TECHNICAL REPORT

A 2-BFGS UPDATING IN A TRUST REGION FRAMEWORK

M.S. Apostolopoulou[a] D.G. Sotiropoulos[b]
P. Pintelas[c]

No. TR07-01

University of Patras
Department of Mathematics
GR-265 04 Patras, Greece.
http://www.math.upatras.gr/

[a]University of Patras, Department of Mathematics, GR-265 04 Patras, Greece (msa@math.upatras.gr)
[b]Ionian University, Department of Informatics, GR-491 00 Corfu, Greece (dgs@ionio.gr)
[c]University of Patras, Department of Mathematics, GR-265 04 Patras, Greece (pintelas@math.upatras.gr)

Technical Report
No. TR07-01

A 2-BFGS updating in a trust region framework

M.S. Apostolopoulou D.G. Sotiropoulos P. Pintelas

June 2007

Abstract. We present a new matrix-free method for the trust region subproblem, assuming that the approximate Hessian is updated by the limited memory BFGS formula with $m = 2$. The resulting updating scheme, called 2-BFGS, give us the ability to determine via simple formulas the eigenvalues of the resulting approximation. The resulting updating scheme gives the ability to determine the eigenvalues of the resulting approximation via simple formulas. Thus, at each iteration, we can construct a positive definite matrix whose inverse can be expressed analytically, without using factorization. Consequently, a direction of negative curvature can be computed immediately by applying the inverse power method. Thus, the computation of the trial step can be obtained by performing a sequence of inner products and vector summations. Furthermore, it immediately follows that the strong convergence properties of trust region methods are preserved. Numerical results are also presented.

1 Introduction

We consider the following large scale *unconstrained optimization* problem

$$\min\{f(x) \mid x \in \mathbb{R}^n\}, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and the dimension n is larger than one thousand. There exist two principal classes of methods for solving (1), namely, line search and trust region methods [20]. Recently, trust region methods [8] have become more popular than line search algorithms due to their strong convergence and robustness [8, 22]. Under some mild conditions it can be proved that the sequence of points $\{x_k\}$ generated by the trust region algorithm converges to a point which satisfies both the first and the second order necessary conditions [17, 28].

At each iteration x_k, a trial step d_k is usually obtained by solving the following quadratic subproblem

$$\min_{d\in\mathbb{R}^n} \phi_k(d) = g_k^T d + \frac{1}{2} d^T B_k d, \quad \text{s.t.} \quad \|d\| \leq \Delta_k, \tag{2}$$

where $g_k = \nabla f(x_k)$, $B_k \in \mathbb{R}^{n\times n}$ is an approximate Hessian of f at x_k, $\|\cdot\|$ is the Euclidean norm, and $\Delta_k > 0$ is the trust region radius. The step d_k is either accepted or rejected and the trust-region radius is kept, expanded or contracted, depending on the value of the ratio

$$r_k = \frac{f(x_k) - f(x_k + d_k)}{\phi_k(0) - \phi_k(d_k)}.$$

The above ratio measures the ability of the quadratic model to accurately predict the behavior of the objective function f. If the step d_k is rejected, that is if $f(x_k + d_k) \geq f(x_k)$, then the trust region radius Δ_k is decreased and either the subproblem (2) is resolved or a backtracking line search ([12, 21]) is performed in order to obtain a new trial step. When the number of variables is very large, it could be very costly to solve (2) exactly. Therefore, various methods for calculating an approximate solution of (2) have been developed such as the dogleg method [23], the two-dimensional subspace minimization methods [7, 27], the truncated CG methods [29], etc. Nearly exact solutions to subproblem (2) have been proposed by Gay [11], Sorensen [28], and Moré and Sorensen [17]. Methods of nearly exact solutions are based on the fact that the solution d satisfies the linear system $(B_k + \lambda I)d = -g_k$, where the Lagrange multiplier $\lambda \geq 0$ ensures the positive semi-definiteness of $B_k + \lambda I$. The main disadvantage of this approach is that requires the factorization of $B_k + \lambda I$, each time that the subproblem is solved. In addition, since B_k is allowed to be indefinite, a reasonably well approximation of the most negative eigenvalue of B_k must be obtained when the matrix is not positive definite. Moreover, in the so called *hard case*, a direction of negative curvature must also be produced. Therefore, this method can be very costly and even prohibitively expensive when it is applied in very large problems.

The aim of this work is to calculate the eigenvalues of B_k with high accuracy, as well as to avoid both the factorization and the storage of any matrix. To this end, the computation of the approximate Hessian B_k in the quadratic model (2) is obtained using the L-BFGS

formula [19, 15], for $m = 1, 2$, where m is the number of the vector pairs that is used for updating B_k. The L-BFGS method is particularly suited for large scale problems since the storage of matrices is avoided.

The study of the approximate Hessian matrix results that its characteristic equation can be obtained via simple formulas. When the exact eigenvalues are known, the construction of a positive semi-definite matrix, at each iteration, is immediate. Furthermore, the characteristic equation constitutes the basic tool in order to express the inverse of $B_k + \lambda I$ in a closed form, avoiding by this way the Cholesky factorization for the solution of the linear system $(B_k + \lambda I)d = -g_k$. The calculation of the eigenvector corresponding to the most negative eigenvalue of B_k is achieved either by simple algebraic computations or by using the method of inverse iteration [2, 14], providing a closed form for its expression. In this case, λ is an exact eigenvalue of B_k, therefore, the power inverse method converges in a single iteration, providing an analytical form for the corresponding eigenvector. For the above reasons, the computation of the step can be obtained by performing a sequence of inner products and vector summations. Clearly, the solution of the trust region subproblem is not computationally expensive even if the dimension of the problem is too large. Furthermore, the trial step can also be obtained with high accuracy, providing an exact solution to the trust region subproblem, if such a solution is desirable.

The paper is organized as follows: In Section 2 we discuss the trust region subproblem, while in Section 3 we study the properties of the updated matrix. In Section 4 we describe in detail the step computing function which results by our approach. In Section 5 we present an algorithm based on the results of the previous analysis, that incorporates both the standard and the hard case for the computation of the trial step. Finally, in Section 6 we present numerical results, indicating that our approach is promising and inexpensive for the solution of the large scale trust region subproblem. Our numerical experiments are gained by implementing a pure trust region algorithm and a trust region algorithm with backtracking line search.

Notation. Throughout the paper $\|\cdot\|$ denotes the Euclidean norm. The Moore-Penrose generalized inverse of a matrix A is denoted by $A^\dagger$. For a symmetric $A \in \mathbb{R}^{n\times n}$, let $\lambda_1 \leq \ldots \leq \lambda_n$ be its eigenvalues sorted into increasing order. We indicate that a matrix is positive semi-definite by $A \geq 0$.

2 The Subproblem

In this section we recall some known properties of the trust region subproblem (2) that will be used in the sequel. Most of the existing methods for solving (2) are based on the following theorem due to Gay [11] and Moré & Sorensen [17]:

Theorem 2.1 ([28]) *A feasible vector d^* is a solution to (2) with corresponding Lagrange multiplier λ^* if and only if d^*, λ^* satisfy $(B + \lambda^* I)\, d^* = -g$, where $B + \lambda^* I$ is positive semi-definite, $\lambda^* \geq 0$, and $\lambda^*(\Delta - \|d^*\|) = 0$.*

This result provides the theoretical basis for our step computing function d and it can be analyzed in the following cases:

1. If B is positive definite and $\|B^{-1}g\| \leq \Delta$, then $\lambda^* = 0$ and $d^* = B^{-1}g$.

2. If B is positive definite and $\|B^{-1}g\| > \Delta$, then for the unique $\lambda^* > 0$ such that $\| (B + \lambda^* I)^{-1} g\| = \Delta$, $d^* = -(B + \lambda^* I)^{-1} g$.

3. (a) If $\lambda_1 \leq 0$ and there is a $\lambda^* > -\lambda_1$ such that $\|(B + \lambda^* I)^{-1} g\| = \Delta$, then $d^* = -(B + \lambda^* I)^{-1} g$.

   (b) Otherwise, $\lambda^* = -\lambda_1$ and $d^* = -(B - \lambda_1 I)^{\dagger} g + \tau u_1$, where $\tau \in \mathbb{R}$ is such that

$$\| - (B - \lambda_1 I)^{\dagger} g + \tau u_1\| = \Delta$$

   and u_1 is an eigenvector that corresponds to λ_1, such that $\|u_1\| = 1$.

Clearly, the most negative eigenvalue of B must be approximated reasonably well in the case where B is not positive definite. Moreover, when this matrix is indefinite and g is orthogonal to every eigenvector of B corresponding to the most negative eigenvalue λ_1, then $\lambda^* = -\lambda_1$, $(B - \lambda_1 I)$ is singular, and a direction of negative curvature must be produced in order to ensure that $\|d\| = \Delta$. This case is known as the *hard* case (case 3 (b)). For this case, Moré and Sorensen [17] have showed that the choice of τ that ensures $\|d\| = \Delta$, is

$$\tau = \frac{\Delta^2 - \|p\|^2}{p^T u_1 + \operatorname{sgn}(p^T u_1)\sqrt{(p^T u_1)^2 + \Delta^2 - \|p\|^2}}, \tag{3}$$

where $p = -(B - \lambda_1 I)^{\dagger} g$ and the function $\operatorname{sgn}(\cdot)$ is defined as

$$\operatorname{sgn}(t) \begin{cases} 1, & \text{if } t \geq 0; \\ -1, & \text{otherwise.} \end{cases}$$

When $\lambda^* \in (-\lambda_1, \infty)$ (the *standard* case) and $\lambda^* \neq 0$, (2) has a solution on the boundary of its constraint set, i.e., $\|d^*\| = \Delta$ and the given n-dimensional constrained optimization problem is reduced into a zero-finding problem in a single scalar variable λ, namely, $\|d(\lambda)\| - \Delta = 0$, where $d(\lambda)$ is a solution of $(B + \lambda I)\, d = -g$. Moré and Sorensen [17] have proved that it is more convenient to solve the equivalent equation (secular equation)

$$\psi(\lambda) \equiv \frac{1}{\Delta} - \frac{1}{\|d(\lambda)\|} = 0$$

that exploits the rational structure of $\|d(\lambda)\|^2$ when Newton's method is applied. The resulting iteration is

$$\lambda^{\ell+1} = \lambda^{\ell} + \frac{\|d(\lambda)\|}{\|d(\lambda)\|'} \left(\frac{\Delta - \|d(\lambda)\|}{\Delta} \right), \quad \ell = 0, 1, 2, \ldots \tag{4}$$

where the derivative of $\|d(\lambda)\|$ is of the form

$$\|d(\lambda)\|' = -\frac{\|d(\lambda)\|^T (B + \lambda I)^{-1} \|d(\lambda)\|}{\|d(\lambda)\|}.$$

Clearly, the trial step $d(\lambda)$ must be recomputed in every iteration for every evaluation of λ. However, an important ingredient of Newton's iteration (4) is the safeguarding required to ensure that a solution is found. The safeguarding depends on the fact that ψ is convex and strictly decreasing in $(-\lambda_1, \infty)$. It ensures that $-\lambda_1 \leq \lambda^\ell$, and therefore $B + \lambda^\ell I$ is always semi positive definite. For a more thorough discussion on these issues, see [17].

