Exact linesearch limited-memory quasi-Newton methods for minimizing a quadratic function

David Ek* and Anders Forsgren*

Abstract

The main focus in this paper is exact linesearch methods for minimizing a quadratic function whose Hessian is positive definite. We give a class of limited-memory quasi-Newton Hessian approximations which generate search directions parallel to those of the method of preconditioned conjugate gradients, and hence give finite termination on quadratic optimization problems in exact arithmetic. With the framework of reduced-Hessians this class provides a dynamical framework for the construction of limited-memory quasi-Newton methods. We give an indication of the performance of the methods within this framework by showing numerical simulations on sequences of related systems of linear equations, which originate from the CUTEst test collection.

In addition, we give a compact representation of the Hessian approximations in the full Broyden class for the general unconstrained optimization problem. This representation consists of explicit matrices and gradients only as vector components.

Keywords: method of conjugate gradients, quasi-Newton method, unconstrained quadratic program, limited-memory method, exact linesearch method.

1 Introduction

In this work we mainly study the behavior of limited-memory quasi-Newton methods on unconstrained quadratic optimization problems on the form

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^T H x + c^T x,$$

(QP)

where $H = H^T$ and $H > 0$. (Throughout, “$>$” is used to denote positive definite and we denote the corresponding set of matrices by $S^+_n$, i.e. $S^+_n = \{ B_0 \in \mathbb{R}^{n \times n} : B_0 = B_0^T$ and $B_0 > 0 \}$.) In particular, we study exact linesearch limited-memory quasi-Newton methods that generate search directions parallel to those of the method of preconditioned conjugate gradients (PCG). Under exact linesearch parallel search directions imply identical iterates.

The motivation for this work originates from applications in which it is desired to solve or approximately solve a sequence of related systems of linear equations. In particular systems where the matrix is symmetric positive definite. Such sequences occur when solving unconstrained nonlinear optimization problems with Newton’s

*Optimization and Systems Theory, Department of Mathematics, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden (daviek@kth.se, andersf@kth.se).
Research supported by the Swedish Research Council (VR).
method and in interior-point methods, which constitute some of the most widely used methods in numerical optimization. As the problems become larger the arising systems of linear equations typically become increasingly computationally expensive to solve and iterative methods may be considered. In exact arithmetic, our model method is the method of preconditioned conjugate gradients, but this method may be too inaccurate in finite precision. Quasi-Newton methods may be expected to be significantly more accurate, but the computational cost is typically too high. In consequence, we aim for less computationally expensive limited-memory versions of quasi-Newton methods that are more accurate than the method of preconditioned conjugate gradients. The goal is to provide better understanding of whether it is viable and/or efficient to use such methods to solve or approximately solve systems of linear equations that arise as Newton’s method or interior-point methods converge.

We envisage the use of the limited-memory quasi-Newton methods as an accelerator for a direct solver when solving a sequence of systems of linear equations. E.g., when the direct solver and the iterative solver can be run in parallel, and where the preconditioner is updated when the direct solver is faster for some system of linear equations in the sequence.

We mainly propose a framework for the construction of reduced-Hessian limited-memory quasi-Newton methods. To give an indication of their potential we construct two examples in this framework for which we give numerical results.

Limited-memory quasi-Newton methods have previously been studied by various authors, e.g., as memory-less quasi-Newton methods by Shanno [23], limited-memory BFGS (L-BFGS) by Nocedal [18] and more recently as limited-memory reduced-Hessian methods by Gill and Leonard [13]. In contrast, we specialize to exact linesearch methods for problems on the form (QP). The model method is PCG, which is interpreted as a particular quasi-Newton method as is done by e.g., Shanno [23] and Forsgren and Odland [11]. We start from a result by Forsgren and Odland [11], which provides necessary and sufficient conditions on the Hessian approximation for exact linesearch methods on (QP) to generate search directions that are parallel to those of PCG. The focus is henceforth directly on Hessian approximations with this property. The approximations are described by a novel compact representation which contains explicit matrices together with gradients and search directions as vector components. The framework for the compact representation is first given for the full Broyden class where we consider unconstrained optimization problems on the form

\[ \min_{x \in \mathbb{R}^n} f(x), \]  

(P)

where the function \( f : \mathbb{R}^n \to \mathbb{R} \) is assumed to be smooth. Compact representations of quasi-Newton matrices have previously been used by various authors but were first introduced by Byrd, Nocedal and Schnabel [1]. They were thereafter extended to the convex Broyden class by Erway and Marcia [6,7], and to the full Broyden class by DeGuchy, Erway and Marcia [3]. In contrast, we give an alternative compact representation of the Hessian approximations in the full Broyden class which only
contains explicit matrices and gradients as vector components. In addition we
discuss how exact linesearch is reflected in this representation.

Compact representations of limited-memory Hessian approximations in the Broy-
den class are also discussed by Byrd, Nocedal and Schnabel [1] and Erway and Mar-
cia [7]. In contrast, our discussion is on limited-memory representations of Hessian
approximations intended for exact linesearch methods for problems on the form
(QP), and the approximations are not restricted to the Broyden class. In addition,
our alternative representation provides a dynamical framework for the construction
of limited-memory approximations for the mentioned purpose.

In Section 2 we provide a brief background to quasi-Newton methods, uncon-
strained quadratic optimization problems (QP) and to the groundwork that pro-
vides the basis for this study. Section 3 contains the alternative compact represen-
tation for the full Broyden class. In Section 4 we present results which include a
class of limited-memory Hessian approximations together with a discussion of how
to solve the systems of linear equations that arise using reduced-Hessian methods.
Section 5 contains numerical results on randomly generated problems on the form
(QP) and on systems of linear equations which originate from the CUTEst test
collection [5]. Finally in Section 6 we give some concluding remarks.

1.1 Notation
Throughout, \( \mathcal{R}(M) \) and \( \mathcal{N}(M) \) denote the range and the nullspace of a matrix \( M \)
respectively. Moreover, \( e_i \) denotes the \( i \)th unit vector of the appropriate dimension
and \( |S| \) denotes the cardinality of a set \( S \).

2 Background

In this section we give a short introduction to quasi-Newton methods for uncon-
strained optimization problems on the form \( (P) \). Thereafter, we give a background
to unconstrained quadratic optimization problems (QP) and to the groundwork
that provides the basis for this study.

2.1 Background on quasi-Newton methods

Quasi-Newton methods were first introduced as variable metric methods by Davi-
don [2] and later formalized by Fletcher and Powell [10]. For a thorough intro-
duction to quasi-Newton methods see, e.g., [8, Chapter 3] and [19, Chapter 6]. In
quasi-Newton methods the search direction, \( p_k \), at iteration \( k \) is generated by

\[
B_k p_k = -g_k, \tag{1}
\]

where \( B_k \) is an approximation of the true Hessian \( \nabla^2 f(x_k) \) and \( g_k \) is the gradient
\( \nabla f(x_k) \). The symmetric two-parameter class of Huang [17] satisfies the scaled
secant condition
\[ B_k s_{k-1} = \sigma_k y_{k-1}, \]  
(2)

where \( s_{k-1} = x_k - x_{k-1}, y_{k-1} = g_k - g_{k-1} \) and \( \sigma_k \) is one of the free parameters. The most well-known quasi-Newton class is obtained if \( \sigma_k = 1 \) in (2), namely the one-parameter Broyden class which updates \( B_{k-1} \) to
\[
B_k = B_{k-1} - \frac{1}{s_{k-1}^TB_{k-1}s_{k-1}} B_{k-1}s_{k-1}s_{k-1}^TB_{k-1} + \frac{1}{y_{k-1}s_{k-1}} y_{k-1}y_{k-1}^T + \phi_{k-1} \omega_{k-1} \omega_{k-1}^T,
\]
(3)

where
\[
\omega_{k-1} = (s_{k-1}^TB_{k-1}s_{k-1})^{1/2} \left( \frac{1}{y_{k-1}s_{k-1}} y_{k-1} - \frac{1}{s_{k-1}s_{k-1}B_{k-1}s_{k-1}} s_{k-1} \right),
\]

with \( \phi_{k-1} \) as the free parameter [9]. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) update scheme is obtained if \( \phi_{k-1} = 0 \) and Davidon-Fletcher-Powell (DFP) if \( \phi_{k-1} = 1 \). In this work we study Hessian approximations described by compact representations with gradients and search directions as vector components. We will therefore throughout this work explicitly use the quantities \( g, p \) and the steplength \( \alpha \) in all equations. In this notation, the Broyden class Hessian approximations in (3) may be written as
\[
B_k = B_{k-1} + \frac{1}{g_{k-1}p_{k-1}} g_{k-1}g_{k-1}^T + \frac{1}{\alpha_{k-1} (g_k - g_{k-1})^T p_{k-1}} (g_k - g_{k-1}) (g_k - g_{k-1})^T + \phi_{k-1} \omega_{k-1} \omega_{k-1}^T,
\]
(4)

where
\[
\omega_{k-1} = (-g_{k-1}^T p_{k-1})^{1/2} \left( \frac{1}{(g_k - g_{k-1})^T p_{k-1}} (g_k - g_{k-1}) - \frac{1}{g_{k-1}^T p_{k-1}} g_{k-1} \right).
\]

As shown in (4), the previous Hessian approximation is in general updated by a rank-two matrix with range equal to the space spanned by the current and the previous gradient. Furthermore, it is well known that under exact linesearch all Broyden class updates generates identical iterates, as shown by Dixon [4]. The case \( \phi_{k-1} = 0 \) in (4), i.e., the BFGS update, will have a particular role in part of our analysis. We will refer to quantities \( B_k, p_k \) and \( \alpha_k \) corresponding to this case as \( B_k^{BFGS}, p_k^{BFGS} \) and \( \alpha_k^{BFGS} \).

