Global Convergence of a New Hybrid Gauss-Newton Structured BFGS Method for Nonlinear Least Squares Problems

Weijun Zhou[1] and Xiaojun Chen[2]

Revised 10 April 2010

Abstract. In this paper, we propose a hybrid Gauss-Newton structured BFGS method with a new update formula and a new switch criterion for the iterative matrix to solve nonlinear least squares problems. We approximate the second term in the Hessian by a positive definite BFGS matrix. Under suitable conditions, global convergence of the proposed method with a backtracking line search is established. Moreover, the proposed method automatically reduces to the Gauss-Newton method for zero residual problems and the structured BFGS method for nonzero residual problems in a neighborhood of an accumulation point. Locally quadratic convergence rate for zero residual problems and locally superlinear convergence rate for nonzero residual problems are obtained for the proposed method. Some numerical results are given to compare the proposed method with some existing methods.

Keywords. Nonlinear least squares, Gauss-Newton method, BFGS method, structured quasi-Newton method, global convergence, quadratic convergence.
AMS subject classification. 90C06, 65K05

1 Introduction

This paper is devoted to solving the following nonlinear least squares problems

$$\min f(x) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(x) = \frac{1}{2}\|r(x)\|^2, \quad x \in R^n, \tag{1.1}$$

where $r(x) = (r_1(x), \cdots, r_m(x))^T$, $r_i : R^n \to R$ are twice continuously differentiable for $i = 1, \cdots, m$, and $||\cdot||$ denotes the Euclidean norm. It is clear that

$$\nabla f(x) = J(x)^T r(x), \quad \nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^{m} r_i(x)\nabla^2 r_i(x), \tag{1.2}$$

where $J(x)$ is the Jacobian matrix of $r(x)$. Throughout the paper, we denote

$$g(x) = \nabla f(x), \quad S(x) = \sum_{i=1}^{m} r_i(x)\nabla^2 r_i(x),$$
$$g_k = g(x_k), \quad J_k = J(x_k), \quad r_k = r(x_k), \quad s_k = x_{k+1} - x_k.$$

[1]Department of Mathematics, Changsha University of Science and Technology, Changsha 410004, China (weijunzhou@126.com). This author's work was supported by the Hong Kong Polytechnic University Postdoctoral Fellowship Scheme, the NSF foundation (10901026 and 10771057) of China and a project (09B001) of Scientific Research Fund of Hunan Provincial Education Department.

[2]Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China (maxjchen@polyu.edu.hk). This author's work was supported in part by the Hong Kong Research Grant Council.

Nonlinear least squares problems have wide applications such as data fitting, parameter estimate, function approximation, et al. [2, 31]. Most iterative methods using a line search are variants of Newton's method, which can be written in a general form:

$$x_{k+1} = x_k + \alpha_k d_k,$$

where $\alpha_k > 0$ is a stepsize given by some line search and d_k is a search direction satisfying

$$B_k d = -g_k,$$

where $B_k \in R^{n\times n}$ is an approximation of $\nabla^2 f(x_k)$.

The aim of this paper is to design a globally and locally fast convergent structured quasi-Newton algorithm with a backtracking line search for nonlinear least squares problems. Although trust region methods have been used to solve nonlinear least squares problems [6, 8, 31], which do not require a positive definite iteration matrix. For example, Dennis, Gay and Welsch [8] presented a quasi-Newton algorithm NL2SOL with trust region strategy. Numerical experiments show that quasi-Newton algorithm NL2SOL is efficient for large residual problems and the performance of NL2SOL is similar to that of the Levenberg-Marquardt algorithm for small residual problems [31]. However, in this paper, we only focus on line search approaches. Hence the need for B_{k+1} to be positive definite is necessary.

Traditional structured quasi-Newton methods are focused on the local convergence analysis. Their global convergence results have not been established. Li and Fukushima [22, 23] proposed two globally convergent modified BFGS methods for nonconvex unconstrained optimization. However, the Li-Fukushima methods have no quadratic convergence rate for least squares problems with zero residual problems, and the special structure of $\nabla^2 f(x_k)$ is not considered in their methods.

We recall some existing methods, especially structured quasi-Newton methods for solving nonlinear least squares problems. Nonlinear least squares problems can be regarded as a special case for unconstrained minimization with a special structure, and hence may be solved by unconstrained minimization methods. However, the cost of providing the complete Hessian matrix is often expensive. To reduce the cost, some methods use only the first derivative information such as the quasi-Newton method, in which B_{k+1} is given by

$$B_{k+1} = B_k + \text{Update}(s_k, y_k, B_k, v_k) \tag{1.3}$$

and satisfies the quasi-Newton equation $B_{k+1}s_k = y_k$ with

$$\text{Update}(s, y, B, v) = \frac{(y - Bs)v^T + v(y - Bs)^T}{v^T s} - \frac{(y - Bs)^T s}{(v^T s)^2} v v^T. \tag{1.4}$$

If $y_k = g_{k+1} - g_k$ and $v_k = y_k + \sqrt{\frac{y_k^T s_k}{s_k^T B_k s_k}} B_k s_k$, then B_{k+1} in (1.3) reduces to the standard BFGS formula for unconstrained optimization, that is, $B_{k+1} = \text{bfgs}(s_k, B_k, y_k)$, where

$$\text{bfgs}(s, B, y) = B - \frac{B s s^T B^T}{s^T B s} + \frac{y y^T}{y^T s}. \tag{1.5}$$

The BFGS formula has been regarded as one of the most efficient quasi-Newton methods in practical computations [4, 7, 11, 22, 23, 24]. A very nice property of the BFGS update is that, if B is symmetric and positive definite, then $B_+ = \text{bfgs}(s, B, y)$ is also symmetric and positive definite whenever $y^T s > 0$. However, this method ignores the special structure of the Hessian and does not use the available term $J_k^T J_k$ in $\nabla^2 f(x_k)$.

Past methods improve local convergence properties by exploiting the presence of the first order term $J_k^T J_k$ in Hessian; for example, the Gauss-Newton type methods(or the Levenberg-Marquardt type methods) [16, 19, 31, 36] are typical methods using the special structure of the Hessian matrix, whose iterative matrix is given by $B_k = J_k^T J_k + \mu_k I$ with $\mu_k \geq 0$. It is well-known that these methods have locally quadratic convergence rate for zero residual problems and linear convergence rate for small residual problems. However, these methods may perform poorly, even diverge for large residual problems [1], since they only use the first order information of f.

There are two ways to overcome this difficulty. One way is to combine the term $J_k^T J_k$ with the BFGS formula to improve convergence rate for zero residual problems and the efficiency of the BFGS method for general unconstrained optimization, for instance, hybrid methods in [1, 17, 18]. Specifically, Fletcher and Xu [18] proposed an efficient hybrid method for solving (1.1), that is, the matrix B_{k+1} is updated by the following rule: for a given constant $\epsilon \in (0, 1)$,

$$B_{k+1} = \begin{cases} J_{k+1}^T J_{k+1} + \|r_{k+1}\| I, & \text{if} \quad (f(x_k) - f(x_{k+1}))/f(x_k) \geq \epsilon, \\ \text{bfgs}(s_k, B_k, \hat{y}_k), & \text{otherwise}, \end{cases} \tag{1.6}$$

where

$$\hat{y}_k = J_{k+1}^T J_{k+1} s_k + (J_{k+1} - J_k)^T r_{k+1} \approx \nabla^2 f(x_{k+1}) s_k.$$

Suppose that $x_k \to x^*$ and $\nabla^2 f(x^*)$ is positive definite. If $f(x^*) \neq 0$, then

$$\lim_{k \to \infty} (f(x_k) - f(x_{k+1}))/f(x_k) = 0.$$

If $f(x^*) = 0$ and $x_k \to x^*$ superlinearly, then

$$\lim_{k \to \infty} (f(x_k) - f(x_{k+1}))/f(x_k) = 1.$$

Hence, the role of the term $(f(x_k) - f(x_{k+1}))/f(x_k)$ is to switch between zero residual and nonzero residual problems. This method converges quadratically for zero residual problems and superlinearly for nonzero residual problems. However, global convergence results for this method have not been given in [18].

The other way is to use the second order information of f sufficiently. For instance, structured quasi-Newton methods in [9, 13]. An important concept for structured quasi-Newton methods for nonlinear least squares problems is the Structure Principle [9].

- Structure Principle : Given $B_k = J_k^T J_k + A_k$ as an approximation to $\nabla^2 f(x_k)$, we want $B_{k+1} = J_{k+1}^T J_{k+1} + A_{k+1}$ to be an approximation of $\nabla^2 f(x_{k+1})$.

