algorithms

MDPI

Article

An Efficient Algorithm for the Separable Nonlinear Least Squares Problem

Yunqiu Shen * and Tjalling J. Ypma *

Department of Mathematics, Western Washington University, Bellingham, WA 98225-9063, USA
* Correspondence: yunqiu.shen@wwu.edu (Y.S.); tjalling.ypma@wwu.edu (T.J.Y.); Tel.: +1-360-650-3785 (T.J.Y.)

Received: 7 June 2017; Accepted: 1 July 2017; Published: 10 July 2017

Abstract: The nonlinear least squares problem $min_{y,z}\|A(y)z + b(y)\|$, where $A(y)$ is a full-rank $(N + \ell) \times N$ matrix, $y \in \mathbf{R}^n$, $z \in \mathbf{R}^N$ and $b(y) \in \mathbf{R}^{N+\ell}$ with $\ell \geq n$, can be solved by first solving a reduced problem $min_y\|f(y)\|$ to find the optimal value y^* of y, and then solving the resulting linear least squares problem $min_z\|A(y^*)z + b(y^*)\|$ to find the optimal value z^* of z. We have previously justified the use of the reduced function $f(y) = C^T(y)b(y)$, where $C(y)$ is a matrix whose columns form an orthonormal basis for the nullspace of $A^T(y)$, and presented a quadratically convergent Gauss–Newton type method for solving $min_y\|C^T(y)b(y)\|$ based on the use of QR factorization. In this note, we show how LU factorization can replace the QR factorization in those computations, halving the associated computational cost while also providing opportunities to exploit sparsity and thus further enhance computational efficiency.

Keywords: separable equations; nonlinear least squares; full-rank matrices; QR factorization; over-determined systems; Gauss–Newton method; least squares solutions; LU factorization; quadratic convergence

MSC: 65H10

1. Introduction

Many applications [1–3] lead to the need to find a solution (y^*, z^*) of the separable nonlinear overdetermined least squares problem

$$min_{y,z}\|F(y,z)\| \equiv min_{y,z}\|A(y)z + b(y)\|, \tag{1}$$

where the full-rank matrix $A(y) \in \mathbf{R}^{(N+\ell)\times N}$ and the vector $b(y) \in \mathbf{R}^{N+\ell}$ are twice Lipschitz continuously differentiable functions of $y \in \mathbf{R}^n$, $z \in \mathbf{R}^N$ and $\ell \geq n$. Here, we are using the Euclidean norm. The classic Golub–Pereyra variable projection method [4] replaces the problem (1) by the simpler least squares problem

$$min_y\|f(y)\| \equiv min_y\|[I - A(y)A^+(y)]b(y)\|. \tag{2}$$

Once y^* has been obtained by solving (2), z^* can be found by solving the resulting linear least squares problem

$$min_z\|A(y^*)z + b(y^*)\|; \tag{3}$$

formally,

$$z^* = -A^+(y^*)b(y^*), \tag{4}$$

where $A^+(y^*)$ is the pseudo-inverse of $A(y^*)$. An alternative method that reduces (1) to a smaller least squares problem $min_y\|f(y)\|$ was proposed in [5,6] for the case $\ell = n$, and the associated iterative

technique was then applied to the case $\ell > n$ in [7] but without a complete analytical justification. The reduced system is

$$min_y \|f(y)\| \equiv min_y \|C^T(y)b(y)\|, \tag{5}$$

where $C(y)$ is an $(N+\ell) \times N$ matrix whose columns form an orthonormal basis for $Null(A^T(y))$. In [8], we presented a quadratically convergent Gauss–Newton type iterative method, incorporating second derivative terms, to solve this problem, and provided a complete theoretical justification for our technique. In particular, we showed in [8] that our Gauss–Newton type method, and also the standard Gauss–Newton method [9–11], which omits the second derivative terms and is not quadratically convergent, are both invariant with respect to the specific basis matrix $C(y)$ that is used at any particular point in the iteration. This makes it possible to substitute, at each point of the iteration, any easily computed local orthonormal basis for the nullspace of $A^T(y)$. This observation also makes it possible to compute both the first and the second derivatives involved in the computation very cheaply.

Many instances of (1), such as those arising from the discretization of linear differential equations with only a few nonlinear parameters, involve $N \to \infty$, while ℓ and n are fixed and small. In this case, the main computational cost in each step of our quadratically convergent Gauss–Newton type method of [8] is the QR factorization of $A(y)$, costing approximately $4N^3/3$ flops [12] per iteration. In this note, we show how the relevant computations can instead be performed by using one LU factorization of $A(y)$ per iteration, so that the computational cost of the computation is essentially halved to $2N^3/3$ flops [12]. We note that, in the case of discretizations of differential equations, the matrix $A(y)$ is typically very sparse and strongly patterned. Since LU factorization more easily exploits and preserves such sparsity and patterns than does QR factorization, this technique also opens up opportunities for further reductions in the computational cost in such applications.

