replacing in (1.1) the right hand side $\mathbf{y}_k$ with a different vector $\tilde{\mathbf{y}}_k$ defined also in terms of the function values $f(\mathbf{x}_k)$ and $f(\mathbf{x}_{k+1})$.

However, for large scale optimization problems, conjugate gradient or "conjugate gradient-type" algorithms are in general preferred to $BFGS$ since they have a lower complexity per step and require a lower amount of high speed storage on a computer. Among the latter algorithms, the $L\text{-}BFGS$ (Limited memory $BFGS$) methods [33], [34] have been studied extensively. The $L\text{-}BFGS$ algorithms update continuously a Hessian approximation by using the most recent second order information available in the form of the vectors $\mathbf{s}_j$, $\mathbf{y}_j$, $j = k - m + 1, \ldots, k$. The rate of convergence of $L\text{-}BFGS$ methods can be improved if more information (corresponding to a larger m) is exploited and/or if the matrices updated are suitably chosen [1], [31], [33], [34], [47].

Other minimization algorithms particularly efficient when n is large have been considered in [2], [23], [32], [39]. The method introduced by Shanno [39], having $O(n)$ complexity per step, is a simple memory-less modification of the $BFGS$ method, in which the identity matrix I is used, instead of B_k, to compute the new approximation B_{k+1}. In [2] and [23] it is shown that suitable variants of this method, named OSS and $OSSV$, can be extremely competitive with the original $BFGS$ method to perform optimal learning in a large Multi-Layer Perceptron (MLP) network. Unfortunately, by the very nature of the memory-less approach, the amount of second order information contained in $OSS\text{-}OSSV$ is considerably reduced in comparison with the standard $BFGS$ method and thus the number of iterations required for the desired approximation must be encreased. In [32] iterative schemes updating the diagonal or the block-diagonal part of B_k were introduced to improve the total efficiency.

The main problem connected to the calculation of the Hessian approximation B_{k+1} is in general to minimize the computational complexity per iteration, by maintaining a QN rate of convergence.

In this paper we introduce the following generalized iterative scheme, where B_{k+1} is defined by utilizing an arbitrary matrix $\tilde{B}_k$ instead of B_k ($BFGS$) or I (OSS-$OSSV$):

$$B_{k+1} = \tilde{B}_k + \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \mathbf{y}_k \mathbf{y}_k^T - \frac{1}{\mathbf{s}_k^T \tilde{B}_k \mathbf{s}_k} \tilde{B}_k \mathbf{s}_k \mathbf{s}_k^T \tilde{B}_k. \tag{1.2}$$

This scheme leads to two main classes of algorithms named *secant* ($\mathcal{S}$) and *nonsecant* ($\mathcal{NS}$), respectively. In principle the matrix $\tilde{B}_k$ should retain as many information on B_k as possible, in order to maintain a "$BFGS$-type" local convergence behaviour. Clearly, $\tilde{B}_k$ should be at least *pd* like B_k. On the other hand, the choice of a "simple" matrix $\tilde{B}_k$ is recommended to minimize the complexity per step.

The above properties are essentially obtained in the $\mathcal{L}QN$ *methods*, which represent a new class of QN *algorithms with memory* involving suitable approximations of the *whole Hessian matrix*. More precisely, in $\mathcal{L}QN$ methods $\tilde{B}_k = \mathcal{L}_{B_k}$ where $\mathcal{L}_{B_k}$ is the best least-squares fit (in the Frobenius norm) of B_k by an algebra $\mathcal{L}$ of matrices simultaneously diagonalized by a fast unitary transform (FFT, Hartley-type [9], [17], [22], [6] or trigonometric [43]).

The most interesting property of $\mathcal{L}QN$ methods depends upon the fact that they require $O(n \log n)$ flops per step (i.e. the same computational cost of the fast transform involved) and $O(n)$ memory allocations. The strong reduction of space complexity is obtained since all iterative formulas exploited in $\mathcal{L}QN$ methods involve single-indexed arrays only. This allows to solve minimization problems for large values of n, which are unsuitable or even prohibitive for the application of $BFGS$. Moreover, it is important to emphasize that a theoretical convergence result can be obtained for a subclass of $\mathcal{L}QN$ methods, in contrast to the heuristic convergence regarding OSS-$OSSV$ (see Sect. 5).

A fundamental property of $\mathcal{L}_{B_k}$, which yields descent directions in $\mathcal{L}QN$ methods, is that $\mathcal{L}_{B_k}$ *is pd whenever* B_k *is pd*. We point out that the latter property was already implemented in preconditioning techniques to reduce the number of steps of Conjugate Gradient (CG) methods for *pd* linear systems $B\mathbf{x} = \mathbf{v}$ [4], [12], [13], [22], [28], [29], [36], [40]. Moreover, the class of algebras of $\mathcal{L}$-type, for which the above property holds, has been also used to define more efficient direct methods solving the system $B\mathbf{x} = \mathbf{v}$. By utilizing simple displacement decompositions of B^{-1} in terms of matrices of $\mathcal{L}$-type, it is, in fact, possible to compute $B^{-1}\mathbf{v}$ via a finite number of fast unitary transforms [5], [8], [17], [21], [30].

Convergence properties of the novel minimization methods are investigated: it is shown that under suitable assumptions on $\tilde{B}_k$ some global convergence results known for the $BFGS$ method (due to Powell [38]), hold unchanged for the general $\mathcal{S}$ and $\mathcal{NS}$ algorithms. In particular, the same

criterion of Powell in constructing $\mathbf{x}_k$ such that a subsequence of $\{\nabla f(\mathbf{x}_k)\}$ converges to the null vector can be applied to the $BFGS$-type methods under suitable conditions on $\tilde{B}_k$. These conditions are satisfied by the $\mathcal{L}QN$ methods, where $\tilde{B}_k = \mathcal{L}_{B_k}$.

Work is in progress in order to obtain $\mathcal{L}QN$ convergence theorems under less restrictive assumptions on f. In particular, a sort of "weak form of discrete convexity" might be sufficient to guarantee the convergence of the $\mathcal{NS}$ $\mathcal{L}QN$ approximating sequence (see Sect. 5 and [38]). Recall that some analogous conditions, implying the convergence of the sequence generated by a minimization algorithm, are considered in [3].

Numerical experiences, reported in [7], show the particular efficiency of the new methods in training MLP-networks where, in most operational cases, n is in the range of at least several hundred [26]. More precisely, by using the $\mathcal{S}$ $\mathcal{L}QN$ method, an improvement factor between $2/3$ and $1/3$, with respect to OSS–$OSSV$ methods, is obtained in terms of CPU time. In this context, the $\mathcal{L}QN$ algorithms turn out to be competitive also with L–$BFGS$ [7]. Moreover, the preliminary experiences performed on general type large–scale unconstrained minimization problems confirm a major efficiency of $\mathcal{S}$ $\mathcal{L}QN$ in comparison to L–$BFGS$.

All these theoretical-experimental results show the strong competitivity of $\mathcal{L}QN$ methods in solving large scale minimization problems with respect to the most efficient methods available.

2 A generalized $BFGS$-type iterative scheme

In this section we wish to introduce a generalized $BFGS$–type scheme as the most suitable framework for a new class of quasi–Newtonian methods of low complexity, the $\mathcal{L}QN$ algorithms. In the $\mathcal{L}QN$ methods, described in detail in Sects. 4–6, a structure is injected in the classical updating $BFGS$ formula, by picking up suitable approximations of the Hessian from an algebra of matrices diagonalized by a fast transform.

Let $f : \mathbb{R}^n \to \mathbb{R}$ and consider the minimum problem

$$\text{find } \mathbf{x}_* \text{ such that } f(\mathbf{x}_*) = \min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})\,. \tag{2.1}$$

Denote by $\mathbf{g}(\mathbf{x})$ and by $G(\mathbf{x})$, respectively, the gradient vector and the Hessian matrix of f in $\mathbf{x}$. A well known efficient iterative quasi–Newton method to solve (2.1) is the $BFGS$ method [16]. The $BFGS$ iterates $\{\mathbf{x}_k\}$ are defined as in the classical Newton method except that the Hessians are replaced with quasi–Hessians obtained by the updating formula

$$\Phi(B, \mathbf{s}, \mathbf{y}) = B + \frac{1}{\mathbf{y}^T\mathbf{s}}\mathbf{y}\mathbf{y}^T - \frac{1}{\mathbf{s}^T B\mathbf{s}} B\mathbf{s}\mathbf{s}^T B, \tag{2.2}$$

$B \in \mathbb{R}^{n\times n}$, $\mathbf{s}, \mathbf{y} \in \mathbb{R}^n$. More precisely, given an approximation $\mathbf{x}_k$ of $\mathbf{x}_*$ and a real symmetric positive definite (*pd*) approximation B_k of $G(\mathbf{x}_*)$, the next $BFGS$ approximations $\mathbf{x}_{k+1}$ and B_{k+1} are defined by the identities $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k$ and

$$B_{k+1} = \Phi(B_k, \mathbf{s}_k, \mathbf{y}_k), \tag{2.3}$$

where $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, $\mathbf{g}_k = \mathbf{g}(\mathbf{x}_k)$. The vector $\mathbf{d}_k$, the search direction, is chosen to minimize the local positive definite quadratic model $f(\mathbf{x}_k) + \mathbf{g}_k^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T B_k \mathbf{d}$ of the function $f(\mathbf{x}_k + \mathbf{d})$, $\mathbf{d} \in \mathbb{R}^n$, and the scalar $\lambda_k > 0$ is chosen to assure that:

(i) $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ and
(ii) the matrix B_{k+1} in (2.3), the $BFGS$ update of B_k, is positive definite.