Initial bounds for λ have been also proposed by Nocedal and Yuan [21]. The following lemma gives the bounds of λ, when the trust region subproblem (2) is solved exactly.

Lemma 2.2 ([21]) *If vector d^* is a solution of (2), $\|d^*\| = \Delta$, and $\lambda^* \geq 0$ satisfies $(B + \lambda^* I)d^* = -g$, with $(B + \lambda^* I) \geq 0$ then*

$$0 \leq \lambda^* \leq \|g\|/\Delta - \lambda_1,$$

where λ_1 is the smallest eigenvalue of B.

For an approximate solution to (2), $\|B\| + (1 + \epsilon)\|g\|/\Delta$ forms another upper bound for λ, where $\epsilon > 0$ is a small number that ensures that $B + \lambda I$ is positive definite [21]. Therefore, λ can be bounded as

$$\max(0, -\lambda_1 + \epsilon) \leq \lambda \leq \|B\| + (1 + \epsilon)\|g\|/\Delta. \tag{5}$$

3 The updated matrix

The computation of the matrix $B \approx \nabla^2 f(x)$ in the quadratic model (2), is accomplished using the L-BFGS philosophy with $m = 2$. Denoting by $B_k^{(0)}$ the initial matrix, the matrix B_{k+1} is updated using $B_k^{(0)} = (1/\theta_k)I$, $\theta_k \in \mathbb{R}$. Motivated by the work of Barzilai and Borwein [3] and Birgin and Martínez [4], we use the scalar parameter

$$\theta_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}. \tag{6}$$

This parameter is the inverse of the Rayleigh quotient $s^T G s / s^T s$, which lies between the largest and the smallest eigenvalues of the average Hessian G. Rigorous analysis of methods exclusively based on this approximation may be found in [5, 25, 26].

More analytically, for $k \geq 2$, the matrix B_{k+1} defined as

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}, \tag{7}$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$, is updated using the matrix B_k defined by

$$B_k = \frac{1}{\theta_k} I - \frac{s_{k-1} s_{k-1}^T}{\theta_k s_{k-1}^T s_{k-1}} + \frac{y_{k-1} y_{k-1}^T}{s_{k-1}^T y_{k-1}}. \tag{8}$$

Thus, B_k is computed by applying the 1-BFGS update on $B_k^{(0)}$ (the stored vector set of pairs contains *one pair* $\{(s_{k-1}, y_{k-1})\}$). Consequently, B_{k+1} is computed by applying the 2-BFGS update on $B_k^{(0)}$ (the stored vector set of pairs contains *two pairs*, $\{(s_{k-1}, y_{k-1}), (s_k, y_k)\}$).

It is known [20] that the trace and the determinant of BFGS matrices are given by the formulas $\mathrm{tr}(B_{k+1}) = \mathrm{tr}(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \frac{\|y_k\|^2}{s_k^T y_k}$ and $\det(B_{k+1}) = \det(B_k) \frac{s_k^T y_k}{s_k^T B_k s_k}$, respectively. When $B_{k+1} \equiv B_{k+1}^{(1)}$, its trace is given by

$$\mathrm{tr}(B_{k+1}^{(1)}) = \frac{a_k + n - 2}{\theta_{k+1}}, \tag{9}$$

where

$$a_k = 1 + \theta_{k+1} \frac{y_k^T y_k}{s_k^T y_k}, \tag{10}$$

while its determinant is expressed as

$$\det(B_{k+1}^{(1)}) = \frac{1}{\theta_{k+1}^n}. \tag{11}$$

When $B_{k+1} \equiv B_{k+1}^{(2)}$, taking into account Eqs. (9) and (11), it is straightforward that

$$\mathrm{tr}\left(B_{k+1}^{(2)}\right) = \frac{a_{k-1} + n - 2}{\theta_k} - \frac{\|B_k^{(1)} s_k\|^2}{s_k^T B_k^{(1)} s_k} + \frac{\|y_k\|^2}{s_k^T y_k}. \tag{12}$$

and

$$\det\left(B_{k+1}^{(2)}\right) = \frac{1}{\theta_k^n} \frac{s_k^T y_k}{s_k^T B_k^{(1)} s_k}. \tag{13}$$

Observing the updating scheme, we can see that B_k is updated by the addition of two rank-one matrices. The following lemma, due to Wilkinson [31, pp. 94–98], states that the eigenvalues of a matrix which is updated by a rank-one matrix, interlace with the eigenvalues of the original matrix.

Lemma 3.1 *If symmetric matrices A and A^* differ by a matrix of rank 1, then their eigenvalues λ and λ^* interpolate each other in a week sense. In particular, if $A^* = A + \sigma v v^T$, where σ is scalar and if $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and if $\lambda_1^* \geq \lambda_2^* \geq \ldots \geq \lambda_n^*$, then*
(1) if $\sigma \geq 0$, $\lambda_1^* \geq \lambda_1 \geq \lambda_2^* \geq \lambda_2 \geq \ldots \geq \lambda_n^* \geq \lambda_n$
(2) if $\sigma \leq 0$, $\lambda_1 \geq \lambda_1^* \geq \lambda_2 \geq \lambda_2^* \geq \ldots \geq \lambda_n \geq \lambda_n^*$

3.1 1-BFGS update

We recall that the update matrix $B_{k+1}^{(1)}$ is given by the expression

$$B_{k+1}^{(1)} = \frac{I}{\theta_{k+1}} - \frac{s_k s_k^T}{\theta_{k+1} s_k^T s_k} + \frac{y_k y_k^T}{s_k^T y_k} \tag{14}$$

where $B_{k+1}^{(0)} = (1/\theta_{k+1})I$. The following lemma gives the general form of the characteristic polynomial of $B_{k+1}^{(1)}$:

Lemma 3.2 *Let the symmetric matrix $B_{k+1}^{(1)} \in \mathbb{R}^{n\times n}$. Then, the characteristic polynomial of $B_{k+1}^{(1)}$ has the general form*

$$p_1(\lambda) = (\lambda - \frac{1}{\theta_{k+1}})^{n-2}(\lambda^2 - \frac{a_k}{\theta_{k+1}}\lambda + \frac{1}{\theta_{k+1}^2}). \tag{15}$$

Moreover, if $a_k > 2$, then $\lambda_1 < 1/\theta_{k+1} < \lambda_n$, where λ_1 and λ_n are the smallest and largest eigenvalues of $B_{k+1}^{(1)}$, respectively.

Proof. Proof First we show that $B_{k+1}^{(1)}$ has at most two distinct eigenvalues. To this end, we consider the matrix $\bar{B} = (1/\theta_{k+1})I - s_k s_k^T/\theta_{k+1} s_k^T s_k$ with rank $(n-1)$. Using Lemma 3.1, it is easy to see that $\bar{B}$ besides the zero eigenvalue, has one more eigenvalue equals to $1/\theta_{k+1}$ of multiplicity $(n-1)$. If $B_{k+1}^{(0)}$ is positive definite, then the addition of the term $y_k y_k^T/s_k^T y_k$ on $\bar{B}$ yields (cf. Lemma 3.1)

$$\lambda_n^{B^{(1)}} \geq 1/\theta_{k+1} \geq \lambda_{n-1}^{B^{(1)}} \geq 1/\theta_{k+1} \geq \ldots \geq \lambda_2^{B^{(1)}} \geq 1/\theta_{k+1} \geq \lambda_1^{B^{(1)}} \geq 0;$$

where $\lambda_i^{B^{(1)}}$ denotes the eigenvalues of $B_{k+1}^{(1)}$, otherwise,

$$0 \geq \lambda_n^{B^{(1)}} \geq 1/\theta_{k+1} \geq \lambda_{n-1}^{B^{(1)}} \geq 1/\theta_{k+1} \geq \ldots \geq \lambda_2^{B^{(1)}} \geq 1/\theta_{k+1} \geq \lambda_1^{B^{(1)}}.$$

In both cases, it is obvious that $\lambda_2^{B^{(1)}} = \ldots = \lambda_{n-1}^{B^{(1)}} = 1/\theta_{k+1}$, and

$$\lambda_1^{B^{(1)}} \geq 1/\theta_{k+1} \geq \lambda_n^{B^{(1)}}. \tag{16}$$

Relation (16) implies that $B_{k+1}^{(1)}$ has at most two distinct eigenvalues and one eigenvalue equals to $1/\theta_{k+1}$ of multiplicity at least $(n-2)$. Denoting by λ_1 and λ_2 the two unknown distinct eigenvalues, the characteristic polynomial of $B_{k+1}^{(1)}$ has the form $p_1(\lambda) = (\lambda - 1/\theta_{k+1})^{n-2}\big[\lambda^2 - (\lambda_1+\lambda_2)\lambda + \lambda_1\lambda_2\big]$. Since $\mathrm{tr}(B_{k+1}^{(1)}) = \lambda_1 + \lambda_2 + (n-2)/\theta_{k+1}$, and $\det(B_{k+1}^{(1)}) = \lambda_1\lambda_2/\theta_{k+1}^{n-2}$, from Eqs. (9) and (11) we obtain $\lambda_1 + \lambda_2 = a_k + 2/\theta_{k+1}$, while $\lambda_1\lambda_2 = 1/\theta_{k+1}^2$. Consequently, relation (15) follows immediately.

Note that the parameter a_k is bounded from below by 2, since

$$a_k = 1 + \theta_{k+1}\frac{y_k^T y_k}{s_k^T y_k} = 1 + \frac{\|s_k\|^2\|y_k\|^2}{(s_k^T y_k)^2} = 1 + \frac{1}{cos^2\phi} \geq 2, \tag{17}$$

where ϕ is the angle between s_k and y_k.
If $a_k = 2$, then the characteristic polynomial is reduced to $p_1(\lambda) = (\lambda - 1/\theta_{k+1})^n$; thus $B^{(1)}_{k+1} = (1/\theta_{k+1})I = B^{(0)}_{k+1}$.
In different case (when $a_k > 2$), the characteristic polynomial becomes $p_1(\lambda) = (\lambda - 1/\theta_{k+1})^{n-2}(\lambda - \lambda_1)(\lambda - \lambda_2)$, where the eigenvalues

$$\lambda_{1,2} = \frac{a_k \pm \sqrt{a_k^2 - 4}}{2\theta_{k+1}}$$

are distinct. From inequalities (16) follows that

$$\min(\lambda_1, \lambda_2) < 1/\theta_{k+1} < \max(\lambda_1, \lambda_2).$$

□

Lemma 3.3 *Let Λ be the set of eigenvalues of $B^{(1)}_{k+1}$ with opposite signs. Then, for any $\lambda \in \mathbb{R}^* \setminus \Lambda$, the inverse of $(B^{(1)}_{k+1} + \lambda I)$ can be expressed by the following closed-form*

$$\left(B^{(1)}_{k+1} + \lambda I\right)^{-1} = \frac{1}{\gamma}\left[\left(B^{(1)}_{k+1}\right)^2 - \gamma_1 B^{(1)}_{k+1} + \gamma_0 I\right] \tag{18}$$

where $\gamma = (1/\theta_{k+1} + \lambda)(\lambda^2 + a_k\lambda/\theta_{k+1} + 1/\theta^2_{k+1})$, $\gamma_1 = \lambda + (a_k + 1)/\theta_{k+1}$ and $\gamma_0 = \lambda^2 + (a_k + 1)\lambda/\theta_{k+1} + (a_k + 1)/\theta^2_{k+1}$ are functions of λ.