Because $\nabla^2 f(x_k) = J_k^T J_k + S(x_k)$ from (1.2), by the structure principle, A_k and A_{k+1} are approximations of $S(x_k)$ and $S(x_{k+1})$, respectively. A popular way to compute B_{k+1} was given in [9], that is,

$$B_{k+1} = B_k^s + \text{Update}(s_k, y_k^s, B_k^s, v_k),$$

$$B_k^s = J_{k+1}^T J_{k+1} + A_k, \quad y_k^s = \bar{y}_k + J_{k+1}^T J_{k+1} s_k,$$

where $\bar{y}_k$ is an approximation of $S(x_{k+1})s_k$ and is often chosen as $\bar{y}_k = (J_{k+1} - J_k)^T r_{k+1}$, and $\text{Update}(s_k, y_k^s, B_k^s, v_k)$ is given by (1.4). The structure principle can be achieved by updating A_{k+1} with the following secant update formula:

$$A_{k+1} = A_k + \text{Update}(s_k, \bar{y}_k, A_k, v_k).$$

The structured quasi-Newton methods possess only locally superlinear convergence rate for both zero and nonzero residual problems. In order to improve convergence rate of the structured quasi-Newton method for zero residual problems, Huschens [21] proposed a product structure type update, that is, B_k and B_{k+1} are defined by

$$B_k = J_k^T J_k + \|r_k\| A_k, \quad B_{k+1} = J_{k+1}^T J_{k+1} + \|r_{k+1}\| A_{k+1}.$$

This update formula was proved to have quadratic convergence rate for zero residual problems and superlinear convergence rate for nonzero residual problems. Although these methods possess locally fast convergence rate, the iterative matrix B_{k+1} can not preserve positive definiteness even if B_k is positive definite. Hence the search direction may not be a descent direction of f. Particularly, the Wolfe line search and Armijo line search [31] can not be used directly. Therefore, global convergence is not easy to be obtained.

To guarantee the positive definite property of $J_k^T J_k + A_k$, some factorized structured quasi-Newton methods were proposed in [26, 32, 33, 34] where

$$B_k = (J_k + L_k)^T (J_k + L_k),$$

and L_k is updated according to certain quasi-Newton formula. Then B_k is at least semi-positive definite.

Under suitable conditions, the matrix $(J_k + L_k)^T (J_k + L_k)$ is positive definite if the initial point is close to a solution point. These methods also have locally superlinear convergence rate for both zero and nonzero residual problems, but do not possess quadratic convergence rate for zero residual problems. In [37], Zhang et al. proposed a family of scaled factorized quasi-Newton methods based on the idea of [21]

$$B_k = (J_k + \|r_k\| L_k)^T (J_k + \|r_k\| L_k),$$

which not only has superlinear convergence rate for nonzero residual problems, but also have quadratic convergence rate for zero residual problems. However, global convergence has not been studied in [21, 37].

There are two main obstacles for global convergence of the above structured quasi-Newton methods with some line search. One is that the iterative matrices B_k may not be positive definite if the point x_k is far from the solution points. Another is that the iterative matrices B_k and their inverses B_k^{-1} are not uniformly bounded. So far, the study of structured quasi-Newton methods is focused on the local convergence rate [33, 34, 35, 37], but global convergence results have not been established.

In this paper, we propose a globally and locally fast convergent hybrid structured BFGS method. The idea of the paper is to approximate the second term in the Hessian, $S(x_k)$, by a positive definite BFGS matrix. The proposed strategy is using a combination of [18] and [9, 13], i.e. not only seeks to reduce to the Gauss-Newton method for zero residual problems as in [18] using a hybridization scheme, but also uses the BFGS method to estimate the second-order term $S(x)$ within the Hessian as in [9, 13], i.e. applying the structure principle. Further, a novel switch for between the Gauss-Newton method and the BFGS method is being proposed.

In the next section, we explain the approach in our method and present the algorithm in detail. Moreover, we prove that this method not only converges globally, but also converges quadratically for zero residual problems and superlinearly for nonzero residual problems. We present numerical results to compare its performance with the Gauss-Newton method and the Fletcher-Xu hybrid method [18], in Section 3 and Appendix.

2 Algorithm and convergence analysis

In this section, we present a new hybrid Gauss-Newton structured BFGS method for the problem (1.1), and give global and local convergence analysis for the method. We first illustrate our approach which is mainly based on the following consideration.

Since $J_k^T J_k$ is available in $\nabla^2 f(x_k)$, we hope to preserve this term unchanged in B_k. According to the structure principle, we approximate $S(x_k)$ using first order information and BFGS updates. From the observation:

$$S(x_{k+1})s_k = \Big(\sum_{i=1}^{m} r_i(x_{k+1})\nabla^2 r_i(x_{k+1})\Big)s_k \approx (J_{k+1} - J_k)^T r_{k+1}\|r_{k+1}\|/\|r_k\|,$$

we have the following lemma.

Lemma 2.1. *Let*

$$z_k = (J_{k+1} - J_k)^T r_{k+1}\|r_{k+1}\|/\|r_k\|, \tag{2.1}$$

then $z_k \approx S(x_{k+1})s_k$.

The first order term z_k was also used as a good approximation of $S(x_{k+1})s_k$ in [21, 37]. Moreover, in our numerical experiments, using z_k is more efficient than using the standard term $\bar{y}_k = (J_{k+1} - J_k)^T r_{k+1}$. Hence we construct

$$A_{k+1} = \text{bfgs}(s_k, A_k, z_k), \quad B_{k+1} = J_{k+1}^T J_{k+1} + A_{k+1}.$$

Lemma 2.2. *Suppose* $\nabla^2 r_i(x_k)$ *is bounded for* $i = 1, 2, \cdots, m$ *and* $x_k \to x^*$. *If* $f(x^*) = 0$, *then* $(z_k^T s_k)/(s_k^T s_k) \to 0$ *as* $k \to \infty$. *If* $f(x^*) \neq 0$ *and* $S(x^*)$ *is positive definite, then there is a positive constant* ϵ *such that* $(z_k^T s_k)/(s_k^T s_k) \geq \epsilon$ *for sufficiently large* k.

Proof. If $f(x^*) = 0$, then from $f(x_{k+1}) < f(x_k)$ we have that

$$(z_k^T s_k)/(s_k^T s_k) = \|r_{k+1}\|/\|r_k\| \sum_{i=1}^{m} r_i(x_{k+1})(\nabla r_i(x_{k+1}) - \nabla r_i(x_k))^T s_k / s_k^T s_k \to 0.$$

If $f(x^*) \neq 0$ and $S(x^*)$ is positive definite, then for sufficiently large k

$$(z_k^T s_k)/(s_k^T s_k) \geq \epsilon,$$

where $0 < \epsilon \leq \frac{1}{2}\lambda_{min}(S(x^*))$ is a constant and $\lambda_{min}(S(x^*))$ is the smallest eigenvalue of the matrix $S(x^*)$. □

Lemma 2.2 implies that $(z_k^T s_k)/(s_k^T s_k)$ can play a similar role as $(f(x_k) - f(x_{k+1}))/f(x_k)$ in (1.6). Therefore we can use the term $(z_k^T s_k)/(s_k^T s_k)$ to construct some hybrid methods. Moreover, the condition $(z_k^T s_k)/(s_k^T s_k) \geq \epsilon$ also gives a way to ensure the positive definiteness of the update matrix B_{k+1}. Now we give the definition of the update matrix.

Definition 2.1.

$$B_{k+1} = \begin{cases} J_{k+1}^T J_{k+1} + A_{k+1}, & \textit{if} \quad (z_k^T s_k)/(s_k^T s_k) \geq \epsilon, \\ J_{k+1}^T J_{k+1} + \|r_{k+1}\| I, & \textit{otherwise}, \end{cases} \tag{2.2}$$

where

$$A_{k+1} = \begin{cases} A_k - \frac{A_k s_k s_k^T A_k^T}{s_k^T A_k s_k} + \frac{z_k z_k^T}{z_k^T s_k}, & \textit{if} \quad (z_k^T s_k)/(s_k^T s_k) \geq \epsilon, \\ A_k, & \textit{otherwise}. \end{cases} \tag{2.3}$$

Since $A_{k+1} s_k = z_k$ when $(z_k^T s_k)/(s_k^T s_k) \geq \epsilon$, A_{k+1} is an approximation of $S(x_{k+1})$. Based on the above discussion, we now can present the hybrid Gauss-Newton structured BFGS method with a backtracking line search for nonlinear least squares problems (1.1).

Algorithm 2.1 (GN-SBFGS Method)

Step 1. Give a starting point $x_0 \in R^n$, a symmetric and positive definite matrix $A_0 \in R^{n \times n}$, scalars δ, $\rho \in (0,1)$, $\epsilon > 0$. Set $B_0 = J_0^T J_0 + A_0$. Let $k := 0$.

Step 2. Compute d_k by solving the following linear equations

$$B_k d = -g_k. \tag{2.4}$$

Step 3. Compute the stepsize α_k by the following backtracking line search, that is, $\alpha_k = \max\{\rho^0, \rho^1, \cdots\}$ satisfying

$$f(x_k + \rho^m d_k) \leq f(x_k) + \delta \rho^m g_k^T d_k. \tag{2.5}$$

Step 4. Let $x_{k+1} = x_k + \alpha_k d_k$.

Step 5. Update B_{k+1} by the formulas (2.2) and (2.3).

Step 6. Let $k := k + 1$ and go to Step 2.

