A Secant Method for Nonlinear Least-Squares Minimization

Thomas F. Coleman[1], Wei Xu[2] and Gang Liu[3]

[1] Department of Combinatorics and Optimization
University of Waterloo
Waterloo, On. Canada, N2L 3G1.

[2,3] Software School , Fudan University
Shanghai, China, 200433.

[1] tfcoleman@uwaterloo.ca
[2] xuw5@fudan.edu.cn
[3] liu_gang@fudan.edu.cn

Abstract

Quasi-Newton methods have played a prominent role, over many years, in the design of effective practical methods for the numerical solution of nonlinear minimization problems and in multi-dimensional zero-finding. There is a wide literature outlining the properties of these methods and illustrating their performance [e.g., [8]]. In addition, most modern optimization libraries house a quasi-Newton collection of codes and they are widely used. The quasi-Newton contribution to practical nonlinear optimization is unchallenged.

In this paper we propose and investigate an efficient quasi-Newton (secant) approach to the nonlinear least-squares problem, made practical due to the selective application of automatic differentiation (AD) technology. We also observe that AD technology can increase the efficiency of the standard quasi-Newton (positive definite secant) approach to the full nonlinear minimization approach to this problem and we compare these two AD-assisted methods. Finally, we compare the AD-assisted approaches to a standard globalized Gauss-Newton method.

1 Introduction

The multi-dimensional zero-finding problem is

$$\text{solve} \quad F(x) = 0, \tag{1.1}$$

where $F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^T$ and each $f_i(x)$ maps real n-vectors x into scalars in a continuous and differentiable manner. A quasi-Newton method approximates the true n-by-n Jacobian matrix, $J(x) = (\frac{\partial f}{\partial x_j})_{n\times n}$, by successive updates. For example if x_k is the current approximate solution to (1.1), and the current approximation to the Jacobian matrix $J_k = J(x_k)$ is B_k, then a quasi-Newton method involves an update function U, $B_{k+1} = U(B_k, y_k = F_{k+1} - F_k, s_k = x_{k+1} - x_k)$ where x_{k+1} is the new iterate, $F_k = F(x_k)$, $F_{k+1} = F(x_{k+1})$. As indicated, U typically depends only on the current Jacobian approximation, the change in the iterate x, and the change in the function F. Hence, no 'extra' information needs to be computed in order to update B_k to yield B_{k+1}. The most successful quasi-Newton update for the nonlinear equations problem, often referred to as the *secant update* [2, 10], is the matrix solution to the convex optimization problem,

$$\min_B \{\|B - B_k\|_2 \ : \ Bs_k = y_k\}, \tag{1.2}$$

i.e., assuming s_k is not the zero vector,

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)s_k^T}{s_k^T s_k}. \tag{1.3}$$

Acceptable practical convergence properties typically require use of a merit function (to ensure and measure progress toward a solution), often $f_2(x) = \frac{1}{2}\|F(x)\|_2^2$ is used; a globalization scheme such as a linesearch or dogleg/trust region method is used in conjunction with the merit function. Consequently, the core loop of a typical globalized secant method for the multi-dimensional zero-finding problem can be framed as follows.

Algorithm 1 *(Secant Quasi-Newton for zero-finding problems.)*

1. *Solve $B_k d_k = -F(x_k)$ (i.e., determine a quasi-Newton direction).*
2. *Use a globalization scheme involving x_k, d_k, $B_k^T F(x_k)$ to determine x_{k+1} such that $f_2(x_{k+1})$ is 'sufficiently less than' $f_2(x_k)$. Set $y_k = F_{k+1} - F_k$.*
3. *Determine B_{k+1} via (1.3).*

A robust quasi-Newton implementation will also attend to other (important) details such as:

1. Choice of starting matrix approximation B_0.
2. Occasional refreshment of the Jacobian approximation B_k for some values of k (when progress is deemed too slow).

It is important to note that the gradient of the merit function f_2, i.e., $\nabla f_2(x) = J(x)^T F(x)$, is never computed in this popular approach. Generally, modern optimization methods for continuously differentiable problems expect an accurate gradient determination. However, in this situation the gradient is not computed - it is deemed too expensive (e.g., if the gradient computation involves computing the square Jacobian matrix at the current point then the cost of the gradient computation will be a factor of n times the cost to evaluate $F(x)$).

Why is the merit function gradient not necessary in this case? There are two main reasons:

(a). The nonsingularity of B_k ensures that the quasi-Newton direction d_k is a descent direction for the merit function f_2, i.e., $\nabla f_2(x)^T d_k < 0$.

(b). If $\bar{x}$ satisfies $B^T F(\bar{x}) = 0$ for any nonsingular matrix B, then $\nabla f_2(\bar{x}) = 0$; alternatively, if $\nabla f_2(\bar{x}) = 0$ and $J(\bar{x})$ is nonsingular, then $F(\bar{x}) = 0$.