Proof. Proof Without loss of generality, we assume that $B^{(1)}_{k+1}$ has two distinct eigenvalues λ_1 and λ_2. Consequently, $(B^{(1)}_{k+1} + \lambda I)$ has also two distinct eigenvalues, $\lambda_1 + \lambda$ and $\lambda_2 + \lambda$. Using Lemma 3.2, after some algebraic calculations, the characteristic polynomial of $(B^{(1)}_{k+1} + \lambda I)$ is $q(x) = (x - c)^{n-2}(x^2 - c_1 x + c_0)$ where $c = 1/\theta_{k+1} - \lambda$, $c_1 = a_k/\theta_{k+1} + 2\lambda$, and $c_0 = \lambda^2 + a_k\lambda/\theta_{k+1} + 1/\theta^2_{k+1}$. Therefore, its minimal polynomial is $q_m(x) = (x - c)(x^2 - c_1 x + c_0)$. Using the Caley-Hamilton theorem, relation (18) follows immediately. □

It is easy to see that in the special case where $a_k = 2$, Eq (18) is reduced to

$$\left(B^{(1)}_{k+1} + \lambda I\right)^{-1} = \left(\frac{1}{\theta_{k+1}} + \lambda\right)^{-1} I. \tag{19}$$

Moreover, under the assumption that $B^{(1)}_{k+1}$ is invertible, Eq. (18) holds even if $\lambda = 0$.

3.2 2-BFGS update

We study now the matrix $B^{(2)}_{k+1}$, which contains curvature information from the two previous iterations, i.e., the stored vector set of pairs contains the pairs $\{(s_{k-1}, y_{k-1}), (s_k, y_k)\}$. We recall that the updated matrix $B^{(2)}_{k+1}$ is given by

$$B^{(2)}_{k+1} = B^{(1)}_k - \frac{B^{(1)}_k s_k s_k^T B^{(1)}_k}{s_k^T B^{(1)}_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}$$

where $B^{(1)}_k$ is defined in Eq. (8).

Lemma 3.4 *Let the symmetric matrix $B_{k+1}^{(2)} \in \mathbb{R}^{n\times n}$. The characteristic polynomial of $B_{k+1}^{(2)}$ has the general form*

$$p_2(\lambda) = (\lambda - 1/\theta_k)^{n-4}\,(\lambda^4 - \beta_3\lambda^3 + \beta_2\beta_0\lambda^2 - \beta_1\beta_0\lambda + \beta_0), \tag{20}$$

where

$$\begin{aligned}
\beta_0 &= \frac{1}{\theta_k^4}\frac{s_k^T y_k}{s_k^T B_k^{(1)} s_k}, \quad \beta_1 = (a_{k-1}+2)\theta_k - 2\frac{s_k^T w_k}{s_k^T y_k} + b_k\theta_{k+1},\\
\beta_2 &= \theta_k(\beta_1 - a_{k-1}\theta_k)(a_{k-1}+2) - 2\theta_k^2 + \left(\frac{s_k^T w_k}{s_k^T y_k}\right)^2 - \theta_{k+1}\frac{w_k^T w_k}{s_k^T y_k} + 2\frac{s_k^T\left(B_k^{(1)}\right)^{-1} w_k}{s_k^T y_k}\\
&\quad -b_k\frac{s_k^T\left(B_k^{(1)}\right)^{-1} s_k}{s_k^T y_k},\\
\beta_3 &= \frac{a_{k-1}+2}{\theta_k} - \frac{\|B_k^{(1)} s_k\|^2}{s_k^T B_k^{(1)} s_k} + \frac{\|y_k\|^2}{s_k^T y_k} \quad \textit{and}\\
w_k &= \left(B_k^{(1)}\right)^{-1} y_k, \quad b_k = 1 + \frac{y_k^T w_k}{s_k^T y_k}.
\end{aligned} \tag{21}$$

Proof. Proof. We first show that $B_{k+1}^{(2)}$ has at most four distinct eigenvalues. Without loss of generality, we assume that $B_k^{(1)}$ and $B_{k+1}^{(2)}$ are positive definite matrices (in different case, the proof follows by similar arguments). Let $\lambda_i^{B^{(1)}}$ and $\lambda_i^{\bar{B}}$ be the eigenvalues of $B_k^{(1)}$ and

$$\bar{B} = B_k^{(1)} - \frac{B_k^{(1)} s_k s_k^T B_k^{(1)}}{s_k^T B_k^{(1)} s_k},$$

respectively. Since $\lambda_2^{B^{(1)}} = \ldots = \lambda_{n-1}^{B^{(1)}} = 1/\theta_k$, by Lemma 3.1 we have that

$$\lambda_n^{B^{(1)}} \geq \lambda_n^{\bar{B}} \geq 1/\theta_k \geq \lambda_{n-1}^{\bar{B}} \geq 1/\theta_k \geq \ldots \geq 1/\theta_k \geq \lambda_2^{\bar{B}} \geq \lambda_1^{B^{(1)}} \geq \lambda_1^{\bar{B}}.$$

The addition of the term $\dfrac{y_k y_k^T}{s_k^T y_k}$ to the matrix $\bar{B}$ yields

$$\lambda_n^{B^{(2)}} \geq \lambda_n^{\bar{B}} \geq \lambda_{n-1}^{B^{(2)}} \geq 1/\theta_k \geq \cdots \geq 1/\theta_k \geq \lambda_2^{B^{(2)}} \geq \lambda_2^{\bar{B}} \geq \lambda_1^{B^{(2)}} \geq \lambda_1^{\bar{B}},$$

where $\lambda_i^{B^{(2)}}$ are the eigenvalues of $B_{k+1}^{(2)}$. The above inequalities imply that

$$\lambda_n^{B^{(2)}} \geq \lambda_{n-1}^{B^{(2)}} \geq 1/\theta_k \geq \lambda_2^{B^{(2)}} \geq \lambda_1^{B^{(2)}}, \tag{22}$$

and thus, the matrix $B_{k+1}^{(2)}$ has at most four distinct eigenvalues. If we denote by λ_1, λ_2, λ_3 and λ_4 the four unknown eigenvalues, the characteristic polynomial of $B_{k+1}^{(2)}$ takes the form

$$p_2(\lambda) = (\lambda - 1/\theta_k)^{n-4}\,(\lambda^4 - c_3\lambda^3 + c_2\lambda^2 - c_1\lambda + c_0),$$

where $c_3 = \sum_{i=1}^{4} \lambda_i$, $c_2 = \sum_{\substack{i,j \\ i<j}} \lambda_i\lambda_j$, $c_1 = \sum_{\substack{i,j,\ell \\ i<j<\ell}} \lambda_i\lambda_j\lambda_\ell$, and $c_0 = \prod_{i=1}^{4} \lambda_i$, and moreover, $\prod_{i=1}^{4} \lambda_i = \det\left(B_{k+1}^{(2)}\right)\theta_k^{n-4}$ and $\sum_{i=1}^{4} \lambda_i = \operatorname{tr}\left(B_{k+1}^{(2)}\right) - \frac{n-4}{\theta_k}$. Utilizing Eqs. (13) and (12), after some algebraic manipulations, we yield the expression in Eq. (20) with the parameters in Eqs. (21). □

In case where $B_{k+1}^{(2)}$ has four distinct eigenvalues, these can be computed by analytically solving the quartic equation $\lambda^4 - \beta_3\lambda^3 + \beta_2\beta_0\lambda^2 - \beta_1\beta_0\lambda + \beta_0 = 0$, using standard methods [6, 10]. However, the exact number of the distinct eigenvalues is strongly depends on the way the vector pairs (s_i, y_i) in the vector set are related. The following proposition establishes sufficient conditions for the exact number of distinct eigenvalues.

Proposition 3.5 *Let the symmetric matrix $B_{k+1}^{(2)} \in \mathbb{R}^{n\times n}$ as defined in Eq. (3.2).*

1. *If $a_{k-1} = 2$ then $B_{k+1}^{(2)}$ has at most two distinct eigenvalues. Moreover,*
   - *i. if $a_k > 2$, then two distinct eigenvalues exist,*
   - *ii. if $a_k = 2$ and $\theta_k \neq \theta_{k+1}$, then only one distinct eigenvalue exists, and*
   - *iii. if $a_k = 2$ and $\theta_k = \theta_{k+1}$, then all the eigenvalues are equal to $1/\theta_k$.*
2. *If $a_{k-1} > 2$, then $B_{k+1}^{(2)}$ has at least three distinct eigenvalues. Moreover,*
   - *i. if $a_k = 2$, then only three distinct eigenvalues exist, and*
   - *ii. if $a_k > 2$, then only four distinct eigenvalues exist.*

Proof. Proof. Assume that $a_{k-1} = 2$. Then, in Lemma 3.2 we have proved that, in this case, $B_k^{(1)} = (1/\theta_k)I$. Consequently, $B_{k+1}^{(2)}$ has the following form

$$B_{k+1}^{(2)} = \frac{1}{\theta_k} I - \frac{1}{\theta_k}\frac{s_k s_k^T}{s_k^T s_k} + \frac{y_k y_k^T}{s_k^T y_k}, \tag{23}$$

while its characteristic polynomial becomes

$$p_2(\lambda) = \left(\lambda - \frac{1}{\theta_k}\right)^{n-2}\left[\lambda^2 - \left(\frac{a_k - 1}{\theta_{k+1}} + \frac{1}{\theta_k}\right)\lambda + \frac{1}{\theta_k\theta_{k+1}}\right]. \tag{24}$$

In view of Eq. (24), it follows immediately that $B_{k+1}^{(2)}$ has at most two distinct eigenvalues. Consider now the quadratic equation $\lambda^2 - \left(\frac{a_k-1}{\theta_{k+1}} + \frac{1}{\theta_k}\right)\lambda + \frac{1}{\theta_k\theta_{k+1}} = 0$, which has the following two solutions

$$\lambda_{1,2} = \frac{\theta_{k+1} + \theta_k\,(a_k - 1) \pm \sqrt{(\theta_k - \theta_{k+1})^2 + \theta_k(a_k - 2)(a_k\theta_k + 2\theta_{k+1})}}{2\theta_k\theta_{k+1}}$$

Let $a_k > 2$ and suppose that either λ_1 or λ_2 equals to $1/\theta_k$. Solving the equations $\lambda_i - 1/\theta_k = 0$ with respect to a_k we obtain $a_k = 2$, which is a contradiction. Thus, when $a_k > 2$ the matrix

has exactly two distinct eigenvalues. When $a_k = 2$, the characteristic polynomial becomes $p_2(\lambda) = (\lambda - 1/\theta_k)^{n-1}(\lambda - 1/\theta_{k+1})$. Therefore, in case where $\theta_k \neq \theta_{k+1}$, the only distinct eigenvalue is $1/\theta_{k+1}$; otherwise $B_{k+1}^{(2)}$ does not have distinct eigenvalues.