In a previous paper [3], we presented the use of LU factorizations in the context of solving separable systems of nonlinear equations and zero-residual separable nonlinear least squares problems. We mentioned there that extending that approach to nonzero residual problems remained an open question. The current paper resolves that question, while also improving on the technique of [3] in other ways. In particular, our new method does not involve the singular value decomposition, and in contrast to the use of the standard Gauss–Newton method in [3], it retains quadratic convergence even in the case of nonzero residual problems. The latter is achieved by retaining the second derivative terms, and we show here how those terms can be computed very efficiently. This is in contrast to the usual use of the Gauss–Newton method, which sacrifices the quadratic convergence for the convenience of not computing the second derivatives.

In the next section, we present the relevant results from [8] relating to our technique for solving (5) by our Gauss–Newton type method using QR factorization, and show how those results and methods can be modified to use LU factorization. The resulting algorithm and some numerical examples to illustrate the method are given in the following two sections. The numerical results exhibit the quadratic convergence expected from our Gauss–Newton type method.

2. Analysis

We begin by presenting the relevant background results from [8]. Assume that $A(y)$ and $b(y)$ are twice Lipschitz continuously differentiable in a neighborhood of the least squares solution y^* of (1). As shown in [5–8], there exists a smoothly differentiable $(N+\ell) \times \ell$ matrix $C(y)$ whose columns form an orthonormal basis of $Null(A^T(y))$ in a neighborhood of y^*. Then, finding the least squares solution of (1) can be reduced to finding the least squares solution of (5). Our Gauss–Newton type method for solving (5) takes the form

$$\begin{aligned} & [(f'(y^{(m)}))^T f'(y^{(m)}) + \Sigma_{i=1}^{\ell} f_i(y^{(m)}) H_i(y^{(m)})](y^{(m+1)} - y^{(m)}) \\ = \; & -(f'(y^{(m)}))^T f(y^{(m)}), \end{aligned} \tag{6}$$

where $f(y) \equiv C^T(y)b(y)$ and $f'(y)$, $f_i(y)$ and $H_i(y)$ denote the derivative of $f(y)$, the i-th component of $f(y)$, and the Hessian matrix of $f_i(y)$, respectively.

In [8], we proved that, at any point $y^{(m)}$ in the iteration (6), the particular matrix $C(y^{(m)})$ whose columns form an orthonormal basis for the nullspace of $A^T(y^{(m)})$ can be replaced by any other orthonormal basis $\widetilde{C}(y^{(m)})$ for this space, without changing $y^{(m+1)}$. Thus, instead of using the function $f(y) \equiv C^T(y)b(y)$ in (6), we may use a different function $g(y) \equiv \widetilde{C}^T(y)b(y)$ at each iteration point $y^{(m)}$. This freedom to use any orthonormal basis for the nullspace of $A^T(y)$ at the point $y^{(m)}$ is key to our numerical technique. It also permits us to write simply $C(y^{(m)})$ and $f(y^{(m)})$ instead of $\widetilde{C}(y^{(m)})$ and $g(y^{(m)})$ in the rest of this note, regardless of the particular matrix $C(y)$ being used, and for simplicity we write $\overline{y}$ instead of $y^{(m)}$. The analysis below centers on the efficient computation of the terms required to use (6) at the particular point $y^{(m)} = \overline{y}$.

The following results are derived and used in [5–8] and are required for our analysis below. With the terms as defined above, and writing $C_i(y)$ for the i-th column of the matrix $C(y)$,

$$C^T(y)A(y) = 0, \quad C^T(y)C(y) = I, \tag{7}$$

$$f'(y) = C^T(y)b'(y) + \left[\begin{array}{ccc} [C^T(y)]_1'b(y) & \ldots & [C^T(y)]_n'b(y) \end{array} \right], \tag{8}$$

and the (j,k)-th entry of the $n \times n$ Hessian matrix $H_i(y)$ is

$$[C_i^T(y)]''_{j,k}b(y) + [C_i^T(y)]'_j[b(y)]'_k + [C_i^T(y)]'_k[b(y)]'_j + C_i^T(y)[b(y)]''_{j,k}. \tag{9}$$

Here, we use $[C(y)]'_j$ and $[C(y)]''_{j,k}$ to denote the termwise first derivative w.r.t. y_j and second derivative w.r.t. y_j, y_k, respectively, of $C(y)$, for $j,k = 1, \ldots, n$.