2.2 Background on quadratic problems

Solving (QP) is equivalent to solving the linear system
\[
H x + c = 0,
\]
(5)

4
Limited-memory quasi-Newton methods for a quadratic function

which has a unique solution if \( H > 0 \). The problem (QP), and hence (5), is in this work solved by an exact linesearch method on the following form. The steplength, iterate and gradient at iteration \( k \) is updated as

\[
\alpha_k = -\frac{g_k^T p_k}{p_k^T H p_k}, \quad x_{k+1} = x_k + \alpha_k p_k, \quad g_{k+1} = g_k + \alpha_k H p_k,
\]

which together with a specific formula for \( p_k \) constitute the particular exact linesearch method. The model method is summarized in the Algorithm 1 below.

**Algorithm 1** An exact linesearch method for solving (QP).

\[
k \leftarrow 0, \quad x_k \leftarrow \text{Initial point}, \quad g_k \leftarrow H x_k + c
\]

While \( \|g_k\| \neq 0 \) do

\[
p_k \leftarrow \text{search direction}
\]

\[
\alpha_k \leftarrow -\frac{g_k^T p_k}{p_k^T H p_k}
\]

\[
x_{k+1} \leftarrow x_k + \alpha_k p_k
\]

\[
g_{k+1} \leftarrow g_k + \alpha_k H p_k
\]

\[
k \leftarrow k + 1
\]

End

The search direction in Algorithm 1 may be calculated using PCG with a symmetric positive definite preconditioner \( M \). The corresponding algorithm for solving (5) can be formulated using the Cholesky factor \( L \) defined by \( M = LL^T \). This is equivalent to application of the method of conjugate gradients (CG) to the preconditioned linear system

\[
L^{-1} H L^{-T} \hat{x} + L^{-1} c = 0,
\]

with \( \hat{x} = L^T x \), see, e.g., Saad [22, Chapter 9.2]. If all quantities generated by CG on (6) are denoted by \( \cdot \hat{\cdot} \), then these quantities will relate to those from CG on (5) as, \( \hat{g} = L^{-1} g \) and \( \hat{p} = L^T p \). The iteration space when \( M = I \) or when \( M \) is an arbitrary symmetric positive definite matrix will thus be related through a linear transformation. In this work the following PCG update is considered,

\[
p_{P CG}^k = \begin{cases} 
-M^{-1} g_0 & k = 0, \\
-M^{-1} g_k + \frac{g_k^T M^{-1} g_k}{g_{k-1}^T M^{-1} g_{k-1}} p_{k-1} & k \geq 1.
\end{cases}
\]

The discussion in this work is mainly on Hessian approximations \( B_k \) that generate \( p_k \) parallel to \( p_{P CG}^k \). We will therefore hereinafter only consider the preconditioner \( M = B_0 \) where \( B_0 \in S^n_+ \). If no preconditioner is used, i.e. \( B_0 = I \), then (7) is the update referred to as Fletcher-Reeves, which together with the exact linesearch method of Algorithm 1 is equivalent to the method of conjugate gradients by Hestenes and Stiefel [16]. If the search direction (7) is used in Algorithm 1, the method terminates when \( \|g_r\| = 0 \) for some \( r \) where \( r \leq n \) and \( x_r \) solves (QP). The search directions generated by the method are mutually conjugate with respect to \( H \) and satisfy \( p_i \in \text{span}(\{B_0^{-1} g_0, \ldots, B_0^{-1} g_r\}) \), \( i = 0, \ldots, r \). In addition,
it holds that the generated gradients are mutually conjugate with respect to $B_0^{-1}$, i.e. $g_i^T B_0^{-1} g_j = 0$, $i \neq j$. By expanding (7), the search direction of PCG may be expressed as

$$p_k^{PCG} = -g_k^T B_0^{-1} g_k \sum_{i=0}^{k-1} \frac{1}{g_i^T B_0^{-1} g_i} B_0^{-1} g_i. \quad (8)$$

Forsgren and Odland [11] have provided necessary and sufficient conditions on $B_k$ for an exact linesearch method to generate $p_k$ parallel to $p_k^{PCG}$. This result provides the basis of our work and is therefore reviewed. Under exact linesearch and $B_0 \in S_+^n$ it holds that $g_k^T p_k^{PCG} = -g_k^T B_0^{-1} g_k$. With $p_{k-1}^{PCG} = p_{k-1}/\delta_{k-1}$, (7) can be written as

$$p_k^{PCG} = -C_k^{-1} B_0^{-1} g_k, \quad (9)$$

where

$$C_k^{-1} = I + \frac{1}{g_k^T \! p_{k-1} g_k}, \text{which imply } C_k = I - \frac{1}{g_k^T \! p_{k-1} g_k}. \quad (10)$$

Insertion of $p_k^{PCG} = p_k/\delta_k = -B_k^{-1} g_k/\delta_k$ into (9) gives

$$B_k C_k^{-1} B_0^{-1} g_k = \frac{1}{\delta_k} g_k. \quad (11)$$

Premultiplication by $C_k^{-T}$ while noting that $C_k^{-T} g_k = g_k$ and letting $W_k = C_k^{-T} B_k C_k^{-1}$ yield

$$B_k = C_k^{-T} W_k C_k, \text{ with } W_k B_0^{-1} g_k = (1/\delta_k) g_k, \quad (12)$$

for $W_k$ nonsingular. Finally, it holds that $B_k \succ 0$ if and only if $W_k \succ 0$. For further details, see [11, Proposition 4].

With the exact linesearch method of Algorithm 1 for solving (QP), parallel search directions imply identical iterates, and therefore search directions parallel to those of PCG imply finite termination. Huang has shown that the quasi-Newton Huang class, the Broyden class and PCG generate parallel search directions [17].

Finally we review a result which is related to the conjugacy of the search directions. Part of the result is similar to those given by Fletcher in [8]. The result will have a central part the analysis to come.

**Lemma 2.1.** Consider iteration $k$, $1 \leq k < r$, of the exact linesearch method of Algorithm 1 for solving (QP). For $B_0 \in S_+^n$ and $p_i = \delta_i p_i^{PCG}$, $\delta_i \neq 0$, $i = 0, \ldots, k-1$, then $p_k = \delta_k p_k^{PCG}$, $\delta_k \neq 0$ if and only if

$$g_i^T p_k = c_k \neq 0, \quad i = 0, \ldots, k, \quad (13)$$

with $c_k = -\delta_k g_k^T B_0^{-1} g_k$ and $p_k \in \text{span}\{B_0^{-1} g_0, \ldots, B_0^{-1} g_k\}$. 

6
Proof. Note that by the assumptions, \( g_i, i = 0, \ldots, k \), are identical to those generated by PCG. We first show the only-if direction. Premultiplication of \( p_k^{PCG} \) in (8) by \( g_i^T \) while taking into account the conjugacy of the \( g_j \)’s with respect to \( B_0^{-1} \) gives \( g_i^T p_k^{PCG} = -g_k^T B_0^{-1} g_k \), so that \( g_i^T (\delta_k p_k^{PCG}) = c_k \) for \( c_k = -\delta_k g_k^T B_0^{-1} g_k \). In addition, (8) shows that \( p_k \in \text{span} \{ B_0^{-1} g_0, \ldots, B_0^{-1} g_k \} \). To show the other direction, let

\[
p_k = \sum_{j=0}^{k} \gamma_j B_0^{-1} g_j,
\]

Premultiplication of (14) by \( g_i^T \) while taking into account the conjugacy of the \( g_j \)’s with respect to \( B_0^{-1} g_i \) gives \( g_i^T p_k = \gamma_i g_i^T B_0^{-1} g_i \), \( i = 0, \ldots, k \), hence if \( p_k \) satisfies (13) with \( c_k = -\delta_k g_k^T B_0^{-1} g_k \), then it follows that

\[
\gamma_i = -\delta_k g_k^T B_0^{-1} g_k / g_i^T B_0^{-1} g_i, \quad i = 0, \ldots, k.
\]

Insertion of (15) into (14) gives \( p_k = \delta_k p_k^{PCG} \), with \( p_k^{PCG} \) given by (8).

For further details on methods of conjugate gradients and related analyses, see e.g. [21].

3 A compact representation of Broyden class Hessian approximations

In this section we consider unconstrained optimization problems on the form (P) and give a compact representation of the Hessian approximations in the full Broyden class. The representation contains only explicit matrices and gradients as vector components. In this section, the gradient of \( f \) in (P) will be denoted by \( g \), in contrast to all other sections where \( g \) refers to the gradient of the objective function of (QP). We first give the general representation without exact linesearch and then discuss how exact linesearch is reflected in the representation.