Remark 2.1. Since A_0 is symmetric and positive definite and A_{k+1} is defined by the BFGS formula, A_{k+1} is also symmetric and positive definite whenever $z_k^T s_k > 0$. Therefore for every k, A_k and B_k in Algorithm 2.1 are symmetric and positive definite. Hence the search direction d_k is a descent direction, that is, $g_k^T d_k < 0$. This also shows that Algorithm 2.1 is well-defined.

In the global convergence of Algorithm 2.1, we use the following assumption.

Assumption A.

(I) The level set $\Omega = \{x \in R^n | f(x) \leq f(x_0)\}$ is bounded.

(II) In an open set N containing Ω, there exists a constant $L_1 > 0$ such that

$$\|J(x) - J(y)\| \leq L_1 \|x - y\|, \quad \forall x, y \in N. \tag{2.6}$$

It is clear that the sequence $\{x_k\}$ generated by Algorithm 2.1 is contained in Ω, and the sequence $\{f(x_k)\}$ is a descent sequence and has a limit f^*, that is,

$$\lim_{k\to\infty} f(x_k) = f^*. \tag{2.7}$$

In addition, we get from Assumption A that there are two positive constants L and γ, such that

$$\|g(x) - g(y)\| \le L\|x - y\|, \quad \|g(x)\| \le \gamma, \quad \forall\, x, y \in \Omega. \tag{2.8}$$

Now we give the following useful lemmas for our global convergence analysis.

Lemma 2.3. *Let Assumption A hold. Then we have*

$$\lim_{k\to\infty} \alpha_k g_k^T d_k = 0. \tag{2.9}$$

Proof. It follows directly from the line search (2.5), (2.7) and $g_k^T d_k < 0$. $\square$

Lemma 2.4. [3, Lemma 4.1] *There exists a constant $c_1 > 0$ such that*

$$\alpha_k = 1 \quad or \quad \alpha_k \ge c_1(-g_k^T d_k)/\|d_k\|^2. \tag{2.10}$$

Lemma 2.5. *Let Assumption A hold. Then for any $p \in (0, 1)$, there are positive constants $\beta_i, i = 1, 2, 3, 4$ such that*

$$\beta_1\|s_j\| \le \|A_j s_j\| \le \beta_2\|s_j\|, \quad \beta_3\|s_j\|^2 \le s_j^T A_j s_j \le \beta_4\|s_j\|^2 \tag{2.11}$$

hold for at least $\lceil pk \rceil$ values of $j \in [1, k]$.

Proof. By (2.6) and $\|r_{k+1}\| < \|r_k\|$, there exists a positive constant c_2 such that $(z_k^T s_k)/(s_k^T s_k) \le c_2$. Then the conclusion follows directly from the update formula A_{k+1} in (2.3) and Theorem 2.1 in [3]. $\square$

Theorem 2.1. *Let Assumption A hold and the sequence $\{x_k\}$ be generated by Algorithm 2.1. Then we have*

$$\liminf_{k\to\infty} \|g_k\| = 0.$$

Proof. Suppose the conclusion is not true, then there exist three positive constants η_1, η_2 and ε_0 such that for all k,

$$\eta_1 \ge \|r_k\| \ge \eta_2, \quad \|g_k\| \ge \varepsilon_0. \tag{2.12}$$

In fact, if $\liminf_{k\to\infty} \|r_k\| = 0$, then $\liminf_{k\to\infty} \|g_k\| = 0$.

Denote $K = \{k | z_{k-1}^T s_{k-1}/s_{k-1}^T s_{k-1} < \epsilon\}$. Since for any $k \in K$, $B_k = J_k^T J_k + \|r_k\| I$, by (2.4) we have

$$-g_k^T d_k = \|J_k d_k\|^2 + \|r_k\|\|d_k\|^2 \ge \eta_2\|d_k\|^2. \tag{2.13}$$

If K is infinite, then by Lemma 2.3, we have $\lim_{k\to\infty, k\in K} \alpha_k g_k^T d_k = 0$.

If $\liminf_{k\to\infty, k\in K} \alpha_k > 0$, then $\lim_{k\to\infty, k\in K} g_k^T d_k = 0$. Hence from (2.13) we get that $\lim_{k\to\infty, k\in K} \|d_k\| = 0$. On the other hand, from (2.4) and the first inequality of (2.12) we have that for $k \in K$,

$$\|g_k\| \le \|J_k^T J_k\|\|d_k\| + \eta_1\|d_k\| \to 0.$$

This leads to a contradiction to the second inequality of (2.12). If $\liminf_{k\to\infty,k\in K}\alpha_k = 0$, then from Lemma 2.4 we have that $-g_k^T d_k/\|d_k\|^2 \to 0$, contradicting (2.13).

Now we assume K is finite, then there exists an integer k_0 such that for all $k > k_0$, $B_k = J_k^T J_k + A_k$. By Lemma 2.5 and $s_k = \alpha_k d_k$, we have for infinite $k > k_0$,

$$-g_k^T d_k = \|J_k d_k\|^2 + d_k^T A_k d_k \geq \beta_3 \|d_k\|^2,$$

$$\|g_k\| \leq \|J_k^T J_k\|\|d_k\| + \|A_k d_k\| \leq \|J_k^T J_k\|\|d_k\| + \beta_2\|d_k\|.$$

Using the similar argument as the above, we also can get the same contradiction as the case that K is infinite. This finishes the proof. □

Theorem 2.1 shows that Algorithm 2.1 is globally convergent for nonlinear least squares problems (1.1). Now we turn to discussing local convergence rate of Algorithm 2.1. To do this, we need the following assumptions.

Assumption B.

(I) $\{x_k\}$ converges to x^* where $g(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite.

(II) $\nabla^2 f$ is Lipschitz continuous near x^*, that is, there exists a constant L_2 such that

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L_2\|x - y\| \tag{2.14}$$

for any x, y in a neighborhood of x^*.

We first present the following local convergence theorem of Algorithm 2.1 for zero residual problems.

Theorem 2.2. *Let Assumption B hold. Suppose problem (1.1) is a zero residual problem, then Algorithm 2.1 reduces to the Gauss-Newton method and x_k converges to x^* quadratically.*

Proof. Because $f(x^*) = 0$, we have $r(x^*) = 0$. By (2.1) and $f(x_{k+1}) < f(x_k)$, we have

$$\|z_k\| = \left\| \|r_{k+1}\|/\|r_k\|(J_{k+1} - J_k)^T r_{k+1} \right\| < \|r_{k+1}\| L_1 \|s_k\|.$$

Hence

$$\left| (z_k^T s_k)/(s_k^T s_k) \right| \leq L_1\|r_{k+1}\| \to 0, \tag{2.15}$$

which shows that there exists an integer k_1 such that for all $k > k_1$,

$$(z_k^T s_k)/(s_k^T s_k) < \epsilon.$$

This implies $B_k = J_k^T J_k + \|r_k\| I$ for $k > k_1$, that is, Algorithm 2.1 reduces to the Gauss-Newton method. Since

$$\nabla^2 f(x^*) = J(x^*)^T J(x^*) + S(x^*) = J(x^*)^T J(x^*) + \sum_{i=1}^m r_i(x^*)\nabla^2 r_i(x^*) = J(x^*)^T J(x^*),$$

Assumption B implies $J(x^*)^T J(x^*)$ is positive definite. Therefore quadratic convergence of the proposed method follows directly from the corresponding theory of the standard Gauss-Newton method, for example, see [31]. This completes the proof. □

In the rest of this section, we assume $f(x^*) \neq 0$, that is, the problem (1.1) is a nonzero residual problem. Firstly, we give the following result on the boundedness of B_k.

Lemma 2.6. *There exist some positive constants $\beta_i, i = 5, 6, 7$ such that*

$$\|B_j s_j\| \le \beta_5 \|s_j\|, \quad \beta_6 \|s_j\|^2 \le s_j^T B_j s_j \le \beta_7 \|s_j\|^2 \tag{2.16}$$

hold for at least $\lceil \frac{k}{2} \rceil$ values of $j \in [1, k]$.

Proof. Denote $K = \left\{ j \in [1, k] \middle| z_{j-1}^T s_{j-1} / s_{j-1}^T s_{j-1} < \epsilon \right\}$. Then for all $j \in K$, $B_j = J_j^T J_j + \|r_j\| I$ is uniformly positive definite since $f(x^*) \neq 0$ implies that there exist two positive constants η_3 and η_4 such that $\eta_3 \le \|r_j\| \le \eta_4$. Hence the inequalities in (2.16) hold for all $j \in K$ from the semi-positive definiteness of $J_j^T J_j$. If $|K| \ge \lceil \frac{k}{2} \rceil$, then we obtain the desirable results.

Now we suppose $|K| < \lceil \frac{k}{2} \rceil$. For $j \notin K$, we have $B_j = J_j^T J_j + A_j$. It follows from Lemma 2.5 that the inequalities in (2.16) hold for at least $\lceil \frac{k - |K|}{2} \rceil \ge \lceil \frac{k}{2} \rceil - |K|$ indices j in $[1, k]$.