Assume now that $a_{k-1} > 2$. In Lemma 3.2 we have proved that $B_k^{(1)}$ has two distinct eigenvalues. Thus, the matrix $\bar{B} = B_k^{(1)} - \frac{B_k^{(1)} s_k s_k^T B_k^{(1)}}{s_k^T B_k^{(1)} s_k}$ besides the zero eigenvalue, has two more distinct eigenvalues. Consequently, $B_{k+1}^{(2)}$ has at least three distinct eigenvalues. If $a_k = 2$, then $s_k = \theta_{k+1} y_k$ and $B_{k+1}^{(2)}$ becomes

$$B_{k+1}^{(2)} = B_k^{(1)} - \frac{B_k^{(1)} y_k y_k^T B_k^{(1)}}{y_k^T B_k^{(1)} y_k} + \frac{y_k y_k^T}{\theta_{k+1} y_k^T y_k}. \tag{25}$$

Hence, $\bar{B} = B_k^{(1)} - \frac{B_k^{(1)} y_k y_k^T B_k^{(1)}}{y_k^T B_k^{(1)} y_k}$, and therefore, the addition of $y_k y_k^T / \theta_{k+1} y_k^T y_k$ changes the zero eigenvalue of $\bar{B}$ to $1/\theta_{k+1}$ and leaves the others unchanged. Thus, $B_{k+1}^{(2)}$ has as many distinct eigenvalues as $\bar{B}$, i.e., three. Since one of them is $1/\theta_{k+1}$, utilizing Lemma 3.4, the characteristic polynomial of $B_{k+1}^{(2)}$ becomes

$$p_2(\lambda) = (\lambda - 1/\theta_k)^{n-3} (\lambda - 1/\theta_{k+1})(\lambda^2 - c_1\lambda + c_2),$$

where $c_1 = a_{k-1}/\theta_k - \|B_k^{(1)} y_k\|^2 / y_k^T B_k^{(1)} y_k$ and $c_2 = y_k^T y_k / (\theta_k^2 \theta_{k+1} y_k^T B_k^{(1)} y_k)$. By solving the quadratic equation $\lambda^2 - c_1\lambda + c_2 = 0$, we obtain the other two distinct eigenvalues. Finally, if $a_k > 2$, then y_k is not an eigenvector of $\bar{B} = B_k^{(1)} - \frac{B_k^{(1)} s_k s_k^T B_k^{(1)}}{s_k^T B_k^{(1)} s_k}$, and thus, the addition of the term $y_k y_k^T / s_k^T y_k$ results one more distinct eigenvalue for $B_{k+1}^{(2)}$. □

Lemma 3.6 *If $B_{k+1}^{(2)}$ has at least two distinct eigenvalues, then*

$$\lambda_1 < 1/\theta_k < \lambda_n, \tag{26}$$

where λ_1 and λ_n are the smallest and largest eigenvalues of $B_{k+1}^{(2)}$, respectively.

Proof. Proof. When $B_{k+1}^{(2)}$ has at least three distinct eigenvalues, from relation (22) it is evident that (26) holds. Now, if $B_{k+1}^{(2)}$ has two distinct eigenvalues, then one pair of the vector set $\{(s_{k-1}, y_{k-1}),\ (s_k, y_k)\}$ must be collinear. Due to Proposition (3.5), these two vectors are s_{k-1} and y_{k-1}, which implies that $B_k^{(1)}$ has no distinct eigenvalues. Under these circumstances, Lemma 3.1 implies relation (26), which completes the proof. □

Lemma 3.7 *Let Λ be the set of eigenvalues of $B_{k+1}^{(2)}$ with opposite signs. For any $\lambda \in \mathbb{R}^* \setminus \Lambda$, the inverse of $\left(B_{k+1}^{(2)} + \lambda I\right)$ can be expressed by the following closed-form*

$$\left(B_{k+1}^{(2)} + \lambda I\right)^{-1} = \frac{1}{\nu}\left[\left(B_{k+1}^{(2)}\right)^4 - \nu_3 \left(B_{k+1}^{(2)}\right)^3 + \nu_2 \left(B_{k+1}^{(2)}\right)^2 - \nu_1 B_{k+1}^{(2)} + \nu_0 I\right] \tag{27}$$

where $\nu_3 = \lambda + \beta_3 + 1/\theta_k$, $\nu_2 = \lambda\nu_3 + \beta_2\beta_0 + \beta_3/\theta_k$, $\nu_1 = \lambda\nu_2 + \beta_1\beta_0 + \beta_2\beta_0/\theta_k$, $\nu_0 = \lambda\nu_1 + \beta_0 + \beta_0\beta_1/\theta_k$ *and* $\nu = \lambda\nu_0 + \beta_0/\theta_k$ *are functions of* λ, *while the parameters* β_0, β_1, β_2 *and* β_3 *are defined as in Lemma 3.4.*

Proof. Proof. From Proposition 3.5 we know that $B_{k+1}^{(2)}$ has at most four distinct eigenvalues, λ_i, $i = 1, \ldots, 4$. Thus, $(B_{k+1}^{(2)} + \lambda I)$ has also at most four distinct eigenvalues, $\lambda_i + \lambda$. Utilizing Lemma 3.4, the characteristic polynomial of $(B_{k+1}^{(2)} + \lambda I)$ takes the form

$$q(x) = (x - c)^{n-4}\,(x^4 - c_3x^3 + c_2x^2 - c_1x + c_0),$$

where $c = \theta_k^{-1} + \lambda$, $c_3 = 4\lambda + \beta_3$, $c_2 = 6\lambda^2 + 3\beta_3\lambda + \beta_2\beta_0$, $c_1 = 4\lambda^3 + 3\beta_3\lambda^2 + 2\beta_2\beta_0\lambda + \beta_1\beta_0$, and $c_0 = \lambda^4 + \beta_3\lambda^3 + b_2\beta_0\lambda^2 + \beta_1\beta_0\lambda + \beta_0$, where the constants β_i are defined in Lemma 3.4. In the general case, its minimal polynomial is

$$q_m(x) = (x - c)(x^4 - c_3x^3 + c_2x^2 - c_1x + c_0).$$

Therefore, using the Caley-Hamilton theorem, we obtain Eq. (27), which completes the proof. □
See that if $B_{k+1}^{(2)}$ is nonsingular, Eq. (27) holds even if $\lambda = 0$.

4 The step computing function

In Section 2 we have seen that when the Lagrange multiplier $\lambda \in (-\lambda_1,\ \infty)$ (*standard* case), then the trial step is computed by the form $d(\lambda) = -\big(B + \lambda I\big)^{-1} g$, otherwise, when the hard case occurs, an eigenvector corresponding to λ_1 must be computed in order to ensure that $\|d\| = \Delta$. In case where B is indefinite and λ_1 is a multiple eigenvalue, due to the structure of B, this computation is achieved by simple algebraic computation. When λ_1 is a distinct eigenvalue, since it is known exactly, we can find the analytical expression of the desirable eigenvector using the inverse iteration [2]. In what follows, we give in detail our approach for computing a solution of the trust region subproblem.

4.1 Computation of the step using the 1-BFGS update

4.1.1 The standard case

First we consider the the *standard* case, where $\lambda \in (-\lambda_1,\ \infty)$. According to Lemma 3.2, two cases may occur: either $a_k > 2$ or $a_k = 2$.
When $a_k > 2$, the trial step $d_k(\lambda) = -\left(B_k^{(1)} + \lambda I\right)^{-1} g_k$ is computed using Eq. (18). This computation gives,

$$d_{k+1}(\lambda) = -\frac{1}{\gamma(\lambda)}\left[\left(B_{k+1}^{(1)}\right)^2 - \gamma_1(\lambda)B_{k+1}^{(1)} + \gamma_0(\lambda)I\right] g_{k+1}.$$

The vectors $B_{k+1}^{(1)} g_{k+1}$ and $\left(B_{k+1}^{(1)}\right)^2 g_{k+1}$ are computed by the iterative scheme

$$v_{i+1} \equiv B_{k+1}^{(1)} v_i = \frac{1}{\theta_{k+1}} v_i - \frac{s_k^T v_i}{\theta_{k+1} s_k^T s_k} s_k + \frac{y_k^T v_i}{s_k^T y_k} y_k, \quad i = 0, 1, \tag{28}$$

with $v_0 = g_{k+1}$. Then, the trial step can be expressed by the form

$$d_{k+1}(\lambda) = -\frac{1}{\gamma(\lambda)} \left[v_2 - \gamma_1(\lambda) v_1 + \gamma_0(\lambda) v_0 \right].$$

By substituting the vectors v_i in the above equation, we finally obtain the general form of the trial step:

$$d_{k+1}(\lambda) = -\gamma_g(\lambda)\ g_{k+1} + \gamma_s(\lambda)\ s_k - \gamma_y(\lambda)\ y_k, \tag{29}$$

where

$$\begin{aligned}
\gamma_g(\lambda) &= \left[1 - \gamma_1(\lambda)\ \theta_{k+1} + \gamma_0(\lambda)\ \theta_{k+1}^2\right] / \left[\gamma(\lambda)\ \theta_{k+1}^2\right], \\
\gamma_s(\lambda) &= \left\{\left[1 - \gamma_1(\lambda)\ \theta_{k+1}\right] s_k^T g_{k+1} + \theta_{k+1} y_k^T g_{k+1}\right\} / \left[\gamma(\lambda)\ \theta_{k+1}^2 s_k^T s_k\right], \quad \text{and} \\
\gamma_y(\lambda) &= \left\{\left[1 - \gamma_1(\lambda)\ \theta_{k+1} + a_k\right] \theta_{k+1} y_k^T g_{k+1} - s_k^T g_{k+1}\right\} / \left[\gamma(\lambda)\ \theta_{k+1}^2 s_k^T y_k\right].
\end{aligned}$$

Moreover, for $\lambda = 0$, we yield

$$\begin{aligned}
\gamma_g(0) &= \theta_{k+1}, \\
\gamma_s(0) &= (\theta_{k+1} y_k^T g_{k+1} - a_k s_k^T g_{k+1}) / s_k^T y_k, \quad \text{and} \\
\gamma_y(0) &= -\theta_{k+1} s_k^T g_{k+1} / s_k^T y_k.
\end{aligned}$$

Therefore,

$$d_{k+1}(0) \equiv -\left(B_{k+1}^{(1)}\right)^{-1} g_{k+1} = -\gamma_g(0)\ g_{k+1} + \gamma_s(0)\ s_k - \gamma_y(0)\ y_k. \tag{30}$$

For the second case, i.e., when $a_k = 2$, from Lemma 3.2 we have $B_{k+1}^{(1)} = (1/\theta_{k+1}) I$, and hence the trial step is immediately obtained via the formula

$$d_{k+1}(\lambda) = -\left(\frac{1}{\theta_{k+1}} + \lambda\right)^{-1} g_{k+1}. \tag{31}$$

For the computation of λ we can follow the ideas described in Moré and Sorensen [17] by applying Newton's method as given in Eq. (4). Easily can be verified that the resulting iteration is

$$\lambda^{\ell+1} = \lambda^{\ell} + \frac{\|d_{k+1}(\lambda)\|^2}{d(\lambda)^T \left(B_{k+1}^{(1)} + \lambda I\right)^{-1} d(\lambda)} \left(\frac{\|d_{k+1}(\lambda)\| - \Delta}{\Delta}\right), \quad \ell = 0, 1, 2, \ldots \tag{32}$$