Define the $(N+\ell) \times (N+\ell)$ nonsingular matrix

$$M(y) \equiv \left[\begin{array}{cc} A(y) & C(y) \end{array} \right]. \tag{10}$$

Then, we showed in [8] that

$$[C(\overline{y})]'_j M(\overline{y}) = \left[\begin{array}{cc} -C^T(\overline{y})[A(\overline{y})]'_j & 0_{(N+\ell)\times\ell} \end{array} \right] \tag{11}$$

and

$$[C^T(\overline{y})]''_{j,k}M(\overline{y}) = \left[-[C^T(\overline{y})]'_j[A(\overline{y})]'_k - [C^T(\overline{y})]'_k[A(\overline{y})]'_j - C^T(\overline{y})[A(\overline{y})]''_{j,k} \mid [K(\overline{y})]''_{j,k} \right], \tag{12}$$

where the (p,q)-entry $\left[K(\overline{y})]''_{j,k}\right]_{(p,q)}$ of the upper triangular matrix $\left[K(\overline{y})]''_{j,k}\right]$ is

$$\left[K(\overline{y})]''_{j,k}\right]_{(p,q)} = \begin{cases} 0 & \text{if } p > q, \\ -[C_p^T(\overline{y})]'_j[C_p(\overline{y})]'_k & \text{if } p = q, \\ -[C_p^T(\overline{y})]'_j[C_q(\overline{y})]'_k - [C_p^T(\overline{y})]'_k[C_q(\overline{y})]'_j & \text{if } p < q. \end{cases}$$

Throughout the rest of this paper, we assume not only that $A(y)$ has full rank N, but also that it is sufficiently well-conditioned that LU factorization with partial pivoting can be performed safely. For separable nonlinear systems with a rank-deficient $A(y)$, the technique of bordered matrices [13–16] may be applied to produce a full rank matrix.

The following theorem lays the foundation for our approach to using the LU factorization.

Theorem 1. *Let the LU factorization of the rectangular matrix $A(\overline{y})$ with some permutation matrix $P(\overline{y})$ be*

$$P(\overline{y})A(\overline{y}) = L(\overline{y})U(\overline{y}), \tag{13}$$

where $A(\overline{y})$ is $(N+\ell)\times N$, $P(\overline{y})$ is an $(N+\ell)\times(N+\ell)$ permutation matrix, $L(\overline{y})$ is an $(N+\ell)\times N$ unit lower triangular matrix and $U(\overline{y})$ is an $N\times N$ nonsingular upper triangular matrix.

(a) The matrix

$$\overline{M}(\overline{y}) \equiv [A(\overline{y})|S(\overline{y})], \tag{14}$$

where

$$S(\overline{y}) \equiv P^T(\overline{y}) \begin{bmatrix} 0 \\ I_\ell \end{bmatrix},$$

is nonsingular.

(b) Define

$$\Psi(\overline{y}) \equiv [\overline{M}^T(\overline{y})]^{-1} \begin{bmatrix} 0 \\ I_\ell \end{bmatrix}. \tag{15}$$

Then, the thin positive QR factorization

$$\Psi(\overline{y}) \equiv C(\overline{y})R(\overline{y}) \tag{16}$$

produces a thin orthonormal $(N+\ell)\times\ell$ matrix $C(\overline{y})$ and a small $\ell\times\ell$ nonsingular upper triangular matrix $R(\overline{y})$, and

$$C^T(\overline{y})A(\overline{y}) = 0, \quad C^T(\overline{y})C(\overline{y}) = I_\ell. \tag{17}$$

Proof. (a) By direct computation, $\overline{M}(\overline{y})$ is a product of nonsingular matrices:

$$\begin{aligned} \overline{M}(\overline{y}) &= P^T(\overline{y}) \left[\begin{array}{c|c} L(\overline{y})U(\overline{y}) & \begin{matrix} 0 \\ I_\ell \end{matrix} \end{array}\right] \\ &= P^T(\overline{y}) \left[\begin{array}{c|c} L(\overline{y}) & \begin{matrix} 0 \\ I_\ell \end{matrix} \end{array}\right] \begin{bmatrix} U(\overline{y}) & 0 \\ 0 & I_\ell \end{bmatrix}. \end{aligned} \tag{18}$$

(b) By (15) and (16),

$$\begin{aligned} C^T(\overline{y})A(\overline{y}) &= R^{-T}(\overline{y})\Psi^T(\overline{y})A(\overline{y}) = R^{-T}(\overline{y}) \begin{bmatrix} 0 & I_\ell \end{bmatrix} [\overline{M}(\overline{y})]^{-1}(\overline{y})A(\overline{y}) \\ &= R^{-T}(\overline{y}) \begin{bmatrix} 0 & I_\ell \end{bmatrix} \begin{bmatrix} I_N \\ 0 \end{bmatrix} = 0. \end{aligned}$$

□

From (17), we see that the matrix $C(\overline{y})$ constructed in the theorem satisfies (7) at $\overline{y}$, and thus, as shown in [8], Equations (11) and (12) hold for this matrix at $\overline{y}$.