Property (a) helps yield downhill steps with respect to the merit function, and property (b) allows the gradient $\nabla f_2(x) = J(x)^T F(x)$ to be replaced with $B^T F(x)$ with respect to first-order convergence results.

Why is there no secant method specifically for low residual nonlinear least-squares problems? A nonlinear least squares problem is a nonlinear minimization problem of the form, $\min_x f_2(x) = \frac{1}{2}\|F(x)\|_2^2$, where F represents a rectangular mapping,

$$F(x) = (f_1(x),\ f_2(x), \cdots,\ f_m(x))^T. \tag{1.4}$$

Each $f_i(x)$ maps real n-vectors x into scalars in a continuous and differentiable manner, and $m \geq n$. We are particularly interested in the case where $m > n$ since the equality case is discussed above. Generally, approaches to the nonlinear least-squares problem fall into two camps: those that treat the minimization of $f_2(x)$ as a general smooth nonlinear minimization problem (and ignore structure) and those that exploit the specific structure of this problem. The most popular approach in the latter camp is to ignore the terms of the Hessian matrix of f_2 that involve multiplication by 'residual' terms (e.g., $f_i(x)$) since they are expected to be small at the solution. Consequently, the linear approximation, $F(x + d) \cong F(x) + J(x)d$ is used in the step determination procedure at point x. For example, a trust region approach to the minimization of f_2 will involve the determination of a trial step d_k at point x_k:

$$d_k :\ \min_d\{\|F(x_k) + J_k d\|_2^2\ :\ \|d\|_2 \leq \Delta_k\}. \tag{1.5}$$

The question we explore in this paper is the potential use of a secant approximation B_k to the Jacobian approximation J_k, for the (low residual) nonlinear least-squares problem.

2 A Rectangular Secant Approximation

Let B_k be an m-by-n approximation to the m-by-n Jacobian matrix of the nonlinear vector-valued function F given in (1.4). Clearly problem (1.2) can be stated in the case where $m > n$ and the solution is the update given in (1.3) - now rectangular - provided the step s_k is not zero. Therefore in principle a globalized secant method can be designed similar to that used in multi-dimensional zero-finding.

Algorithm 2 *(Secant Quasi-Newton Method for nonlinear square problems.)*

1. *Use the linear approximation $F(x_k + d) \cong F(x_k) + B_k d$ to determine a direction d_k.*

2. *Use a globalization scheme involving x_k, d_k, $B_k^T F(x_k)$ to determine x_{k+1} such that $f_2(x_{k+1})$ is 'sufficiently less than' $f_2(x_k)$.*

3. *Determine B_{k+1} via (1.3).*

The updated matrix B_{k+1} via (1.3) maintains full column rank with the similar conditions as for the square matrices.

Theorem 2.1 *Assume that $B_k \in R^{m\times n}$ $(m \geq n)$ is full column rank and $s_k^T B_k^\dagger y_k \neq 0$, where $B_k^\dagger$ is the Moore-Penrose inverse [19], then B_{k+1} updated by (1.3) maintains full column rank.*

(Refer to the proof in Appendix.)

Note that the framework we have presented here, in analogy to the zero-finding situation, does not involve the gradient of the function f_2, i.e., $\nabla f_2(x) = J(x)^T F(x)$. In the zero-finding case this approach succeeds essentially because $F(x) \to 0$, as $x \to x_*$, a zero of F. However, in the rectangular case we are forcing $\nabla f_2(x) = J(x)^T F(x) \to 0$ *without the expectation that* $F(x) \to 0$. Therefore it is difficult to see how sufficient progress (step 2) toward a solution can be ensured without the gradient computation.

However, automatic differentiation technology allows for the computation of the gradient, $\nabla f_2(x) = J^T(x)F(x)$, in time proportional to the time required to evaluate $f_2(x)$, i.e., proportional to the time to evaluate $F(x)$ - there is no need to compute the Jacobian itself [13]. *Therefore, without significant extra expense the globalization scheme (step 2) can also involve the additional information $\nabla f_2(x_k) = J^T(x_k)F(x_k)$.*

To follow is an efficient dogleg/secant method for the (low-residual) nonlinear least-squares problem. We note that the 'nonlinear computations' in each iteration take time $O(\omega(F(x)))$ where $\omega(F(x))$ is the time (or work) required to evaluate the argument.