Since the matrix B_{k+1} in (2.3) solves the *secant equation*

$$X\mathbf{s}_k = \mathbf{y}_k, \tag{2.4}$$

the method $BFGS$ belongs to the class of secant methods. Under suitable assumptions on f and $\mathbf{x}_0$ the $BFGS$ method has both global and local convergence properties. In particular, the choice $\lambda_k = 1$ for "large" values of k leads to a local superlinear rate of convergence of the sequence $\{\mathbf{x}_k\}$ to $\mathbf{x}_*$ (even in cases where the corresponding sequence $\{\|B_k - G(\mathbf{x}_*)\|\}$ remains bounded away from zero). This is essentially due to the fact that the $BFGS$ search direction, $-B_k^{-1}\mathbf{g}_k$, converges to the Newton direction $-G(\mathbf{x}_k)^{-1}\mathbf{g}_k$ as $k \to +\infty$ [10], [15], [16]. See Dennis–Schnabel [16, chapters 8 and 9] for an analysis of the properties of Φ and of the fast local convergence of $BFGS$.

The lower rate of convergence of $BFGS$ with respect to a modified Newton-Raphson method, is compensated by the fact that the computation of $\mathbf{d}_k$ in $BFGS$ does not require any Hessian evaluation and can be performed in only $O(n^2)$ flops. In fact, if H_k denotes the inverse of B_k, then $\mathbf{d}_{k+1} = -B_{k+1}^{-1}\mathbf{g}_{k+1} = -H_{k+1}\mathbf{g}_{k+1}$ can be computed by using the identity

$$H_{k+1} = \Psi(H_k, \mathbf{s}_k, \mathbf{y}_k) \tag{2.5}$$

where, for $H \in \mathbb{R}^{n\times n}$, $\mathbf{s}, \mathbf{y} \in \mathbb{R}^n$,

$$\Psi(H, \mathbf{s}, \mathbf{y}) = \left(I - \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{y}^T\mathbf{s}}\right)^T H \left(I - \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{y}^T\mathbf{s}}\right) + \frac{\mathbf{s}\mathbf{s}^T}{\mathbf{y}^T\mathbf{s}} \tag{2.6}$$

which follows by the Sherman-Morrison-Woodbury formula (see [16] for a more stable procedure updating the Cholesky factors of B_k).

A possible alternative, named M–$BFGS$, has been recently introduced by replacing in (2.3) the matrix $\Phi(B_k, \mathbf{s}_k, \mathbf{y}_k)$ with $\Phi(B_k, \mathbf{s}_k, \tilde{\mathbf{y}}_k)$, where $\tilde{\mathbf{y}}_k$ is defined in the following way

$$\tilde{\mathbf{y}}_k = \left[1 + \left(3(\mathbf{g}_{k+1} + \mathbf{g}_k)^T\mathbf{s}_k - 6(f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k))\right)\frac{1}{\mathbf{s}_k^T\mathbf{y}_k}\right]\mathbf{y}_k$$

(see [46], [47]). As a consequence one obtains the equality $B_{k+1}\mathbf{s}_k = \tilde{\mathbf{y}}_k$, that is a modified version of the classical secant equation (1.1), giving rise to a better approximation B_{k+1} of $G(\mathbf{x}_{k+1})$. More precisely, it can be proved that $\mathbf{s}_k^T B_{k+1}\mathbf{s}_k = \mathbf{s}_k^T G(\mathbf{x}_{k+1})\mathbf{s}_k$ whenever $f(\mathbf{x})$ is a cubic function. This method seems to lead to a 10 or 15% saving of computational time with respect to the standard $BFGS$ algorithm.

Both $BFGS$ and M–$BFGS$ could be inefficient, however, for large values of n, taking in account, besides the number of flops, the $O(n^2)$ memory allocations required by their implementation.

In order to reduce time and space complexity of $BFGS$ method, one can try to replace B_k by a suitable simpler matrix $\tilde{B}_k$ through the following steps:

1. construct an iterative formula, more general than (2.3), in terms of a generic positive definite matrix $\tilde{B}_k$ (replacing B_k):

$$B_{k+1} = \Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k). \tag{2.7}$$

A natural choice of $\tilde{B}_k$ could be in the set of matrices *approximating* B_k.

2. Look for *best* choices of $\tilde{B}_k$ in order to minimize the complexity by maintaining a quasi-Newtonian rate of convergence.

Of course, the same strategy could be applied to M–$BFGS$ method.

The iterative scheme (2.7) leads to the following general $BFGS$–type method (including $BFGS$ as a particular case):

$$\begin{array}{l} \mathbf{x}_0 \in \mathbb{R}^n, \ B_0 = pd\ n \times n\ matrix. \\ For\ k = 0, 1, \ldots : \\ \left\{ \begin{array}{l} \tilde{B}_k = pd\ n \times n\ matrix \\ \mathbf{d}_k = -B_k^{-1}\mathbf{g}_k \\ \mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k \\ \mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \\ B_{k+1} = \Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k) \end{array} \right. \end{array} \tag{2.8}$$

In the new $\mathcal{L}QN$ methods described in detail in Sections 4–6 the matrix $\tilde{B}_k$ is chosen as the best least–squares fit (in Frobenius norm) of B_k in a suitable matrix algebra $\mathcal{L}$. The main idea is to reduce the complexity per step to a small number of fast transforms diagonalizing the matrices of $\mathcal{L}$. In this way one obtains $O(n \log n)$ flops instead of the $O(n^2)$ of $BFGS$. Also the space complexity is reduced, from $O(n^2)$ to $O(n)$.

Notice that the limited–memory $BFGS$ method (L–$BFGS$) [1], [31], [33], [34], [47] and the OSS–$OSSV$ methods [2], [23], [39] turn out to be particular cases of the algorithm (2.8). The main idea in L–$BFGS$ method is to use second order $BFGS$ information only from the most recent iterations. The L–$BFGS$ updating formula is exactly the (2.7) where $\tilde{B}_k$ depends on m different $\mathbf{y}_j$ with $j = k, \ldots, k-m+1$. More precisely we have

$$\tilde{B}_k = \Phi(\Phi(\cdots \Phi(B_k^0, \mathbf{s}_{k-m+1}, \mathbf{y}_{k-m+1}) \cdots, \mathbf{s}_{k-2}, \mathbf{y}_{k-2}), \mathbf{s}_{k-1}, \mathbf{y}_{k-1}) \tag{2.9}$$

where B_k^0 is usually equal to the inverse of $(\mathbf{s}_k^T\mathbf{y}_k/\mathbf{y}_k^T\mathbf{y}_k)I$ [34]. For $m = k-1$ one obtains the $BFGS$ updating formula, i.e. $\tilde{B}_k = B_k$. The OSS–$OSSV$ methods [2], [23] are defined by the memory–less updating formula (2.7) with $\tilde{B}_k = I$ and can be implemented with $O(n)$ flops per step. As L–$BFGS$, the OSS–$OSSV$ methods represent a useful alternative to $BFGS$ when n is large; in particular, if $f(\mathbf{x})$ in (2.1) is the error function of a large MLP network [26]. From experimental results the convergence of these methods seem to be guaranteed provided that $\mathbf{d}_k$ is redefined as the steepest descent direction $-\mathbf{g}_k$, either every n steps in OSS, or following a suitable adaptive procedure [42] in $OSSV$.

Observe that in (2.8) the secant equation $B_{k+1}\mathbf{s}_k = \mathbf{y}_k$ is verified if B_{k+1} has the expression (2.7). So the method (2.8) will be referred as *secant* algorithm ($\mathcal{S}$). If $\tilde{B}_k$ is an "approximation" of B_k, then an alternative *nonsecant* algorithm ($\mathcal{NS}$) is obtained by changing the definition of $\mathbf{d}_k$ as follows:

$$\mathbf{d}_k = -\tilde{B}_k^{-1}\mathbf{g}_k. \tag{2.10}$$

A well known property of the updating formula Φ used in both $\mathcal{S}$ and $\mathcal{NS}$ algorithms is stated in the following

Proposition 2.1 [16]. *Let $\tilde{B}$ be a pd $n \times n$ matrix and let $\mathbf{s}$, $\mathbf{y}$ belong to $\mathbb{R}^n$. Then the matrix $\Phi(\tilde{B}, \mathbf{s}, \mathbf{y})$ is a well defined pd $n \times n$ matrix iff $\mathbf{y}^T\mathbf{s} > 0$.*

By Proposition 2.1 it is possible to state that, if the positive parameters λ_k are properly chosen, then both $\mathcal{S}$ and $\mathcal{NS}$ algorithms yield well defined, strictly decreasing sequences $\{f(\mathbf{x}_k)\}$. In particular, for a continuously differentiable, lower bounded function f, such a sequence is obtained if the step length λ_k satisfies the Armijo-Goldstein (AG) prescriptions [16] (see also [24], [25], [37], [44]), that is, λ_k belongs to the set Λ_k defined here below.