To determine the computational cost, recall that we assume that $N \to \infty$, while ℓ and n are fixed and small. Then, the thin QR factorization in (16) is cheap since the matrix $\Psi(\overline{y})$ has only ℓ columns. More specifically, the LU factorization of the $(N+\ell)\times N$ dimensional $A(\overline{y})$ in (13) costs $(N+\ell)N^2 - N^3/3 \approx 2N^3/3$ flops, with an additional $O(N^2)$ comparisons to produce $P(\overline{y})$ [12], while computing $S(\overline{y})$ only costs $O(N)$ flops and the thin QR factorization of the $(N+\ell)\times\ell$ dimensional matrix $S(\overline{y})$ takes only $(N+\ell)\ell^2 - \ell^3 = O(N)$ flops [12].

Notice that solving (11) and (12) involves solving several equations of the form $u^T M(\overline{y}) = v^T$. However, the matrix $M(y)$ defined in (10) and which is used in (11) and (12), differs from the matrix $\overline{M}(\overline{y})$ defined in (14) whose LU factors are given in (18). To reduce the cost of solving (11) and (12) from $O(N^3)$ to $O(N^2)$ operations, we can exploit the factorization (18) of the matrix $\overline{M}(\overline{y})$, as below. This result is essentially an application of the Sherman–Morrison–Woodbury formula.

Theorem 2. *Let $P(\overline{y})A(\overline{y}) = L(\overline{y})U(\overline{y})$ be the LU factorization of $A(\overline{y})$ with a permutation matrix $P(\overline{y})$ as in (13), and let the matrices $\overline{M}(\overline{y})$, $S(\overline{y})$ and $C(\overline{y})$ be defined as in (14)–(16), while $M(y)$ is defined by (10). Denote*

$$W(\overline{y}) = I_{N+\ell} - [C(\overline{y}) - S(\overline{y})]C^T(\overline{y}). \tag{19}$$

Then,

(a) $M^{-1}(\overline{y}) = \overline{M}^{-1}(\overline{y})W(\overline{y})$;

(b) *For $u, v, w \in \mathbf{R}^{N+\ell}$, $u^T M(\overline{y}) = v^T$ if and only if*

$$w^T\overline{M}(\overline{y}) = v^T, \quad u^T = w^T W(\overline{y}). \tag{20}$$

Proof. (a) By (18),

$$\begin{aligned} & W(\overline{y})M(\overline{y}) = W(\overline{y}) \left[\begin{array}{c|c} A(\overline{y}) & C(\overline{y}) \end{array} \right] \\ = & \left[\{I_{N+\ell} - [C(\overline{y}) - S(\overline{y})]C^T(\overline{y})\}A(\overline{y}) \mid \{I_{N+\ell} - [C(\overline{y}) - S(\overline{y})]C^T(\overline{y})\}C(\overline{y}) \right] \\ = & [A(\overline{y}) \mid C(\overline{y}) - [C(\overline{y}) - S(\overline{y})]] = [A(\overline{y}) \mid S(\overline{y})] = \overline{M}(\overline{y}). \end{aligned}$$

Then,

$$\begin{aligned} & M^{-1}(\overline{y}) = \overline{M}^{-1}(\overline{y})[\overline{M}(\overline{y})]M^{-1}(\overline{y}) \\ = & \overline{M}^{-1}(\overline{y})[W(\overline{y})M(\overline{y})]M^{-1}(\overline{y}) = \overline{M}^{-1}(\overline{y})W(\overline{y}). \end{aligned}$$

Part (b) follows directly from (a). □

Finally, we show how the value z^* defined by (3) can be computed efficiently using the LU factorization of $A(y^*)$.

Theorem 3. *$z^* = -A^+(y^*)b(y^*)$ if and only if*

$$z^* = -\left[\begin{array}{cc} I_N & 0_{N\times\ell} \end{array} \right] \left[\begin{array}{cc} A(y^*) & C(y^*) \end{array} \right]^{-1} b(y^*). \tag{21}$$

Proof. Let the QR factorization of $A(y^*)$ be

$$A(y^*) = QR \equiv \left[\begin{array}{cc} (Q_1)_{(N+\ell)\times N} & (Q_2)_{(N+\ell)\times\ell} \end{array} \right] \left[\begin{array}{c} R_1 \\ 0 \end{array} \right].$$

Since $A(y^*)$ is full rank, we see that

$$\begin{aligned} & A^+(y^*) = [A^T(y^*)A(y^*)]^{-1}A^T(y^*) \\ = & \left(\left[\begin{array}{cc} (R_1)^T & 0 \end{array} \right] Q^T Q \left[\begin{array}{c} R_1 \\ 0 \end{array} \right] \right)^{-1} \left[\begin{array}{cc} (R_1)^T & 0 \end{array} \right] Q^T \\ = & R_1^{-1}((R_1)^T)^{-1} \left[\begin{array}{cc} (R_1)^T & 0 \end{array} \right] Q^T = \left[\begin{array}{cc} R_1^{-1} & 0 \end{array} \right] Q^T. \end{aligned}$$

