SIAM J. NUMER. ANAL.
Vol. 49, No. 4, pp. 1417–1435

HESSIAN MATRIX VS. GAUSS–NEWTON HESSIAN MATRIX*

PEI CHEN†

Abstract. In this paper, we investigate how the Gauss–Newton Hessian matrix affects the basin of convergence in Newton-type methods. Although the Newton algorithm is theoretically superior to the Gauss–Newton algorithm and the Levenberg–Marquardt (LM) method as far as their asymptotic convergence rate is concerned, the LM method is often preferred in nonlinear least squares problems in practice. This paper presents a theoretical analysis of the advantage of the Gauss–Newton Hessian matrix. It is proved that the Gauss–Newton approximation function is the only nonnegative convex *quadratic approximation* that retains a critical property of the original objective function: taking the minimal value of zero on an $(n-1)$-dimensional manifold (or affine subspace). Due to this property, the Gauss–Newton approximation does not change the **zero-on-$(n-1)$-D** "structure" of the original problem, explaining the reason why the Gauss–Newton Hessian matrix is preferred for nonlinear least squares problems, especially when the initial point is far from the solution.

Key words. least squares problem, Hessian matrix, Gauss–Newton Hessian matrix

AMS subject classifications. 65Y10, 65Y20

DOI. 10.1137/100799988

1. Introduction.

1.1. Nonlinear least squares problem and parameter estimation.

Most parameter estimation problems can be reduced to solving a nonlinear least squares problem, specifically, minimizing the following objective function:

$$f(\theta) = \sum f_i(\theta) = \sum r^2(\mathbf{x}_i;\theta), \tag{1.1}$$

where data points $\{\mathbf{x}_i\}$ are to be fitted by the model $r(\mathbf{x};\theta)$. Usually, each term $f_i(\theta)$ in (1.1) is associated with the constraint of $r(\mathbf{x}_i;\theta)=0$, denoting the penalty for the discrepancy from the model. Supposing $\theta \in R^n$, the nonlinear least squares problem in (1.1) has the following structure:

(a) When there are less than n constraints, the parameter estimation problem is *underdetermined*, precisely, there are infinite solutions where the objective function (1.1) takes the minimal value of zero.
(b) When there are n independent constraints, there exists a unique solution where the objective function (1.1) takes the minimal value of zero.
(c) When there are at least n independent constraints, the parameters can be estimated by minimizing the objective function (1.1).

The assertions (a)–(c) above are consequences of the property below:

- For a single constraint associated with $f_i(\theta)$, $f_i(\theta)=0$ holds on an $(n-1)$-dimensional manifold.

The property above is intrinsic with most parameter estimation problems. Formally, we define such a property as follows.

DEFINITION 1.1. *Given a function $\mathbb{f}(\theta) \geq 0$, with $\theta \in R^n$, we call it* **zero-on-$(n-1)$-D** *if $\mathbb{f}(\theta)=0$ holds on an $(n-1)$-dimensional manifold.*

*Received by the editors June 23, 2010; accepted for publication (in revised form) May 9, 2011; published electronically July 19, 2011. This work was supported by the Ministry of Science and Technology, People's Republic of China, under the 973 Program 2006CB303104.
http://www.siam.org/journals/sinum/49-4/79998.html
†School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China (chenpei@mail.sysu.edu.cn).

Iterate
1. Compute the gradient vector $\mathbf{b}$ and the Hessian matrix $\mathbf{H}$;
2. Compute the increment $\widehat{\delta\theta}$ by (1.3);
3. Update the parameter $\theta \leftarrow \theta + \widehat{\delta\theta}$;

Until convergence.

FIG. 1.1. *The Newton algorithm or the Gauss–Newton algorithm. When the full Hessian matrix* (1.4) *is used in step* 2, *it is the Newton algorithm; on the other hand, it is called the Gauss–Newton algorithm when the Gauss–Newton Hessian matrix* (1.5) *is used in step* 2.

In this paper, we are interested in a class of minimization problems (1.1), where each term in the summation is **zero-on-**$(n-1)$**-D**. In many parameter estimation problems, such as ellipse fitting and fundamental matrix estimation [23, 22, 20, 19, 24], $f_i(\theta)$ in (1.1) is associated with a linearized constraint of $\theta^T\mathbf{w}(\mathbf{x}_i) = 0$, where $\mathbf{w}(\mathbf{x})$ is a nonlinear mapping of the elements of $\mathbf{x}$. For example, the nonlinear mapping in ellipse fitting is $[\ x^2 \quad y^2 \quad xy \quad x \quad y \quad 1\ ]$ for the point $[\ x \quad y\ ]$. In these problems, each term of the objective function in (1.1) is **zero-on-**$(n-1)$**-D**.[1]

1.2. Newton-type algorithms. Mature algorithms are available for the optimization problem (1.1); see [14, 15, 28, 29, 3, 17]. Reinvestigated in this paper are the Newton-type algorithms, such as the Levenberg–Marquardt (LM) method [31].