Lemma 3.1. Consider iteration \( k \) of solving (P) by a quasi-Newton method. Let \( B_0 \) be a given nonsingular matrix. Assume that \( p_i, i = 0, \ldots, k - 1 \), has been given by \( B_i p_i = -g_i \) with \( B_i, i = 1, \ldots, k - 1 \), as any nonsingular matrix on the form (4). Any Hessian approximation in the Broyden class can then be written as

\[
B_k = B_0 + \sum_{i=0}^{k-1} \left[ \frac{1}{g_i^T p_i} g_i g_i^T + \frac{1}{\alpha_i (g_{i+1} - g_i)^T p_i} (g_{i+1} - g_i)(g_{i+1} - g_i)^T + \phi_i \omega_i \omega_i^T \right]
\]

where

\[
\omega_i = \left( -g_i^T p_i \right)^{1/2} \left( \frac{1}{(g_{i+1} - g_i)^T p_i} (g_{i+1} - g_i) - \frac{1}{g_i^T p_i} g_i \right),
\]

or equivalently

\[
B_k = B_0 + G_k T_k G_k^T,
\]
where $G_k = [g_0 \ g_1 \ \ldots \ g_{k-1} \ g_k] \in \mathbb{R}^{n \times (k+1)}$ and $T_k \in \mathbb{R}^{(k+1) \times (k+1)}$ is a symmetric tridiagonal matrix on the form $T_k = T_k^C + T_k^\phi$, with

$$e_1^T T_k^C e_1 = \frac{1}{g_0^T p_0} + \frac{1}{\alpha_0 (g_1 - g_0)^T p_0}, \quad (17a)$$

$$e_{i+1}^T T_k^C e_{i+1} = \frac{1}{g_i^T p_i} + \frac{1}{\alpha_{i-1} (g_i - g_{i-1})^T p_{i-1}} + \frac{1}{\alpha_i (g_{i+1} - g_i)^T p_i}, \quad i = 1, \ldots, k - 1, \quad (17b)$$

$$e_{i+1}^T T_k^C e_i = -e_i^T T_k^C e_{i+1} = -\frac{1}{\alpha_{i-1} (g_i - g_{i-1})^T p_{i-1}}, \quad i = 1, \ldots, k, \quad (17c)$$

$$e_{k+1}^T T_k^C e_k = -\frac{1}{\alpha_{k-1} (g_k - g_{k-1})^T p_{k-1}}, \quad (17d)$$

$$e_1^T T_k^\phi e_1 = -\phi_0 g_0^T p_0 \left( \frac{1}{(g_1 - g_0)^T p_0} + \frac{1}{g_0^T p_0} \right)^2, \quad (17e)$$

$$e_{i+1}^T T_k^\phi e_{i+1} = -\phi_{i-1} g_{i-1}^T p_{i-1} \left( \frac{1}{(g_i - g_{i-1})^T p_{i-1}} \right)^2 - \phi_i g_i^T p_i \left( \frac{1}{(g_{i+1} - g_i)^T p_i} + \frac{1}{g_i^T p_i} \right)^2, \quad i = 1, \ldots, k - 1, \quad (17f)$$

$$e_{i+1}^T T_k^\phi e_i = e_i^T T_k^\phi e_{i+1} = \phi_{i-1} \frac{g_{i-1}^T p_{i-1}}{((g_i - g_{i-1})^T p_{i-1})^2} + \frac{1}{(g_i - g_{i-1})^T p_{i-1}}, \quad i = 1, \ldots, k, \quad (17g)$$

$$e_{k+1}^T T_k^\phi e_k = -\phi_{k-1} g_{k-1}^T p_{k-1} \left( \frac{1}{(g_k - g_{k-1})^T p_{k-1}} \right)^2. \quad (17h)$$

**Proof.** The result follows directly from telescoping (4) and writing it on outer product form. \qed

The compact representation in Lemma 3.1 requires storage of $(k+1)$ gradient vectors and an explicit component matrix, $T_k$, of size $(k+1) \times (k+1)$. In comparison to compact representations given in [1], [6] and [7] which require storage of $2k$ vector-pairs $(B_0 s_i, y_i)$, $i = 0, \ldots, k - 1$, and an implicit $2k \times 2k$ component matrix. An expression for the inverse of the proposed representation (16) can be obtained with Sherman-Morrison-Woodbury formula [14].

One of the most commonly used quasi-Newton update schemes is the BFGS
update. For \( \phi_i = 0, i = 1, \ldots, k - 1 \), it follows from Lemma 3.1 that

\[
B_{k}^{\text{BFGS}} = B_0 + \sum_{i=0}^{k-1} \left[ \frac{1}{g_i^T p_i} g_i g_i^T + \frac{1}{\alpha_i (g_{i+1} - g_i) p_i} (g_{i+1} - g_i) (g_{i+1} - g_i)^T \right],
\]

or equivalently

\[
B_{k}^{\text{BFGS}} = B_0 + G_k T_k^{\text{BFGS}} G_k^T,
\]

where \( G_k = \begin{bmatrix} g_0 & g_1 & \cdots & g_k \end{bmatrix} \in \mathbb{R}^{n \times (k+1)} \) and \( T_k^{\text{BFGS}} \in \mathbb{R}^{(k+1) \times (k+1)} \) is a symmetric tridiagonal matrix with elements given in (17a)-(17d).

Under exact linesearch the step length is chosen such that \( g_i^T p_{i-1} = 0 \), \( i = 1, \ldots, k \). All non-diagonal terms of \( T_k^{\phi} \) become zero and the diagonal terms may be simplified to

\[
\phi_{k-1} \omega_{k-1} \omega_{k-1}^T = -\frac{\phi_{k-1}}{g_{k-1}^T p_{k-1}} g_k g_k^T.
\]

The choice of Broyden member is thus only reflected in the diagonal of \( T_k \) in Lemma 3.1. This can be observed directly in (17e) - (17h) by making use of the exact linesearch condition \( g_i^T p_{i-1} = 0 \), \( i = 1, \ldots, k \). All non-diagonal terms of \( T_k^{\phi} \) become zero and the diagonal terms may be simplified to

\[
e_i^T T_k^{\phi} e_{i+1} = \begin{cases} 0 & i = 0, \\ -\frac{\phi_{i-1}}{g_{i-1}^T p_{i-1}} & i = 1, \ldots, k. \end{cases}
\]

Any Hessian approximation in the Broyden class may in fact be written as \( B_k = B_k^{\text{BFGS}} - (\phi_{k-1} / g_{k-1}^T p_{k-1}) g_k g_k^T \). In consequence, \( B_k \) is independent of \( \phi_i \) for \( i = 0, \ldots, k - 2 \) and the choice of Broyden member only affects the scaling of the search direction. This property follows solely from exact linesearch, in comparison to the properties that stem from exact linesearch on quadratic optimization problems (QP), which are discussed in Section 4.

## 4 Quadratic problems

In this section we consider quadratic problems on the form (QP) and start from the requirement that \( p_k \) generated by the exact linesearch method of Algorithm 1 shall be parallel to \( p_k^{\text{PCG}} \). Motivated by the performance of the Broyden class, we start by considering Hessian approximations \( B_k = B_{k-1} + U_k \) where \( U_k \) is a symmetric rank-two matrix with \( \mathcal{R}(U_k) = \text{span} \{ g_{k-1}, g_k \} \) and thereafter look at generalizations. A characterization of all such update matrices \( U_k \) is provided as well as a multi-parameter Hessian approximation that generates \( p_k = \delta_k p_k^{\text{PCG}} \) for nonzero scalar \( \delta_k \). Thereafter, we consider limited-memory Hessian approximations with this property, discuss potential extensions and how to solve the arising systems with a reduced-Hessian method.
Proposition 4.1. Consider iteration \( k, 1 \leq k < r \), of the exact linesearch method of Algorithm 1 for solving \((QP)\). Assume that \( p_i = \delta_i p_i^{PCG} \) with \( \delta_i \neq 0 \) for \( i = 0, \ldots, k-1 \), where \( p_i^{PCG} \) is the search direction of PCG with associated \( B_0 \in S^n_+ \), as stated in (7). Let \( B_{k-1} \) be a nonsingular matrix such that \( B_{k-1}g_{k-1} = -g_k \) and \( B_{k-1}B_0^{-1}g_k = g_k \). Moreover, let \( U_k = B_k - B_{k-1} \) and assume that \( B_k \) and \( p_k \) satisfy \( B_kp_k = -g_k \) with \( B_k \) nonsingular. Then, if \( U_k \) is symmetric, rank-two with \( \mathcal{R}(U_k) = \text{span}\{g_{k-1}, g_k\} \) it holds that \( p_k = \delta_k p_k^{PCG}, \delta_k \neq 0 \), if and only if

\[
U_k = \frac{1}{g_{k-1}^T p_{k-1}} g_{k-1} g_{k-1}^T + \rho_{k-1} (g_k - g_{k-1})(g_k - g_{k-1})^T + \frac{1}{g_k^T B_0^{-1} g_k} g_k g_k^T,
\]

where \( \rho_{k-1} \) is a free parameter.

Proof. As stated in [11, Proposition 5], the assumptions in the proposition together with (11) and \( B_k = B_{k-1} + U_k \) give the following necessary and sufficient condition on \( U_k \) such that \( p_k = \delta_k p_k^{PCG} \) for a scalar \( \delta_k \neq 0 \),

\[
U_k \left( B_0^{-1} g_k + \frac{g_k^T B_0^{-1} g_k}{p_{k-1}^T g_{k-1}} p_{k-1} \right) = \left( \frac{1}{\delta_k} - 1 \right) g_k + \frac{g_k^T B_0^{-1} g_k}{p_{k-1}^T g_{k-1}} g_{k-1}. \tag{18}
\]

Any symmetric rank-two matrix, \( U_k \), with \( \mathcal{R}(U_k) = \text{span}\{g_{k-1}, g_k\} \) can be written as

\[
U_k = \begin{pmatrix} g_{k-1} & g_k \end{pmatrix} \begin{pmatrix} \eta_{k-1} + \rho_{k-1} & -\rho_{k-1} \\ -\rho_{k-1} & \varphi_k + \rho_{k-1} \end{pmatrix} \begin{pmatrix} g_k^T \\ g_k \end{pmatrix}, \tag{19}
\]

for parameters \( \eta_{k-1}, \rho_{k-1} \) and \( \varphi_k \). Insertion of (19) into (18), taking into account \( g_k^T B_0^{-1} g_k = 0 \) and \( g_k^T p_{k-1} = 0 \) gives

\[
\varphi_k g_k^T B_0^{-1} g_k g_k + \eta_{k-1} g_k^T B_0^{-1} g_{k-1} = \left( \frac{1}{\delta_k} - 1 \right) g_k + \frac{g_k^T B_0^{-1} g_k}{p_{k-1}^T g_{k-1}} g_{k-1},
\]

which is independent of \( \rho_{k-1} \). Identification of coefficients for \( g_k \) and \( g_{k-1} \) respectively gives

\[
\varphi_k = \left( \frac{1}{\delta_k} - 1 \right) \frac{1}{g_k^T B_0^{-1} g_k}, \quad \text{and} \quad \eta_{k-1} = \frac{1}{g_{k-1}^T p_{k-1}}.
\]

Insertion of \( \varphi_k \) and \( \eta_{k-1} \) into (19) gives \( U_k \) as stated in the lemma, with \( \rho_{k-1} \) free.