Definition. *Set $\chi_k(\lambda) = f(\mathbf{x}_k + \lambda\mathbf{d}_k)$ and fix two constants c_1, c_2, $0 < c_1 < c_2 < 1$. Then the AG set Λ_k is the set of all $\lambda \in \mathbb{R}^+$ such that*

$$\begin{cases} \chi_k(\lambda) \le \chi_k(0) + \lambda c_1 \chi_k'(0), \\ \chi_k'(\lambda) \ge c_2 \chi_k'(0). \end{cases} \tag{2.11}$$

In fact, since $\mathbf{d}_k$ is a descent direction in $\mathbf{x}_k$ ($\chi_k'(0) = \mathbf{g}_k^T\mathbf{d}_k < 0$), the set Λ_k is nonempty, and the choice $\lambda_k \in \Lambda_k$ yields the inequalities $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ and $\mathbf{s}_k^T\mathbf{y}_k = \lambda_k(\chi_k'(\lambda_k) - \chi_k'(0)) > 0$. So, by Proposition 2.1, B_{k+1} in (2.8) is a well defined *pd* matrix and

$$\chi_{k+1}'(0) = \mathbf{g}_{k+1}^T\mathbf{d}_{k+1} = \begin{cases} -\mathbf{g}_{k+1}^T B_{k+1}^{-1}\mathbf{g}_{k+1} & \mathcal{S} \\ -\mathbf{g}_{k+1}^T \tilde{B}_{k+1}^{-1}\mathbf{g}_{k+1} & \mathcal{NS} \end{cases} < 0$$

(unless $\mathbf{g}_{k+1} = \mathbf{0}$), i.e. $\mathbf{d}_{k+1}$ is a well defined descent direction in $\mathbf{x}_{k+1}$.

3 Global convergence results for the $\mathcal{NS}$ methods

In this section global convergence properties of the $\mathcal{NS}$ algorithm are investigated assuming that the step-lengths λ_k satisfy the AG conditions (2.11). More precisely, the well known result of Powell on the global convergence of the $BFGS$ method is extended to the $\mathcal{NS}$ $BFGS$–type algorithm, by adding few simple hypotheses on $\tilde{B}_k$.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a lower bounded, continuously differentiable function. Consider the basic steps of the $\mathcal{NS}$ algorithm to minimize f by choosing λ_k in the AG set Λ_k:

$$
\begin{array}{l}
\mathbf{x}_0 \in \mathbb{R}^n,\ B_0 \in \mathbb{R}^{n\times n}\ pd.\\
For\ k = 0, 1, \ldots :\\
if\ \nabla f(\mathbf{x}_k) \neq \mathbf{0},\ then\\
\left\{\begin{array}{l}
define\ a\ pd\ matrix\ \tilde{B}_k \in \mathbb{R}^{n\times n}\\
\mathbf{d}_k = -\tilde{B}_k^{-1}\nabla f(\mathbf{x}_k)\\
\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k, \quad \lambda_k \in \Lambda_k\\
B_{k+1} = \Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k)
\end{array}\right.
\end{array}
\tag{3.1}
$$

Denote by $\mathcal{I}_0$ the level set $\{\mathbf{x} : f(\mathbf{x}) \leq f(\mathbf{x}_0)\}$. As a consequence of Proposition 2.1 and the subsequent considerations illustrated in Section 2, we can state

Proposition 3.1. *The $\mathcal{NS}$ algorithm yields a sequence of points $\mathbf{x}_{k+1}$, $k = 0, 1, \ldots$, such that*

$$f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k) \quad and \quad \mathbf{y}_k^T \mathbf{s}_k > 0.$$

Therefore, $\mathbf{x}_{k+1}$ belongs to the set $\mathcal{I}_0$ and the matrix $B_{k+1} = \Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k)$ is well defined and pd, until $\nabla f(\mathbf{x}_k) = \mathbf{0}$.

From now on assume that $\nabla f(\mathbf{x}_k) \neq \mathbf{0}$, $\forall k$ (otherwise the algorithm terminates in a finite number of steps at a stationary point for f). Since $\{f(\mathbf{x}_k)\}$ is a lower bounded strictly decreasing sequence, obviously $\lim_{k\to\infty} f(\mathbf{x}_k) \geq \inf f(\mathbf{x})$.

In the following fundamental theorem we prove that under special prescriptions on the trace and on the determinant of $\tilde{B}_k$, a subsequence of $\{\nabla f(\mathbf{x}_k)\}$ converges to the null vector, provided that the ratios $\frac{\|\mathbf{y}_k\|^2}{\mathbf{y}_k^T \mathbf{s}_k}$ are upper bounded.

Theorem 3.2. *Let $\tilde{B}_k$ in algorithm* (3.1) *satisfy the conditions*

$$
\left\{\begin{array}{l}
\mathrm{tr} B_k \geq \mathrm{tr}\tilde{B}_k\\
\det B_k \leq \det \tilde{B}_k
\end{array}\right.
\tag{3.2}
$$

If $\exists M > 0$*:*

$$\frac{\|\mathbf{y}_k\|^2}{\mathbf{y}_k^T \mathbf{s}_k} \leq M \tag{3.3}$$

then

$$\liminf \|\nabla f(\mathbf{x}_k)\| = 0. \tag{3.4}$$

Proof. The points $\mathbf{x}_{j+1}$ are in the level set $\mathcal{I}_0$ and satisfy the inequalities

$$f(\mathbf{x}_{j+1}) \leq f(\mathbf{x}_j) + \lambda_j c_1 \mathbf{g}_j^T \mathbf{d}_j, \tag{3.5}$$

$$\mathbf{g}_{j+1}^T \mathbf{d}_j \geq c_2 \mathbf{g}_j^T \mathbf{d}_j. \tag{3.6}$$

By (3.5), $j = 0, 1, \ldots, k$, we have $c_1 \sum_{j=0}^{k} \lambda_j (-\mathbf{g}_j^T \mathbf{d}_j) \leq f_0 - f_{k+1} \leq f_0 - \inf f(\mathbf{x})$. Thus the series $\sum_{j=0}^{+\infty} \lambda_j (-\mathbf{g}_j^T \mathbf{d}_j)$ is convergent and we obtain that

$$-\mathbf{g}_j^T \mathbf{s}_j = \lambda_j (-\mathbf{g}_j^T \mathbf{d}_j) \to 0. \tag{3.7}$$

On the other hand, by (3.2)

$$\begin{aligned} \operatorname{tr}(B_{j+1}) &= \operatorname{tr}(\tilde{B}_j) + \frac{1}{\mathbf{y}_j^T \mathbf{s}_j} \mathbf{y}_j^T \mathbf{y}_j - \frac{1}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} (\tilde{B}_j \mathbf{s}_j)^T (\tilde{B}_j \mathbf{s}_j) \\ &\leq \operatorname{tr}(B_j) + \frac{\|\mathbf{y}_j\|^2}{\mathbf{y}_j^T \mathbf{s}_j} - \frac{\|\tilde{B}_j \mathbf{s}_j\|^2}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j}, \quad j = 0, 1, \ldots, k. \end{aligned}$$

Hence:

$$\operatorname{tr}(B_{k+1}) \leq \operatorname{tr}(B_0) + \sum_{j=0}^{k} \frac{\|\mathbf{y}_j\|^2}{\mathbf{y}_j^T \mathbf{s}_j} - \sum_{j=0}^{k} \frac{\|\tilde{B}_j \mathbf{s}_j\|^2}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j}.$$

Thus, by (3.3), we have $\operatorname{tr}(B_{k+1}) \leq \operatorname{tr}(B_0) + M(k+1) \leq c_3(k+1)$ and by the geometric/arithmetic mean inequality

$$\det(B_{k+1}) = \prod_{i=1}^{n} \nu_i(B_{k+1}) \leq \left(\frac{\sum_{i=1}^{n} \nu_i(B_{k+1})}{n} \right)^n \leq \left(\frac{c_3(k+1)}{n} \right)^n \tag{3.8}$$

where $\nu_i(B_{k+1})$ are the eigenvalues of B_{k+1}. Moreover

$$\begin{aligned} \sum_{j=0}^{k} \frac{\|\tilde{B}_j \mathbf{s}_j\|^2}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} &\leq \operatorname{tr}(B_0) - \operatorname{tr}(B_{k+1}) + \sum_{j=0}^{k} \frac{\|\mathbf{y}_j\|^2}{\mathbf{y}_j^T \mathbf{s}_j} \\ &\leq \operatorname{tr}(B_0) + \sum_{j=0}^{k} \frac{\|\mathbf{y}_j\|^2}{\mathbf{y}_j^T \mathbf{s}_j} \leq c_3(k+1). \end{aligned}$$

Again, by the geometric/arithmetic mean inequality, we have

$$\prod_{j=0}^{k} \frac{\|\tilde{B}_j \mathbf{s}_j\|^2}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \leq c_3^{k+1}. \tag{3.9}$$

Furthermore, the following inequalities hold:

$$\det(B_{j+1}) = \frac{\mathbf{s}_j^T \mathbf{y}_j}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \det(\tilde{B}_j) \geq \frac{\mathbf{s}_j^T \mathbf{y}_j}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \det(B_j), \quad j = 0, 1, \ldots, k$$

(see [35] or Lemma 7.6 in [15] and (3.2)). So

$$\prod_{j=0}^{k} \frac{\mathbf{s}_j^T \mathbf{y}_j}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \leq \frac{\det(B_{k+1})}{\det(B_0)}. \tag{3.10}$$