Using Eq. (18), the scalar quantity $d_{k+1}(\lambda)^T \left(B_{k+1}^{(1)} + \lambda I\right)^{-1} d_{k+1}(\lambda)$ in the denominator of (32) becomes

$$\begin{aligned} d_{k+1}(\lambda)^T \left(B_{k+1}^{(1)} + \lambda I\right)^{-1} d_{k+1}(\lambda) &= -\frac{\gamma_1(\lambda)}{\gamma(\lambda)} d_{k+1}(\lambda)^T B_{k+1}^{(1)} d_{k+1}(\lambda) \\ &+ \frac{\gamma_0(\lambda)}{\gamma(\lambda)} \|d_{k+1}(\lambda)\|^2 + \frac{\|B_{k+1}^{(1)} d_{k+1}(\lambda)\|^2}{\gamma(\lambda)}. \end{aligned}$$

Since $B_{k+1}^{(1)} d_{k+1}(\lambda) = -\left[\lambda\, d_{k+1}(\lambda) + g_{k+1}\right]$, the above relation can be written as

$$\begin{aligned} d_{k+1}(\lambda)^T \left(B_{k+1}^{(1)} + \lambda I\right)^{-1} d_{k+1}(\lambda) &= \frac{1}{\gamma(\lambda)} \big\{ \left[\lambda^2 + \lambda\gamma_1(\lambda) + \gamma_0(\lambda)\right] \|d_{k+1}(\lambda)\|^2 + \\ &\quad \left[2\lambda + \gamma_1(\lambda)\right] d_{k+1}(\lambda)^T g_{k+1} + \|g_{k+1}\|^2 \big\}. \end{aligned} \tag{33}$$

In the special case where $a_k = 2$, utilizing Eq. (31), Eq. (33) is reduced to

$$d_{k+1}(\lambda)^T \left(B_{k+1}^{(1)} + \lambda I\right)^{-1} d_{k+1}(\lambda) = \left(\frac{1}{\theta_{k+1}} + \lambda\right)^{-3} \|g_{k+1}\|^2. \tag{34}$$

4.1.2 The hard case

We recall that in the hard case, g is perpendicular to all eigenvectors corresponding to λ_1, therefore $\lambda^* = -\lambda_1$ and a direction of negative curvature must be computed. When $a_k > 2$, from Lemma 3.2 we know that λ_1 is distinct and is also known exactly. Therefore, we can use the inverse power method [30] for the computation of the eigenvector.

Given a non-zero starting vector $u^{(0)}$, inverse iteration generates a sequence of vectors $u^{(i)}$, generated recursively by the formula

$$u^{(i)} = \left(B - \hat{\lambda} I\right)^{-1} \frac{u^{(i-1)}}{\|u^{(i-1)}\|},$$

$i \geq 1$, where $\hat{\lambda} = \lambda + \epsilon$, λ is a distinct eigenvalue and $\epsilon \to 0$. The sequence of iterates $u^{(i)}$ converges to an eigenvector associated with an eigenvalue closest to $\hat{\lambda}$. Usually, the starting vector $u^{(0)}$ is chosen to be the normalized $(1, 1, \ldots, 1)^T$. Moreover, if this particular eigenvalue λ is known exactly, this method converges in a single iteration [14].

Since λ_1 is known exactly, we can find the analytical form for the corresponding eigenvector using the inverse iteration. The inverse iteration along with Eq. (18) yields

$$\begin{aligned} \hat{u_1} &= \left[\left(B_{k+1}^{(1)}\right)^2 - \gamma_1(\hat{\lambda}) B_{k+1}^{(1)} + \gamma_0(\hat{\lambda}) I\right] \frac{u}{\gamma(\hat{\lambda})} \\ &= -\gamma_u(\hat{\lambda})\, u + \gamma_{us}(\hat{\lambda})\, s_k - \gamma_{uy}(\hat{\lambda})\, y_k, \end{aligned} \tag{35}$$

where $\hat{\lambda} = -\lambda_1 + \epsilon$, $u = u^{(0)}/\|u^{(0)}\|$,

$$\begin{aligned}
\gamma_u(\hat{\lambda}) &= \left[1 - \gamma_1(\hat{\lambda})\,\theta_{k+1} + \gamma_0(\hat{\lambda})\,\theta_{k+1}^2\right] / \left[\gamma(\hat{\lambda})\,\theta_{k+1}^2\right], \\
\gamma_{us}(\hat{\lambda}) &= \left\{\left[1 - \gamma_1(\hat{\lambda})\,\theta_{k+1}\right] s_k^T u + \theta_{k+1} y_k^T u\right\} / \left[\gamma(\hat{\lambda})\,\theta_{k+1}^2 s_k^T s_k\right], \quad \text{and} \\
\gamma_{uy}(\hat{\lambda}) &= \left\{\left[1 - \gamma_1(\hat{\lambda})\,\theta_{k+1} + a_k\right] \theta_{k+1} y_k^T u - s_k^T u\right\} / \left[\gamma(\hat{\lambda})\,\theta_{k+1}^2 s_k^T y_k\right].
\end{aligned}$$

Therefore, $u_1 = \hat{u_1}/\|\hat{u_1}\|$ and the trial step is computed by the formula

$$d_{k+1} = -\gamma_g(\hat{\lambda})\,g_{k+1} + \gamma_s(\hat{\lambda})\,s_k - \gamma_y(\hat{\lambda})\,y_k + \tau u_1,$$

where τ is obtained by Eq. (3). In case where $a_k = 2$, using the eigendecomposition of $B^{(1)}$ we have $B^{(1)} = U\Lambda U^T$, where $U = I$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_1, \ldots, \lambda_1)$. Easily can be verified that a corresponding eigenvector to λ_1 is $u_1 = e_1 = (1, 0, \ldots, 0)^T$.

4.2 Computation of the step using the 2-BFGS update

4.2.1 The standard case

When $\lambda \in (\lambda_1, \infty)$ and the 2-BFGS update is performed, the trial step $d(\lambda) = -\left(B + \lambda I\right)^{-1} g$ is computed using Eq. (27), by the following procedure:

Procedure 4.1 *(Compute d)*

$d^{(4)} = g$;
for $j = 4, \ldots, 1$ **do**
  $v \leftarrow B^{(2)} d^{(j)}$;
  $d^{(j-1)} \leftarrow v + (-1)^{j-1} \nu_{j-1}(\lambda)\, g$;
end for
return $-d^{(0)}/\nu(\lambda)$;

The vector $v \in \mathbb{R}^n$ is computed by the formula

$$v \equiv B_{k+1}^{(2)} d^{(j)} = B_k^{(1)} d^{(j)} - \frac{s_k^T B_k^{(1)} d^{(j)}}{s_k^T B_k^{(1)} s_k} B_k^{(1)} s_k + \frac{y_k^T d^{(j)}}{s_k^T y_k} y_k,$$

while the vectors $B_k^{(1)} d^{(j)}$ and $B_k^{(1)} s_k$ are computed by means of (28), substituting v_i with $d^{(j)}$ and s_k, respectively. In case where $\lambda = 0$, then

$$\begin{aligned}
d_{k+1}(0) \equiv -\left(B_{k+1}^{(2)}\right)^{-1} g_{k+1} &= -\left(B_k^{(1)}\right)^{-1} g_{k+1} + \frac{w_k^T g_{k+1} - b_k s_k^T g_{k+1}}{s_k^T y_k} s_k \\
&+ \frac{s_k^T g_{k+1}}{s_k^T y_k} w_k
\end{aligned} \tag{36}$$

where $w_k = \left(B_k^{(1)}\right)^{-1} y_k$. In the special case where $a_{k-1} = a_k = 2$, $B_{k+1}^{(2)}$ has the form

$$B_{k+1}^{(2)} = \frac{1}{\theta_k} I + \left(\frac{1}{\theta_{k+1}} - \frac{1}{\theta_k}\right) \frac{y_k y_k^T}{y_k^T y_k}, \tag{37}$$

therefore, the trial step is obtained by the following equation:

$$d_{k+1}(\lambda) = -\left[\left(\lambda + \frac{1}{\theta_k} + \frac{1}{\theta_{k+1}}\right) I - B_{k+1}^{(2)}\right] \frac{g_{k+1}}{\left(\lambda + \frac{1}{\theta_k}\right)\left(\lambda + \frac{1}{\theta_{k+1}}\right)}.$$

Using the 2-BFGS update, the expression $d(\lambda)^T (B + \lambda I)^{-1} d(\lambda)$ of the denominator in Newton's iteration (32) becomes

$$\begin{aligned}
& d_{k+1}(\lambda)^T \left(B_{k+1}^{(2)} + \lambda I\right)^{-1} d_{k+1}(\lambda) = \\
& \frac{1}{\nu^2(\lambda)} \Big\{ \left[\lambda^4 + \lambda^3 \nu_3(\lambda) + \lambda^2 \nu_2(\lambda) + \lambda \nu_1(\lambda) + \nu_0(\lambda)\right] \|d_{k+1}(\lambda)\|^2 \\
+ \quad & \left[4\lambda^3 + 3\lambda^2 \nu_3(\lambda) + 2\lambda \nu_2(\lambda) + \nu_1(\lambda)\right] d_{k+1}(\lambda)^T g_{k+1} \\
- \quad & \left[2\lambda + \nu_3(\lambda)\right] g_{k+1}^T B_{k+1}^{(2)} g_{k+1} + \left[3\lambda^2 + 2\lambda \nu_3(\lambda)\right. \\
+ \quad & \left.\nu_2(\lambda)\right] \|g_{k+1}\|^2 + \|B_{k+1}^{(2)} g_{k+1}\|^2 \Big\},
\end{aligned}$$

while when $a_{k-1} = a_k = 2$, the above equation yields

$$d_{k+1}(\lambda)^T \left(B_{k+1}^{(2)} + \lambda I\right)^{-1} d_{k+1}(\lambda) = \frac{\left(2\lambda + \frac{1}{\theta_k} + \frac{1}{\theta_{k+1}}\right) \|d_{k+1}(\lambda)\|^2 + d_{k+1}(\lambda)^T g_{k+1}}{\left(\lambda + \frac{1}{\theta_k}\right)\left(\lambda + \frac{1}{\theta_{k+1}}\right)}.$$

4.2.2 The hard case

Proposition 3.5 along with Lemma 3.6 reveals that if λ_1 is a distinct eigenvalue of $B^{(2)}$, then either $B^{(2)}$ has at least two distinct eigenvalues or it has exactly one equals to $\lambda_1 = 1/\theta_{k+1}$. Therefore, in the hard case where the trial step is of the form

$$d_{k+1} = -\left(B_{k+1}^{(2)} + \hat{\lambda} I\right)^{-1} g_{k+1} + \tau u_1,$$

where $\hat{\lambda} = -\lambda_1 + \epsilon$, $\epsilon \to 0^+$, and τ is computed using Eq (3), in the first case (at least two distinct eigenvalues) we can find the eigenvector corresponds to λ_1 by applying the inverse iteration. The resulting eigenvector is $u_1 = \hat{u_1}/\|\hat{u_1}\|$, where

$$\hat{u_1} = \left[\left(B_{k+1}^{(2)}\right)^4 + \sum_{i=0}^{3} (-1)^i \nu_i(\hat{\lambda}) \left(B_{k+1}^{(2)}\right)^i\right] \frac{u}{\nu(\hat{\lambda})}, \tag{38}$$

$\hat{\lambda} = -\lambda_1 + \epsilon$ and $u = (1, \ldots, 1)^T/\|u\|$. The computation of $\hat{u_1}$ can be obtained using Procedure 4.1. In the latter case (one distinct eigenvalue), from Eq (37) it is straightforward to see that $u_1 = y_k/\|y_k\|$.
When λ_1 is a multiple eigenvalue, then $a_{k-1} = a_k = 2$ and $\lambda_1 = 1/\theta_k$. If $\theta_k \neq \theta_{k+1}$, from Eq (37) easily can be verified that the resulting eigenvector is of the form $u_1 = \hat{u_1}/\|\hat{u_1}\|$, where

$$\hat{u_1} = (-\frac{y_k^{(n)}}{y_k^{(1)}}, 0, \ldots, 0, 1)^T \tag{39}$$

and $y_k^{(i)}$ denotes the ith component of y_k. In the contrast, if $\theta_k = \theta_{k+1}$, then $u_1 = e_1 = (1, 0, \ldots, 0)^T$.