The result in Proposition 4.1 provides a two-parameter update matrix, \( U_k \). If the conditions of Proposition 4.1 apply then it follows directly from \( U_k \) that the iterates satisfy the scaled secant condition (2). This can be seen by considering \( B_k \alpha_{k-1} p_{k-1} \) with \( B_k = B_{k-1} + U_k \) which gives

\[
(B_{k-1} + U_k) \alpha_{k-1} p_{k-1} = -\rho_{k-1} \alpha_{k-1} g_{k-1}^T p_{k-1} (g_k - g_{k-1}).
\]
Consequently the characterization in Proposition 4.1 provides a class which under exact linesearch is equivalent to the symmetric Huang class. The scaling in the secant condition does neither affect the search direction nor its scaling. Utilizing the secant condition sets the parameter \( \rho_{k-1} = -1/(\alpha_{k-1}g_k^T p_{k-1}) \), that together with the change of variable

\[
\varphi_k = \left( \frac{1}{\delta_k} - 1 \right) \frac{1}{g_k^T B_0^{-1} g_k} = -\frac{\phi_{k-1}}{g_k^T p_{k-1}},
\]

(20)
gives the exact linesearch form of the Broyden class matrices in (4). Hence, as expected, utilizing the secant condition fixes one of the parameters and gives the Broyden class.

The result of Proposition 4.1 motivates further study of the structure in the corresponding Hessian approximations.

**Lemma 4.2.** Consider iteration \( k, 1 \leq k < r \), of the exact linesearch method of Algorithm 1 for solving (QP). Assume that \( B_i p_i = -g_i, i = 0, \ldots, k-1 \), where \( B_0 \in S^+_n \) and

\[
B_i = B_{i-1} + \frac{1}{g_{i-1}^T p_{i-1}} g_{i-1} g_{i-1}^T + \rho_{i-1} (g_i - g_{i-1}) (g_i - g_{i-1})^T + \varphi_i g_i g_i^T, \quad i = 1, \ldots, k,
\]

(21)
with \( \rho_{i-1} \) and \( \varphi_i \) chosen such that \( B_i \) is nonsingular. Then \( B_k \) takes the form

\[
B_k = B_0 + \sum_{i=0}^{k-1} \left(-\frac{1}{g_i^T B_0^{-1} g_i} g_i g_i^T + \rho_i (g_{i+1} - g_i)(g_{i+1} - g_i)^T\right) + \varphi_k g_k g_k^T.
\]

(22)

**Proof.** With the assumptions in the proposition, the update of (21) satisfies the requirements of Proposition 4.1 and hence for each \( i, i = 0, \ldots, k-1 \), it follows that \( p_i = \delta_i p_i^{PCG} \) where \( \delta_i = 1/\left(1 + \varphi_i g_i^T B_0^{-1} g_i\right) \) and \( \varphi_0 = 0 \). Premultiplication of \( p_i = \delta_i p_i^{PCG} \) by \( g_i^T \) gives

\[
g_i^T p_i = \frac{1}{1 + \varphi_i g_i^T B_0^{-1} g_i} g_i^T p_i^{PCG}, \quad i = 0, \ldots, k-1.
\]

(23)

Inverting (23) and taking into account that \( g_i^T p_i^{PCG} = -g_i^T B_0^{-1} g_i, i = 0, \ldots, k-1 \), gives

\[
\frac{1}{g_i^T p_i} + \varphi_i = -\frac{1}{g_i^T B_0^{-1} g_i}, \quad i = 0, \ldots, k-1.
\]

(24)

By telescoping (21) at iteration \( k \) we obtain

\[
B_k = B_0 + \sum_{i=0}^{k-1} \left(\left(\frac{1}{g_i^T p_i} + \varphi_i\right) g_i g_i^T + \rho_i (g_{i+1} - g_i)(g_{i+1} - g_i)^T\right) + \varphi_k g_k g_k^T.
\]

(25)

Insertion of (24) into (25) gives (22). □
Lemma 4.2 and (22) show that if \( p_k \) is given by (1) with \( B_k \) as in (22), then \( p_k \) is independent of all \( \rho_i, i = 0, \ldots, k - 1 \), as long as \( B_k \) is nonsingular. This result is formalized in the following proposition.

**Proposition 4.3.** Consider iteration \( k, 1 \leq k < r \), of the exact linesearch method of Algorithm 1 for solving (QP). Assume that \( p_i = \delta_i p_i^{PCG} \) with \( \delta_i \neq 0 \) for \( i = 0, \ldots, k - 1 \), where \( p_i^{PCG} \) is the search directions of PCG with associated \( B_0 \in S_+ \), as stated in (7). Let \( p_k \) satisfy \( B_k p_k = -g_k \) where

\[
B_k = B_0 + \sum_{i=0}^{k-1} \left( \frac{-1}{g_i^T B_0^{-1} g_i} g_i g_i^T + \rho_i^{(k)} (g_{i+1} - g_i)(g_{i+1} - g_i)^T \right) + \varphi_k g_k g_k^T, \tag{26}
\]

with \( \rho_i^{(k)}, i = 0, \ldots, k - 1 \), and \( \varphi_k \) chosen such that \( B_k \) is nonsingular. Then,

\[
p_k = \frac{1}{1 + \varphi_k g_k^T B_0^{-1} g_k} p_k^{PCG}.
\]

In particular, if \( \rho_i^{(k)} > 0, i = 0, \ldots, k - 1 \), and \( \varphi_k > -1/(g_k^T B_0^{-1} g_k) \), then \( B_k \succ 0 \).

**Proof.** From Proposition 4.1 and Lemma 4.2 it follows that \( B_k \) given by (22) generates \( p_k = \delta_k p_k^{PCG} \) where \( \delta_k = 1/(1 + \varphi_k g_k^T B_0^{-1} g_k) \) and hence satisfies

\[
(g_{i+1} - g_i)^T p_k = 0, \quad i = 0, \ldots, k - 1,
\]

by Lemma 2.1. If \( \rho_i, i = 0, \ldots, k - 1 \), and \( \varphi_k \) chosen such that \( B_k \) is nonsingular then the solution is unique and independent of \( \rho_i, \rho_i = \rho_i^{(k)}, i = 0, \ldots, k - 1 \). Moreover, if \( \rho_i^{(k)} > 0, i = 0, \ldots, k - 1 \) and \( \varphi_k = 0 \) then the matrix of (26) is positive definite by Lemma A.2. It then follows from Lemma A.1 that \( B_k \succ 0 \) for \( \varphi_k > -1/(g_k^T B_0^{-1} g_k) \).

The result in Proposition 4.3 together with the exact linesearch method of Algorithm 1 provide a multiple-parameter class that generates search directions parallel to those of PCG. In the framework of updates on the form \( B_k = B_{k-1} + U_k \) this class allows update matrices with \( \mathcal{R}(U_k) = \text{span}\{g_0, \ldots, g_k\} \) and reduces to the symmetric Huang class if \( \mathcal{R}(U_k) = \text{span}\{g_{k-1}, g_k\} \) is required. The Hessian approximation in (26) of Proposition 4.3 can also be viewed as a matrix composed of three parts

\[
B_k = B_0 + V_k^\varphi + V_k^\rho, \tag{27}
\]

where

\[
V_k^\varphi = \sum_{i=0}^{k-1} \frac{-1}{g_i^T B_0^{-1} g_i} g_i g_i^T + \varphi_k g_k g_k^T,
\]

and

\[
V_k^\rho = \sum_{i=0}^{k-1} \rho_i^{(k)} (g_{i+1} - g_i)(g_{i+1} - g_i)^T. \tag{28}
\]
The direction is determined solely by \( B_0 + V^\varphi_k \), compare with (8). The parameter \( \varphi_k \) gives a scaling and the parameters \( \rho_i^{(k)}, \ i = 0, \ldots, k - 1 \), have no effect on the direction. Certain choices of these parameters merely guarantee nonsingularity and may provide numerical stability.

### 4.1 Limited-memory Hessian approximations

In this section we extend the above discussion to limited-memory Hessian approximations. The goal is to obtain approximations such that (1) gives search directions parallel to those of PCG. From (8) it follows directly that \( p_k^{PCG} \in \text{span}(\{ B_0^{-1}g_0, \ldots, B_0^{-1}g_k \}) \) and that \( p_k^{PCG} \) has a nonzero component in every direction \( B_0^{-1}g_i, \ i = 0, \ldots, k \). In consequence, gradient information can not be discarded if \( B_k \) is on the form of (27) where \( \mathcal{R}(V^\varphi_k + V^\rho_k) = \text{span}(\{ g_0, \ldots, g_k \}) \). As long as \( B_k \) remains nonsingular, gradient information can be discarded from \( V^\rho_k \) but not from the part essential for the direction, namely \( V^\varphi_k \). However, parallel directions can be generated if a specific correction term is added to the right hand side of (1), as is shown in Appendix, Theorem A.3. It can also be done as e.g. in [1], by at each iteration \( k \) recalculating the basis vectors from the \( m \) latest vector pairs \((s_i, y_i)\), \( i = k - m, \ldots, k - 1 \).

In light of the above, the discussion will now be extended to consider Hessian approximations on the form \( B_k = B_0 + V_k + V_k^\rho \) where \( V_k \) is not restricted to have an outer-product form consisting of only gradients. Theorem 4.4 below provides conditions on \( V_k \) and \( V_k^\rho \) such that \( B_k \) is positive definite and generates directions parallel to those of PCG.