By (3.6) we have $\mathbf{y}_j^T \mathbf{s}_j \geq (1 - c_2)(-\mathbf{g}_j^T \mathbf{s}_j)$. Moreover

$$\tilde{B}_j \mathbf{s}_j = \tilde{B}_j(\mathbf{x}_{j+1} - \mathbf{x}_j) = \tilde{B}_j(-\lambda_j \tilde{B}_j^{-1} \mathbf{g}_j) = -\lambda_j \mathbf{g}_j.$$

Thus, by (3.8), (3.9) and (3.10) we obtain

$$\begin{aligned}(1 - c_2)^{k+1} \prod_{j=0}^{k} \frac{\|\mathbf{g}_j\|^2}{\mathbf{s}_j^T(-\mathbf{g}_j)} &\leq \prod_{j=0}^{k} \frac{\|\tilde{B}_j \mathbf{s}_j\|^2}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \frac{\mathbf{s}_j^T \mathbf{y}_j}{\mathbf{s}_j^T \tilde{B}_j \mathbf{s}_j} \\ &\leq c_3^{k+1} \left(\frac{c_3(k+1)}{n} \right)^n \frac{1}{\det(B_0)} \leq c_4^{k+1},\end{aligned}$$

i.e.

$$\prod_{j=0}^{k} \frac{\|\mathbf{g}_j\|^2}{\mathbf{s}_j^T(-\mathbf{g}_j)} \leq c_5^{k+1}. \tag{3.11}$$

Since (3.7) and (3.11) hold simultaneously, a subsequence of $\{\|\mathbf{g}_j\|^2\}$ must be convergent to zero. □

Clearly , the above proof fails for the $\mathcal{S}$ method, as $\tilde{B}_j \mathbf{s}_j$ is no longer equal to $-\lambda_j \mathbf{g}_j$ ($\tilde{B}_j \mathbf{s}_j = -\lambda_j \tilde{B}_j B_j^{-1} \mathbf{g}_j$).

It is important to emphasize the fundamental role played by the condition (3.3) in a general global convergence result. The hypothesis (3.3) is actually a sort of "weak form of discrete convexity" guaranteeing the convergence of the approximating sequence. In fact we have the following

Corollary 3.3. *Let* (3.2) *and* (3.3) *hold and let* $\mathcal{I}_0$ *be bounded. Then a subsequence of* $\{\mathbf{x}_k\}$ *converges to a stationary point* $\mathbf{x}_*$ *of* f *and* $f(\mathbf{x}_k) \to f(\mathbf{x}_*)$.

Proof. By Theorem 3.2 we have $\liminf \|\nabla f(\mathbf{x}_k)\| = 0$. Let i_k, $\|\nabla f(\mathbf{x}_{i_k})\| \to 0$, $k \to \infty$. As $\{\mathbf{x}_{i_k}\}$ is bounded, there exists $\{\mathbf{x}_{j_{i_k}}\}$, $\mathbf{x}_{j_{i_k}} \to \mathbf{x}_* \in \mathcal{I}_0$. Moreover, $f(\mathbf{x}_{j_{i_k}}) \to f(\mathbf{x}_*)$, $\|\nabla f(\mathbf{x}_{j_{i_k}})\| \to \|\nabla f(\mathbf{x}_*)\|$ and $\|\nabla f(\mathbf{x}_{j_{i_k}})\| \to 0$. Thus $\nabla f(\mathbf{x}_*) = \mathbf{0}$ and $f(\mathbf{x}_k) \to f(\mathbf{x}_*)$. □

A sufficient condition to fulfil the inequality (3.3) is shown in the next Corollary 3.5, which requires some suitable convexity assumptions on f. These assumptions and the boundness of $\mathcal{I}_0$, allow us also to deduce from (3.4) that $f(\mathbf{x}_k) \to \min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$. The proof of Corollary 3.5 is based on the following result.

Proposition 3.4 [38]. *Assume that f is convex and has continuous and bounded second derivatives in a convex set $\mathcal{I} \subset \mathbb{R}^n$. Then, for all $\mathbf{x},\mathbf{y} \in \mathcal{I}$,*

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2 \leq M(\nabla f(\mathbf{x}) - \nabla f(\mathbf{y}))^T(\mathbf{x} - \mathbf{y})$$

where $\|\nabla^2 f(\mathbf{x})\| \leq M$ in $\mathcal{I}$.

Now the condition (3.3) on our algorithm can be easily derived by using Proposition 3.1 and by applying Proposition 3.4 for $\mathcal{I} = \mathcal{I}_0$.

Corollary 3.5. *Let f be a twice continuously differentiable convex function in the level set $\mathcal{I}_0$. Assume $\mathcal{I}_0$ convex and bounded. Let $\tilde{B}_k$ in* (3.1) *satisfy the conditions* (3.2). *Then $\{f(\mathbf{x}_k)\}$ converges to the least value of f.*

In Section 5 we will prove that the conditions (3.2) are fulfilled for $\tilde{B}_k = \mathcal{L}_{B_k}$ where $\mathcal{L}_{B_k}$ is the best least squares fit of B_k from suitable matrix algebra $\mathcal{L}$.

4 $\mathcal{L}QN$ methods

In the previous section a convergence result has been obtained for *general* nonsecant algorithms under the only conditions (3.2), (3.3). This result may seem surprising as a generic $\tilde{B}_k$ is used instead of B_k in the updating formula, but is fully justified by the fact that the convergence of the quasi-Newtonian methods does not depend necessarily on the convergence of B_k to the Hessian $G(\mathbf{x}_*)$.

So, we can try to obtain the maximum gain from this independence by choosing $\tilde{B}_k$ in an algebra $\mathcal{L}$ of matrices diagonalized (or quasi-diagonalized) by a fast transform $U_{\mathcal{L}}$, in order to reduce the computational complexity. A natural choice of $\tilde{B}_k$ in $\mathcal{L}$ is $\tilde{B}_k = \mathcal{L}_{B_k}$ where $\mathcal{L}_{B_k}$ is the best least squares fit to B_k in $\mathcal{L}$. The latter assumption does not compromise the convergence properties.

Let $M_n(\mathbb{C})$ be the set of all $n \times n$ matrices with complex entries and let $B \in M_n(\mathbb{C})$. Let $\mathcal{L}$ be a subspace of $M_n(\mathbb{C})$ of dimension m. The set $M_n(\mathbb{C})$ is

a Hilbert space with the inner product $(X, Y) = \sum_{i,j=1}^{n} \overline{x}_{ij} y_{ij}$ and the norm induced by $(\cdot, \cdot)$ is the Frobenius norm $\|X\|_F = (\sum_{i,j=1}^{n} |x_{ij}|^2)^{1/2}$. Thus, by the Hilbert projection theorem, there exists a unique element $\mathcal{L}_B \in \mathcal{L}$ such that

$$\|\mathcal{L}_B - B\|_F \leq \|X - B\|_F, \ \forall X \in \mathcal{L}, \tag{4.1}$$

or, equivalently, such that

$$(B - \mathcal{L}_B, X) = 0, \ \forall X \in \mathcal{L}. \tag{4.2}$$

The matrix $\mathcal{L}_B$ is called *the best least squares (l.s.) fit to B from* $\mathcal{L}$ [22].

If $\{J_1, J_2, \ldots, J_m\}$ is a basis of $\mathcal{L}$, then an explicit formula for $\mathcal{L}_B$ can be obtained by solving the system of normal equations

$$W\mathbf{z} = \mathbf{c}, \quad w_{ij} = (J_i, J_j), \ c_i = (J_i, B), \tag{4.3}$$

in fact, by the orthogonality condition (4.2), we have

$$\mathcal{L}_B = \sum_{k=1}^{m} [W^{-1}\mathbf{c}]_k J_k. \tag{4.4}$$

The representation (4.4) of $\mathcal{L}_B$ implies that $\mathcal{L}_B$ is linear, that is,

$$\mathcal{L}_{\alpha X + \beta Y} = \alpha \mathcal{L}_X + \beta \mathcal{L}_Y. \tag{4.5}$$

Moreover, by using (4.4), one easily realizes that $\mathcal{L}_B$ is real whenever B is real and $\mathcal{L}$ is spanned by real matrices.

The $\mathcal{L}QN$ *methods* are defined by setting $\tilde{B}_k = \mathcal{L}_{B_k}$ in the $\mathcal{S}$ and $\mathcal{NS}$ algorithms :

$$\begin{array}{l} \mathbf{x}_0 \in \mathbb{R}^n, \ B_0 = pd \ n \times n \ matrix. \\ For \ k = 0, 1, \ldots : \\ \left\{ \begin{array}{l} \mathbf{d}_k = \left\{ \begin{array}{ll} -B_k^{-1}\mathbf{g}_k & \mathcal{S} \\ -\mathcal{L}_{B_k}^{-1}\mathbf{g}_k & \mathcal{NS} \end{array} \right. \\ \mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k \\ \mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \ \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \\ B_{k+1} = \Phi(\mathcal{L}_{B_k}, \mathbf{s}_k, \mathbf{y}_k) \end{array} \right. \end{array} \tag{4.6}$$

and by requiring that $\mathcal{L}$ satisfies the condition

$$B \ pd \ \Rightarrow \ \mathcal{L}_B \ pd. \tag{4.7}$$

Notice that the implication in (4.7) assures that $-B_{k+1}^{-1}\mathbf{g}_{k+1}$ in $\mathcal{S}$ and $-\mathcal{L}_{B_{k+1}}^{-1}\mathbf{g}_{k+1}$ in $\mathcal{NS}$ are both descent directions provided that B_k is *pd* and λ_k is such that $\mathbf{y}_k^T \mathbf{s}_k > 0$ (see Proposition 2.1). So $\mathcal{L}QN$ methods always yield well defined, strictly decreasing sequences $\{f(\mathbf{x}_k)\}$ (e.g. by choosing $\lambda_k \in AG$).