The trust region subproblem to be solved at each iteration to generate a trial step s_k is

$$\min_s \{s^T \nabla f_2(x_k) + \frac{1}{2} s^T B_k^T B_k s \ : \ \|s\|_2 \leq \Delta_k, s \in P_k\}, \tag{2.6}$$

where $\Delta_k > 0$ is the trust region radius, and P_k is the 'dogleg' piecewise linear path connecting x_k to CP, the Cauchy Point (i.e., the minimizer of the quadratic function in (2.6) along the negative gradient direction $-\nabla f_2(x_k)$), and then connecting CP to x_k^N, where $x_k^N = x_k + s_k^N$, and

$$[B_k^T B_k] s_k^N = -\nabla f_2(x_k). \tag{2.7}$$

The overall dogleg/trust region approach, which guarantees our proposed Algorithm 2 is able to converge to a first-order point from any starting point (e.g., Theorem 6.4.5 and 6.4.6 [1]), is:

```
For $k = 1, \cdots$
    Solve (2.6) for $s_k$, set $qp_k \leftarrow s_k^T \nabla f_2(x_k) + \frac{1}{2} s_k^T B_k^T B_k s_k$
    Set $new_k \leftarrow f_2(x_k + s_k),\ ratio_k \leftarrow \frac{new_k - f_2(x_k)}{qp_k}$
    Adjust $\Delta_k$:
        If $ratio_k < \tau_1$
            $\Delta_{k+1} \leftarrow \frac{\|s_k\|}{\gamma_1}$
        Elseif $ratio_k > \tau_2$ and $\|s_k\| = \Delta_k$
            $\Delta_{k+1} \leftarrow \gamma_2 \Delta_k$
        Else
            $\Delta_{k+1} \leftarrow \Delta_k$
        Endif
    Update $x$
        If $ratio_k \le 0$
            $x_{k+1} \leftarrow x_k$
        Else
            $x_{k+1} \leftarrow x_k + s_k$,
            update $B_k \rightarrow B_{k+1}$ via (1.3).
        Endif
```

The constants in this approach satisfy: $0 < \tau_1 < \tau_2 < 1$, $\gamma_1 > 1$, $\gamma_2 > 1$. Typical choices are: $\tau_1 = 0.25,\ \tau_2 = 0.75,\ \gamma_1 = 4,\ \gamma_2 = 2$.

3 Local Convergence

This section is devoted to analyzing the local convergence property for Algorithm 2 proposed in Section 2 for two different initial approximation of Jacobians. The results are similar to those in [4].

Theorem 3.1 *Let x^* be a least squares solution of the nonlinear least square problem, $\min_x f_2(x)$, where $f_2(x) = \frac{1}{2}\|F(x)\|_2^2$. Assume that there exists a neighborhood N of x^* such that*

1. *$J(x_0)$ satisfies a Lipschitz condition on N of order one with constant L,*
2. *$x_0 \in N$, where x_0 is the initial approximation to the solution and*

$$\|e_0\|_2 \le \mu_1/\alpha L,$$

   where $e_0 = x_0 - x^*$, $\mu_1 \simeq 0.244$, $\alpha = \|J^\dagger\|_2$, where J is the Jacobian of $F(x)$ at x^* and has full column rank,

3.

$$\|E_0\|_2^2 \le \left(\frac{\theta_1 - 0.5\alpha L\|e_0\|_2}{\alpha(1+\theta_1)}\right)^2 - \frac{L^2\|e_0\|_2^2}{1-\theta_1^2},$$

where $E_0 = B_0 - J$, where B_0 is the initial approximation Jacobian and $\theta_1 \simeq 0.671$.

Then, if x_i is the ith approximation to the solution generated by Algorithm 2 and $e_i = x_i - x^*$, we have

$$\|e_i\|_2 \leq \theta_1^i \|e_0\|_2.$$

Proof. Let B_i be the approximation of the Jacobian of ith iteration at x_i and define the matrix error E_i by $E_i = B_i - J$. From (1.3), we have

$$E_{i+1} = E_i - \frac{(y_i - B_i s_i)s_i^T}{s_i^T s_i},$$

so that

$$\begin{aligned} E_{i+1} &= E_i - \frac{[y_i - (J + E_i)s_i]s_i^T}{s_i^T s_i} \\ &= E_i(I - \frac{s_i s_i^T}{s_i^T s_i}) - \frac{(y_i - Js_i)s_i^T}{s_i^T s_i}. \end{aligned}$$

Since $F(x_{i+1}) \simeq F(x_i) + J(x_{i+1})s_i$, that is $y_i \simeq J(x_{i+1})s_i$, it follows that

$$E_{i+1} \simeq E_i(I - \frac{s_i s_i^T}{s_i^T s_i}) - \frac{(J(x_{i+1}) - J)s_i s_i^T}{s_i^T s_i},$$

so that,

$$\|E_{i+1}\|_2^2 \leq \|E_i\|_2^2 + \|J(x_{i+1}) - J\|_2^2 \leq \|E_1\|_2^2 + L^2\|e_{i+1}\|_2^2 \leq \|E_1\|_2^2 + L^2\|e_i\|_2^2, \tag{3.8}$$

since $\|e_{i+1}\|_2 < \|e_i\|_2$ and condition (1) of the theorem implies that, for $x, y \in N$,

$$\|J(x) - J(y)\|_2 \leq L\|x - y\|_2. \tag{3.9}$$

Now, if we define r by

$$r_i = F(x_i) - Je_i, \tag{3.10}$$

it follows from (3.9) and a Lemma 1. in [7] that

$$\|r_i\|_2 \leq 0.5L\|e_i\|_2^2.$$

Now, $x_{i+1} = x_i - B_i^\dagger F(x_i)$, so that, from (3.10), and the definition of x_i,