**Theorem 4.4.** Consider iteration \( k, \ 1 \leq k < r \), of the exact linesearch method of Algorithm 1 for solving (QP). Assume that \( p_i = \delta_i p_i^{PCG} \) with \( \delta_i \neq 0 \) for \( i = 0, \ldots, k - 1 \), where \( p_i^{PCG} \) is the search direction of PCG with associated \( B_0 \in S^n_+ \), as stated in (7). Let \( p_k \) satisfy \( B_k p_k = -g_k \) with \( B_k = B_0 + V_k + V_k^\rho \) where \( V_k^\rho \) is defined by (28) and \( V_k \) is symmetric such that \( B_0 + V_k \succeq 0 \). If, in addition, \( \rho_i^{(k)} \geq 0, \ i = 0, \ldots, k - 1, \) and \( \mathcal{N}(B_0 + V_k) \cap \mathcal{N}(V_k^\rho) = \emptyset \), then \( B_k \succeq 0 \), and, for \( \delta_k \neq 0 \), it holds that \( p_k = \delta_k p_k^{PCG} \) if and only if

\[
V_k \left( B_0^{-1}g_k + \frac{g_k^T B_0^{-1}g_k}{p_k^{(k)} g_k} p_{k-1} \right) = \left( \frac{1}{\delta_k} - 1 \right) g_k - \frac{g_k^T B_0^{-1}g_k}{p_k^{(k)} g_k} B_0 p_{k-1}. \tag{29}
\]

In particular, (29) with \( \delta_k = 1, \ B_0 + V_k \succeq 0 \) and \( \mathcal{N}(B_0 + V_k) \cap \mathcal{N}(V_k^\rho) = \emptyset \) are satisfied for

\[
V_k = C_k^T B_0 C_k - B_0, \ \text{with} \ C_k \ \text{defined as in (10)}, \tag{30}
\]

and

\[
V_k = -\frac{1}{p_k^{(k)} B_0 p_{k-1}} B_0 p_{k-1} p_k^T B_0, \ \text{with} \ \rho_k^{(k)} > 0. \tag{31}
\]
Proof. The parameters $\rho_i^{(k)} \geq 0$, $i = 0, \ldots, k - 1$ give $V^\rho_k \succeq 0$. It then follows from $\mathcal{N}(B_0 + V_k) \cap \mathcal{N}(V^\rho_k) = \emptyset$ that $B_k \succ 0$. Insertion of $B_k = B_0 + V_k + V^\rho_k$ into (11) gives the if and only if condition

$$(B_0 + V_k + V^\rho_k) \left( B_0^{-1}g_k + \frac{g_k^TB_0^{-1}g_k}{p_{k-1}^Tg_{k-1}}p_{k-1} \right) = \frac{1}{\delta_k} g_k,$$  \quad (32)

for $p_k = \delta_k p_{PCG}^k$, $\delta_k \neq 0$. Using the identity

$$V_k^\rho \left( B_0^{-1}g_k + \frac{g_k^TB_0^{-1}g_k}{p_{k-1}^Tg_{k-1}}p_{k-1} \right) = 0,$$

which follows from Lemma 2.1, in (32) and moving terms corresponding to $B_0$ to the right-hand side gives if and only if condition (29) on $V_k$. In particular, with $B_0 + V_k = C_k^T B_0 C_k$, as given by (30), letting $W_k = B_0$ in (12) gives $p_k = p_{PCG}^k$. In addition, since $C_k$ is nonsingular and $B_0 \succ 0$, it follows that $B_0 + V_k = C_k^T B_0 C_k \succ 0$, so that $B_0 + V_k + V^\rho_k \succ 0$ for $\rho_i^{(k)} \geq 0$, $i = 0, \ldots, k - 1$. The null-space of $B_0 + V_k$ with $V_k$ as in (31) is one-dimensional and spanned by $p_{k-1}$. Since $B_0 \succ 0$, we have $B_0 + V_k \succeq 0$. In addition, if $\rho_i^{(k)} > 0$, then

$$V_k^\rho p_{k-1} = \rho_i^{(k)}(g_k - g_{k-1})^T p_{k-1}(g_k - g_{k-1}) = -\rho_i^{(k)} g_k^T p_{k-1}(g_k - g_{k-1}) \neq 0,$$

since $g_k^T p_{k-1} \succeq 0$ and $p_{k-1} = \delta_k^{-1} p_{PCG}^{k-1}$, with $\delta_k^{-1} \neq 0$, by assumption. Therefore, $B_0 + V_k + V^\rho_k \succ 0$ if $\rho_i^{(k)} > 0$ and $\rho_i^{(k)} \geq 0$, $i = 0, \ldots, k - 2$. Finally, $V_k$ of (31) satisfies (29) for $\delta_k = 1$ since $g_k^T p_{k-1} = 0$, as required. \qed

The result in Theorem 4.4 provides a class of multi-parameter limited-memory Hessian approximations where the memory usage can be changed between iterations. The choices of $V_k$ in (30) and (31) are merely two examples of members in the class. The matrix $V_k$ of (30) is of rank-two, and if used in $B_k = B_0 + V_k + V^\rho_k$, with $\rho_i^{(k)} = 0$, $i = 0, \ldots, k - 1$, then $B_k p_k = -g_k$ may viewed as a symmetric PCG update, compare with (9). The matrix $V_k$ of (31) is the matrix of least rank that satisfies (29) with $\delta_k = 1$. In general, there is little restriction on which iterations to include information from in $V^\rho_k$. Information can thus be included from iterations that are believed to be of importance. All information may also be expressed in terms of search directions and the current gradient $g_k$. This provides the ability to reduce the amount of storage when the arising systems are solved by reduced-Hessian methods, described in Section 4.2, with search directions in the basis.

### 4.2 Solving the systems

In this section we discuss solving systems of linear equations using reduced-Hessian methods. These methods provide an alternative procedure for solving systems
arising in quasi-Newton methods. We follow Gill and Leonard [12, 13] and refer to their work for a thorough introduction.

Assume that a Hessian approximation of Theorem 4.4 is given and used together with the exact linesearch method of Algorithm 1 for solving (QP). The search direction at iteration $k$ then satisfies $p_k = \delta_k p_k^{PCG}$ for a scalar $\delta_k$ and hence by (7) $p_k \in \text{span} \left( \{ p_{k-1}, B_0^{-1}g_k \} \right)$. Define $\Psi_k^\text{min} = \{ p_{k-1}, B_0^{-1}g_k \}$ and let $\Psi_k$ be a subspace such that $\Psi_k^\text{min} \subseteq \Psi_k$. Furthermore let $Q_k$ be a matrix whose columns span $\Psi_k$ and $Z_k$ be the matrix whose columns are the vectors obtained from the Gram-Schmidt process on the columns of $Q_k$. The search direction can then be written as $p_k = Z_k u_k$ for some vector $u_k$. Premultiplication of (1) by $Z_k^T$ together with $p_k = Z_k u_k$ gives

$$Z_k^T B_k Z_k u_k = -Z_k^T g_k,$$

which has a unique solution if $B_k$ is positive definite. Hence $p_k = Z_k u_k$ where $u_k$ satisfies (33). Note that the analogous procedure is also applicable for the result in Appendix, Theorem A.3 where the Hessian approximation is given by (A.6a) and $p_k$ is generated by $B_k p_k = -N_k g_k$. The minimal space required is $\Psi_k = \Psi_k^\text{min}$ but other feasible choices are for example $\Psi_k = \{ B_0^{-1}g_0, \ldots, B_0^{-1}g_k \}$, by (8), or $\Psi_k = \{ p_{t-1}, B_0^{-1}g_t, \ldots, B_0^{-1}g_k \}$ where $0 < t < k$.

### 4.3 Construction of methods and complexity

The Hessian approximations proposed in Theorem 4.4 combined with the reduced-Hessian framework of Section 4.2 provide freedom in the construction of limited-memory quasi-Newton methods. A summary of the quantities that can be chosen is shown in Table 1 below.

| Quantity | Description |
|----------|-------------|
| $V_k$    | Must satisfy the conditions of Theorem 4.4. |
| $\rho_i^{(k)}$, $i = 0, \ldots, k-1$ | Parameter values. |
| $m_k$    | # non-zero $\rho_i^{(k)}$, $i = 0, \ldots, k-1$, of $V_k^\rho$ in (28). |
| $Q_k$    | Provides the space for $p_k$, columns must span $\Psi_k^\text{min}$. |
| $\hat{m}_k$ | # columns of $Q_k$ and dimension of the reduced-Hessian. |

The complexity of the method is essentially determined by the construction and solution of (33). Each iteration $k$ requires a Gram-Schmidt process as well as the construction and factorization of the matrix $Z_k^T (B_0 + V_k + V_k^\rho) Z_k$. If no information is re-used this gives the worst case complexity

$$O(n\hat{m}_k^2 + [(n^2\hat{m}_k + n\hat{m}_k^2) + \text{rank}(V_k)(n\hat{m}_k + \hat{m}_k^2) + (nm\hat{m}_k + m\hat{m}_k^2)] + \hat{m}_k^3),$$

where constants have been omitted. However, the overall complexity can be reduced with particular choices of the quantities in Table 1. We will demonstrate this
by constructing two relatively simple quasi-Newton methods with fixed limited-memory $m = \hat{m} > 3$ for which numerical results are given in Section 5. The methods will be denoted by symPCGs and Vsr1 for which $V_k$ is given by (30) of Theorem 4.4 and (31) respectively. Both use Hessian approximations on the form $B_k = B_0 + V_k + V_\rho^k$ with $V_\rho^k$ as in (28) and parameters defined by

$$
\rho_i^{(k)} = \begin{cases} 
\rho_i^B & i = 0, \ldots, m - 4, k - 3, k - 2, k - 1, \\
0 & i = m - 5, \ldots, k - 4,
\end{cases}
$$

where $\rho_i^B$, $i = 0, \ldots, k - 1$, are the quantities which corresponds to the secant condition. The basis is computed from

$$Q_k = (p_0 \cdots p_{m-4} p_{k-2} p_{k-1} B_0^{-1} g_k).$$

Hence systems of size at most $m$ has to be solved at every iteration with information from the $(m-3)$-first and the 3-latest iterations for $k > m$. In consequence part of the matrices $Z_k$ and $V_\rho^k$ stay constant and the reduced-Hessian may be updated instead of recomputed at every iteration. The computational complexity is then dominated by $\max\{n^2, m^3\}$, see Appendix for motivation. The choice $m = n^{2/3}$ gives complexity $n^2$ which is the same as that of PCG with exact linesearch.