The underlying theory of the Newton-type algorithms is to approximate the objective function f around θ_0 by a *quadratic* function. Specifically, the Newton algorithm uses the second-order Taylor expansion

$$f(\theta_0 + \delta\theta) \approx f(\theta_0) + \mathbf{b}^T\delta\theta + \frac{1}{2}\delta\theta^T\mathbf{H}\delta\theta, \tag{1.2}$$

where the gradient vector $\mathbf{b} = \frac{\partial f}{\partial\theta}|_{\theta_0}$ and the Hessian matrix $\mathbf{H} = \frac{\partial^2 f}{\partial\theta^2}|_{\theta_0}$. The second-order Taylor expansion in (1.2) is called the Newton approximation of function f, as it is related to the Newton algorithm. Similarly, when $\mathbf{H}$ is replaced by the Gauss–Newton Hessian matrix $\mathbf{H}_{GN}$ in (1.5), the quadratic approximation in (1.2) is called the Gauss–Newton approximation.

With an initial solution of θ_0, the Newton algorithm iteratively estimates the parameter θ by minimizing the approximated quadratic function (1.2). The new estimate (precisely, the increment $\widehat{\delta\theta}$) is computed by the requirement that the gradient of the approximated quadratic function at the minimum be zero:

$$\widehat{\delta\theta} = -\mathbf{H}^{-1}\mathbf{b}. \tag{1.3}$$

The Newton algorithm is shown in Figure 1.1.

There are many variants of the Newton algorithm, for instance, the Newton algorithm with a trust region step control [14, 15, 28, 29]. In this paper, we are particularly interested in the choice of $\mathbf{H}$ in (1.3). A comprehensive review about the Hessian matrix $\mathbf{H}$ and its approximation can be found in [29, 1]. Here, we give an overview of its Gauss–Newton approximation and the LM algorithm.

[1] In these problems, we do not minimize $\sum||\theta^T\mathbf{w}(\mathbf{x}_i)||^2$ in order to estimate the maximum-likelihood solution. Instead, other objective functions are to be minimized, such as the Sampson error [32, 20, 36]. Nevertheless, each term of the Sampson error is still **zero-on-**$(n-1)$**-D**.

$\lambda \leftarrow 10^{-3}$;
Repeat
1. Compute the gradient vector $\mathbf{b}$ and the Hessian matrix $\mathbf{H}$;
2. Compute the increment $\widehat{\delta\theta} = -(\mathbf{H} + \lambda I)^{-1}\mathbf{b}$;
3. If $f(\theta + \widehat{\delta\theta}) < f(\theta)$
   - $\lambda \leftarrow \frac{\lambda}{10}$;

   Else
   - Repeat
     - $\lambda \leftarrow \lambda \times 10$;
     - Compute the increment $\widehat{\delta\theta} = -(\mathbf{H} + \lambda I)^{-1}\mathbf{b}$;

     Until $f(\theta + \widehat{\delta\theta}) < f(\theta)$;

   Endif
4. $\theta \leftarrow \theta + \widehat{\delta\theta}$;

Until convergence.

FIG. 1.2. *The LM algorithm.*

The full Hessian matrix of f in (1.1) is[2,3]

$$\mathbf{H}_N = 2\sum \frac{\partial r_i}{\partial \theta}\left(\frac{\partial r_i}{\partial \theta}\right)^T + r_i \frac{\partial^2 r_i}{\partial \theta^2}. \tag{1.4}$$

The Gauss–Newton approximation of the Hessian matrix[4] is obtained by omitting the second-order derivative in (1.4):

$$\mathbf{H}_{GN} = \sum \mathbf{H}_{i,GN} = 2\sum \frac{\partial r_i}{\partial \theta}\left(\frac{\partial r_i}{\partial \theta}\right)^T. \tag{1.5}$$

Since Newton-type methods may take step sizes which are too long, the LM algorithm can be used as a trust region method. In the LM method for Gauss–Newton,[5] $\mathbf{H}$ takes an augmented form of $\mathbf{H}_{GN}$ (or $\mathbf{H}_N$, in some situations):

$$\mathbf{H}_{LM} = \mathbf{H}_{GN} + \lambda \mathbf{I} \tag{1.6}$$

with $\lambda \geq 0$. When λ is very small (precisely, much smaller than the least nonzero singular value of $\mathbf{H}_{GN}$), the estimated increment is almost the same as that of the Gauss–Newton method. On the other hand, when λ is large so that $\lambda\mathbf{I}$ dominates $\mathbf{H}_{GN}$, the method reduces to the gradient descent method [31, 20, 1]. The value of λ varies at each iteration, depending on whether the objective function decreases. See a detailed implementation of the LM algorithm in [31, 20]. The LM algorithm is included for completeness in Figure 1.2.