Since $C(y^*)$ is an orthonormal matrix, there exists an $\ell \times \ell$ orthogonal matrix Θ such that

$$\begin{bmatrix} I_N & 0_{N\times\ell} \end{bmatrix} \begin{bmatrix} A(y^*) & C(y^*) \end{bmatrix}^{-1} = \begin{bmatrix} I_N & 0_{N\times\ell} \end{bmatrix} \left(Q \begin{bmatrix} R_1 & 0 \\ 0 & \Theta_{\ell\times\ell} \end{bmatrix} \right)^{-1}$$
$$= \begin{bmatrix} I_N & 0_{N\times\ell} \end{bmatrix} \left(\begin{bmatrix} R_1^{-1} & 0 \\ 0 & \Theta_{\ell\times\ell}^T \end{bmatrix} Q^T \right) = \begin{bmatrix} R_1^{-1} & 0 \end{bmatrix} Q^T.$$

Hence, the conclusion follows. □

Computing z^* via (21) involves the matrix $M(y^*)$ of (10), but the technique of Theorem 2 may again be used to replace this by the known LU factorization (18) of $\overline{M}(y^*)$ obtained from the LU factorization of $A(y^*)$. Thus, the cost of this step is also only $2N^3/3 + O(N^2)$ flops.

3. Algorithm

We now give a detailed Algorithm 1 based on the analysis from the last section.

Algorithm 1: Given the function $F(y,z) = A(y)z + b(y)$ with a matrix $A(y)$ that is full rank in a neighborhood of y^*, a small positive real number ϵ, a positive integer m_0, and a point $y^{(0)}$ near the solution y^*. For $m = 0, 1, 2, ..., m_0$, do steps (a)–(l):

(a) Compute the LU factorization (13) of $A(y^{(m)})$:

$$P(y^{(m)})A(y^{(m)}) = L(y^{(m)})U(y^{(m)});$$

(b) Form the matrix $S(y^{(m)})$ and the matrix $\overline{M}(y^{(m)})$ as in (14);
(c) Use (16) and the factorization (18) to form the matrix $C(y^{(m)})$;
(d) Compute $f(y^{(m)}) = C^T(y^{(m)})b(y^{(m)})$;
(e) Use (11) and (20) to solve for $[C(y^{(m)})]'_j$, $j = 1, ..., n$;
(f) Use (8) to form $f'(y^{(m)})$;
(g) Use (12) and (20) to solve for $[C^T(y^{(m)})]''_{j,k}$, $j, k = 1, ..., n$;
(h) Use (9) to find the (j,k)-th entry of $H_i(y^{(m)})$, $i = 1, \ldots, \ell$;
(i) Form the matrix $[(f'(y^{(m)}))^T f'(y^{(m)}) + \Sigma_{i=1}^{\ell} f_i(y^{(m)})H_i(y^{(m)})]$ for the left-hand side of (6);
(j) Form the vector $-(f'(y^{(m)}))^T f(y^{(m)})$ for the right side of (6);
(k) Compute $y^{(m+1)}$ by solving (6);
(l) If $||y^{(m+1)} - y^{(m)}|| < \epsilon$ stop, output $y^* \approx y^{(m+1)}$ and z^* from (21) using (19) and (20). Otherwise, if $m < m_0$, replace m by $m + 1$ and go to (a); if $m = m_0$, output that the method fails to obtain y^* within given m_0 and ϵ.

Note that the cost of solving (6) for $y^{(m+1)} - y^{(m)}$ is only $O(1)$ since the matrix involved is only $\ell \times \ell$ and we assume $\ell = O(1)$. Thus, the overall cost per iteration remains $2N^3/3 + O(N^2)$, the primary cost being the LU factorization of $A(y^{(m)})$. The standard Gauss–Newton method, which omits the second derivative terms, omits steps *(g)* and *(h)*, but the resulting method is no longer quadratically convergent.

4. Examples

We present three examples to illustrate our algorithm for computing the least squares solution of overdetermined separable equations. Our examples have large N and small n, ℓ, approximating the theoretical characteristic that $n, \ell = O(1)$ as $N \to \infty$. Similar results hold if we use much larger N.