## 5 Numerical results

In this section we first show the convergence behavior of quasi-Newton-like methods and PCG for randomly generated quadratic optimization problems. Thereafter we give performance profiles [5] for the two limited-memory methods described in Section 4.3. As mentioned in Section 4.3, these methods are referred to as symPCGs and Vsr1 respectively. These are also compared to our own Matlab implementations of PCG, with search direction given by (7), BFGS, with search direction given by (3) with $\phi_{k-1} = 0$ for all $k$, and L-BFGS with search direction as proposed by Nocedal in [18]. All methods use exact linesearch, as defined by Algorithm 1, with their particular search direction update. We refer to our implementation of these methods as PCG, BFGS and L-BFGS respectively. The performance profiles are in terms of number of iterations for solving systems of linear equations which originate from the CUTEst test collection [15]. The benchmark problems were initially processed using the Julia packages CUTEst.jl and NLPmodels.jl by Orban and Siqueira [20].

The purpose of the first part is partly to illustrate the difference between quasi-Newton-like methods and PCG to give an indication of the round-off error effects. Convergence for a member in the class of quasi-Newton Hessian approximations in (26) of Proposition 4.3, here denoted by MuP, is shown in Figure 1. The figure also contains the convergence of the BFGS method and PCG in both finite and exact arithmetic, all with $B_0 = I$. In this study we consider exact arithmetic PCG as the original but with 512 digits precision. The parameters of (26) were chosen as
follows, \( \varphi_k = 0 \) for all \( k \) and

\[
\rho_i^{(k)} = \xi_i^{(k)} \rho_i^B, \quad i = 0, \ldots, k - 1,
\]

where \( \xi_i^{(k)} \), \( i = 0, \ldots, k - 1 \), are normally distributed random variables and \( \rho_i^B \), \( i = 0, \ldots, k - 1 \), are the quantities corresponding to the secant condition. Note that the scaling of \( \rho_i^{(k)} \), \( i = 0, \ldots, k - 1 \), are randomly changed for every \( k \).

\[\begin{array}{c}
\end{array}\]

\textbf{Figure 1}: Convergence for solving a randomly generated quadratic problem with 300 variables and condition number in the order of \( 10^4 \). The convergence corresponds to representative results based on 100 simulations with parameters \( \xi_i^{(k)} \in [10^{-1}, 10^8] \), \( i = 0, \ldots, k - 1 \).

The three methods compared in Figure 1 are with the exact linesearch method of Algorithm 1 equivalent in exact arithmetic on (QP). However, in finite arithmetic this is not the case. As can be seen in the figure, PCG suffers from round-off errors while BFGS behaves like the exact arithmetic PCG. The maximum error from all simulations between the iterates of BFGS and exact PCG was \( 5.1 \cdot 10^{-14} \), i.e.

\[
\max_i \| x_{i}^{\text{BFGS}} - x_{i}^{\text{PCG}} \| = 5.1 \cdot 10^{-14}.
\]

Consequently, the BFGS method does not suffer from round-off errors on these randomly generated quadratic problems. By the result of Proposition 4.3 it is not required to fix the parameters \( \rho_i^{(k)} \), \( i = 0, \ldots, k - 1 \), and as Figure 1 shows there is an interval where this result also holds in finite arithmetic. The secant condition is expected to provide an appropriate scaling of the quantities since it gives the true Hessian in \( n \) iterations. Our results indicate that there is no particular benefit for the quadratic case to choose the values given by the secant condition. This freedom may be useful, since values of \( \rho_i^{(k)} \) close to zero for some \( i \) may make the Hessian approximation close to singular, and such values could potentially be avoided.

The purpose of the next part is to give a first indication of the performance of the proposed class of limited-memory quasi-Newton methods for solving a sequence
of related systems of linear equations. The sequences of systems were generated by considering unconstrained nonlinear CUTEst problems of the order $10^3$. For such a problem, a sequence of matrix and right-hand side pairs $\nabla^2 f(x_j)$ and $-\nabla f(x_j)$, $j = 1, \ldots, J$, was accepted if an initial point $x_0$ for which Newton’s method with unit step converged to a point $x_J$ such that $\|\nabla f(x_J)\| \leq 10^{-6}$ and $\lambda_{\min}(\nabla^2 f(x_J)) > 10^{-6}$. In addition, it was required for each $j$ that $\nabla^2 f(x_j)d_j = -\nabla f(x_j)$, was solvable with accuracy at least $10^{-7}$ by the built-in solver in Julia and that $\nabla^2 f(x_j)$ was positive definite. These conditions reduced the test set to 21 problems giving 21 sequences of linear equations, corresponding to 220 systems of linear equations in total. The performance measure in terms of number of iterations is defined as follows. Let $N_{p,s}$ be the number of iterations required by method $s \in S$ on problem $p \in P$. If the method failed to converge within a maximum number of iterations this value is defined as infinity. The measure $P_s(\tau)$ is defined as

$$P_s(\tau) = \frac{|\{p \in P : r_{p,s} \leq \tau\}|}{|P|}, \text{ where } r_{p,s} = \frac{N_{p,s}}{\min\{N_{p,s} : s \in S\}}.$$  

Performance profiles are shown in Figure 2 for SymPCGs, Vsr1, PCG, BFGS, and L-BFGS. All with $B_0 = \nabla^2 f(x_{j-1})$, i.e. when the Newton system at iteration $j$ is preconditioned with the previous Hessian. The figure contains results for $m = [n^{\frac{1}{3}}, n^{\frac{2}{3}}, n^{\frac{4}{3}}]$ and two different tolerances in the stopping criterion. The first corresponds to the criterion $\|g_k\| \leq 10^{-7}$, namely when an approximate solution is desired. The other corresponds to when a more accurate solution is desired and the criterion $\|g_k\| \leq \max\{\epsilon^{DS}, 10^{-13}\}$, where $\epsilon^{DS}$ is the solution accuracy of the built-in direct solver on the preconditioned Newton system with preconditioner $B_0$. The two stopping criteria will be referred to as low and high accuracy respectively in Table 2-3. For the two first $m$-values the overall computational complexity is $O(n^2)$ whereas for the third it is $O(n^{\frac{4}{3}})$, i.e. a case where more computational work is allowed if needed. In Table 3 we show a measure of the mean computational work done for solving systems on the form (33) per iteration. The measure is in terms of the average size of the reduced-Hessian relative to $m$. Moreover, the average number of iterations are shown in Table 2. The maximum number of iterations was set to $5n$ in all simulations.

We also give performance profiles when no regard is taken to the fact that the systems of linear equations within a sequence are related, namely when $B_0 = I$. The corresponding performance profiles and averaged quantities are shown in Figure 3 and Table 2-3 respectively.
Limited-memory quasi-Newton methods for a quadratic function

Figure 2: Performance profiles with $B_0 = \nabla^2 f(x_{j-1})$. The left figure corresponds to stopping criteria $\|g_k\| < 10^{-7}$ and the right to $\|g_k\| \leq \max\{\varepsilon_D S, 10^{-13}\}$.

Table 2: Average number of iterations required per system of linear equations.

| $B_0$ | Accuracy | BFGS | PCG | L-BFGS | SymPCGs | Vsr1 |
|-------|----------|------|-----|--------|---------|------|
| $I$   | low      | 223  | 845 | 842, 753, 704 | 860, 616, 589 | 858, 618, 581 |
|       | high     | 319  | 994 | 997, 930, 912 | 1020, 819, 778 | 1022, 815, 770 |
| $\nabla^2 f$- | low  | 24   | 125 | [127, 37, 30] | [118, 26, 24] | [125, 27, 24] |
|       | high     | 29   | 141 | [135, 45, 35] | [132, 32, 29] | [140, 32, 30] |

Table 3: Average size relative to $m$ of the systems solved per iteration.

| $B_0$ | Accuracy | $n_1^\frac{1}{2}$ | $n_2^\frac{2}{3}$ | $n_3^{21/36}$ | $n_1^\frac{1}{2}$ | $n_2^\frac{2}{3}$ | $n_3^{21/36}$ |
|-------|----------|-----------------|-----------------|----------------|-----------------|-----------------|----------------|
| $I$   | low      | 0.846           | 0.939           | 0.924          | 0.984           | 0.949           | 0.923          |
|       | high     | 0.854           | 0.948           | 0.933          | 0.985           | 0.948           | 0.933          |
| $\nabla^2 f$- | low  | 0.929           | 0.443           | 0.335          | 0.933           | 0.446           | 0.342          |
|       | high     | 0.927           | 0.457           | 0.345          | 0.931           | 0.461           | 0.355          |
The results in Figure 2-3 and Table 2 show that the BFGS method has the best overall performance and PCG the worst. With preconditioner SymPCGs and Vsr1 outperform L-BFGS for the two larger limited memory values $m$. This difference is not as distinct in the non-preconditioned case, as can be seen in Figure 3 even though SymPCGs and Vsr1 are able to solve more problems within the maximum number of iterations compared to L-BFGS. Numerical experiments have shown that, depending on the particular instance, small $m$-values can lead to loss of convergence rate close the solution. Tendencies to this can be seen in Figure 2-3 and Table 2 for the limited-memory methods with $m = n^{\frac{2}{3}}$, which is around 30 for problems of size $10^3$. Further decreasing $m$ also increases this tendency. The memory parameter $m$ might be considered to be large however, as discussed in Appendix in regards to complexity, this value does not have a significant impact on the complexity when $m \leq n^{\frac{2}{3}}$. As Table 3 also shows, most systems solved in the preconditioned case is of size less than $m$. The slight difference between the low and high tolerances in Table 2-3 indicates that the bulk of the work is made prior to reaching low tolerance level. Numerical experiments has further shown that the behavior on a specific system of linear equations not only depends on the choices when designing
the method but also on the individual properties of the linear system. We choose to give the results for the methods denoted by SymPCGs and Vsr1 since we believe that these are representative for the proposed limited-memory quasi-Newton class, given the complexity restrictions, on the test set. However we have observed that the results can be improved and dis-improved for different choices of the quantities in Table 1. We conclude that there is a potential for accelerating the process of solving sequences of related systems of linear equations with iterative methods in the proposed framework.