Therefore the inequalities in (2.16) hold for at least $\lceil \frac{k}{2} \rceil - |K| + |K| = \lceil \frac{k}{2} \rceil$ indices j in $[1, k]$. This completes the proof. □

Lemma 2.7. *There exist two positive constants η_5 and η_6 such that at each iteration either*

$$f(x_k + \alpha_k d_k) \le f(x_k) - \eta_5 (g_k^T d_k)^2 / \|d_k\|^2,$$

or

$$f(x_k + \alpha_k d_k) \le f(x_k) + \eta_6 g_k^T d_k.$$

Proof. It follows from Lemma 2.4, the line search (2.5) and $g_k^T d_k < 0$ directly. □

Lemma 2.8. [3, Theorem 3.1] *Let Assumption B hold. Then we have*

$$\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty. \tag{2.17}$$

Lemma 2.9. *Let Assumption B hold. We also suppose that $S(x^*)$ is positive definite and $\nabla^2 r_i(x)$ is Lipschitz continuous near x^* for $i = 1, 2, \cdots, m$. Then for sufficiently large k there exists a positive constant M such that*

$$\|z_k - S(x^*) s_k\| / \|s_k\| \le M \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\}.$$

Proof. By (2.1) we have

$$\begin{aligned}
& \|z_k - S(x^*) s_k\| \\
= & \left\| \|r_{k+1}\| / \|r_k\| \sum_{i=1}^m r_i(x_{k+1}) (\nabla r_i(x_{k+1}) - \nabla r_i(x_k)) - \sum_{i=1}^m r_i(x^*) \nabla^2 r_i(x^*) s_k \right\| \\
\le & \left\| \Big(\|r_{k+1}\| / \|r_k\| - 1 \Big) \sum_{i=1}^m r_i(x_{k+1}) (\nabla r_i(x_{k+1}) - \nabla r_i(x_k)) \right\| \\
& + \left\| \sum_{i=1}^m (r_i(x_{k+1}) - r_i(x^*)) (\nabla r_i(x_{k+1}) - \nabla r_i(x_k)) \right\| \\
& + \left\| \sum_{i=1}^m r_i(x^*) ((\nabla r_i(x_{k+1}) - \nabla r_i(x_k)) - \nabla^2 r_i(x^*) s_k) \right\| \\
\stackrel{\triangle}{=} & A_1 + A_2 + A_3.
\end{aligned}$$

From the assumptions, there exist a small positive number δ_0 and some constants $c_i > 0$, $i = 0, 1, 2, 3, 4$ such that $\|r(x)\| > c_0$, $|r_i(x)| \le c_1$, $\|r(x) - r(y)\| \le c_2\|x - y\|$, $\|\nabla r_i(x) - \nabla r_i(y)\| \le c_3\|x - y\|$, $\|\nabla^2 r_i(x) - \nabla^2 r_i(y)\| \le c_4\|x - y\|$ for all $x, y \in \{u | \|u - x^*\| \le \delta_0\}$ and $i = 1, \cdots, m$. Therefore for sufficiently large k we have

$$\begin{aligned}
A_1 &\le \|r_{k+1} - r_k\|/\|r_k\| \sum_{i=1}^{m} |r_i(x_{k+1})| \|\nabla r_i(x_{k+1}) - \nabla r_i(x_k)\| \\
&\le \frac{mc_1c_2c_3}{c_0} \|x_{k+1} - x_k\|^2 \\
&\le \frac{2mc_1c_2c_3}{c_0} \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \|s_k\|, \\
A_2 &\le mc_2c_3 \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \|s_k\|, \\
A_3 &\le c_1 \sum_{i=1}^{m} \|\nabla r_i(x_{k+1}) - \nabla r_i(x_k) - \nabla^2 r_i(x^*) s_k\| \\
&= c_1 \sum_{i=1}^{m} \Big\| \int_0^1 \big(\nabla^2 r_i(x_k + ts_k) - \nabla^2 r_i(x^*)\big) s_k dt \Big\| \\
&\le 2mc_1c_4 \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \|s_k\|.
\end{aligned}$$

Set $M = \frac{2mc_1c_2c_3}{c_0} + mc_2c_3 + 2mc_1c_4$, then we have

$$\|z_k - S(x^*)s_k\|/\|s_k\| \le M \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\}.$$

This finishes the proof. □

Lemma 2.10. [3, Theorem 3.2] *Under the assumptions of Lemma 2.9, we have*

$$\lim_{k\to\infty} \|(A_k - S(x^*))s_k\|/\|s_k\| = 0.$$

Moreover, the sequences $\{\|A_k\|\}$ *and* $\{\|A_k^{-1}\|\}$ *are uniformly bounded.*

The following lemma shows that the Dennis-Moré condition holds.

Lemma 2.11. *Suppose the assumptions of Lemma 2.9 hold and the positive constant* ϵ *in Algorithm 2.1 satisfies* $\epsilon \le \frac{1}{2}\lambda_{min}(S(x^*))$. *Then we have*

$$\lim_{k\to\infty} \|(B_k - \nabla^2 f(x^*))s_k\|/\|s_k\| = 0. \tag{2.18}$$

Moreover, the sequences $\{\|B_k\|\}$ *and* $\{\|B_k^{-1}\|\}$ *are uniformly bounded.*

Proof. It is clear that the assumptions imply that for all sufficiently large k, $B_k = J_k^T J_k + A_k$, that is, Algorithm 2.1 reduces to a structured BFGS method. Hence, from Lemma 2.10 we have

$$\begin{aligned}
&\lim_{k\to\infty} \|(B_k - \nabla^2 f(x^*))s_k\|/\|s_k\| \\
=\ &\lim_{k\to\infty} \|(J_k^T J_k - J(x^*)^T J(x^*))s_k + (A_k - S(x^*))s_k\|/\|s_k\| \\
\le\ &\lim_{k\to\infty} \|J_k^T J_k - J(x^*)^T J(x^*)\| + \lim_{k\to\infty} \|(A_k - S(x^*))s_k\|/\|s_k\| \\
=\ &0.
\end{aligned}$$

Moreover, the sequences $\{\|B_k\|\}$ and $\{\|B_k^{-1}\|\}$ are uniformly bounded since $J_k^T J_k$ is semi-positive definite. The proof is then finished. □

The following theorem shows that Algorithm 2.1 converges superlinearly.

Theorem 2.3. *Suppose the assumptions of Lemma 2.11 hold. If the parameter δ in the line search (2.5) is chosen to satisfy $\delta \in (0, \frac{1}{2})$, then $\{x_k\}$ converges to x^* superlinearly.*

Proof. By Lemma 2.11, we only need to prove $\alpha_k = 1$ for all sufficiently large k in the line search (2.5). In fact, by Lemma 2.11, we have $\|d_k\| = \|B_k^{-1} g_k\| \to 0$. From Taylor's expansions we have

$$\begin{aligned} & f(x_k + d_k) - f(x_k) - \delta g_k^T d_k \\ = \; & (1-\delta) g_k^T d_k + \frac{1}{2} d_k^T \nabla^2 f(x_k + \theta_k d_k) d_k \\ = \; & -(1-\delta) d_k^T B_k d_k + \frac{1}{2} d_k^T \nabla^2 f(x_k + \theta_k d_k) d_k \\ = \; & -(\frac{1}{2} - \delta) d_k^T B_k d_k - \frac{1}{2} d_k^T (B_k - \nabla^2 f(x_k + \theta_k d_k)) d_k \\ = \; & -(\frac{1}{2} - \delta) d_k^T \nabla^2 f(x^*) d_k + o(\|d_k\|^2), \end{aligned}$$

where $\theta_k \in (0, 1)$ and the last equality follows from the Dennis-Moré condition (2.18). Thus $f(x_k + d_k) - f(x_k) - \delta g_k^T d_k \le 0$ for all sufficiently large k, which implies $\alpha_k = 1$ for all sufficiently large k. Therefore according to the well-known characterization result of Dennis and Moré [10], we conclude that the proposed method converges superlinearly. □

3 Numerical experiments

In this section, we compare the performance of the following three methods with the same line search (2.5) for some nonlinear least squares problems

- The Gauss-Newton method: $B_k = J_k^T J_k + \|r_k\| I$;
- The hybrid Gauss-Newton structured BFGS method: Algorithm 2.1 with $\epsilon = 10^{-6}$;
- The Fletcher-Xu hybrid (FXhybrid) method: B_k is specified by (1.6) with $\epsilon = 0.2$ which was recommended in [17, 18].

All codes were written in Matlab 7.4. We set $\delta = 0.1$ and $\rho = 0.5$ in the line search (2.5). For the three methods, we set the initial matrix $B_0 = J_0^T J_0 + 10^{-4} \|r_0\| I$. We stopped the iteration if one of the following conditions is satisfied

(i) $\|g_k\| \le 10^{-5}$;

(ii) $f(x_k) - f(x_{k+1}) \le 10^{-15} \max(1, f(x_k))$;

(iii) $f(x_k) \le 10^{-8}$;

(iv) The total number of iterations exceeds 500.