$$e_{i+1} = e_i - B_i^\dagger(Je_i + r). \tag{3.11}$$

Since $E_i = B_i - J$ and from Theorem 2.1, B_i is column full rank, that is $B_i^\dagger B_i = I$, it follows that

$$B_i^\dagger J = I - B_i^\dagger E_i. \tag{3.12}$$

Inserting (3.12) into (3.11), we have

$$e_{i+1} = (I - B_i^\dagger J)e_i - B_i^\dagger r = B_i^\dagger E_i e_i - B_i^\dagger r.$$

Taking the 2-norm on the both sides of the above equation for e_{i+1}, it follows that

$$\|e_{i+1}\|_2 \leq \|B_i^\dagger\|_2 \|E_i\|_2 \|e_i\|_2 + \|B_i^\dagger\|_2 \|r\|_2. \tag{3.13}$$

From Corollary 8.6.2 of [12] and $\alpha = \|J^\dagger\|_2$, if $\alpha\|E_i\|_2 < 1$, we have

$$\|B_i^\dagger\|_2 = \frac{1}{\sigma_{min}(B_i)} \leq \frac{1}{\sigma_{min}(J) - \|E_i\|_2} = \frac{\alpha}{1 - \alpha\|E_i\|_2},$$

where $\sigma_{min}(A)$ is the smallest nonzero singular value of matrix A.
Equation (3.13) can be rewritten as

$$\|e_{i+1}\|_2 \leq \frac{\alpha(\|E_i\|_2\|e_i\|_2 + \|r\|_2)}{1 - \alpha\|E_i\|_2} \leq \frac{\alpha(\|E_i\|_2\|e_i\|_2 + 0.5L\|e_i\|_2^2)}{1 - \alpha\|E_i\|_2},$$

that is,

$$\frac{\|e_{i+1}\|_2}{\|e_i\|_2} \leq \frac{\alpha(\|E_i\|_2 + 0.5L\|e_i\|_2)}{1 - \alpha\|E_i\|_2}. \tag{3.14}$$

Next, the proof proceeds inductively, and we use the subscript kth to denote kth approximation. Assume that for $i = 1, 2, \cdots, k$, $\|e_i\|_2 \leq \theta\|e_{i-1}\|_2$, where $0 \leq \theta \leq 1$. Then, from (3.8), we have

$$\begin{aligned} \|E_k\|_2^2 &\leq \|E_0\|_2^2 + L^2\|e_0\|_2^2(1 + \theta^2 \cdots + \theta^{2k-2}) \\ &< \|E_0\|_2^2 + \frac{L^2\|e_0\|_2^2}{1 - \theta^2}. \end{aligned}$$

So, from (3.14), $\|e_{k+1}\|_2 \leq \theta\|e_k\|_2$ is true only if

$$\|E_k\|_2 \leq \frac{(\theta - 0.5\alpha L\|e_0\|_2)}{\alpha(1 + \theta)}, \tag{3.15}$$

and it follows that from (3.14), that $\|e_{k+1}\|_2 \leq \theta\|e_k\|_2$, if

$$\|E_0\|_2^2 + \frac{L^2\|e_0\|_2^2}{1 - \theta} \leq \left(\frac{\theta - 0.5\alpha L\|e_0\|_2}{\alpha(1 + \theta)}\right)^2, \tag{3.16}$$

or

$$\alpha^2\|E_0\|_2^2 \leq \left(\frac{\theta - 0.5\mu}{1 + \theta}\right)^2 - \frac{\mu^2}{1 - \theta^2},$$

where $\mu = \alpha L\|e_0\|_2$.
It is only possible to satisfy (3.16) if

$$\left(\frac{\theta - 0.5\mu}{1 + \theta}\right)^2 - \frac{\mu^2}{1 - \theta^2} \geq 0,$$

i.e.,

$$\mu \leq \frac{2\theta(1-\theta^2)^{1/2}}{(1-\theta^2)^{1/2}+2(1+\theta)}.$$

We would like to obtain the largest bounds for e_0, thus the maximum value of μ, denoted as μ_1, is about 0.244, and the corresponding θ, denoted as θ_1, is about 0.671.