5 The trial step algorithm

In the previous section we have discussed how Newton's method can be applied for solving subproblem (2), when the approximate Hessian is computed by the memoryless BFGS formula. In case where $B - \lambda_1 I$ is singular, we have perturbed λ_1 by $\hat{\lambda} = -\lambda_1 + \epsilon$, where $\epsilon \to 0^+$, such that $B + \hat{\lambda} I$ is positive definite. This perturbation does not affect the eigenvector that corresponds to λ_1 (see [18]), while it does not influence the behavior of $d(\lambda)$ when $\lambda \to -\lambda_1$. Indeed, using the eigendecomposition of $B^{(m)} + \lambda I$ we have that $B^{(m)} + \lambda I = U\,(\Lambda + \lambda I)\,U^T$, where U is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. Thus, for $\lambda \neq \lambda_i$ we have

$$\|d(\lambda)\|^2 = \sum_{i=1}^{n} \frac{(u_i^T g)^2}{(\lambda_i + \lambda)^2},$$

where u_i denotes the ith column of U. Therefore, if $u_1^T g \neq 0$ for at least one eigenvector corresponding to λ_1, and $\lambda = -\lambda_1 + \epsilon$, then $\lim_{\lambda \to -\lambda_1} \|d(\lambda)\| = \lim_{\epsilon \to 0^+} \|d(-\lambda_1 + \epsilon)\| = \infty$.

The following algorithm is a unified algorithm that incorporates both the standard and the hard case and computes an approximate solution of the subproblem (2). In addition, the safeguarding scheme required for Newton's iteration (32) uses the parameters λ_L and λ_U such that $[\lambda_L, \lambda_U]$ is an interval of uncertainty which contains λ^*. Since the extreme eigenvalues of $B^{(m)}$ are known, from Eq. (5) we have that $\lambda_L = \max(0,\ -\lambda_1 + \epsilon)$, and $\lambda_U = \max|\lambda_i| + (1 + \epsilon)\|g\|/\Delta$, $i = 1, \ldots, n$. Clearly, the lower bound for λ is greater than $-\lambda_1$, which ensures that $B^{(m)} + \lambda I$ is always positive definite.

Algorithm 1 *(computation of the trial step)*

Step 1 *Compute the eigenvalues λ_i of $B^{(m)}$; given $\epsilon \to 0^+$, set $\lambda_L := \max(0, -\lambda_1 + \epsilon)$ and $\lambda_U := \max|\lambda_i| + (1 + \epsilon)\|g\|/\Delta$.*

Step 2 *If $\lambda_1 > 0$, then initialize λ by setting $\lambda := 0$ and compute $d(\lambda)$; if $\|d\| \leq \Delta$ stop; else go to Step 4.*

Step 3 *Initialize λ by setting $\lambda := -\lambda_1 + \epsilon$ such that $B + \lambda I$ is positive definite and compute $d(\lambda)$;*

a. if $\|d\| > \Delta$ go to Step 4;

b. if $\|d\| = \Delta$ stop;

c. if $\|d\| < \Delta$ compute τ and u_1 such that $\| - (B^{(m)} + \hat{\lambda} I)g + \tau u_1\| = \Delta$; set $d := d + \tau u_1$ and stop;

Step 4 *Use Newton to find $\lambda \in [\lambda_L, \lambda_U]$ and compute $d(\lambda)$;*

Step 5 *If $\|d\| \leq \Delta$ stop; else update λ_L and λ_U such that $\lambda_L \leq \lambda \leq \lambda_U$ and go to Step 4.*

When $m = 1$, the eigenvalues in Step 1 of Algorithm 1 can be computed by Eq. (15). In Step 2, the trial step is obtained by Eq. (30), while in Step 3 and 4 by means of Eq. (29). In Step 3(c) the computation of τ is obtained from Eq. (3), while if $a_k > 2$, u_1 is obtained using Eq. (35), otherwise we set $u_1 = e_1$.

When $m = 2$, the eigenvalues can be computed by means of Eq. (20). The computation of d in Step 2 is obtained by Eq. (36), while in Steps 3 and 4 by means of Procedure 4.1. In Step 3(c), τ is obtained from Eq. (3). Moreover, if $a_{k-1} > 2$ or $a_k > 2$, then u_1 is computed using Eq. (38), else if $a_{k-1} = a_k = 2$, then:
(1.) $u_1 = y_k/\|y_k\|$, if λ_1 is distinct eigenvalue,
(2.) $u_1 = \hat{u_1}/\|\hat{u_1}\|$, where u is defined in Eq (39), if λ_1 has multiplicity $n - 1$,
(3.) $u_1 = e_1$, if λ_1 has multiplicity n.
As we can see, the computation of the step does not require the Cholesky factorization of $B^{(m)} + \lambda^{(\ell)} I$. Thus, Algorithm 1 can solve inexpensively the subproblem, and therefore it can be iterate until convergence to the optimal λ is obtained with high accuracy. Moreover, the knowledge of the extreme eigenvalues, along with Lemma 2.2, results in a straightforward safeguarding procedure for λ.

The following lemma is established by Powell [24] and gives a lower bound for the maximum reduction $\phi(0) - \phi(d^*)$ of the quadratic model within the trust region $\|d\| \leq \Delta$.

Lemma 5.1 ([24]) *If d^* is a solution of the subproblem (2), then*

$$\phi(0) - \phi(d^*) \geq \frac{1}{2}\|g\| \min\{\Delta,\ \|g\|/\|B\|\}.$$

Global convergence theory of trust region algorithms requires that the trial step d_k must satisfy the following inequality

$$\phi_k(0) - \phi(d_k) \geq c\|g_k\| \min\{\Delta_k, \|g_k\|/\|B_k\|\} \tag{40}$$

for all k, where c is a positive constant [8, 20].

The following theorem shows that relation (40) is satisfied if the trial step is computed by Algorithm 1.

Theorem 5.2 *Assume that $\|B_k^{(m)}\| \leq M < \infty$, $m = 1, 2$, for all k, where M is a positive constant. If the trial step d_k is computed approximately by Algorithm 1, then*

$$\phi_k(0) - \phi_k(d_k(\lambda)) \geq \frac{1}{8}\|g_k\| \min\{\Delta_k,\ \|g_k\|/\|B_k^{(m)}\|\} \tag{41}$$

Proof. Proof. We recall that at each iteration k, the quadratic model is of the form

$$\phi_k(d_k) = d_k^T g_k + \frac{1}{2} d_k^T B_k^{(m)} d_k. \tag{42}$$

We consider the following two cases:
(i) If $B_k^{(m)} > 0$, then from Step 2, we have that if $\lambda = 0$ and $\|d_k\| \leq \Delta_k$, the algorithm stops. In this case,

$$g_k^T d_k = -g_k \left(B_k^{(m)}\right)^{-1} g_k \leq \frac{-\|g_k\|^2}{\|B_k^{(m)}\|}. \tag{43}$$

Then, from (42) and (43) we have that

$$\begin{aligned}
\phi_k(0) - \phi_k(d_k(\lambda)) &= -d_k(\lambda)^T g_k - \frac{1}{2} d_k^T(\lambda) B_k^{(m)} d_k(\lambda) \\
&= -d_k(\lambda)^T g_k - \frac{1}{2} d_k^T(\lambda)(B_k^{(m)} + \lambda I) d_k(\lambda) + \frac{1}{2}\lambda \|d_k(\lambda)\|^2 \\
&= -\frac{1}{2} d_k(\lambda)^T g_k + \frac{1}{2}\lambda \|d_k(\lambda)\|^2 \\
&\geq \frac{1}{2} \frac{\|g_k\|^2}{\|B_k^{(m)}\|}.
\end{aligned} \tag{44}$$

If $\lambda = 0$ and $\|d_k\| > \Delta_k$, Steps 4 and 5 of Algorithm 1 yield λ and d such that

$$0 < \lambda \leq \lambda_U \quad \text{and} \quad \|d_k(\lambda)\| \leq \Delta_k,$$

respectively. Since $B_k^{(m)}$ is positive definite, we have that $\|B_k^{(m)} + \lambda I\| = \lambda_n + \lambda = \|B_k^{(m)}\| + \lambda$, where λ_n is the largest eigenvalue of $B_k^{(m)}$. Therefore, taking into account that $\lambda \leq \|B_k^{(m)}\| + (1+\epsilon)\|g_k\|/\Delta_k$ and

$$\|B_k^{(m)} + \lambda I\| = \|B_k^{(m)}\| + \lambda \leq 2\|B_k^{(m)}\| + (1+\epsilon)\|g_k\|/\Delta_k \leq 2\left(\|B_k^{(m)}\| + \|g_k\|/\Delta_k\right),$$

we have that

$$\begin{aligned}
g_k^T d_k(\lambda) &\leq -\frac{\|g_k\|^2}{\|\left(B_k^{(m)} + \lambda I\right)^{-1}\|} \\
&\leq -\frac{1}{2} \frac{\|g_k\|^2}{\|B_k^{(m)}\| + \|g_k\|/\Delta_k} \\
&\leq -\frac{1}{4}\|g_k\| \min\left(\Delta_k, \|g_k\|/\|B_k^{(m)}\|\right).
\end{aligned} \tag{45}$$

Thus, the reduction of the model yields

$$\begin{aligned}\phi_k(0) - \phi_k(d_k(\lambda)) &= -\frac{1}{2} d_k(\lambda)^T g_k + \frac{1}{2}\lambda \|d_k(\lambda)\|^2 \\ &\geq \frac{1}{8}\|g_k\| \min\left(\Delta_k, \|g_k\|/\|B_k^{(m)}\|\right). \end{aligned} \tag{46}$$