6 Conclusion

In this work we have given a class of limited-memory quasi-Newton Hessian approximations which on (QP) with the exact linesearch method of Algorithm 1 generate \( p_k \) parallel to \( p_{PCG}^k \). With the framework of reduced-Hessians this provides a dynamical framework for the construction of limited-memory quasi-Newton methods. In addition, we have characterized all symmetric rank-two update matrices, \( U_k \) with \( \mathcal{R}(U_k) = \text{span}(\{g_{k-1}, g_k\}) \) which gives \( p_k \) parallel to \( p_{PCG}^k \) in this setting. The Hessian approximations were described by a novel compact representation whose framework was first presented in Section 3 for the full Broyden class on unconstrained optimization problems (P). The representation of the full Broyden class consists only of explicit matrices and gradients as vector components.

Numerical simulations on randomly generated unconstrained quadratic optimization problems have shown that for these problems our suggested multiparameter class, with parameters within a certain range, is equivalent to the BFGS method in finite arithmetic. Simulations on sequences of related systems of linear equations which originate from the CUTEst test collection have given an indication of the potential of the the proposed class of limited-memory methods. The results indicate that there is a potential for accelerating the process of solving sequences of related systems of linear equations with iterative methods in the proposed framework.

The results of this work are meant to contribute to the theoretical and numerical understanding of limited-memory quasi-Newton methods for minimizing a quadratic function. We envisage that they can lead to further research on limited-memory methods for unconstrained optimization problems. In particular, limited-memory methods for minimizing a near-quadratic function and for systems arising as interior-point methods converge.

Acknowledgements

We thank Elias Jarlebring for valuable discussions on finite precision arithmetic. In addition, we thank the referees and the associate editor for many helpful comments and suggestions which significantly improved the presentation.
Appendix

The following lemma relates a symmetric rank-one update in the matrix to a scaling of the solution when the rank-one vector is a multiple of the right-hand side.

**Lemma A.1.** If $Ax = b$, with $A$ nonsingular then

$$(A + \gamma bb^T) y = b,$$

for $y = \frac{1}{1 + \gamma b^T x} x$, if $1 + \gamma b^T x \neq 0$. If, in addition, $b^T x \neq 0$, it holds that

$$\frac{1}{b^T y} = \frac{1}{b^T x} + \gamma.$$  

(A.1)

Finally, if $A = A^T \succ 0$, then $b^T x > 0$ and $A + \gamma bb^T \succ 0$ if and only if

$$\gamma > -\frac{1}{b^T x}.$$  

Proof. Assume that $Ax = b$ where $A$ is nonsingular. Premultiplication of $(A + \gamma bb^T) y = b$ by $A^{-1}$ gives

$$(I + \gamma A^{-1} bb^T) y = A^{-1} b.$$  

(A.2)

Insertion of $x = A^{-1} b$ into (A.2) and rearranging gives

$$y = (1 - \gamma b^T y) x.$$  

(A.3)

Insertion of $y = \alpha x$ into (A.3) and solving for $\alpha$ yields

$$\alpha = \frac{1}{1 + \gamma b^T x}, \quad 1 + \gamma b^T x \neq 0.$$  

The result in (A.1) follows by premultiplication of $y = \frac{1}{1 + \gamma b^T x} x$ by $b^T$ and rearranging. For the final result, note that $b^T x = x^T A x > 0$ since $A \succ 0$ and that

$$(A + \gamma bb^T) = A^{1/2} \left( I + \gamma A^{-1/2} bb^T A^{-1/2} \right) A^{1/2},$$

which is a congruent transformation and hence $I + \gamma A^{-1/2} bb^T A^{-1/2} \succ 0$ if and only if $A + \gamma bb^T \succ 0$. Then consider the similarity transformation

$$A^{-1/2} \left( I + \gamma A^{-1/2} bb^T A^{-1/2} \right) A^{1/2} = I + \gamma xb^T,$$

where the only eigenvalue not equal to unity is $1 + \gamma b^T x$, which is positive only if $\gamma > -\frac{1}{b^T x}$. \qed

Next, we give a nonsingularity condition on a class of quasi-Newton matrices that we consider.

**Lemma A.2.** Let $B_0 \in S^n_+$ and let $g_i$, $i = 0, \ldots, k$, be nonzero vectors that are conjugate with respect to $B_0^{-1}$. Define $B_k$ as

$$B_k = B_0 + \sum_{i=0}^{k-1} \left( -\frac{1}{g_i^T B_0^{-1} g_i} g_i g_i^T + \rho_i (g_{i+1} - g_i) (g_{i+1} - g_i)^T \right),$$

where $\rho_i \in \mathbb{R}$, $i = 0, \ldots, k - 1$. Then $B_k \succ 0$ if $\rho_i > 0$, $i = 1, \ldots, k - 1$.  

22
Limited-memory quasi-Newton methods for a quadratic function

Proof. Any vector \( p \) in \( \mathbb{R}^n \) can be written as

\[
p = \sum_{i=0}^{k} \alpha_i B_0^{-1} g_i + B_0^{-1} u, \quad \text{with} \quad g_i^T B_0^{-1} u = 0, \quad i = 0, \ldots, k. \tag{A.4}
\]

Insertion of (A.4) into \( p^T B_k p \) gives

\[
p^T B_k p = p^T \left( B_0 + \sum_{i=0}^{k-1} \left( -\frac{1}{g_i^T B_0^{-1} g_i} g_i g_i^T + \rho_i (g_{i+1} - g_i) (g_{i+1} - g_i)^T \right) \right) p
\]

\[
= p^T B_0 p - \sum_{i=0}^{k-1} \frac{(g_i^T p)^2}{g_i^T B_0^{-1} g_i} + \sum_{i=0}^{k-1} \rho_i \left( (g_{i+1} - g_i)^T p \right)^2
\]

\[
= \sum_{i=0}^{k} \alpha_i^2 g_i^T B_0^{-1} g_i + u^T B_0^{-1} u - \sum_{i=0}^{k-1} \frac{(\alpha_i g_i^T B_0^{-1} g_i)^2}{g_i^T B_0^{-1} g_i} + \sum_{i=0}^{k-1} \rho_i \left( (g_{i+1} - g_i)^T p \right)^2
\]

\[
= \alpha_k g_k^T B_0^{-1} g_k + u^T B_0^{-1} u + \sum_{i=0}^{k-1} \rho_i \left( \alpha_{i+1} g_{i+1}^T B_0^{-1} g_{i+1} - \alpha_i g_i^T B_0^{-1} g_i \right)^2.
\]

For the remainder of the proof, let \( \rho_i > 0, i = 0, \ldots, k-1 \). Then \( B_k \) is positive semidefinite with \( p^T B_k p = 0 \) only if

\[
\alpha_k^2 g_k^T B_0^{-1} g_k = 0, \tag{A.5a}
\]

\[
u^T B_0^{-1} u = 0, \tag{A.5b}
\]

\[
\alpha_{i+1} g_{i+1}^T B_0^{-1} g_{i+1} - \alpha_i g_i^T B_0^{-1} g_i = 0, \quad i = 0, \ldots, k-1. \tag{A.5c}
\]

From the positive definiteness of \( B_0 \), (A.5a) gives \( \alpha_k = 0 \), which in combination with (A.5c) gives \( \alpha_i = 0, i = 0, \ldots, k \). In addition, (A.5b) gives \( u = 0 \). Therefore, \( p^T B_k p = 0 \) only if \( p = 0 \), proving that \( B_k \) is positive definite.

In Theorem 4.4, it has been shown how search directions parallel to PCG may be generated in a limited-memory setting if search directions are included in the basis in addition to gradients. The following theorem gives a way of using gradients only by modifying the right-hand side.