Therefore, if e_0 and E_0 satisfy condition (3) of the theorem, and $\|e_{i+1}\|_2 \leq \|\theta_1\|e_i\|_2$, for $i = 1, 2, \cdots, k$, then $\|e_{k+1}\|_2 \leq \theta_1\|e_k\|_2$, which implies that $\|e_{k+j}\|_2 \leq \theta_1^j\|e_k\|_2$, for $i = 1, 2, \cdots$. Also, if E_0 satisfies (3.15) with $k = 0$, then $\|e_1\|_2 \leq \theta_1\|e_0\|_2$, which implies that $\|e_i\|_2 \leq \theta_1^i\|e_0\|_2$. □

Theorem 3.2 *If $x_0 \in N$, $B_0 = J(x_0)$ and $\|e_0\|_2 \leq \mu_2/\alpha L$, where $\mu_2 \simeq 0.205$, then $\|e_i\|_2 \leq \theta_2^i\|e_0\|_2$ where $\theta_2 \simeq 0.724$.*

The proof of Theorem 3.2 is similar to the one for Theorem 2 in [4]. □

The above two theorems establish that our proposed method for rectangular problems has the same convergence properties as the Broyden method in [2, 4].

4 Numerical Experiments

In this section, we will compare the performance of three nonlinear least squares solver approaches.

Method 1.(Secant Quasi-Newton) The secant quasi-Newton method is proposed in Section 2. In this method, we use the secant update for the Jacobian matrix and the reverse mode AD to evaluate $\nabla f_2(x)$. In order to reduce the number of iterations, we use the criterion in [14] to occasionally refresh the exact Jacobian matrix. That is we recompute the exact Jacobian matrix by forward AD, rather than the Broyden update, when $\|\nabla f_2(x_{i+1})\|_{inf} \geq 0.9 * \|\nabla f_2(x_i)\|_{inf}$, where $\|\cdot\|_{inf}$ is the infinite norm.

Method 2.(Powell's dogleg) We choose the Powell's dogleg method [17] developed by Nielsen in their toolbox, 'immoptibox',[16]. Users have to supply their own Jacobian computation method for the dogleg method, thus we use the forward mode AD approach in ADMAT 2.0 [6] as our derivative computation method.

Method 3.(Standard Quasi-Newton) We choose the standard quasi-Newton method 'fminunc' in MATLAB optimization toolbox. Before calling 'fminunc' function, we have to set two flags of input argument 'options'— 'Gradobj' to on and 'Largescale' to off (See MATLAB function 'fminunc' documentation for details). The first flag will call the reverse mode AD in ADMAT-2.0 [6] (or user specified differentiation method) to compute gradients. The second flag is set to choose the BFGS quasi-Newton method [3, 9, 11, 18] with a mixed quadratic and cubic line search procedure to solve nonlinear least squares problems.

All experiments were taken on a laptop with Intel Duo Core processor T2300 1.66GHz and 1G RAM running Matlab 7.0 under Windows XP. MATLAB AD toolbox ADMAT 2.0 [6] carries the forward and reverse mode AD computations. Then, these three approaches are used to solve following three nonlinear typical least squares problems with various problem sizes.

Problem 1. (Data fitting problem) Consider the following data fitting problem.

$$M(\mathbf{x}, t) = x_1 e^{\frac{x_2}{t_i + x_3}} + e^{x_{\min\{i,n\}}},$$

where $i = 1, 2, \cdots, m$ and n is the number of entries in $\mathbf{x}$. We assume that there exists an $\mathbf{x}^\sharp$ so that

$$y_i = M(\mathbf{x}^\sharp, t_i) + \epsilon_i, \qquad i = 1, 2, \cdots, m,$$

where the $\{\epsilon_i\}$ are errors on the data ordinates, assumed to behave like 'white-noise'. Then, for any choice of $\mathbf{x}$, we can compute the residuals, $F = (f_1, f_2, \cdots, f_m)$,

$$f_i(\mathbf{x}) = y_i - M(\mathbf{x}, t_i).$$

The least squares fit is to determine a minimizer $\mathbf{x}^*$, such that

$$\|F(\mathbf{x}^*)\|_2 = \min \|F(\mathbf{x})\|_2.$$

Here, we use these three methods to solve 50 problems with n varying from 40 to 2000 and m varying from 50 to 2500. The vector $\mathbf{x}^\sharp$ is chosen as a random vector with the corresponding size. The initial guess, $\mathbf{x}_0$, is a random vector equally distributed on interval $[0, 1]$. Table 1 displays a part of the performance results of these three solvers. Figure 1 illustrates the execution time of these three solvers in seconds.

Problem 2. (Variably dimensioned function) In this example, we tested the three approaches on the variably dimensioned function in the Moré-Garbow-Hillstrom collection [15], which is widely used in testing unconstrained optimization software.