Tables 2-5 in the Appendix list numerical results of these three methods, where "Biter/Iter" and "Nf" stand for the total number of BFGS update/all iterations and the function evaluations respectively; $f(x_k)$ and r_k mean the functional evaluation and the residual at the stopping point respectively. In Tables 2-5, $\lambda_{\min}$ is the smallest eigenvalue of $S(x)$ at the stopping point.

Table 2 reports the numerical results of the three methods for 28 zero or small residual problems [25] and the BOD problem [2] with 6 different initial points. Table 3 lists the numerical results of the three methods for solving 40 large residual problems where "Froth", "Jensam" and "Cheb" are from [25] and the others are given as follows.

•**Trigonometric Problem** (Trigo) [1]:

$$r_i(x) = -d_i + \tilde{r}_i(x)^2, \quad i = 1, 2, \cdots, m,$$

where

$$\tilde{r}_i(x) = -e_i + \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j), \quad i = 1, 2, \cdots, m,$$

with $x = (x_1, \cdots, x_n)^T$, a_{ij}, b_{ij} are random integers in $[-10, 10]$, e_i are random numbers in $[0, 1]$ and $d = (d_1, d_2, \cdots, d_m)^T = (1, 2, \cdots, m)^T$. We choose the initial point x_0 as a random vector whose elements are in $[-100, 0]$.

•**Signomial Problem** (Sig) [1]:

$$r_i(x) = -e_i + \sum_{k=1}^{l} c_{ik} \prod_{j=1}^{n} x_j^{a_{ijk}}, \quad i = 1, 2, \cdots, m,$$

where a_{ijk} are random integers in $[0, 3]$, c_{ik} and e_i are random numbers in $[-100, 100]$ and $[-10, 10]$ respectively. We choose $l = 8$, and the initial point x_0 as a random vector whose elements are in $[-5, 5]$.

•**Parameterized Problem** (Para) [21]:

$$r_1(x) = x_1 - 2, \quad r_2(x) = (x_1 - 2\psi)x_2, \quad r_3(x) = x_2 + 1,$$

where $x = (x_1, x_2)^T$ and ψ is a parameter. If $\psi \neq 1$, then this problem is a nonzero residual problem. We choose different values of ψ and initial points x_0 in our test. For details, see Table 2.

•**Nonlinear regression problem** (BOD) [2, pp.305]: The nonlinear regression model based on the data on biochemical oxygen demand(BOD) can be converted into the nonlinear least square problem (1.1) where $r(x) = (r_1(x), \cdots, r_8(x))^T$, $x = (x_1, x_2)^T$ and

$$r_1(x) = x_1(1 - e^{x_2}) - 0.47; \quad r_2(x) = x_1(1 - e^{2x_2}) - 0.74;$$

$$r_3(x) = x_1(1 - e^{3x_2}) - 1.17; \quad r_4(x) = x_1(1 - e^{4x_2}) - 1.42;$$

$$r_5(x) = x_1(1 - e^{5x_2}) - 1.60; \quad r_6(x) = x_1(1 - e^{7x_2}) - 1.84;$$

$$r_7(x) = x_1(1 - e^{9x_2}) - 2.19; \quad r_8(x) = x_1(1 - e^{11x_2}) - 2.17.$$

Table 4 and Table 5 list some numerical results of the three methods for solving a special class of nonlinear least square problems.

• **Convex variational regularization problem**: Suppose that $F : R^n \to R^m$ is a map. The convex variational regularization problem is the following minimization problem

$$\min_{x \in R^n} f(x) = \frac{1}{2}\|F(x)\|^2 + \frac{\mu}{2} h(x), \tag{3.1}$$

where $h : R^n \to R$ is a convex function and μ is a regularization parameter. Many practical problems can be converted into solving this problems such as ill-posed problems, inverse problems, some constrained optimization problems and model parameter estimation [27, 30, 28, 29, 14, 15].

Ill-posed problems occur frequently in science and engineering. Regularization methods for computing stabilized solutions to the ill-posed problems have been extensively studied [20]. In this paper, we chose two convex variational regularization problems which come from ill-posed problems. In our test, we chose $h(x) = \sum_{i=1}^{n}(x_i^2)^2$ in (3.1). Therefore (3.1) reduces to the nonlinear least squares problem (1.1) with the form:

$$r(x) = (F(x), \sqrt{\mu}x_1^2, \cdots, \sqrt{\mu}x_n^2)^T.$$

Now we chose two ill-posed problems as follows. One is linear and another is nonlinear.

(i) Ill-posed problem 1 (The linear ill-conditioned problem): $F(x) = Ax - b$, where $A = (a_{ij})_{n\times n}$ with $a_{ij} = \frac{1}{i+j-1}$ is the Hilbert matrix. In our code, we set $b = A * ones(n, 1) + 10^{-4} * ones(n, 1)$ and the initial point $x_0 = (10, \cdots, 10)^T$.

(ii) Ill-posed problem 2 (The nonlinear inverse problem): The Fredholm integral equation of the first kind has the following version

$$\int_a^b K(t, s, u(s))ds = g(t), \quad c \le t \le d, \tag{3.2}$$

where the right-hand side g and the kernel K are given, and x is an unknown solution. We use the composite quadrature method to approximate the integral by a weighted sum

$$\int_a^b K(t, s, u(s))ds \approx I_n(t) = \sum_{i=1}^{n} w_i K(t, s_i, u(s_i)).$$

Collocation in the m points $t_1, \cdots, t_m$ leads to the requirements $I_n(t_j) = g(t_j), j = 1, \cdots, m$. It is a finite dimensional nonlinear ill-posed problem. To obtain a meaningful solution, it is often converted into solving a regularization solution of (3.1) where

$$F_j(x) = I_n(t_j) - g(t_j), j = 1, \cdots, m, x = (u(s_1), \cdots, u(s_n))^T.$$

In our test we chose the following data [5]

$$[a, b] = [c, d] = [0, 1], \quad K(t, s, u(s)) = se^{(t+1)u(s)}, \quad g(t) = \frac{e^{t+1} - 1}{2(t + 1)}.$$

Integral equation (3.2) with these data has an analytical solution as $u(s) = s^2$ on $[0, 1]$. In our numerical experiment, we chose $t_j = \frac{j-1}{m-1}$ for $j = 1, \cdots, m$. We set the initial point $x_0 = (0.1, \cdots, 0.1)^T$.

Table 1 summaries the data in Tables 2-5, in which "#Bestiter", "#BestNf" and "#Bestfv" are the number of test problems that the method wins over the rest of the methods on the number of iterations, function evaluations and the best final objective function value performance in all 138 test problems, respectively; "Probability" roughly means the probability that the method wins over the rest of the methods.

It is clear from Table 1 that Algorithm 2.1 is the best method among these three methods. In order to show the number of iterations or function evaluations performance

Table 1: Summary over Tables 2-5 in the Appendix.

	Gauss-Newton	Algorithm 2.1	FXhybrid
#Bestiter	28	85	49
Probability	$\frac{28}{138} \approx 20\%$	$\frac{85}{138} \approx 62\%$	$\frac{49}{138} \approx 36\%$
#BestNf	23	108	26
Probability	$\frac{23}{138} \approx 17\%$	$\frac{108}{138} \approx 78\%$	$\frac{26}{138} \approx 19\%$
#Bestfv	92	100	81
Probability	$\frac{23}{138} \approx 67\%$	$\frac{108}{138} \approx 72\%$	$\frac{26}{138} \approx 58\%$

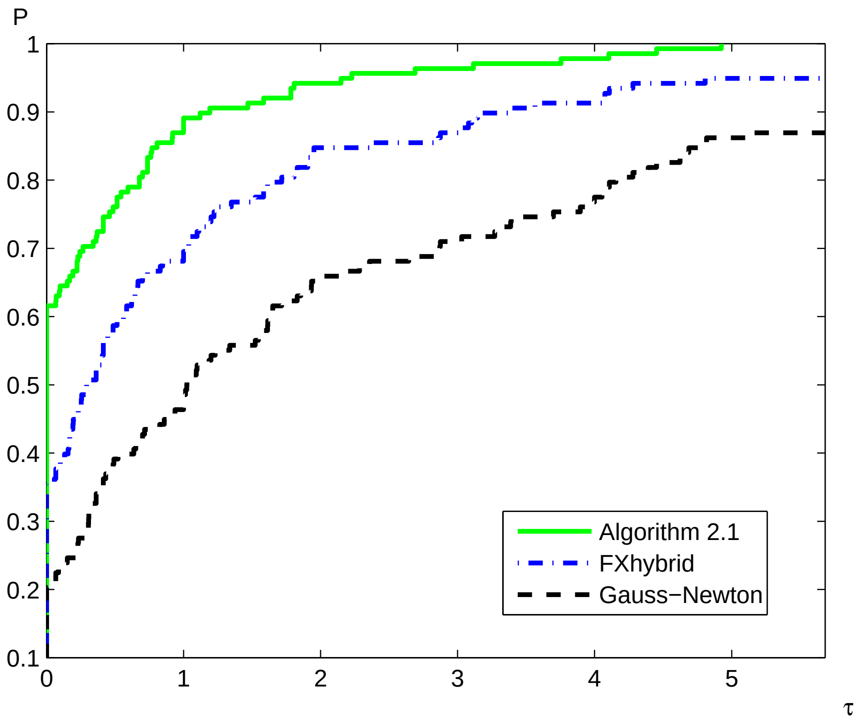

Figure 1: Performance profiles with respect to the number of iterations.