**Theorem A.3.** Consider iteration \( k, 1 \leq k < r \), of the exact-linesearch method of Algorithm 1 for solving (QP). Assume that \( \rho_i = \delta_i p_{PCG} \) with \( \delta_i \neq 0 \) for \( i = 0, \ldots, k-1 \), where \( p_{PCG} \) is the search direction of PCG with associated \( B_0 \in S^n_+ \), as stated in (7). Let \( A_k = \{ j_1, \ldots, j_{m_k} \} \subseteq \{ 0, \ldots, k \} \) with \( j_1 < j_2 < \cdots < j_{m_k} \) such that \( j_{m_k} = k \) and let \( I_k = \{ 0, \ldots, k \} \setminus A_k \). Furthermore, let \( p_k \) satisfy \( B_k p_k = -N_k g_k \) where

\[
B_k = B_0 + \sum_{i=1}^{m_k-1} \left( -\frac{1}{g_j^T B_0^{-1} g_j} g_j g_j^T + \rho_j^{(k)} (g_{j+i} - g_j) (g_{j+i} - g_j)^T \right)
\]

\[
+ \left( \frac{1}{\delta_k} - 1 \right) \frac{1}{g_k^T B_0^{-1} g_k} g_k g_k^T,
\]

and

\[
N_k = I + \delta_k \sum_{i \in I_k} \left( \frac{1}{g_i^T B_0^{-1} g_i} \right) g_i g_i^T B_0^{-1},
\]

where \( \delta_k \) and \( \rho_j^{(k)} \), \( i = 1, \ldots, m_k - 1 \), are chosen such that \( B_k \) is nonsingular. Then

\[
p_k = \delta_k p_{PCG}^{(k)}.
\]

In particular, if \( \rho_j^{(k)} > 0 \), \( i = 1, \ldots, m_k - 1 \), and \( \delta_k > 0 \), then \( B_k \succ 0 \).
Proof. We will show that \( p_k = \delta_k p_{k}^{PCG} \), \( \delta_k \neq 0 \) satisfies \( B_k p_k = -N_k g_k \). Note that by Lemma 2.1 it follows that \( g_i^T p_{k}^{PCG} = -g_i^T B_0^{-1} g_k \), \( i = 0, \ldots, k \) and hence

\[
\delta_k B_k p_k^{PCG} = \delta_k \left( B_0 p_k^{PCG} + \sum_{i=1}^{m_k-1} g_i^T B_0^{-1} g_j g_j \left( \frac{1}{\delta_k} - 1 \right) g_k \right) \quad \text{(A.7)}
\]

Insertion of \( p_k^{PCG} \) as in (8) with \( M = B_0 \) into (A.7) gives

\[
\delta_k B_k p_k^{PCG} = \delta_k \left( -\sum_{i=0}^{k} g_i^T B_0^{-1} g_i g_i + \sum_{i=1}^{m_k-1} g_i^T B_0^{-1} g_j g_j \left( \frac{1}{\delta_k} - 1 \right) g_k \right)
\]

\[
= -\delta_k \sum_{i \in I_k} g_i^T B_0^{-1} g_i g_i - g_k = -N_k g_k,
\]

with \( N_k \) given by (A.6b). Thus \( p_k = \delta_k p_{k}^{PCG} \), \( \delta_k \neq 0 \) is a solution to \( B_k p_k = -N_k g_k \), since \( B_k \) is nonsingular this is also the unique solution. The matrix \( B_k \) is positive definite with \( \delta_k = 1 \) and \( \rho_k^{(k)} > 0 \), \( i = 1, \ldots, m_k - 1 \) by Lemma A.2. The steps to show that \( \delta_k > 0 \) if in addition holds that \( \delta_k > 0 \) is analogous to the last steps of Lemma A.1.

If all indices are chosen to be active in the Hessian approximation of (A.6a) then it is equivalent to (26) of Proposition 4.3 where \( \varphi_k \) relates to \( \delta_k \) as in (20). Conversely, if the indices corresponding to all previous gradients, i.e. \( i = 0, \ldots, k - 1 \) are inactive, and \( \delta_k = 1 \) then (A.6) with \( B_k p_k = -N_k g_k \) is equivalent to the PCG update (8). The update scheme of Theorem A.3 contains only gradients as vector components and with the exact linesearch method of Algorithm 1 the finite termination property is maintained. However, this is at the expense of adding a correction term on the right hand side. Moreover, the update in Theorem A.3 relies heavily on the result in Lemma 2.1 which is exact on quadratic problems. For non-quadratic problems other more accurate approximations and modifications may be considered to improve the method.

### Complexity of symPCGs and Vsr1

It is sufficient to consider the complexity for iteration \( k \geq m \). Note that the first \((m - 3)\) columns of \( Z_k \) remain constant since the first \((m - 3)\) columns of \( Q_k \) remain the same. The basis matrix may hence be written as \( Z_k = \begin{bmatrix} Z_0 & \bar{Z}_k \end{bmatrix} \) where \( Z_0 \in \mathbb{R}^{m \times (m - 3)} \) and \( \bar{Z}_k \in \mathbb{R}^{m \times 3} \). This reduces the complexity of the Gram-Schmidt process to \( O(mn) \). Moreover, for both \( \text{SymPCG} \)s and \( \text{Vsr1} \) the matrix \( B_k \) can be written as

\[
B_k = B_0 + V_0^\rho + F_k,
\]

where \( V_0^\rho = \sum_{i=0}^{m-4} \rho_i^P (g_i+1 - g_i)(g_i+1 - g_i)^T \) and \( F_k = V_0 + \sum_{i=k-3}^{k-1} \rho_i^P (g_i+1 - g_i)(g_i+1 - g_i)^T \). Note that \( F_k \) has a compact representation of rank-five in \( \text{symPCG} \)s and of rank-four in \( \text{Vsr1} \) respectively. Moreover, note that the matrix multiplication \( MF_N \) where \( M \in \mathbb{R}^{m \times n} \), \( N \in \mathbb{R}^{n \times q_2} \) and \( F \in \mathbb{R}^{n \times n} \) is of rank \( \hat{r} \), with a known compact representation, has complexity \( O(nq_1 \hat{r} + q_1 q_2 \hat{r} + nq_2 \hat{r}) \). The reduced-Hessian can at iteration \( k \) be written as

\[
Z_k^T B_k Z_k = \begin{bmatrix} Z_k^T (B_0 + V_0^\rho) Z_0 + Z_k^T F_k Z_0 & Z_k^T (B_0 + V_0^\rho) \bar{Z}_k + Z_k^T F_k \bar{Z}_k \\ Z_k^T (B_0 + V_0^\rho) Z_0 + Z_k^T F_k Z_0 & Z_k^T (B_0 + V_0^\rho + F_k) \bar{Z}_k \end{bmatrix}.
\]

The matrices \( Z_k^T (B_0 + V_0^\rho) Z_0 \in \mathbb{R}^{(m-3) \times (m-3)} \) and \( (B_0 + V_0^\rho) Z_0 \in \mathbb{R}^{m \times (m-3)} \) have been successively built for iteration \( k < m - 3 \) and can be stored. What remains to be computed and the corresponding complexity, taking into account that \( \hat{r} \leq 5 \), is shown in Table A.1.

24
Table A.1: Complexity for computing quantities in the reduced-Hessian, $Z_k^T B_k Z_k$.

| $Z_0^T F_k Z_0$ | $Z_k^T (B_0 + V_0^p) Z_0$ | $Z_k^T F_k Z_0$ | $Z_k^T (B_0 + V_0^p + F_k) Z_k$ |
|-----------------|-----------------|-----------------|-----------------|
| $O(m^2 + nm)$   | $O(nm)$         | $O(n + m + nm)$ | $O(n + n^2)$    |

It remains to add everything together, which has complexity $O(m^2)$, and to factorize, which has complexity $m^3$. The overall asymptotic complexity is thus dominated by $\max\{n^2, m^3\}$.  

25
Bibliography

[1] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited memory methods. *Math. Programming*, 63(2, Ser. A):129–156, 1994.

[2] W. C. Davidon. Variable metric method for minimization. *SIAM J. Optim.*, 1(1):1–17, 1991.

[3] O. DeGuchy, J. B. Erway, and R. F. Marcia. Compact representation of the full Broyden class of quasi-Newton updates. *ArXiv e-prints*, (arXiv:1705.08306), May 2017.

[4] L. C. W. Dixon. Quasi-Newton algorithms generate identical points. *Math. Programming*, 2:383–387, 1972.

[5] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[6] J. B. Erway and R. F. Marcia. On efficiently computing the eigenvalues of limited-memory quasi-Newton matrices. *SIAM J. Matrix Anal. Appl.*, 36(3):1338–1359, 2015.

[7] J. B. Erway and R. F. Marcia. On solving large-scale limited-memory quasi-Newton equations. *Linear Algebra Appl.*, 515:196–225, 2017.

[8] R. Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, Chichester, New York, Brisbane, Toronto and Singapore, second edition, 1987. ISBN 0-471-91547-5.

[9] R. Fletcher. An overview of unconstrained optimization. In *Algorithms for continuous optimization (Il Ciocco, 1993)*, volume 434 of *NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci.*, pages 109–143. Kluwer Acad. Publ., Dordrecht, 1994.

[10] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Comput. J.*, 6:163–168, 1963/1964.

[11] A. Forsgren and T. Odland. On exact linesearch quasi-Newton methods for minimizing a quadratic function. *Comput. Optim. Appl.*, 69(1):225–241, 2018.

[12] P. E. Gill and M. W. Leonard. Reduced-Hessian quasi-Newton methods for unconstrained optimization. *SIAM J. Optim.*, 12(1):209–237, 2001.

[13] P. E. Gill and M. W. Leonard. Limited-memory reduced-Hessian methods for large-scale unconstrained optimization. *SIAM J. Optim.*, 14(2):380–401, 2003.
[14] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[15] N. I. M. Gould, D. Orban, and P. L. Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Comput. Optim. Appl.*, 60(3):545–557, 2015.

[16] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436 (1953), 1952.

[17] H. Y. Huang. Unified approach to quadratically convergent algorithms for function minimization. *J. Optimization Theory Appl.*, 5:405–423, 1970.

[18] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comp.*, 35(151):773–782, 1980.

[19] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer, New York, second edition, 2006.

[20] D. Orban and A. S. Siqueira. JuliaSmoothOptimizers: Infrastructure and solvers for continuous optimization in Julia, 2019.

[21] R. Pytlak. *Conjugate gradient algorithms in nonconvex optimization*, volume 89 of *Nonconvex Optimization and its Applications*. Springer-Verlag, Berlin, 2009.

[22] Y. Saad. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, second edition, 2003.

[23] D. F. Shanno. Conjugate gradient methods with inexact searches. *Math. Oper. Res.*, 3(3):244–256, 1978.