Consider solving the nonlinear least squares problem,

$$\min_x \|g(\mathbf{x})\|_2,$$

where $\mathbf{y} = g(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$, where $m = n + 2$, is the variably dimensioned function, which is defined as,

```
$\mathbf{y} = \mathbf{x} - 1;$
$tmp = 0;$
for $i = 1 : n$
    $tmp = tmp + i * (\mathbf{x}(i) - 1);$
end
$\mathbf{y}(n + 1) = tmp;$
$\mathbf{y}(n + 2) = tmp^2.$
```

problem size	Secant Quasi-Newton			Powell's dogleg			Standard Quasi-Newton		
$(m \times n)$	time	it	$\|F\|_2$	time	it	$\|F\|_2$	time	it	$\|F\|_2$
100× 80	2.42	8 (1)	1.4356	0.14	6	1.4356	1.25	67	1.4356
200× 160	5.46	10 (2)	1.5381	0.82	7	1.5381	6.01	97	1.5381
400× 320	16.34	10 (1)	2.3425	11.47	7	2.3425	37.99	98	2.3436
800×640	31.92	9 (1)	4.0912	48.34	7	4.0912	123.32	98	4.0912
1600×1240	229.44	13 (2)	5.5856	1245.69	7	5.5856	669.17	98	5.5888
2500×2000	739.33	12 (2)	6.6311	1591.48	7	6.6311	2353.89	98	6.9432

Table 1: The performance and results comparison of the secant Quasi-Newton, Powell's dogleg and standard quasi-Newton approaches for data fitting problem. (Column 'it' for the number of iterations.) Numbers in brackets in column 'it' is the number of Jacobian revaluations.

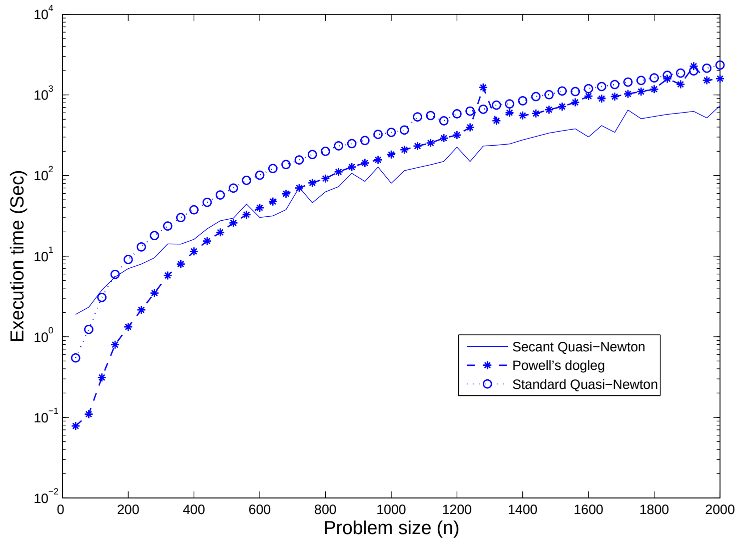

Figure 1: Running times for three approaches to solving varying-sized data fitting problems.

problem	Secant Quasi-Newton			Powell's dogleg		
size (n)	time	it	$\|g\|_2$	time	it	$\|g\|_2$
100	0.08	8 (1)	1.54E−8	0.23	17	1.54E−8
200	0.16	6 (1)	1.52E−8	1.57	19	1.32E−8
400	0.72	6 (1)	2.85E−7	11.34	21	2.85E−7
800	4.82	6 (1)	4.20E−7	97.78	22	5.10E−7
1250	17.59	6 (1)	5.90E−7	371.89	24	5.90E−7
1500	33.39	6 (1)	1.08E−6	637.58	24	1.09E−6

Table 2: The performance and results comparison of the secant Quasi-Newton and Powell's dogleg approaches for variably dimensioned function. (Column 'it' for the number of iterations.) Numbers in brackets in column 'it' is the number of Jacobian revaluations.

The initial guess, $\mathbf{x}_0$ is a random vector distributed on interval $[0, 1]$. As we know, the minimizer of $\min \|g(\mathbf{x})\|_2$ is $\mathbf{x}^* = [1, 1, \cdots, 1, 1]^T$ and $\|g(\mathbf{x}^*)\|_2 = 0$. Similar to Problem 1, we list a part of the performance results in Table 2 and plot the execution times in Figure 2. In Table 2 and Figure 2, we only record the performance results of the secant Quasi-Newton and Powell's dogleg method since the standard Quasi-Newton method failed in solving this problem.

Problem 3. (Penalty function I) This testing problem is also selected from the Moré-Garbow-Hillstrom collection [15].

Consider solving the nonlinear least squares problem,

$$\min_{x} \|g(\mathbf{x})\|_2,$$

where $\mathbf{y} = g(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$, where $m = n + 1$, is the penalty function I, which is defined as,

$$\begin{aligned} &\mathbf{y} = \sqrt{10^{-5}} * (\mathbf{x} - 1); \\ &\mathbf{y}(n+1) = \mathbf{x}^T * \mathbf{x}. \end{aligned}$$

The initial guess, $\mathbf{x}_0$ is also chosen randomly, whose entries are distributed uniformly on interval $[0, 1]$. Table 3 lists some performance results of three approaches while Figure 3 plots all the execution time for the penalty function I with n varying for 50 to 1000.

The numerical results presented above support the thesis that our proposed secant method (in combination with reverse-mode automatic differentiation), can be an efficient way to solve nonlinear least-squares minimization problems, relative to popular alternatives. Reverse mode automatic differentiation plays a crucial role because this allows for the efficient and accurate determination of the gradient of the nonlinear least-squares objective function without assuming the accurate determinate of the (perhaps expensive) Jacobian matrix.