Example 1. *Consider the least squares solution of the overdetermined system*

$$A(y)z + b(y) \equiv \begin{bmatrix} (A_1(y))_{(2k+1)\times(2k+1)} + I_{2k+1} \\ (A_2(y))_{\ell\times(2k+1)} \end{bmatrix} z + \begin{bmatrix} (b_1(y))_{(2k+1)\times 1} \\ (b_2(y))_{\ell\times 1} \end{bmatrix} = 0, \tag{22}$$

where

$$A_1(y) = \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \dots & & \\ & 1 & -2 & 1 \\ & & 1 & -2 \end{bmatrix}, \quad b_1(y) = 0_{(2k+1)\times 1},$$

$$A_2(y) = \begin{bmatrix} e^T{}_{k+1} \\ 0 \end{bmatrix}, \quad b_2(y) = \begin{bmatrix} -1 \\ 0.02\sqrt{\alpha(y, y^*, \beta)} \end{bmatrix},$$

with

$$\alpha(y, y^*, \beta) = (y - y^*)^2 - (y - y^*)sin(2(y - y^*)) - 0.5cos(2(y - y^*)) + \beta$$

and $y \in \mathbf{R}$ *(thus* $n = 1$*), and where* e_{k+1} *is the* $(k+1)$*-th standard unit vector in* $\mathbf{R}^{2k+1}$*. The selected value of* $y^* \in \mathbf{R}$ *is the value of the exact solution* (y^*, z^*)*. Here,* $\ell = 2$*.*

The problem can be regarded as a constrained generalized eigenvalue problem. We choose $n = 21$, $\beta = 9.5$, and $y^* = 0.25csc^2(\pi/44) \approx 49.1228712506303890$. For the eigenvalues of matrices of the form of $A_1(y)$, see [17]. Beginning with $y^{(0)} = 48$ and using Algorithm 1, we obtain the results in Table 1 for $y^{(m)}, m = 1, 2, 3, 4$, which shows quadratic convergence of the method.

Table 1. Gauss–Newton type iterations for Example 1.

m	$y^{(m)}$	$\|\|y^{(m)} - y^*\|\|$
0	4.80000000000000e+001	1.1229e+000
1	4.90312546237768e+001	9.1617e−002
2	4.91243552951534e+001	1.4840e−003
3	4.91228716369764e+001	3.8635e−007
4	4.91228712506304e+001	2.8422e−014

After $y^{(4)}$ is obtained, we compute $z^{(4)} \approx z^*$ using (21), where $z^* \in \mathbf{R}^{21}$, $z_j^* = sin(j * \pi/22)$, $j = 1, \dots, 21$. We find that $\|z^{(4)} - z^*\| = 5.2774 \times 10^{-15}$, and the 2-norm of the residual of $A(y^{(4)})z^{(4)} + b(y^{(4)})$ is ≈0.06.

Example 2. *Consider the least squares solution of the overdetermined system*

$$A(y)z + b(y) \equiv \begin{bmatrix} A_1(y) \\ A_2(y) \end{bmatrix} z + \begin{bmatrix} b_1(y) \\ b_2(y) \end{bmatrix} = 0, \tag{23}$$

where

$$A_1(y) = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \end{bmatrix}_{N\times N}, \quad b_1(y) = \begin{bmatrix} -1 - y_1 \\ 0 \\ \dots \\ 0 \end{bmatrix}_{N\times 1},$$

$$A_2(y) = \begin{bmatrix} y_1 & 0 & 0 & 0 & \dots & 0 \\ y_2 & y_1 & 0 & 0 & \dots & 0 \\ 0 & y_2 & y_1 & 0 & \dots & 0 \end{bmatrix}_{3\times N}, \quad b_2(y) = \begin{bmatrix} 1 - y_1 + y_1^2 \\ 1 + y_1 - y_2 + y_1 y_2 \\ 2 - y_1 + y_2 - y_2^2 \end{bmatrix}.$$

Here, we use $y = (y_1, y_2)$, thus $n = 2$, and $\ell = 3$.

We choose $n = 23$. The least squares solution of (23) is $(y^*)^T = [0, 0]$. Beginning with $[y^{(0)}]^T = [0.1, 0.1]$ and using Algorithm 1, we obtain the results in Table 2 for $y^{(m)}, m = 1, 2, 3, 4$. These results show the expected quadratic convergence. After $y^{(4)}$ is obtained, we compute $z^{(4)} \approx z^*$ using (21). Here, $z^* \in \mathbf{R}^{23}$, $z_j^* = (-1)^{j+1}$, $j = 1, \ldots, 23$, and the exact residual is $A(y^*)z^* + b(y^*) = e_{24} + e_{25} + 2e_{26}$. We find that $\|z^{(4)} - z^*\| = 5.0974 \times 10^{-15}$, while 2-norm of the residual of $A(y^{(4)})z^{(4)} + b(y^{(4)})$ is $2.4494897427831779 \approx \sqrt{6}$.