The $\mathcal{L}QN$ methods represent a sort of optimal compromise between the $BFGS$ and the OSS–$OSSV$ methods. As a matter of fact, each iteration of $\mathcal{L}QN$ methods maintains a larger amount of second order information with respect to the OSS–$OSSV$ methods. In fact, besides $\mathbf{s}_k$ and $\mathbf{y}_k$, the updated matrix, $\mathcal{L}_{B_k}$, depends upon the previous Hessian approximation B_k. Notice that the distribution of the eigenvalues of $\mathcal{L}_{B_k}$ is strictly related to that of B_k. In particular, the following result holds [40], [41]: if B is hermitian and $\mathcal{L}$ is a space of matrices simultaneously diagonalized by a unitary matrix $U_{\mathcal{L}}$ ($\mathcal{L} = SDU_{\mathcal{L}}$), then $\mathcal{L}_B$ is hermitian and

$$\begin{aligned} \nu_1(B) + \cdots + \nu_i(B) &\le \nu_1(\mathcal{L}_B) + \cdots + \nu_i(\mathcal{L}_B), \\ \nu_i(\mathcal{L}_B) + \cdots + \nu_n(\mathcal{L}_B) &\le \nu_i(B) + \cdots + \nu_n(B) \end{aligned} \tag{4.8}$$

where $\nu_i(X)$, $i = 1, \ldots, n$, denote the eigenvalues of X in nondecreasing order. For this reason, the search direction proposed in $\mathcal{L}QN$ methods appears to be (at least for the $\mathcal{S}$ method) much more related to the optimization criteria used by $BFGS$.

Observe that any space $\mathcal{L}$ for which

$$B \text{ } hpd \Rightarrow \mathcal{L}_B \text{ } hpd \tag{4.9}$$

(*hpd* denotes hermitian positive definite), verifies (4.7) whenever $\mathcal{L}$ is spanned by real matrices. From (4.8), $i = 1$ (see also Proposition 5.1) it follows that $SDU_{\mathcal{L}}$ spaces satisfy (4.9). But in order to have *real* approximations $\mathbf{x}_k$ the $SDU_{\mathcal{L}}$ spaces are required to have a real basis, so that (4.7) is satisfied. Let us see an example.

Consider the set $\mathcal{C}_\xi$ of the ξ–circulant matrices

$$\mathcal{C}_\xi(\mathbf{a}) = \sum_{k=1}^{n} a_k (P_\xi)^{k-1},$$

$$P_\xi = \begin{bmatrix} 0 & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \\ \xi & & & & 0 \end{bmatrix}, \quad \xi = e^{-\mathbf{i}\varphi}, \ \varphi \in [0, 2\pi).$$

For the space $\mathcal{C}_\xi$ we have

$$\mathcal{C}_\xi = \{F_\xi \mathrm{diag}(z_k) F_\xi^* : z_k \in \mathbb{C}\}, \quad [F_\xi]_{kj} = \frac{1}{\sqrt{n}} (e^{-\mathbf{i}(\varphi + 2j\pi)/n})^k, \ \mathbf{i} = \sqrt{-1},$$

To prove this identity simply note that $P_\xi F_\xi = F_\xi D_{P_\xi}$ with D_{P_ξ} diagonal. Thus $\mathcal{C}_\xi = SDU_{\mathcal{C}_\xi}$ with $U_{\mathcal{C}_\xi} = F_\xi$ and therefore, by (4.8), it verifies the

condition (4.9). On the other hand, by (4.4) with $J_k = (P_\xi)^{k-1}$, we have

$$
\begin{aligned}
(\mathcal{C}_\xi)_B &= \mathcal{C}_\xi(\mathbf{a}) = \sqrt{n} F_\xi d(F_\xi^T \mathbf{a}) F_\xi^*, \\
a_k &= \frac{1}{n}\Big(\sum_{i=1}^{n-k+1} b_{i,i+k-1} + \overline{\xi} \sum_{i=n-k+2}^{n} b_{i,i+k-1-n}\Big)
\end{aligned} \tag{4.10}
$$

and thus, if B is real, then the matrix $(\mathcal{C}_\xi)_B$ is real iff $\xi = \pm 1$ or iff the matrices $(P_\xi)^k$ are real. So the condition (4.7) is verified for $\mathcal{L} = \mathcal{C}_\xi$ iff $\xi = \pm 1$.

Notice that in Section 5 it is shown that $\mathcal{L}QN$ methods where $\mathcal{L}$ is equal to $SDU_{\mathcal{L}}$ and is spanned by real matrices are globally convergent. This is in particular true for $\mathcal{C}_1 QN$ and $\mathcal{C}_{-1} QN$.

Property (4.9) has been partially investigated in the context where the l.s. approximations $\mathcal{L}_B$ have found their first application, i.e. in preconditioning techniques for minimizing quadratic functions

$$
f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T B\mathbf{x} - \mathbf{x}^T\mathbf{v} + \alpha, \quad \mathbf{v} \in \mathbb{R}^n,\ \alpha \in \mathbb{R},
$$

with a positive definite Hessian B [4], [12], [13], [22], [28], [29], [36], [40]. In fact, when the Hessian is a Toeplitz matrix, $B = (b_{|i-j|})_{i,j=1}^n$, T. Chan [13] suggested as a possible preconditioner the best l.s. fit to B from the space $\mathcal{C} = \mathcal{C}_1$ of circulant matrices. He showed, in some experimental results, that the minimization of the function $\hat{f}(\mathbf{y}) = f(E^{-T}\mathbf{y})$, $EE^T = \mathcal{C}_B$, with a conjugate gradient (CG) type method, requires a lower number of iterations with respect to the direct minimization of $f(\mathbf{x})$. In [12], [28], [36], [40] a theoretical justification of the idea of T. Chan was obtained in terms of the implication (4.9) where $\mathcal{L}$ represents the spaces of matrices simultaneously diagonalized by unitary transforms ($SDU_{\mathcal{L}}$ spaces), and, in particular, the circulant matrices used by T. Chan.

Notice that, in the $\mathcal{L}QN$ algorithms, $\mathcal{L}_{B_k}$ is not used as preconditioner. The matrix $\mathcal{L}_{B_k}$ *replaces* B_k either in both formulas defining respectively the search direction $\mathbf{d}_k$ and the matrix B_{k+1} ($\mathcal{NS}$) or only in the $BFGS$ updating formula defining B_{k+1} ($\mathcal{S}$). Thus, for example, in the $\mathcal{NS}$ $\mathcal{L}QN$ algorithm the system $B_k\mathbf{d}_k = -\mathbf{g}_k$ is replaced by $\mathcal{L}_{B_k}\mathbf{d}_k = -\mathbf{g}_k$ (not preconditioned by $\mathcal{L}_{B_k}$). However an efficient criterion for the choice of the best preconditioner $\mathcal{L}_B$ could be useful in the choice of the best $\mathcal{L}QN$ method, at least in the neighbourhood of $\mathbf{x}_*$ where f can be approximated by a positive quadratic function. In fact some preliminary experimental results show that if $G(\mathbf{x}_*) \in \mathcal{L}$, then $\mathcal{L}QN$ seems to converge faster. Moreover, a definite, *a priori* choice of the algebra $\mathcal{L}$ could be replaced by a more adaptive choice *during* the process.

In [22] it is shown that the class of spaces $\mathcal{L}$ for which the condition (4.9) is verified is greater than $\{SDU_{\mathcal{L}} :\ U_{\mathcal{L}}$ unitary$\}$. In particular it includes any group algebra defined as $\mathbb{C}[\mathcal{G}] = \{X \in M_n(\mathbb{C}) :\ x_{i,j} = x_{ki,kj},\ i, j, k \in \mathcal{G}\}$,

where $\mathcal{G} = \{1, 2, \ldots, n\}$ is a group. So the space $\Delta = \mathbb{C}[\mathcal{G}]$, $\mathcal{G}$ =dihedral group, of the matrices

$$\begin{bmatrix} X & J_m Y \\ J_m Y & X \end{bmatrix}, \quad X, Y \ \ m \times m \ \text{ circulants} \quad (J_m = (\delta_{i,m-j+1})_{i,j=1}^{m}, m = \tfrac{n}{2})$$

verifies (4.9). Moreover, Δ verifies (4.7) since it is obviously spanned by real matrices.