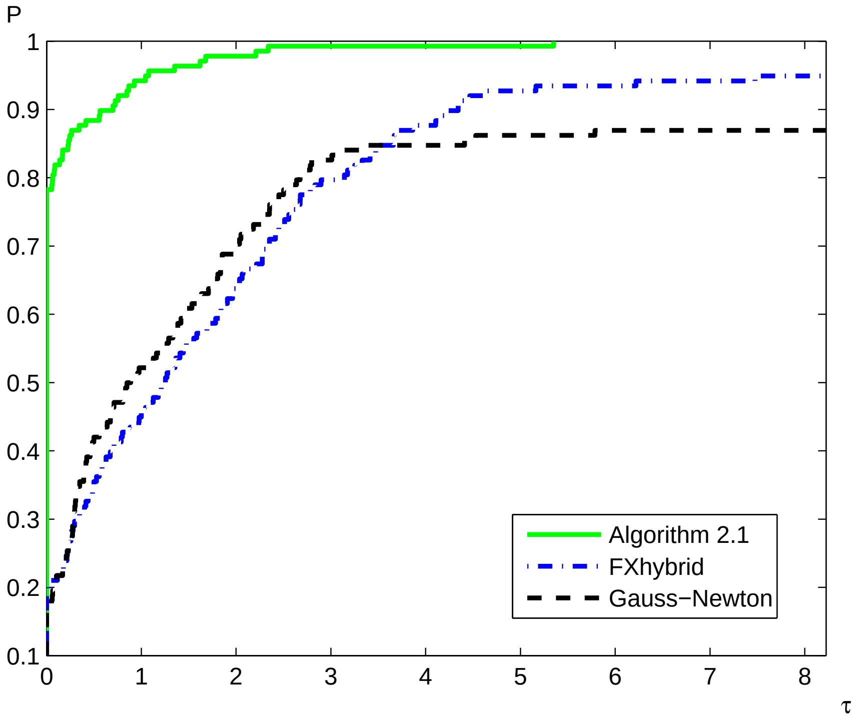

Figure 2: Performance profiles with respect to the number of function evaluations.

of the three methods more clearly, we plotted Figures 1-2 according to the data in Tables 2-5 in the Appendix section by using the performance profiles of Dolan and Moré [12].

Since the top curve in Figures 1-2 corresponds to Algorithm 2.1, it is clear that Algorithm 2.1 is most efficient for solving these 138 test problems among the three methods. We see from Figure 1 that Algorithm 2.1 solves about 62% and 78% (85 and 108 out of 138) of the test problems with the least number of iterations and function evaluations respectively. Figure 1 also shows that the FXhybrid method performs better than the Gauss-Newton method. However, Figure 2 shows that the FXhybrid method needs more function evaluations than the Gauss-Newton method within $0.3 < \tau < 3.5$. Table 2 shows that the Gauss-Newton method is efficient for zero residual problems and using the BFGS update can improve numerical performance. We also note from Table 1 that Algorithm 2.1 has the best final objective value for most problems, which has about 72% (100 out of 138) probability with the best final objective value.

4 Conclusions

In this paper, we proposed a new hybrid Gauss-Newton structured BFGS method for nonlinear least squares problems. We use a new formula (2.2)-(2.3) to update the iterative matrix. The new formula deals with zero or nonzero residual problems in an intelligent way. Global convergence of the proposed method is established. Under suitable conditions, the proposed method possesses quadratic convergence rate for zero residual problems and superlinear convergence rate for nonzero residual problems. Numerical results show that the proposed method is efficient for nonlinear least squares problems compared with the Gauss-Newton method and the Fletcher-Xu hybrid method.

5 Acknowledgement

We would like to thank the referees and the associate editor whose very helpful suggestions led to much improvement of this paper.

References

[1] M. Al-Baali and R. Fletcher, Variational methods for non-linear least squares, J. Oper. Res. Soc., 36 (1985), pp. 405-421.

[2] D. M. Bates and D. G. Watts, Nonlinear Regression Analysis and Its Applications, John Wiley and Sons, New York, 1988.

[3] R. H. Byrd and J. Nocedal, A tool for the analysis of quasi-Newton methods with application to unconstrained minimization, SIAM J. Numer. Anal., 26 (1989), pp. 727-739.

[4] R. H. Byrd, J. Nocedal and Y. Yuan, Global covergence of a class of quasi-Newton method on convex problens, SIAM J. Numer. Anal., 24 (1987), pp. 1171-1190.

[5] A. H. Borzabadi and O. S. Fard, Approximate solution of nonlinear Fredholm integral equations of the first kind via converting to optimization, World Academy of Science, Engineering and Technology, 33 (2007), pp. 70-73.

[6] A. R. Conn, N. I. M. Gould and P. L. Toint, Trust-Region Methods, SIAM, 2000.

[7] Y. H. Dai, Convergence properties of the BFGS algorithm, SIAM J. Optim., 13 (2002), pp. 693-701.

[8] J. E. Dennis, D. M. Gay and R. E. Welsch, An adaptive nonlinear least-squares algorithm, ACM Trans. Math. Software, 7 (1981), pp. 348-368.

[9] J. E. Dennis, H. J. Martínez and R. A. Tapia, Convergence theory for the structured BFGS secant method with application to nonlinear least squares, J. Optim. Theory Appl., 61 (1989), pp. 161-178.

[10] J. E. Dennis and J. J. Moré, A characterization of superlinear convergence and its application to quasi-Newton methods, Math. Comput., 28 (1974), pp. 549-560.

[11] J. E. Dennis and H. J. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[12] E. D. Dolan and J. J. Moré, Benchmarking optimization software with performance profiles, Math. Program., 91 (2002), pp. 201-213.

[13] J. R. Engels and H. J. Martínez, Local and superlinear convergence for partially known quasi-Newton methods, SIAM J. Optim., 1 (1991), pp. 42-56.

[14] H. W. Engl, M. Hanke and A. Neubauer, Regularization of Inverse Problems, Kluwer Academic Publishers, Boston, 1996.

[15] J. Eriksson, P. A. Wedin, M. E. Gulliksson and I. Söderkvist, Regularization methods for uniformly rank-deficient nonlinear least-squares problems, J. Optim. Theory Appl., 127 (2005), pp. 1-26.

[16] J. Fan and Y. Yuan, On the quadratic convergence of the Levenberg-Marquardt method without nonsingularity assumption, Computing, 74 (2005), pp. 23-39.

[17] R. Fletcher, Practical Methods of Optimization, Second Edition, John Wiley & Sons, New York, 1987.

[18] R. Fletcher and C. Xu, Hybrid methods for nonlinear least squares, IMA J. Numer. Anal., 7 (1987), pp. 371-389.

[19] S. Gatton, A. S. Lawless and N. K. Nichols, Approximate Gauss-Newton methods for nonlinear least squares problems, SIAM J. Optim., 18 (2007), pp. 106-132.

[20] P. C. Hansen, Regularization tools version 4.0 for Matlab 7.3, Numer. Algor., 46 (2007), pp. 189-194.

[21] J. Huschens, On the use of product structure in secant methods for nonlinear least squares problems, SIAM J. Optim., 4 (1994), pp. 108-129.

[22] D. H. Li and M. Fukushima, A modified BFGS method and its global convergence in nonconvex minimization, J. Comput. Appl. Math., 129 (2001), pp. 15-35.

[23] D. H. Li and M. Fukushima, On the global convergence of the BFGS method for nonconvex unconstrained optimization problems, SIAM J. Optim., 11 (2001), pp. 1054-1064.

[24] W. F. Mascarenhas, The BFGS method with exact line searches fails for non-convex objective functions, Math. Program., 99 (2004), pp. 49-61.

[25] J. J. Moré, B. S. Garbow and K. E. Hillstrom, Testing unconstrained optimization software, ACM Trans. Math. Software, 7 (1981), pp. 17-41.

[26] H. Ogasawara, A note on the equivalence of a class of factorized Broyden families for nonlinear least squares problems, Linear Algebra Appl., 297 (1999), pp. 183-191.

[27] S. Osher and M. Burger, Convergence rates of convex variational regularization, Inverse Problems, 20 (2004), pp. 1411-1421.

[28] E. Resmerita, Regularization of ill-posed problems in Banach spaces: convergence rates, Inverse Problems, 21 (2005), pp. 1303-1314.

[29] E. Resmerita and O. Scherzer, Error estimates for non-quadratic regularization and the relation to enhancement, Inverse Problems, 22 (2006), pp. 801-814.

[30] A. Tarantola, Inverse Problem Theory and Methods for Model Parameter Estimation, SIAM, Philadelphia, 2005.

[31] W. Sun and Y. Yuan, Optimization Theory and Methods, Springer Science and Business Media, LLC, New York, 2006.