Table 2. Gauss–Newton type iterations for Example 2.

m	$y_1^{(m)} - y_1^*$	$y_2^{(m)} - y_2^*$	$\|\|y^{(m)} - y^*\|\|$
0	1.0000e−001	1.0000e−001	1.4142e−001
1	−3.2975e−002	−1.7299e−002	3.7238e−002
2	8.5227e−004	3.5533e−004	9.2338e−004
3	−1.5227e−008	−6.1721e−009	1.6890e−008
4	−4.3532e−016	−1.8608e−016	4.7342e−016

Our last example arises from a discretization, using the finite element method, of the one-dimensional elliptic interface problem $-(\beta u_x)_x + 2 = 0$, for $x \in (0, 1)$, where $\beta = \beta^-$ for $0 < x < \alpha$ and $\beta = \beta^+$ for $\alpha < x < 1$. The boundary conditions are $u(0) = 0$ and $u(1) = 1/\beta^+ + (1/\beta^- - 1/\beta^+)\alpha^2$, and the interface conditions are $lim_{x\to\alpha^-} u(x) = lim_{x\to\alpha^+} u(x)$ and $lim_{x\to\alpha^-} \beta^- u_x(x) = lim_{x\to\alpha^+} \beta^+ u_x(x)$. For more on interface problems, see [18]. In this discretization, the slopes of the basis tent functions on either side of the interface $x = \alpha$ are modified from $-1/h$ and $1/h$ to $-s_1$, $-s_2$ and s_1, s_2 on the respective line segments. In our setting, β^- and s_2 are given, and $(y_1, y_2) = (s_1, \beta^+)^T$ represent the nonlinear variables.

Example 3. *Consider the least squares solution of the overdetermined system*

$$A(y)z + b(y) = 0, \tag{24}$$

where the $N \times (N + 3)$ *matrix* $A(y)$ *is defined as*

$$A(y) \equiv \begin{bmatrix} \frac{\beta^-}{h} F_{k-1} & -\frac{\beta^-}{h} e_{k-1} & 0_{(k-1)\times 1} & 0_{(k-1)\times(N-k-1)} \\ -\frac{\beta^-}{h} e_{k-1}^T & \frac{\beta^-}{h} + \beta^- y_1 & -\beta^- y_1 & 0_{1\times(N-k-1)} \\ 0_{1\times(k-1)} & -s_2 y_2 & \frac{y_2}{h} + s_2 y_2 & -\frac{y_2}{h} e_1^T \\ 0_{(N-k-2)\times(k-1)} & 0_{(N-k-2)\times 1} & -\frac{y_2}{h} e_1 & \frac{y_2}{h} F_{N-k-1} \\ 0_{1\times(k-1)} & a_{N+1,k}(y_1) & 0 & 0_{1\times(N-k-1)} \\ 0_{1\times(k-1)} & 0 & \beta^- y_1 - s_2 y_2 & 0_{1\times(N-k-1)} \\ 0_{1\times(k-1)} & 0 & 0 & 0_{1\times(N-k-1)} \end{bmatrix},$$

with $a_{N+1,k}(y_1) = (\alpha - x_k)y_1 + (x_{k+1} - \alpha)s_2 - 1$ *and*

$$b(y) \equiv \begin{bmatrix} 2hE_{(k-1)\times 1} \\ 2h + (\alpha - x_k) - y_1(\alpha - x_k)^2 - (x_{k+1} - \alpha) + s_2(x_{k+1} - \alpha)^2 \\ 2h - (\alpha - x_k) + y_1(\alpha - x_k)^2 + (x_{k+1} - \alpha) - s_2(x_{k+1} - \alpha)^2 \\ 2hE_{(N-k-1)\times 1} - \left[(1 - \alpha^2)/h + (\alpha^2/h)y_2/\beta^-\right] e_{N-k-1} \\ 0 \\ 0 \\ (a_{N+1,k}(y))^2 + (\beta_1 y_1 - s_2 y_2)^2 + {\gamma_3}^2 \end{bmatrix},$$

where $h = 1/(N+1)$ *and* $x_j = jh$, $x_k < \alpha < x_{k+1}$, *and the square matrices F are tridiagonal of the form*

$$\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & \cdots & & \\ & -1 & 2 & -1 \\ & & -1 & 2 \end{bmatrix},$$

the vectors E have all entries one, e_j *are the jth standard unit vectors, and*

$$\gamma_1 = \frac{1-(x_{k+1}-\alpha)s_2}{\alpha - x_k}, \quad \gamma_2 = \frac{[1-(x_{k+1}-\alpha)s_2]\beta^-}{(\alpha - x_k)s_2}. \tag{25}$$

The exact least squares solution is $y^* = (\gamma_1, \gamma_2)^T$.