(ii) If $B_k^{(m)} \leq 0$, then $d_k^T(\lambda) B_k^{(m)} d_k(\lambda) \leq 0$. From Step 3(a) along with Steps 4 and 5 we have that $\hat{\lambda} \leq \lambda \leq \lambda_U$, and $\|d_k(\lambda)\| \leq \Delta_k$, where $\hat{\lambda} = -\lambda_1 + \epsilon$. Therefore, using relation (45), for the reduction of the model we obtain

$$\begin{aligned}\phi_k(0) - \phi_k(d_k(\lambda)) &= -d_k(\lambda)^T g_k - \frac{1}{2} d_k^T(\lambda) B_k^{(m)} d_k(\lambda) \\ &\geq -d_k(\lambda)^T g_k \geq \frac{1}{4}\|g_k\| \min\left(\Delta_k, \|g_k\|/\|B_k^{(m)}\|\right). \end{aligned} \tag{47}$$

Finally, in Step 3(c), taking into account that $\lambda = \hat{\lambda}$, $u_1^T g_k = 0$ and $\|B_k^{(m)} + \hat{\lambda} I\| \leq \lambda_n + \hat{\lambda} \leq 2\|B_k^{(m)}\|$, where λ_n is the largest eigenvalue of B_k, we obtain

$$g_k^T d_k(\hat{\lambda}) = -g_k \left(B_k^{(m)} + \hat{\lambda} I\right)^{-1} g_k \leq -\frac{\|g_k\|^2}{\|B_k^{(m)} + \hat{\lambda}\|} \leq -\frac{\|g_k\|^2}{2\|B_k^{(m)}\|}. \tag{48}$$

Therefore, using relation (48) we have that

$$\begin{aligned}\phi_k(0) - \phi_k(d_k(\hat{\lambda})) &= -d_k(\hat{\lambda})^T g_k - \frac{1}{2} d_k^T(\hat{\lambda}) B_k^{(m)} d_k(\hat{\lambda}) \\ &\geq -d_k(\hat{\lambda})^T g_k \geq \frac{1}{2}\frac{\|g_k\|^2}{\|B_k^{(m)}\|}. \end{aligned} \tag{49}$$

Combining relations (44), (46), (47) and (49), relation (41) follows immediately. □

6 Numerical results

In this section we present numerical results to demonstrate the viability of our approach for large scale problems. Algorithm 1 have been used to compute an approximate solution of the trust region subproblem and it have been implemented within two trust region versions:

`PTR` A pure trust region algorithm with a particular strategy for updating the trust region radius (see [8, 20]). In this version, if the trial step d_k results in an increase in the objective function then the trust region radius Δ_k is reduced and the subproblem (2) is resolved.

`TR+BT` A trust region algorithm with backtracking line search proposed by Nocedal and Yuan [21]. Backtracking is performed only when the trial step increases the objective

Table 1: Numerical results for the PTR 1-BFGS algorithm in various dimensions.

	Dimension of the problem	$n = 10^3$	$n = 10^4$	$n = 10^5$	$n = 10^6$
No.	Function name	GE	GE	GE	GE
1.	Extended Beale	44	53	58	55
2.	Extended Himmelblau	18	23	32	28
3.	Extended Block-Diagonal BD1	42	48	49	45
4.	Extended Tridiagonal 1	37	32	34	37
5.	Generalized Tridiagonal 2	54	75	110	162
6.	Generalized quartic GQ1	23	15	14	18
7.	Generalized quartic GQ2	39	46	57	45
8.	Raydan 2	8	2	8	9
9.	Full Hessian	5	4	5	15
10.	Broyden Tridiagonal	74	46	119	313
11.	Diagonal 4	50	38	56	61
12.	Diagonal 6	8	2	8	9
13.	Diagonal 8	10	8	7	66
14.	DIXMAANA (CUTE)	8	10	12	13
15.	DIXMAANB (CUTE)	8	10	12	19
16.	DIXMAANC (CUTE)	9	12	13	21
17.	Extended DENSCHNB (CUTE)	15	18	15	20
18.	Extended DENSCHNF (CUTE)	36	24	30	36

function. Unlike PTR, this version does not resolve the subproblem when the trial step d_k results in an increase in the objective function, but instead it performs a line search procedure based on truncated quadratic interpolation. More analytically, if $f(x_k + d_k) \geq f(x_k)$, then a step length α_k is computed by the formula

$$\alpha_k = \max\left\{0.1,\, 0.5/\left[1 + (f(x_k) - f(x_k + d_k))/d_k^T g_k\right]\right\},$$

and the new direction is set to be $d_k := \alpha_k d_k$. This process is repeated until a lower function value is obtained.

The algorithms have been coded in MATLAB 7.3 and all numerical experiments were performed on a Pentium 1.86 GHz personal computer with 1GB of RAM running Linux operating system. Double precision IEEE floating point arithmetic with machine precision approximately 2.2×10^{-16} was employed. We have selected 18 large-scale unconstrained optimization test problems in extended or generalized form [1, 13, 16]. For each test function we have considered 4 numerical experiments with number of variables $n = 10^3, 10^4, 10^5, 10^6$. In both implementations mentioned above, we calculate the trial step using Algorithm 1 which is based on the analysis described in Sections 3 and 5. Both implementations are terminated when $\|g_k\| \leq 10^{-5}$, either the trust region radius is too small, i.e., $\Delta_k < 10^{-15}$, or the minimum of the quadratic subproblem is too close to zero, i.e., $|\phi(d_k)| \leq 10^{-20}$. Both

algorithms use exactly the same parameters $\Delta_0 = 100$ and $\epsilon = 10^{-5}$, while $B_k^{(m)}$ is allowed to be indefinite. Thus, the condition $s_k^T y_k > 0$ does not always hold and therefore, for B_k to be well defined, the storage of the vector pair $\{s_k, y_k\}$ is skipped if $|s_k^T y_k| \leq 10^{-12} \|s_k\| \, \|y_k\|$.

Table 2: Numerical results for the PTR 2-BFGS algorithm in various dimensions.

	Dimension of the problem	$n = 10^3$	$n = 10^4$	$n = 10^5$	$n = 10^6$
No.	Function name	GE	GE	GE	GE
1.	Extended Beale	32	18	23	24
2.	Extended Himmelblau	20	15	15	22
3.	Extended Block-Diagonal BD1	22	20	17	27
4.	Extended Tridiagonal 1	23	30	43	37
5.	Generalized Tridiagonal 2	47	81	96	326
6.	Generalized quartic GQ1	16	12	15	15
7.	Generalized quartic GQ2	36	37	31	40
8.	Raydan 2	8	2	8	9
9.	Full Hessian	5	4	5	15
10.	Broyden Tridiagonal	41	33	87	135
11.	Diagonal 4	7	12	10	15
12.	Diagonal 6	8	2	8	9
13.	Diagonal 8	10	8	7	66
14.	DIXMAANA (CUTE)	8	10	11	13
15.	DIXMAANB (CUTE)	8	10	12	20
16.	DIXMAANC (CUTE)	9	12	14	22
17.	Extended DENSCHNB (CUTE)	12	12	12	15
18.	Extended DENSCHNF (CUTE)	31	24	17	19

Tables 1 and 2 summarize the results of the PTR 1-BFGS and 2-BFGS algorithm, respectively, applied to 18 different problems with sizes from 10^3 to 10^6. The first row denotes the number of variables of each problem. For each test problem, we only give the number of gradient evaluations (GE), since this equals the number of iterations and function evaluations (i.e., there is exactly one function and gradient evaluation per iteration). To solve all the 18 problems, the CPU time (in seconds) required by the PTR 1-BFGS algorithm is 1.19, 3.94, 56.17, and 874.86, for each dimension, respectively, while the CPU time (in seconds) required by the PTR 2-BFGS algorithm is 1.56, 4.79, 82.41, and 2481.8.

Tables 3 and 4 summarize the results of the TR+BT 1-BFGS and 2-BFGS algorithm, respectively, applied to 18 different problems with sizes from 10^3 to 10^6. We list the number of function (FE) and gradient (GE) evaluations, since the number of iterations equals to the number of gradient evaluations. To solve all the 18 problems, the CPU time (in seconds) required by the TR+BT 1-BFGS algorithm is 1.19, 5.03, 41.57, and 842.4537, for each dimension, respectively. The CPU time (in seconds) required by the TR+BT 2-BFGS algorithm is 1.41, 5.65, 53.39, and 1530.8.

Table 3: Numerical results for the TR+BT 1-BFGS algorithm in various dimensions.

	Dimension of the problem	$n = 10^3$		$n = 10^4$		$n = 10^5$		$n = 10^6$	
No.	Function name	FE	GE	FE	GE	FE	GE	FE	GE
1.	Extended Beale	51	37	58	41	41	28	43	27
2.	Extended Himmelblau	18	15	19	16	36	36	26	24
3.	Extended Block-Diagonal BD1	35	25	33	25	43	33	37	29
4.	Extended Tridiagonal 1	27	21	41	33	39	33	59	50
5.	Generalized Tridiagonal 2	121	94	145	107	91	67	249	193
6.	Generalized quartic GQ1	23	20	15	15	14	14	18	18
7.	Generalized quartic GQ2	52	35	44	35	49	37	48	37
8.	Raydan 2	8	8	2	2	8	8	9	9
9.	Full Hessian	6	5	4	4	6	6	6	6
10.	Broyden Tridiagonal	47	40	105	79	42	34	115	86
11.	Diagonal 4	40	28	40	27	31	23	64	43
12.	Diagonal 6	8	8	2	2	8	8	9	9
13.	Diagonal 8	8	7	7	6	7	7	8	8
14.	DIXMAANA (CUTE)	8	8	10	10	12	12	14	14
15.	DIXMAANB (CUTE)	8	8	10	10	12	12	20	20
16.	DIXMAANC (CUTE)	9	9	12	12	13	13	22	22
17.	Extended DENSCHNB (CUTE)	19	18	18	18	14	14	17	17
18.	Extended DENSCHNF (CUTE)	37	26	29	24	41	33	33	27

The cumulative total for each performance metric (function and gradient evaluations, CPU time) over all problems does not seem to be too informative, since a small number of the most difficult problems can tend to dominate these results. For this reason, we use the performance profile proposed by Dolan and Moré [9] to display the performance of each implementation (solver) on the set of 18 test problems. The performance profile for a solver is the cumulative distribution function for a performance metric. Benchmark results are generated by running a solver on a set $\mathcal{P}$ of problems and recording information of interest such as the number of function or gradient evaluations and the CPU time. We use as a performance metric the CPU time

$$\text{Eeff} = \text{FE} + n \cdot \text{GE}, \tag{50}$$

where n denotes the number of variables. Let $t_{p,s}$ denote the time to solve problem $p \in \{1, \ldots, 18\}$ by solver $s \in \{$PTR 1-BFGS, PTR 2-BFGS, TR+BT 1-BFGS, TR+BT 2-BFGS$\}$. Define the performance ratio as

$$r_{p,s} = \frac{t_{p,s}}{t_p^*}$$

Table 4: Numerical results for the TR+BT 2-BFGS algorithm in various dimensions.