[32] C. Xu, X. Ma and M. Kong, A class of factorized quasi-Newton methods for nonlinear least squares problems, J. Comput. Math., 14 (1996), pp. 143-158.

[33] H. Yabe and T. Takahashi, Factorized quasi-Newton methods for nonlinear least squares problems, Math. Program., 51 (1991), pp. 75-100.

[34] H. Yabe and N. Yamaki, Convergence of a factorized Broyden-like family for nonlinear least squares problems, SIAM J. Optim., 5 (1995), pp. 770-791.

[35] H. Yabe and N. Yamaki, Local and superlinear convergence of structured quasi-Newton methods for nonlinear optimization, J. Oper. Res. Soc. Japan, 39 (1996), pp. 541-557.

[36] N. Yamashita and M. Fukushima, On the rate of convergence of the Levenberg-Marquardt method, Computing (Supp.), 15 (2001), pp. 237-249.

[37] J. Z. Zhang, L. H. Chen and N. Y. Deng, A family of scaled factorized Broyden-like methods for nonlinear least squares problems, SIAM J. Optim., 10 (2000), pp. 1163-1179.

Appendix

Table 2: Test results for 28 some zero or small residual test problems from [25] and the BOD problem with 6 different initial points.

			Gauss-Newton			Algorithm 2.1			FXhybrid		
Prob	n	m	Iter	Nf	$f(x_k)$	Biter/Iter	Nf	$f(x_k)$	Iter	Nf	$f(x_k)$
Rose	2	2	15	24	4.86e-012	18/19	27	7.89e-011	21	87	3.48e-010
Badscp	2	2	500	502	1.77e-006	68/85	97	9.90e-009	500	5308	2.09e-004
Badscb	2	3	500	503	4.97e+011	8/16	43	3.64e-015	33	402	7.09e-012
Beale	2	3	9	10	4.40e-011	2/10	11	2.35e-013	9	10	4.40e-011
Helix	3	3	11	12	3.27e-013	66/71	490	3.17e-009	11	12	3.27e-013
Bard	3	15	148	149	4.11e-003	6/6	7	4.11e-003	71	102	4.11e-003
Gauss	3	15	1	2	5.64e-009	0/1	2	5.64e-009	1	2	5.64e-009
Gulf	3	10	500	502	1.96e-003	13/17	31	1.26e-008	27	61	2.10e-006
Box	3	10	50	51	6.27e-008	4/8	9	7.12e-011	30	31	4.14e-008
Sing	4	4	11	12	6.95e-009	8/9	10	4.09e-009	11	12	6.95e-009
Wood	4	6	330	334	1.04e-014	14/46	48	3.99e-015	389	883	3.99e-011
Kowosb	4	11	24	26	1.54e-004	1/31	33	1.54e-004	24	26	1.54e-004
Biggs	6	13	500	505	9.74e-004	6/317	326	4.90e-006	500	605	9.02e-006
Osb2	11	65	356	357	2.01e-002	31/34	43	2.01e-002	247	380	2.01e-002
Watson	20	31	13	14	2.18e-007	5/8	10	5.93e-008	13	14	2.18e-007
Cheb	5	5	3	5	2.95e-009	2/4	6	1.66e-015	3	5	2.95e-009
Rosex	20	20	22	27	6.34e-011	20/21	32	9.61e-013	22	47	5.29e-009
Singx	20	20	14	15	6.68e-008	8/9	10	2.05e-008	14	15	6.68e-008
Vardim	20	22	11	12	2.15e-017	10/11	12	2.18e-010	11	12	2.15e-017
Trig	20	20	39	137	2.32e-006	5/8	17	2.40e-012	9	55	2.84e-009
Rosex	100	100	39	44	3.45e-014	20/22	32	2.42e-012	44	67	3.62e-010
Singx	100	100	23	24	6.34e-008	8/10	11	1.65e-008	23	24	6.34e-008
Vardim	100	102	15	16	4.42e-010	15/16	17	2.43e-011	15	16	4.42e-010
Trig	100	100	7	17	4.16e-010	5/9	31	8.54e-012	196	3026	1.28e-011
Rosex	500	500	76	81	4.63e-012	20/21	31	6.57e-011	22	73	8.32e-012
Singx	500	500	42	43	1.39e-007	9/11	12	3.55e-008	42	43	1.39e-007
Vardim	500	502	20	21	1.93e-014	19/21	22	2.81e-012	20	21	1.93e-014
Trig	500	500	7	16	1.62e-009	11/14	74	3.96e-010	62	1192	5.09e-011
Prob	n	x_0^T	Iter	Nf	$f(x_k)$	Biter/Iter	Nf	$f(x_k)$	Iter	Nf	$f(x_k)$
BOD	2	$(1,0)$	8	56	0.01	3/6	54	0.01	9	86	0.01
BOD	2	$(100,0)$	7	113	0.45	3/7	85	0.45	8	87	0.45
BOD	2	$(0.01,0.01)$	500	525	0.57	3/4	68	1.75	4	105	1.81
BOD	2	$(10,0.01)$	500	501	0.56	29/32	102	0.49	20	122	0.50
BOD	2	$(100,0.01)$	6	113	0.44	6/7	82	0.45	6	116	0.44
BOD	2	$(-10,-1)$	8	56	0.01	9/13	14	0.01	9	65	0.01

Table 3: Test results about 40 large residual problems.

			Gauss-Newton			Algorithm 2.1			FXhybrid		
Prob	n	m	Iter	Nf	$\|r_k\|$	Biter/Iter	Nf	$\|r_k\|$	Iter	Nf	$\|r_k\|$
Froth	2	2	187	188	6.70	31/89	90	6.70	74	1166	6.70
Jensam	2	4	93	349	2.05	14/46	143	2.05	73	763	2.05
Jensam	2	6	268	1436	4.39	18/18	26	4.39	27	448	4.39
Jensam	2	8	500	3281	7.44	19/19	25	7.45	39	508	7.46
Jensam	2	10	500	3796	11.16	49/49	65	11.20	51	463	11.15
Cheb	8	8	130	502	0.0593	17/34	78	0.0593	177	1724	0.0593
Cheb	10	10	17	23	0.0806	2/13	20	0.0806	20	149	0.0806
Cheb	8	16	22	23	0.2428	2/21	22	0.2428	23	24	0.2428
Trigo	3	6	19	21	8.10	2/12	13	4.10	28	31	86.21
Trigo	3	12	57	67	159.23	13/53	57	159.24	28	30	150.06
Trigo	3	15	21	23	179.21	1/22	24	179.21	31	208	178.87
Trigo	4	8	27	31	49.36	15/20	25	49.36	36	64	40.55
Trigo	4	20	117	119	352.11	17/71	73	353.68	16	203	373.68
Trigo	4	40	47	49	607.09	20/45	47	607.55	38	40	777.12
Trigo	6	8	82	178	19.01	26 /75	86	30.57	16	17	6.06
Trigo	6	12	48	69	24.23	8/35	37	21.14	24	25	23.94
Trigo	6	20	49	51	211.39	25/46	49	211.47	27	29	228.79
Trigo	8	8	95	350	4.99	18/31	77	1.01	79	453	4.30
Trigo	8	16	26	34	135.27	7/25	27	135.27	28	164	141.09
Trigo	8	40	61	63	698.26	25/30	32	698.46	90	661	361.86
Trigo	10	20	69	75	74.91	9/45	47	74.02	42	473	51.88
Trigo	10	40	142	144	770.00	51/117	119	678.06	69	579	570.24
Trigo	10	50	45	50	1041.07	26/36	38	1043.10	32	362	1165.79
Sig	2	6	187	825	52.10	28/75	244	52.10	64	246	52.10
Sig	2	10	47	80	79.36	19/40	61	79.36	49	71	79.36
Sig	2	30	50	52	269.10	17/40	42	269.10	49	51	269.10
Sig	4	8	28	30	11.47	4/24	25	11.47	31	33	11.47
Sig	4	10	251	252	15.56	78/82	84	14.78	251	252	15.56
Sig	4	20	81	84	24.24	9/72	74	24.24	26	29	24.41
Sig	4	30	56	59	26.22	9/40	44	26.22	53	56	26.22
Sig	4	40	52	116	88.51	18/26	28	88.54	29	45	88.51
Sig	6	12	338	893	13.88	50/144	491	14.04	172	334	13.88
Sig	6	24	46	50	21.19	18/24	27	22.64	49	53	21.19
Sig	6	30	96	164	28.22	64/68	73	28.28	45	65	28.22
Prob	ψ	x_0^T	Iter	Nf	$\|r_k\|$	Iter	Nf	$\|r_k\|$	Iter	Nf	$\|r_k\|$
Para	10	$(0,0)$	8	9	1.00	0/8	9	1.00	4	5	1.00
Para	10	(1,1)	14	15	1.00	1/12	13	1.00	14	15	1.00
Para	10	(10,10)	25	26	1.00	8/8	10	1.00	30	64	1.00
Para	100	(0,0)	7	8	1.00	0/7	8	1.00	5	6	1.00
Para	100	(1,1)	13	14	1.00	1/7	8	1.00	14	16	1.00
Para	100	(10,10)	20	21	1.00	4/5	6	1.00	41	215	1.00

Table 4: Test results about the ill-posed problem 1 with the initial point $x_0 = (10, 10, \cdots, 10)^T$ and different regularization parameters.