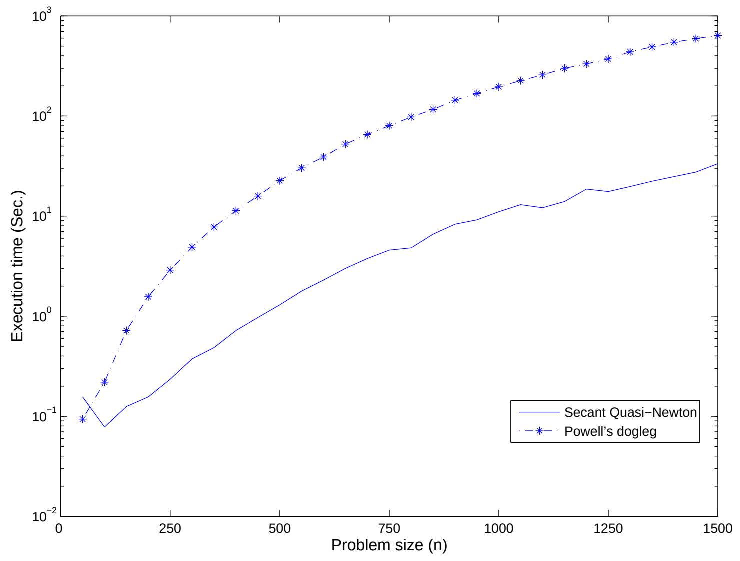

Figure 2: Running times for three approaches to solving variably dimensioned function problems.

problem	Secant Quasi-Newton			Powell's dogleg			Standard Quasi-Newton		
size (n)	time	it	$\|F\|_2$	time	it	$\|F\|_2$	time	it	$\|F\|_2$
100	0.14	23(9)	0.0035	0.13	19	0.0035	1.61	63	0.0035
200	0.39	25(10)	0.0045	0.91	20	0.0045	3.32	69	0.0045
400	2.11	25(8)	0.036	7.55	22	0.036	7.36	61	0.036
800	15.81	25(10)	0.064	58.71	22	0.064	15.59	50	0.064
1000	18.08	17(9)	0.082	113.38	22	0.082	21.79	51	0.082

Table 3: The performance and results comparison of the secant Quasi-Newton, Powell's dogleg and standard quasi-Newton approaches for the penalty function I. (Column 'it' for the number of iterations.) Numbers in brackets in column 'it' is the number of Jacobian revaluations.

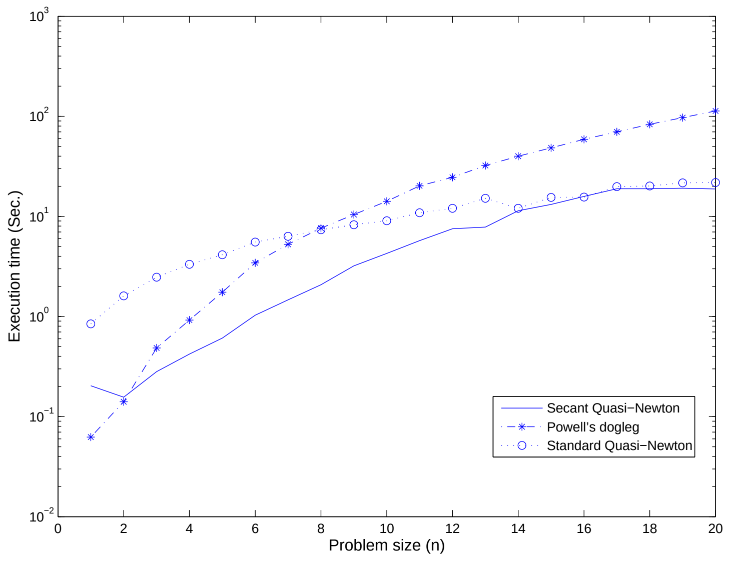

Figure 3: Running times for three approaches to solving the penalty function I problems.

5 Concluding Remarks

We have presented a new secant method for nonlinear least squares, especially for low-residual problems. This method is computationally feasible because of the application of (reverse-mode) automatic differentiation which enables the efficient calculation of the gradient of the objective function without requiring the calculation of the Jacobian matrix (which can be prohibitively expensive).

We have established strong supporting local convergence results, and a standard globalization technique yields good global behavior. The numerical results indicate this new approach can outperform the best existing methods, and is certainly competitive.

We conclude with one caution. The memory demands of reverse mode AD can be large and if fast memory availability is exceeded then the performance of this approach can degrade in a significant way. However, there are techniques available - and this is an active research area - that can often limit memory requirements [5] thus increasing the range of applicability of reverse-mode AD and thus the proposed nonlinear least-squares techniques in this paper.