In [22] it is also shown that if a subspace $\mathcal{L}$ of $M_n(\mathbb{C})$ satisfies (4.9), then $\mathcal{L}$ must be closed under conjugate transposition. An example of space $\mathcal{L}$ closed under conjugate transposition for which (4.9) has not yet been proved is

$$\mathcal{C} + J\mathcal{C} = \{X + JY : \ X, Y \ n \times n \text{ circulants}\}, \quad J = J_n.$$

As the dimension of $\mathcal{C} + J\mathcal{C}$ is about twice the dimension of $\mathcal{C}$ and

$$\|(\mathcal{C} + J\mathcal{C})_B - B\|_F \leq \|\mathcal{C}_B - B\|_F$$

one expects that a $(\mathcal{C} + J\mathcal{C})QN$ method could be more efficient than a $\mathcal{C}QN$ method. Thus it would be interesting to know if (4.9), (4.7) hold for $\mathcal{L} = \mathcal{C} + J\mathcal{C}$.

Once the class of all subspaces of $M_n(\mathbb{C})$ satisfying (4.7) is characterized, any space $\mathcal{L}$ of this class could lead, in principle, to a new effective $\mathcal{L}QN$ method.

The most part of this section can be resumed in the following proposition.

Proposition 4.1. *If $\mathcal{L} = SDU_{\mathcal{L}}$, $U_{\mathcal{L}}$ unitary, and $\mathcal{L}$ is spanned by n real matrices J_k, then* (4.7) *is satisfied and the algorithm* (4.6), $\lambda_k \in AG$, *generates real sequences* $\{\mathbf{x}_k\}$, $\{\mathbf{s}_k^T \mathbf{y}_k\}$ *and* $\{f(\mathbf{x}_k)\}$ *with* $\mathbf{s}_k^T \mathbf{y}_k > 0$ *and* $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ *for every k.*

The same conclusion can be extended to a class of spaces $\mathcal{L}$ including non commutative group algebras, like the dihedral Δ.

In the next Sect. 5 and 6 only the $\mathcal{L}QN$ methods where $\mathcal{L} = SDU_{\mathcal{L}}$ are considered. We shall see that if the unitary matrix $U_{\mathcal{L}}$ associated with $\mathcal{L}$ defines a fast transform, then each step of these methods can be performed in $O(n \log n)$ flops. Moreover a global convergence result for the $\mathcal{NS}\,\mathcal{L}QN$ algorithm will be obtained.

5 Convergence of $(SDU_{\mathcal{L}})QN$ methods

Theorem 5.3 below states that, under suitable conditions on f, the $\mathcal{NS}\,\mathcal{L}QN$ method (4.6) is convergent to the least value of f, *independently of the choices of* $\mathcal{L} = SDU_{\mathcal{L}}$, $\mathbf{x}_0 \in \mathbb{R}^n$ and B_0 *pd*. Some properties of $\mathcal{L}_B$, reported in Proposition 5.2, have an important role in the extension of Powell's proof to

the $\mathcal{L}QN$ case. In fact, they imply that the $BFGS$–type convergence conditions (3.2) are satisfied. The inequality (5.2) is here stated for the first time.

Let $U_{\mathcal{L}}$ be a $n \times n$ unitary matrix, $U_{\mathcal{L}}^* = U_{\mathcal{L}}^{-1}$, and set

$$\mathcal{L} = SDU_{\mathcal{L}} = \{U_{\mathcal{L}} d(\mathbf{z}) U_{\mathcal{L}}^* : \ \mathbf{z} \in \mathbb{C}^n\} \tag{5.1}$$

where $d(\mathbf{z}) = \mathrm{diag}(z_i, \ i = 1, \ldots, n), \mathbf{z} = [z_1 \, z_2 \cdots z_n]^T$. As $\mathcal{L}$ is a subspace of $M_n(\mathbb{C})$, for an arbitrary matrix $B \in M_n(\mathbb{C})$ the best l.s. fit to B from $\mathcal{L}$, $\mathcal{L}_B$, is well defined.

One can state the following properties of $\mathcal{L}_B$.

Proposition 5.1. *i) $\mathcal{L}_B = Ud(\mathbf{z}_B)U^*$ where $[\mathbf{z}_B]_j = [U^*BU]_{jj}$. In particular $\mathbf{z}_{\mathbf{xy}^T} = d(U^*\mathbf{x})U^T\mathbf{y}$, $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$.*

ii) If $B = B^*$, then $\mathcal{L}_B = \mathcal{L}_B^*$ and $\min \nu(B) \leq \nu(\mathcal{L}_B) \leq \max \nu(B)$ where $\nu(X)$ denotes the generic eigenvalue of X. Therefore $\mathcal{L}_B$ is hermitian positive definite whenever B is hermitian positive definite.

Proof. The proof of i) lies in the following equality $\|Ud(\mathbf{z})U^* - B\|_F = \|d(\mathbf{z}) - U^*BU\|_F$. The properties in ii) can be obtained directly from i). □

Let us consider the $\mathcal{NS}$ $\mathcal{L}QN$ algorithm (4.6) where $\lambda_k \in AG$, i.e. (3.1) with $\tilde{B}_k = \mathcal{L}_{B_k}$, and assume that $\mathcal{L} = SDU_{\mathcal{L}}$ is spanned by real matrices, so that the conclusions in Proposition 4.1 hold. Now *the conditions* (3.2) *turn out to be satisfied for* $\tilde{B}_k = \mathcal{L}_{B_k}$:

Proposition 5.2. *Let $\mathcal{L} = SDU_{\mathcal{L}}$ and let $B \in M_n(\mathbb{C})$. Then* $\mathrm{tr}(\mathcal{L}_B) = \mathrm{tr}(B)$. *Moreover, if B is hpd, then*

$$\det(B) \leq \det(\mathcal{L}_B) \tag{5.2}$$

and $\det(B) = \min_{U_{\mathcal{L}}}(\det(\mathcal{L}_B))$.

Proof. The equality $\mathrm{tr}(\mathcal{L}_B) = \mathrm{tr}(B)$ follows from Proposition 5.1,i). In order to prove (5.2) first notice that if $U_{\mathcal{L}} = I$, then (5.2) can be rewritten as

$$\det(B) \leq \prod_{i=1}^{n} b_{ii}. \tag{5.3}$$

This is a known property of *hpd* matrices [27]. The inequality (5.2) follows by applying (5.3) to the *hpd* matrix U^*BU, $U = U_{\mathcal{L}}$:

$$\begin{aligned} \det(B) = \det(U^*BU) &\leq \prod_{i=1}^{n} [U^*BU]_{ii} \\ &= \det(\mathrm{diag}([U^*BU]_{ii}, i = 1, \ldots, n)) \\ &= \det(U\,\mathrm{diag}([U^*BU]_{ii}, i = 1, \ldots, n)U^*) = \det(\mathcal{L}_B). \end{aligned}$$

□

Notice that the inequality (5.2) can be considered as a generalization of the Hadamard determinant inequality (5.3).

So, we can rewrite Theorem 3.2 and Corollaries 3.3 and 3.5 for the $\mathcal{L}QN$ case. In particular we have the following

Theorem 5.3. *Consider the algorithm* (3.1) *where* $\tilde{B}_k = \mathcal{L}_{B_k}$ *and* $\mathcal{L} = SDU_{\mathcal{L}}$ *and assume* $\mathcal{I}_0$ *bounded. Then* $\liminf \|\nabla f(\mathbf{x}_k)\| = 0$, *a subsequence of* $\{\mathbf{x}_k\}$ *converges to a stationary point* $\mathbf{x}_*$ *of* f *and* $f(\mathbf{x}_k) \to f(\mathbf{x}_*)$ *provided that the condition* (3.3) *is verified. This condition is in particular verified under the same assumptions on* f *and* $\mathcal{I}_0$ *of Corollary 3.5. Moreover, under these assumptions,* $f(\mathbf{x}_*) = \min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$.

In [18] local convergence properties of $BFGS$-type algorithms are analized in order to retrieve the $BFGS$ convergence behaviour in the case $\tilde{H}_k \neq H_k$. In particular, it is proved that if $U_{\mathcal{L}}$ is the same unitary matrix diagonalizing $G(\mathbf{x}_*)$, then the corresponding $(SDU_{\mathcal{L}})QN$ method converges superlinearly. Obviously, if $U_{\mathcal{L}}$ defines a fast transform, then clearly $\mathcal{L}QN$ outperforms $BFGS$. Notice, however, that by Theorem 5.3, *no restriction on* $\mathcal{L}$ *is needed to obtain the global convergence of* $\mathcal{L}QN$ *methods*. Thus, only the rate of convergence of $\mathcal{L}QN$ methods may depend upon the choice of $\mathcal{L}$.

6 Computational complexity

In the following Theorem 6.1 it is stated that, if $U = U_{\mathcal{L}}$ defines a fast discrete transform, as for instance Fourier, Jacobi-type [43], Hartley-type [9], [6], then $O(n \log n)$ flops are sufficient to perform one step of both $\mathcal{S}$ and $\mathcal{NS}$ $\mathcal{L}QN$ methods. Moreover, the number of memory allocations required to implement the same methods is $O(n)$. An obvious example of such special matrices U is given by the Fourier matrix $F_{\pm 1}$ (see Section 4). Since the $BFGS$, the L–$BFGS$ and the OSS–$OSSV$ methods require, respectively, $O(n^2)$, $O(mn)$ and $O(n)$ flops per step, in order to evaluate the competitivity of the new methods in terms of time complexity, one should study their rate of convergence. A theoretical convergence result was shown in the previous section only for $\mathcal{NS}$. However, intuitively, the $\mathcal{S}$ version of $\mathcal{L}QN$ is able to minimize a function f by performing a smaller number of iterations with respect to $\mathcal{NS}$. As a matter of fact, in the latter method the $\mathcal{L}_{B_k}$ approximation is used also in the definition of the search direction $\mathbf{d}_k$. Actually, a greater efficiency of $\mathcal{S}$ is shown by the experimental data. Thus, one can expect that $\mathcal{S}$ can be the most competitive $\mathcal{L}QN$ method in comparison with the known methods ($BFGS$, L–$BFGS$, OSS–$OSSV$).