		Gauss-Newton			Algorithm 2.1				FXhybrid		
μ	$m = n$	Iter	Nf	$\|r_k\|$	Biter/Iter	Nf	$\|r_k\|$	$\lambda_{\min}$	Iter	Nf	$\|r_k\|$
1	10	6	55	2.12e+000	6/8	55	2.17e+000	7.79e-001	6	55	2.12e+000
1	50	23	301	1.06e+001	5/7	55	1.09e+001	7.53e-001	23	301	1.06e+001
1	100	15	95	2.12e+001	5/7	58	2.16e+001	7.32e-001	15	95	2.12e+001
1	150	23	218	3.18e+001	6/8	61	3.23e+001	7.09e-001	23	218	3.18e+001
1	200	24	239	4.23e+001	6/8	99	4.44e+001	8.21e-001	24	239	4.23e+001
1	250	32	233	5.29e+001	6/9	58	5.45e+001	7.50e-001	32	233	5.29e+001
10^{-2}	10	44	75	4.61e-002	4/6	51	4.61e-002	1.52e-002	44	75	4.61e-002
10^{-2}	50	63	268	2.30e-001	5/6	47	2.30e-001	1.49e-002	63	268	2.30e-001
10^{-2}	100	118	307	4.60e-001	6/7	48	4.60e-001	1.48e-002	118	307	4.60e-001
10^{-2}	150	133	235	6.91e-001	7/8	46	6.91e-001	1.47e-002	133	235	6.91e-001
10^{-2}	200	138	247	9.21e-001	7/8	51	9.21e-001	1.48e-002	138	247	9.21e-001
10^{-2}	250	136	201	1.15e+000	6/7	49	1.15e+000	1.48e-002	136	201	1.15e+000
10^{-4}	10	251	252	4.97e-004	8/26	70	4.97e-004	1.76e-004	91	273	5.44e-004
10^{-4}	50	500	501	2.50e-003	9/28	109	2.48e-003	1.73e-004	16	66	1.53e-002
10^{-4}	100	500	501	5.30e-003	15/53	81	4.96e-003	1.80e-004	205	501	5.39e-003
10^{-4}	150	500	501	8.87e-003	14/34	68	7.44e-003	1.77e-004	18	79	7.84e-002
10^{-4}	200	500	501	1.36e-002	24/124	164	9.93e-003	1.80e-004	268	616	1.05e-002
10^{-4}	250	500	501	1.99e-002	22/121	159	1.24e-002	1.80e-004	284	624	1.33e-002
10^{-6}	10	253	254	1.18e-005	8/10	11	5.22e-006	1.47e-006	86	140	2.00e-005
10^{-6}	50	500	501	9.49e-005	35/37	38	2.71e-005	1.42e-006	500	520	6.56e-005
10^{-6}	100	500	501	8.01e-004	49/72	73	5.16e-005	1.61e-006	500	540	1.13e-004
10^{-6}	150	500	501	2.85e-003	45/84	85	7.86e-005	1.49e-006	500	536	1.90e-004
10^{-6}	200	500	501	6.90e-003	81/183	184	1.02e-004	1.28e-006	500	585	2.80e-004
10^{-6}	250	500	501	1.36e-002	103/171	172	1.29e-004	8.71e-007	500	531	3.35e-004

Table 5: Test results about the ill-posed problem 2 with the initial point $x_0 = (0.1, 0.1, \cdots, 0.1)^T$ and different regularization parameters.

			Gauss-Newton			Algorithm 2.1				FXhybrid		
μ	n	m	Iter	Nf	$\|r_k\|$	Biter/Iter	Nf	$\|r_k\|$	$\lambda_{\min}$	Iter	Nf	$\|r_k\|$
1	10	10	22	76	8.00e-001	6/14	63	8.00e-001	1.37e-001	7	89	8.02e-001
1	10	50	45	47	8.52e-001	9/16	55	8.52e-001	1.32e-001	10	121	8.54e-001
1	20	20	49	81	1.12e+000	6/21	66	1.12e+000	8.03e-002	6	81	1.12e+000
1	20	100	96	98	1.19e+000	6/10	55	1.19e+000	6.22e-002	6	70	1.20e+000
1	30	30	78	115	1.36e+000	7/52	59	1.36e+000	5.64e-002	6	80	1.37e+000
1	30	150	147	149	1.46e+000	4/8	48	1.46e+000	3.64e-002	7	72	1.47e+000
1	40	40	107	131	1.57e+000	27/81	269	1.57e+000	4.53e-002	6	84	1.57e+000
1	40	200	197	199	1.68e+000	7/31	73	1.68e+000	4.79e-002	9	79	1.68e+000
1	50	50	141	245	1.76e+000	6/86	122	1.76e+000	3.87e-002	5	75	1.76e+000
1	50	250	247	249	1.88e+000	7/31	75	1.88e+000	3.99e-002	9	76	1.88e+000
10^{-2}	10	10	3	54	9.29e-002	4/5	48	9.34e-002	2.00e-004	3	64	9.29e-002
10^{-2}	10	50	6	50	9.50e-002	4/8	48	9.52e-002	-9.80e-008	8	99	9.50e-002
10^{-2}	20	20	3	46	1.30e-001	0/3	46	1.30e-001	2.03e-004	4	52	1.30e-001
10^{-2}	20	100	6	51	1.33e-001	5/10	53	1.34e-001	3.27e-006	8	94	1.33e-001
10^{-2}	30	30	3	46	1.59e-001	0/3	46	1.59e-001	1.85e-004	3	54	1.59e-001
10^{-2}	30	150	7	54	1.63e-001	5/10	51	1.63e-001	1.70e-005	9	118	1.63e-001
10^{-2}	40	40	3	48	1.83e-001	0/3	48	1.83e-001	1.80e-004	3	57	1.83e-001
10^{-2}	40	200	7	53	1.88e-001	5/10	46	1.88e-001	3.14e-005	9	135	1.88e-001
10^{-2}	50	50	3	47	2.05e-001	0/3	47	2.05e-001	1.77e-004	3	47	2.05e-001
10^{-2}	50	250	8	52	2.10e-001	6/11	55	2.11e-001	3.02e-005	9	138	2.10e-001
10^{-4}	10	10	8	46	9.60e-003	3/6	9	9.59e-003	3.09e-006	10	101	9.60e-003
10^{-4}	10	50	60	62	9.63e-003	2/7	9	9.63e-003	3.15e-006	6	62	9.68e-003
10^{-4}	20	20	9	47	1.35e-002	2/7	9	1.35e-002	2.37e-006	10	100	1.35e-002
10^{-4}	20	100	124	126	1.35e-002	2/24	26	1.35e-002	2.25e-006	8	84	1.37e-002
10^{-4}	30	30	11	56	1.65e-002	2/7	9	1.65e-002	2.23e-006	16	161	1.65e-002
10^{-4}	30	150	177	179	1.66e-002	3/151	154	1.66e-002	2.04e-006	180	222	1.66e-002
10^{-4}	40	40	10	47	1.91e-002	2/9	11	1.90e-002	2.17e-006	16	139	1.91e-002
10^{-4}	40	200	232	234	1.91e-002	3/197	200	1.91e-002	1.97e-006	9	97	1.94e-002
10^{-4}	50	50	11	53	2.13e-002	0/11	53	2.13e-002	2.04e-006	13	120	2.13e-002
10^{-4}	50	250	287	289	2.13e-002	3/243	246	2.13e-002	1.94e-006	8	80	2.17e-002
10^{-6}	10	10	7	9	1.19e-003	3/16	19	1.10e-003	2.40e-007	7	9	1.19e-003
10^{-6}	10	50	57	59	9.70e-004	2/46	48	9.70e-004	1.79e-007	49	512	9.69e-004
10^{-6}	20	20	56	58	1.51e-003	2/39	41	1.51e-003	7.92e-008	61	204	1.51e-003
10^{-6}	20	100	77	79	1.37e-003	2/62	64	1.37e-003	8.75e-008	71	310	1.37e-003
10^{-6}	30	30	63	65	1.88e-003	2/49	51	1.88e-003	5.15e-008	65	88	1.88e-003
10^{-6}	30	150	91	93	1.68e-003	3/87	90	1.69e-003	6.42e-008	40	379	1.90e-003
10^{-6}	40	40	68	70	2.19e-003	2/55	57	2.19e-003	3.99e-008	75	155	2.19e-003
10^{-6}	40	200	102	104	1.95e-003	3/96	99	1.95e-003	5.16e-008	103	414	1.95e-003
10^{-6}	50	50	73	75	2.47e-003	2/60	62	2.47e-003	3.40e-008	84	333	2.47e-003
10^{-6}	50	250	112	114	2.18e-003	3/103	106	2.18e-003	4.48e-008	121	265	2.18e-003