We use $n = 39$, $k = 26$, $\ell = 3$, $n = 2$, $\beta^- = 1$, $s_2 = 8$, $\alpha = 2/3$ and $\gamma_3 = 4.775\pi$. Using the Algorithm 1 with $y^{(0)} = (56.5, 7.2)^T$, we obtain the results in Table 3 for $y^{(m)}, m = 1, 2, 3, 4$, which shows quadratic convergence of the method. After $y^{(4)}$ is obtained, we obtain $z^{(4)}$ using (21). Here, $z_j^* = (jh)^2/\beta^-, j = 1, 2, ..., k$ and $z_j^* = (jh)^2/\gamma_2 + (1/\beta^- - 1/\gamma_2)\alpha^2, j = k+1, ..., N$, and the exact residual is $A(y^*)z^* + b(y^*) = \gamma_3^2 e_{N+3}$. We find that $\|z^{(4)} - z^*\| = 1.5081 \times 10^{-15}$ and the 2-norm of the residual of $A(y^{(4)})z^{(4)} + b(y^{(4)})$ is $225.03314884758808 \approx (4.775\pi)^2$.

Table 3. Gauss–Newton type iterations for Example 3.

m	$y_1^{(m)} - y_1^*$	$y_2^{(m)} - y_2^*$	$\|\|y^{(m)} - y^*\|\|$
0	5.0000e−001	2.0000e−001	5.3852e−001
1	1.4160e+000	1.7845e−001	1.4272e+000
2	2.0255e−002	2.5279e−003	2.0412e−002
3	−1.3048e−006	−1.6292e−007	1.3149e−006
4	1.4921e−013	1.8652e−014	1.5038e−013

5. Conclusions

We have shown how LU factorization can be used to solve separable nonlinear least squares problems with nonzero residuals. Our Gauss-Newton type method retains the second derivative terms and is quadratically convergent. The technique is most efficient for situations in which there are few nonlinear terms and variables. In that context we show how the second derivative terms can be computed relatively cheaply. Our numerical examples demonstrate the effectiveness of the technique in some areas of application.

Acknowledgments: Publication costs were provided by Western Washington University, Bellingham, WA 98225, USA.

Author Contributions: Yunqiu Shen conceived the key ideas and drafted the paper; Tjalling Ypma refined the details and the presentation.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Golub, G.H.; Pereyra, V. Separable nonlinear least squares: the variable projection method and its applications. Topic Review. *Inverse Probl.* **2003**, *19*, R1–R26.
2. Mullen, K.M.; van Stokkum, I.M. The variable projection algorithm in time-resolved spectroscopy, microscopy and mass spectrometry applications. *Numer. Algorithms* **2009**, *51*, 319–340.
3. Shen, Y.-Q.; Ypma, T.J. Solving separable nonlinear equations using LU factorization. *ISRN Math. Anal.* **2013**, *2013*, 258072.

4. Golub, G.H.; Pereyra, V. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.* **1973**, *10*, 413–432.
5. Shen, Y.-Q.; Ypma, T.J. Solving nonlinear systems of equations with only one nonlinear variable. *J. Comput. Appl. Math.* **1990**, *30*, 235–246.
6. Ypma, T.J.; Shen, Y.-Q. Solving N+m nonlinear equations with only m nonlinear variables. *Computing* **1990**, *44*, 259–271.
7. Lukeman, G.G. *Separable Overdetermined Nonlinear Systems: An Application of the Shen-Ypma Algorithm*; VDM Verlag: Saarbrucken, Germany, 2009.
8. Shen, Y.; Ypma, T.J. Solving Separable Least Squares Problems using QR factorization. *J. Comp. Appl. Math.* submitted.
9. Dennis, J.E., Jr.; Schnabel, R.B. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Corrected Reprint of the 1983 Original*; SIAM: Philadelphia, PA, USA, 1996.
10. Deuflhard, P.; Hohmann, A. *Numerical Analysis in Modern Scientific Computing*, 2nd ed.; Springer: New York, NY, USA, 2003.
11. Ortega J.M.; Rheinboldt, W.C. *Iterative Solution of Nonlinear Equations in Several Variables*; Academic Press: New York, NY, USA, 1970.
12. Golub, G.H.; Van Loan, C.F. *Matrix Computations*, 3rd ed.; Johns Hopkins: Baltimore, MD, USA, 1996.
13. Shen, Y.-Q.; Ypma, T.J. Newton's method for singular nonlinear equations using approximate left and right nullspaces of the Jacobian. *Appl. Numer. Math.* **2005**, *54*, 256–265.
14. Shen, Y.-Q.; Ypma, T.J. Solving rank-deficient separable nonlinear equations. *Appl. Numer. Math.* **2007**, *57*, 609–615.
15. Shen, Y.-Q.; Ypma, T.J. Numerical bifurcation of separable parameterized equations. *Elect. Trans. Numer. Anal.* **2009**, *34*, 31–43.
16. Shen, Y.-Q.; Ypma, T.J. Rank deficiency and bifurcation into affine subspaces for separable parametrized equations. *Math. Comp.* **2016**, *85*, 271–293.
17. Smith, G.D. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, 3rd ed.; Oxford University Press: New York, NY, USA, 1985.
18. Li, Z.; Ito, K. *The Immersed Interface Method*; SIAM: Philadelphia, PA, USA, 2006.