6 Appendix

Proof of Theorem 2.1.
We first assume that B_{k+1} updated by (1.3) is rank deficient. Thus, there exists a nonzero vector $z \in R^n$, such that

$$0 = B_{k+1}z = B_k z + \frac{s_k^T z}{s_k^T s_k}(y_k - B_k s_k).$$

The above equation can be rewritten as,

$$B_k z = \frac{s_k^T z}{s_k^T s_k}(y_k - B_k s_k) \neq 0,$$

since B_k is full column rank and z is nonzero. For any nonzero vector $q \in \mathcal{N}(B_k^T)$, where $\mathcal{N}(B_k^T)$ is the null space of B_k^T, that is $B_k^T q = 0$ or $q^T B_k = 0$, thus, we have

$$0 = q^T B_k z = \frac{q^T y_k s_k^T z}{s_k^T s_k} - \frac{q^T B_k s_k s_k^T z}{s_k^T s_k},$$

i.e.,

$$q^T y_k = 0.$$

It implies that $y_k \in \mathcal{R}(B_k)$, where $\mathcal{R}(B_k)$ is the range of B_k, that is there exists a nonzero vector $p \in R^n$ such that $y_k = B_k p$. Since B_k is full column rank, it follows that $p = B_k^\dagger y_k$, where $B_k^\dagger$ is the Moore-Penrose inverse.

Then, insert $y_k = B_k p$ into (1.3), we have

$$B_{k+1} = B_k(I + \frac{(p - s_k)s_k^T}{s_k^T s_k}).$$

Since B_{k+1} is rank deficient and B_k is full column rank, the matrix $(I + \frac{(p-s_k)s_k^T}{s_k^T s_k})$ must be rank deficient. According to the Sherman-Morrison formula, this implies that

$$0 = 1 + \frac{s_k^T(p - s_k)}{s_k^T s_k} = s_k^T p = s_k^T B_k^\dagger y_k,$$

which contradicts the assumption that $s_k^T B_k^\dagger y_k \neq 0$. Thus, B_{k+1} is full column rank. □

References

[1] A. R. Conn, N. I. M. Gould and P. L. Toint, *Trust-region methods*, SIAM, Philadelphia, 2000.

[2] C.G. Broyden, *A class of methods for solving nonlinear simultaneous equations*, Math. Comput., vol.19, 577-593, 1965.

[3] C. G. Broyden, *The Convergence of a Class of Double-rank Minimization Algorithms*, Journal of the Institute of Mathematics and Its Applications, vol. 6, 76-90, 1970.

[4] C.G. Broyden, *The convergence of an algorithm for solving sparse nonlinear systems*, Math. Comput., vol. 25, 285-294, 1971.

[5] T.F. Coleman and G.F. Jonsson, *The efficient computation of structured gradients using automatic differentiation*, SIAM J. Sci. Comput. Vol. 20, 1999, 1430–1437.

[6] T.F. Coleman and W. Xu, *ADMAT-2.0*, Available at http://www.math.uwaterloo.ca/CandO_Dept/securedDownloadsWhitelist/index.shtml.

[7] J.E. Dennis, *On convergence of Newton-like methods*, Numerical Methods for Non-linear Algebraic Equations edited by P. Rabinowitz, Gordon & Breach, London, 1970.

[8] J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, Philadelphia, PA, 1996.

[9] R. Fletcher, *A New Approach to Variable Metric Algorithms*, Computer Journal, vol. 13, 317-322, 1970.

[10] D.M. Gay, *Some convergence properties of Broyden's method*, SIAM Num. Anal., vol. 16, 623-630, 1979.

[11] D. Goldfarb, *A Family of Variable Metric Updates Derived by Variational Means*, Mathematics of Computation, vol. 24, 23-26, 1970.

[12] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[13] A. Griewank and G.F. Corliss, eds., *Automatic Differentiation of Algorithms: Theory, Implementation and Applications*, SIAM, Philadelphia, PA, 1991.

[14] K. Madsen,*A Combined Gauss-Newton and Quasi-Newton Method for Non-Linear Least Squares*, IMM, DTU. Report NI-88-10, 1988.

[15] J.J. Moré, B.S. Grabow and K.E. Hillstrom, *Test unconstrained optimization software*, ACM Tran. Math. Soft. vol.7, 17-24, 1981.

[16] H.B. Nielsen, *immoptibox—A MATLAB TOOLBOX FOR OPTIMIZATION*, IMM, Technical University of Denmark, 2006. Available at http://www2.imm.dtu.dk/ hbn/immoptibox/

[17] M.J.D. Powell, *A hybrid method for nonlinear equations*, in Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Gordon and Breach, London, 87-114, 1970.

[18] D. F. Shanno, *Conditioning of Quasi-Newton Methods for Function Minimization*, Mathematics of Computation vol. 24, 647-656, 1970.

[19] G. Wang, Y. Wei and S. Qiao, *Generalized Inverses: Theory and Computations*, Science Press, Beijing/New York, 2004.