	Dimension of the problem	$n = 10^3$		$n = 10^4$		$n = 10^5$		$n = 10^6$	
No.	Function name	FE	GE	FE	GE	FE	GE	FE	GE
1.	Extended Beale	26	18	48	35	31	24	40	27
2.	Extended Himmelblau	18	16	19	17	17	13	16	15
3.	Extended Block-Diagonal BD1	21	18	23	19	22	19	24	22
4.	Extended Tridiagonal 1	30	25	42	25	56	41	30	24
5.	Generalized Tridiagonal 2	97	73	77	59	69	54	289	227
6.	Generalized quartic GQ1	16	16	12	12	15	15	14	14
7.	Generalized quartic GQ2	40	32	42	33	42	33	42	35
8.	Raydan 2	8	8	2	2	8	8	9	9
9.	Full Hessian	6	5	4	4	6	6	6	6
10.	Broyden Tridiagonal	41	34	83	68	42	34	112	86
11.	Diagonal 4	18	13	15	11	15	11	22	16
12.	Diagonal 6	8	8	2	2	8	8	9	9
13.	Diagonal 8	8	7	7	6	7	7	8	8
14.	DIXMAANA (CUTE)	8	8	10	10	11	11	13	13
15.	DIXMAANB (CUTE)	8	8	10	10	12	12	21	21
16.	DIXMAANC (CUTE)	9	9	12	12	14	14	23	23
17.	Extended DENSCHNB (CUTE)	12	11	12	12	14	14	15	15
18.	Extended DENSCHNF (CUTE)	24	19	23	21	17	17	25	25

where t_p^* is the best performance by any solver on problem p. Then, the performance profile is defined as the cumulative distribution function for the performance ratio $r_{p,s}$, which is

$$\rho_s(\tau) = \frac{1}{n_p} \text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\},$$

where n_p is the total number of problems. The function $\rho_s(\tau)$ is the probability for solver s that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio. If $\tau = 1$, then $\rho_s(1)$ represents the probability that the solver wins over the rest of the solvers, while $1 - \rho_s(\tau)$ is the fraction of problems that the solver cannot solve within a factor τ of the best solver.

Figure 1 shows the CPU time performance profiles for various dimensions of the four solvers (PTR 1-BFGS, PTR 2-BFGS, TR+BT 1-BFGS, TR+BT 2-BFGS) in the interval $[\log_2(1), \log_2(1024)]$. The interpretation of the results in Fig. 1 implies that algorithms PTR 1-BFGS and TR+BT 1-BFGS have the highest probability of being the optimal solvers. Fig. 1 shows that the CPU time needed to solve the quadratic subproblem does not affect the performance of PTR algorithm in contrast to TR+BT algorithm which does not resolve the subproblem. More analytically, comparing PTR 1-BFGS and TR+BT 1-BFGS we can see that both algorithms have the same performance, except of the last dimension (10^6) in which TR+BT 1-BFGS is the best solver. However if we choose within a factor of 7 of the

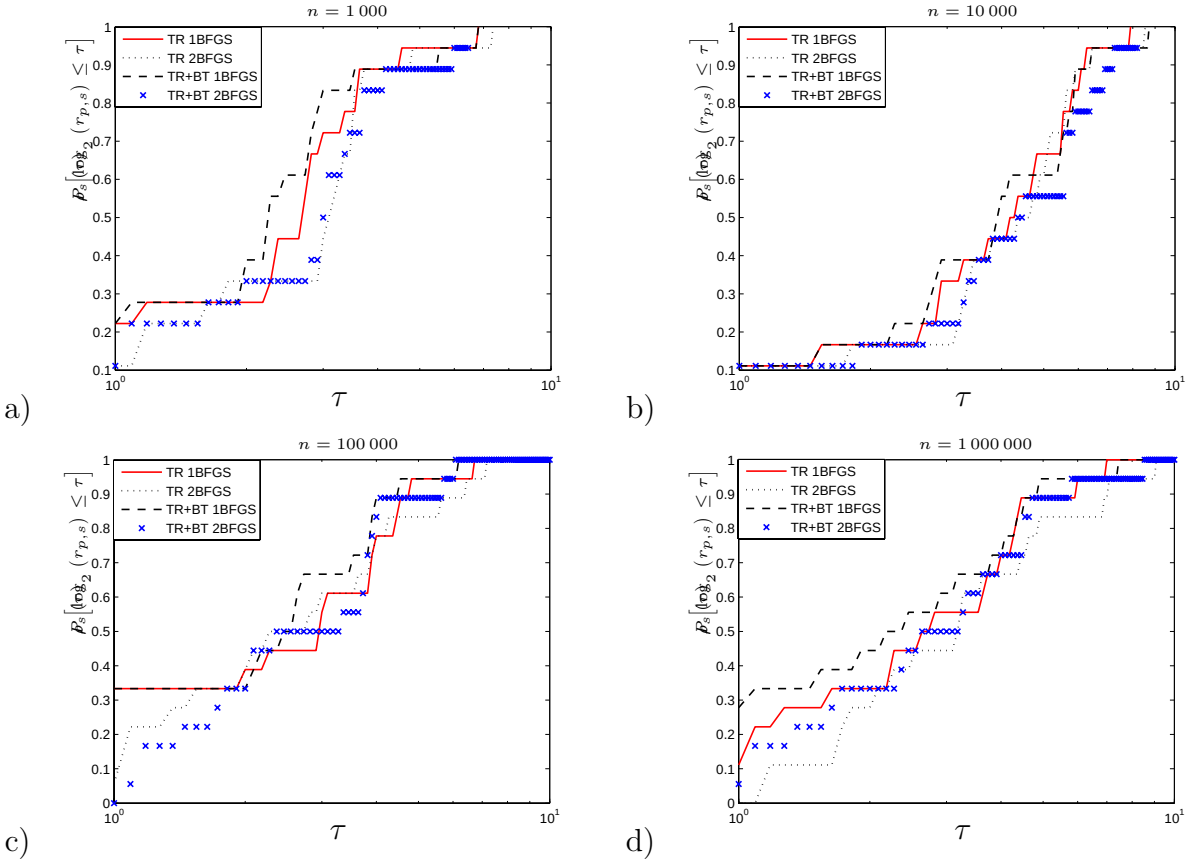

Figure 1: Log_2 scaled performance profile for CPU time

best solver, PTR 1-BFGS has 100% probability to solve all problems instead of TR+BT 1-BFGS which has 90% probability. Comparing PTR 2-BFGS and TR+BT 2-BFGS we can see that both algorithms have the same performance when $n = 10^3$ and $n = 10^4$. In Figure 1c) PTR 2-BFGS is a better solver than TR+BT 2-BFGS, while in the last dimension (10^6) TR+BT 2-BFGS performs better than PTR 2-BFGS.

7 Conclusions

We have shown how to use the 1- BFGS and 2-BFGS method in a trust-region framework. Our results have been based on the properties of the BFGS matrices for $m = 1, 2$. The eigenvalues can immediately be computed with high accuracy while the inverse of $B^{(m)} + \lambda I$ can be expressed in a closed form. The linear system which has to be solved for the computation of the trial step is always positive definite, resulting that the use of the Moore-Penrose generalized inverse is not necessary. Finally, the eigenvector corresponding to the smallest eigenvalue can be computed by applying one step of the power inverse method.

Therefore, the main contribution of this paper is to solve the quadratic subproblem without the use of factorization. Hence, very large problems up to 10^6 variables can be solved inexpensively.

Acknowledgements. We would like to thank Professors J. Nocedal and Y. Yuan for kindly providing us with their implemented algorithm described in [21].

References

[1] N. Andrei. Unconstrained optimization test functions: Algebraic expression. See `http://www.ici.ro/camo/neculai/SCALCG/testuo.pdf`.

[2] O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, 1996.

[3] J. Barzilai and J.M. Borwein. Two point step size gradient method. *IMA J. Numer. Anal.*, 8:141–148, 1988.

[4] E. G. Birgin and J. M. Martínez. A spectral conjugate gradient method for unconstrained optimization. *Appl. Math. Optim.*, 43:117–128, 2001.

[5] E.G. Birgin, J.M. Martínez, and M. Raydan. Nonmonotone spectral projected gradient methods on convex sets. *SIAM J. Optim.*, 10(6):1196–1211, 2000.

[6] P. Borwein and T. Erdélyi. *Polynomials and Polynomial Inequalities*, volume 161 of *Graduate Texts in Mathematics*. Springer-Verlag, NY, 1995.

[7] R. H. Burd, R. B. Schnabel, and G. A. Shuldz. Approximate solution of the trust region problem by minimization over two-dimensional subspaces. *Mathematical Programming*, 40:247–263, 1988.

[8] A.R. Conn, N.I.M. Gould, and Ph.L. Toint. *Trust-Region Methods*. MPS/SIAM Series on Optimization. SIAM, Philadelphia, USA, 2000.

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

[10] W. M. Faucette. A geometric interpretation of the solution of the general quartic polynomial. *American Mathematical Monthly*, 103(1):51–57, January 1996.

[11] D.M. Gay. Computing optimal locally constrained steps. *SIAM J. Sci. Stat. Comput.*, 2(2):186–197, 1981.

[12] E.M. Gertz. A quasi-newton trust-region method. *Mathematical Programming*, 100:447–470, 2004.

[13] I.Bongartz, A.R.Conn, N.I.M. Gould, and Ph.L. Toint. Cute: constrained and unconstrained testing enviroment. *ACM Transactions on Mathematical Software*, 21:123–160, 1995.

[14] I. C. F. Ipsen. Computing an eigenvector with inverse iteration. *SIAM Review*, 39(2):254–291, June 1997.

[15] D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[16] J.J. Moré, B.S. Garbow, and K.E. Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, 4:553–572, 1981.

[17] J.J. Moré and D.C. Sorensen. Computing a trust region step. *SIAM J. Sci. Stat. Comput.*, 4(3):553–572, 1983.

[18] K. Neymeyr. A note on inverse iteration. *Numerical Algebra with Applications*, 12:1–8, 2005.

[19] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comput.*, 35:773–782, 1980.

[20] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, New York, 1999.

[21] J. Nocedal and Y. Yuan. Combining trust region and line search techniques. In Y. Yuan, editor, *Advances in Nonlinear Programming*, pages 153–175. Kluwer, Dordrecht, The Netherlands, 1998.

[22] E. Polak. *Optimization*. Springer-Verlag, Ney York, 1997.

[23] M.J.D. Powell. A new algorithm for unconstrained optimization. In J.B. Rosen, O.L. Mangasarian, and K. Ritter, editors, *Nonlinear Programming*, pages 31–65. Academic Press, New York, NY, 1970.

[24] M.J.D. Powell. Convergence properties of a class of minimization algorithms. In O.L. Mangasarian, R.R. Meyer, and S.M. Robinson, editors, *Nonlinear Programming 2*, pages 1–27. Academic Press, New York, 1975.

[25] M. Raydan. On the Barzilai and Borwein choice of steplength for the gradient-method. *IMA J. Numer. Anal.*, 13(3):321–326, 1993.

[26] M. Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J. Optim.*, 7(1):26–33, 1997.

[27] G. A. Shuldz, R. B. Schnabel, and R. H. Burd. A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties. *SIAM J. Numer. Anal.*, 22(1):47–67, February 1985.

[28] D.C. Sorensen. Newton's method with a model trust region modification. *SIAM J. Numer. Anal.*, 19(2):409–426, 1982.

[29] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20(3):626–637, June 1983.

[30] J.H. Wilkinson. The calculation of the eigenvectors of codiagonal matrices. *Computer J.*, 1:90–96, 1958.

[31] J.H. Wilkinson. *The algebraic eigenvalue problem.* Oxford University Press, 1965.