Concerning the space complexity, the $\mathcal{L}QN$ methods obviously outperform $BFGS$ whose implementation requires $O(n^2)$ memory allocations. Also L–$BFGS$ methods require a greater amount of memory, precisely

$O(mn)$. The OSS–$OSSV$ algorithms have the same space complexity $O(n)$ of $\mathcal{L}QN$, but the second order information in the single–indexed arrays used in $\mathcal{L}QN$ iterations is substantially richer and more significant than the corresponding one in OSS–$OSSV$. More precisely, as it is shown in the next Theorem 6.1 (see (6.1)–(6.3)), in $\mathcal{L}QN$ the search direction can be defined in terms of the eigenvalues of $\mathcal{L}_{B_k}$ only and, by the inequalities (4.8), such eigenvalues are strictly related to the eigenvalues of the original Hessian approximation B_k. The latter property justifies the stronger relationship between $\mathcal{L}QN$ and $BFGS$ search directions.

Theorem 6.1. *The $\mathcal{S}$ and $\mathcal{NS}$ $\mathcal{L}QN$ methods* (4.6) *with $\mathcal{L} = SDU_{\mathcal{L}}$, require at each step the computation of two $U_{\mathcal{L}}$-discrete transforms plus $O(n)$ flops. Thus their time and space complexity are $O(n \log n)$ and $O(n)$, respectively, whenever $U_{\mathcal{L}}$ defines a fast transform.*

Proof. The result can be achieved by rewriting the algorithm (4.6) in terms of the matrix $U = U_{\mathcal{L}}$. By the linearity (4.5) of $\mathcal{L}_B$ and by the definition of $\mathbf{z}_B$ in Proposition 5.1, the updating formula in (4.6) yields

$$\mathbf{z}_{B_{k+1}} = \mathbf{z}_{B_k} + \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \mathbf{z}_{\mathbf{y}_k \mathbf{y}_k^T} - \frac{1}{\mathbf{s}_k^T \mathcal{L}_{B_k} \mathbf{s}_k} \mathbf{z}_{\mathcal{L}_{B_k} \mathbf{s}_k (\mathcal{L}_{B_k}^T \mathbf{s}_k)^T}.$$

By Proposition 5.1 this formula can be rewritten as follows:

$$\mathbf{z}_{B_{k+1}} = \mathbf{z}_{B_k} + \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} |U^* \mathbf{y}_k|^2 - \frac{1}{\mathbf{z}_{B_k}^T |U^* \mathbf{s}_k|^2} d(\mathbf{z}_{B_k})^2 |U^* \mathbf{s}_k|^2, \tag{6.1}$$

where $|\mathbf{z}|^2$ is the vector whose ith entry is $|z_i|^2$. Moreover the following formulas for the search directions $\mathbf{d}_{k+1}$, respectively, in $\mathcal{S}$ and $\mathcal{NS}$ hold:

$$\begin{aligned} U^* \mathbf{d}_{k+1} = {} & -d(\mathbf{z}_{B_k}^{-1}) U^* \mathbf{g}_{k+1} + \frac{\mathbf{s}_k^T \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} d(\mathbf{z}_{B_k}^{-1}) U^* \mathbf{y}_k \\ & + \left[- \left(1 + \frac{(\mathbf{z}_{B_k}^{-1})^T |U^* \mathbf{y}_k|^2}{\mathbf{y}_k^T \mathbf{s}_k} \right) \frac{\mathbf{s}_k^T \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} \right. \\ & \left. + \frac{(\mathbf{z}_{B_k}^{-1})^T d(U^T \mathbf{y}_k) U^* \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} \right] U^* \mathbf{s}_k, \end{aligned} \tag{6.2}$$

$$U^* \mathbf{d}_{k+1} = -d(\mathbf{z}_{B_{k+1}}^{-1}) U^* \mathbf{g}_{k+1} \tag{6.3}$$

where $\mathbf{z}^{-1}$ denotes the vector whose ith entry is z_i^{-1}.

Here below the algorithm (4.6) is rewritten by using (6.1), (6.2), (6.3).

$$
\begin{array}{l}
\mathbf{x}_0 \in \mathbb{R}^n,\ B_0 = pd\ n \times n\ matrix, \\
U^*\mathbf{d}_0 = \begin{cases} -U^* B_0^{-1}\mathbf{g}_0 & \mathcal{S} \\ -d(\mathbf{z}_{B_0}^{-1})U^*\mathbf{g}_0 & \mathcal{NS} \end{cases}, \\
\mathbf{d}_0 = U(U^*\mathbf{d}_0). \\
For\ k = 0, 1, \ldots : \\
\begin{cases}
\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k \\
\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \\
(6.1) \\
\begin{cases} (6.2) & \mathcal{S} \\ (6.3) & \mathcal{NS} \end{cases} \\
\mathbf{d}_{k+1} = U(U^*\mathbf{d}_{k+1})
\end{cases}
\end{array}
\tag{6.4}
$$

From (6.4) it is clear that each step of the $\mathcal{L}QN$ methods (4.6) consists in computing the two transforms $U^* \cdot \mathbf{g}_{k+1}$ and $U \cdot (U^*\mathbf{d}_{k+1})$ (in $\mathcal{S}$, the vector $U^*\mathbf{d}_{k+1}$ can be computed from $U^*\mathbf{s}_k = \lambda_k U^*\mathbf{d}_k$), and in performing $O(n)$ flops. □

In the $\mathcal{S}$ and $\mathcal{NS}$ $\mathcal{L}QN$ algorithms an obvious convenient choice of B_0 is $B_0 = \mathcal{L}_{B_0} = I$ and thus $\mathbf{z}_{B_0} = [1\ 1 \cdots 1]^T$. This choice implies that the search direction in the first step is the steepest descent one. However, if $\mathbf{x}_0$ is in the neighbourhood of a minimum for f, it could be more convenient to choose the matrix B_0 as an "approximation" of the Hessian $G(\mathbf{x}_0)$ in order to increase the quasi-Newton properties of the methods. In this case, an explicit formula for the vector $\mathbf{z}_{B_0}$ or, equivalently, for the eigenvalues of $\mathcal{L}_{B_0}$, is needed. Such a formula can be deduced from (4.10) when $\mathcal{L} = \mathcal{C}_{\pm 1}$ and from [22], [20] for some other special choices of $\mathcal{L}$. Obviously, B_0 should be chosen in such a way that the complexity of the computation of $\mathbf{z}_{B_0}$ is at most $O(n \log n)$, as the complexity of the generic step.

Clearly, the criterion for the choice of the best algebra $\mathcal{L}$ should take into account the Hessian approximation properties of $\mathcal{L}$. In the context of preconditioning techniques for the minimization of positive definite quadratic functions $\frac{1}{2}\mathbf{x}^T B\mathbf{x} - \mathbf{x}^T\mathbf{v} + \alpha$ experimental results show that if the inequality

$$
\|\mathcal{L}_B - B\|_F << \|\mathcal{L}'_B - B\|_F \tag{6.5}
$$

is verified, then $\mathcal{L}_B$ is in general a better preconditioner than $\mathcal{L}'_B$. For example, if B is more similar to a circulant matrix than to a (-1)-circulant, i.e. $\|\mathcal{C}_B - B\|_F << \|(\mathcal{C}_{-1})_B - B\|_F$, then $\mathcal{C}_B$ usually performs better than $(\mathcal{C}_{-1})_B$ and viceversa [22], [28]. In fact, as is shown in [22], a criterion for the choice of the best preconditioner should take into account also possible symmetries of B and the structure of the vector $\mathbf{v}$. One could use (6.5) with the current Hessian approximation B_k in place of B in order to choose $\mathcal{L}QN$ instead of $\mathcal{L}'QN$. Moreover, if B_k is symmetric but not persymmetric, then

choices of $\mathcal{L}$ different from $\mathcal{L} = \mathcal{C}, \mathcal{C}_{-1}$ (e.g. $\mathcal{L}$=Hartley-type [6]) may lead to better $\mathcal{L}QN$ methods since $\mathcal{C}_{B_k}$ and $(\mathcal{C}_{-1})_{B_k}$ are simultaneously symmetric and persymmetric.

The L–$BFGS$ methods [33], [34], [47] are competitors to the novel $\mathcal{L}QN$ algorithms. In L–$BFGS$ the search direction $\mathbf{d}_{k+1}$ is defined by $\mathbf{d}_{k+1} = -\Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k)^{-1}\mathbf{g}_{k+1}$ with $\tilde{B}_k$ as in (2.9). Now, $\mathbf{d}_{k+1}$ can be computed in $4mn + O(m) + O(n)$ flops per step by using the identity $\Phi(\tilde{B}_k, \mathbf{s}_k, \mathbf{y}_k)^{-1} = \Psi(\tilde{H}_k, \mathbf{s}_k, \mathbf{y}_k)$ where

$$\tilde{H}_k = \Psi(\Psi(\cdots\Psi(H_k^0, \mathbf{s}_{k-m+1}, \mathbf{y}_{k-m+1})\cdots, \mathbf{s}_{k-2}, \mathbf{y}_{k-2}), \mathbf{s}_{k-1}, \mathbf{y}_{k-1}),$$
$$H_k^0 = (B_k^0)^{-1},$$

(see [34,p.225]). Thus, the use of L–$BFGS$ with small values of m can reduce the complexity per step of large scale optimization problems; however, the computation of the $\tilde{B}_k$ matrix involves only a small m–part of B_k and the optimal choice of m is problem–dependent [34]. On the contrary, the $\mathcal{L}QN$ methods (4.6), in which $\tilde{B}_k = \mathcal{L}_{B_k}$, can be implemented with a fixed $O(n \log n)$ number of flops per step and a fixed $O(n)$ amount of memory allocations. Notice that this saving of space complexity does not imply that the amount of second order information in $\mathcal{L}QN$ is smaller than the corresponding one in L–$BFGS$. In fact, by (4.8), the array $\mathbf{z}_{B_k}$ of the eigenvalues of $\mathcal{L}_{B_k}$ is in some sense close to the array of the eigenvalues of B_k and the very kernel of the whole Hessian approximation B_k is achieved by a sort of second order information compression in the single–index array $\mathbf{z}_{B_k}$.

References

1. M. Al Baali: Improved Hessian approximations for the limited memory $BFGS$ method. Numerical Algorithms, **22**, 99–112 (1999)
2. R. Battiti: First- and second-order methods for learning: between steepest descent and Newton's method, Neural Computation, **4**, 141–166 (1992)
3. M. Bianchini, S. Fanelli, M. Gori, M. Protasi: Non-suspiciousness: a generalization of convexity in the frame of foundations of numerical analysis and learning. Proc. of the IJCNN (Anchorage, Alaska, May 1998), 1619–1623 (1998)
4. D. Bini, P. Favati: On a matrix algebra related to the discrete Hartley transform SIAM. J. Matrix Anal. Appl., **14**, 500–507 (1993)
5. D. Bini, V. Pan: Polynomial and Matrix Computations, Vol.1 Fundamental Algorithms Birkhäuser, Boston, MA, 1994
6. A. Bortoletti, C. Di Fiore: On a set of matrix algebras related to discrete Hartley-type transforms. Linear Algebra Appl. (to appear)
7. A. Bortoletti, C. Di Fiore, S. Fanelli, P. Zellini: A new class of quasi-Newtonian methods for optimal learning in MLP-networks. IEEE Transactions on Neural Networks (to appear)
8. E. Bozzo, C. Di Fiore: On the use of certain matrix algebras associated with discrete trigonometric transforms in matrix displacement decomposition. SIAM J. Matrix Anal. Appl., **16**, 312–326 (1995)

9. R. N. Bracewell: The fast Hartley transform. Proc. IEEE., **72**, 1010–1018 (1984)
10. C. G. Broyden: The convergence of a class of double-rank minimization algorithms 2. The new algorithm. J. Inst. Maths Applics., **6**, 222–231 (1970)
11. C. G. Broyden, J. E. Dennis, Jr., Jorge J. Moré: On the local and superlinear convergence of quasi-Newton methods. J. Inst. Maths Applics, **12**, 223–245 (1973)
12. R. Chan, X. Jin, M. Yeung: The circulant operator in the Banach algebra of matrices. Linear Algebra Appl., **149**, 41–53 (1991)
13. T. F. Chan: An optimal circulant preconditioner for Toeplitz systems. SIAM J. Sci. Stat. Comput., **9**, 766–771 (1988)
14. W. C. Davidon: Variable metric method for minimization. US Atomic Energy Commission, Research and Development Report ANL-5990 (Rev.), Argonne National Laboratories, Ill. (1959)
15. J. E. Dennis, Jr., Jorge J. Moré: Quasi-Newton methods, motivation and theory. SIAM Review, **19**, 46–89 (1977)
16. J. E. Dennis, Jr., R. B. Schnabel: Numerical Methods for Unconstrained Optimization and Nonlinear Equations Prentice-Hall, Englewood Cliffs, New Jersey, 1983
17. C. Di Fiore: Matrix algebras and displacement decompositions. SIAM J. Matrix Anal. Appl., **21**, 646–667 (2000)
18. C. Di Fiore: Structured matrices in unconstrained minimization methods (submitted)
19. C. Di Fiore, S. Fanelli, P. Zellini: Matrix algebras in quasi-newtonian algorithms for optimal learning in Multi-Layer Perceptrons. Proc. of the ICONIP/ANZIIS/ANNES '99 (Dunedin, New Zealand, Nov. 1999) N. Kasabov, K. Ko (eds.) pp. 27–32 (1999)
20. C. Di Fiore, F. Lepore, P. Zellini: Matrix algebras in displacement and optimization strategies. Linear Algebra Appl. (to appear)
21. C. Di Fiore, P. Zellini: Matrix decompositions using displacement rank and classes of commutative matrix algebras. Linear Algebra Appl., **229**, 49–99 (1995)
22. C. Di Fiore, P. Zellini: Matrix algebras in optimal preconditioning. Linear Algebra Appl., **335**, 1–54 (2001)
23. S. Fanelli, P. Paparo, M. Protasi: Improving performances of Battiti-Shanno's quasi-newtonian algorithms for learning in feed-forward neural networks. Proc. of the 2nd Australian and New Zealand Conference on Intelligent Information Systems (Brisbane, Australia, Nov.-Dec. 1994), 115–119 (1994)
24. R. Fletcher: A new approach to variable metric algorithms. Computer Journal. **13**, 317–322 (1970)
25. A. A. Goldstein: Constructive Real Analysis. New York: Harper & Row 1967
26. J. Hertz, A. Krogh, R. G. Palmer: Introduction to the Theory of Neural Computation. Santa Fe: Addison-Wesley 1991
27. R. A. Horn, C. R. Johnson: Matrix Analysis. New York: Cambridge University Press 1987
28. T. Huckle: Circulant/skewcirculant matrices as preconditioners for hermitian Toeplitz systems. Iterative Methods in Linear Algebra. R. Beauwens, P. de Groen (eds.) North-Holland: Elsevier Science Publishers B. V. IMACS, 563–574 (1992)
29. T. Kailath, V. Olshevsky: Displacement structure approach to discrete trigonometric transform based preconditioners of G. Strang type and of T. Chan type. Calcolo., **33**, 191–208 (1996)
30. T. Kailath, A. H. Sayed: Displacement structure: theory and applications. SIAM Review., **37**, 297–386 (1995)
31. D. C. Liu, J. Nocedal: On the limited memory $BFGS$ method for large scale optimization. Math. Programming., **45**, 503–528 (1989)
32. M. Møller: A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks., **6**(4) 525–533 (1993)

33. J. Nocedal: Updating quasi-Newton matrices with limited storage. Math Comp., **35**, 773–782 (1980)
34. J. Nocedal, S. J. Wright: Numerical Optimization. New York: Springer 1999
35. J. D. Pearson: Variable metric methods of minimisation. Computer Journal., **12**, 171–178 (1969)
36. D. Potts, G. Steidl: Optimal trigonometric preconditioners for nonsymmetric Toeplitz systems. Linear Algebra Appl., **281**, 265–292 (1998)
37. M. J. D. Powell: A view of unconstrained optimization. Optimization in Action, Proc. of Conference. (Univ. of Bristol, Jan. 1975) (L. C. W. Dixon, ed.), London: Academic Press 117–152 (1976)
38. M. J. D. Powell: Some global convergence properties of a variable metric algorithm for minimization without exact line searches. Nonlinear Programming, SIAM-AMS Proc. (New York, March 1975) (R. W. Cottle, C. E. Lemke, eds.), vol. 9, Providence, 1976, pp. 53–72
39. D. F. Shanno: Conjugate gradient methods with inexact searches. Math. Oper. Res., **3**, 244–256 (1978)
40. E. E. Tyrtyshnikov: Optimal and superoptimal circulant preconditioners. SIAM J. Matrix Anal. Appl., **13**, 459–473 (1992)
41. E. E. Tyrtyshnikov: Matrix approximations: theory and applications in numerical analysis, Lectures held in the Dept. of Mathematics and Computerscience of the Univ. of Udine, October, 2000
42. T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, D. L. Alkon: Accelerating the convergence of the back-propagation method. Biological Cybernetics., **59**, 257–263 (1988)
43. Z. Wang: Fast algorithms for the discrete *W* transform and for the discrete Fourier transform. IEEE Trans. Acoust. Speech Signal Processing., ASSP-**32**, 803–816 (1984)
44. P. Wolfe: Convergence conditions for ascent methods. SIAM Review., **11**, 226–235 (1969)
45. J. Z. Zhang, N. Y. Deng, L. H. Chen: New quasi-Newton equation and related methods for unconstrained optimization. J. Optim. Theory and Appl., **102**, 147–167 (1999)
46. J. Zhang, C. Xu: Properties and numerical performance of quasi-Newton methods with modified quasi-Newton equations. J. Comput. Appl. Math., **137**, 269–278 (2001)
47. C. Zhu, R. H. Byrd, P. Lu, J. Nocedal: L–BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Software., **23**, 550–560 (1997)