Theoretically, the Newton algorithm has a locally quadratic convergence rate [14, 29]. On the other hand, the convergence speed of the Gauss–Newton algorithm

[2]Here, $\frac{\partial r_i}{\partial \theta}$ denotes $\frac{\partial r(\mathbf{x}_i;\theta)}{\partial \theta}$ for concise representation. This simplification also applies to r_i and $\frac{\partial^2 r_i}{\partial \theta^2}$.

[3]In the following, $\mathbf{H}_N$ denotes the full Hessian matrix.

[4]The Gauss–Newton approximation of the Hessian matrix is called the Gauss–Newton Hessian matrix. When the Gauss–Newton Hessian matrix, instead of the full Hessian matrix, is used in (1.3), the algorithm is called the Gauss–Newton algorithm.

[5]There is another type of the augmented $\mathbf{H}$ used in the LM algorithm. See the details in [31].

depends on the nonlinearity and the residual magnitude of the problem [14]. For a linear model $r(\mathbf{x};\theta)$ or small residual problems (with small residuals at the solution), the Gauss–Newton algorithm has a rapid convergence speed. However, the Gauss–Newton algorithm tends to perform poorly in highly nonlinear problems or large-residual problems (with large residuals at the solution), with only linear convergence rate [14, 15, 29]. See a detailed comparison between the Newton algorithm and the Gauss–Newton algorithm in Table 10.2.3 in [14].

1.3. Confusing facts. Local convergence rate is an indicator of the convergence speed after the iteration has fallen in a small neighborhood around the solution, where the objective function can be well approximated by a quadratic function. There are other important aspects of numerical algorithms for engineering problems such as the basin of convergence. This is especially important when an initial point that is close to the solution is not available. However, there are confusing facts related to the basin of convergence in the literature.

1.3.1. Newton method vs. Gauss–Newton method. Seemingly, the methods using the full Hessian matrix in (1.4) are expected to have a better performance than those with the Gauss–Newton Hessian matrix in (1.5), because the former is a more sophisticated Hessian matrix that includes the second-order derivative $\frac{\partial^2 r_i}{\partial \theta^2}$. However, it was pointed out that the Gauss–Newton Hessian matrix is preferred to the full Hessian matrix in applications, especially when the data is highly corrupted by noise[6] [1, 27].

In addition, the difficulty of the Newton method was ascribed to the noise in data that is to be fitted. "Inclusion of the second-derivative term can in fact be destabilizing if the model fits badly ..." [31]. "The Newton Hessian also depends on the second derivatives of the template. It appears that the increased noise in estimating the second derivatives of the template outweighs the increased sophistication in the algorithm."[1]

However, the explanations mentioned just above are not convincing. From the experiments in [1], the noise has a similar effect on both the Newton method and the Gauss–Newton method. Moreover, it is not clear whether the superiority of the Gauss–Newton method is restricted only to the applications where such a claim has been demonstrated by experiments. This ambiguity prevents its free extension to other potential applications.

1.3.2. Large-residual problem. As noted above, the Gauss–Newton algorithm and the LM algorithm based on the Gauss–Newton Hessian matrix perform poorer in large-residual problems [14, 15, 29]. For example, the Gauss–Newton algorithm performs poorer on the Brown function[7] and on the trigonometric function; see Table 6.1.3 in [15]. The underlying reason for this phenomenon is that the second-order term in (1.4) is too significant to be ignored when with large residuals r_i.

However, a confusing fact about large-residual problems was observed [29]: "the behavior of both Newton and quasi Newton on early iterations (before the iterations reach a neighborhood of the solution) may be inferior to Gauss–Newton and

[6] The computation of the Gauss–Newton matrix is usually much more efficient than that of the full Hessian matrix. However, in this paper, we are concerned with the frequency of convergence of the algorithms. Thus, when referring to better performance of the Gauss–Newton methods, we mean a higher frequency of convergence for these techniques.

[7] Actually, the Brown function is not **zero-on-**$(n-1)$**-D**. The Brown function has 4 parameters to be estimated, while each term in the Brown function takes the minimum of zero on a 2-D affine subspace, not on a 3-D affine subspace.

Levenberg-Marquardt, ..." Actually, the residuals in early iterations have a larger magnitude than those near the solution. Therefore, we should have observed the opposite: The Newton algorithm and the quasi-Newton algorithm should perform better than the Gauss–Newton, especially on early iterations.

1.4. Main result. The difference between the Newton algorithm and the Gauss–Newton algorithm lies in using the Hessian matrix or the Gauss–Newton Hessian matrix in (1.2). Other modifications of the Hessian matrix have also been proposed to overcome the difficulties with the full Hessian matrix; see [29]. These facts hint at other possibilities of approximating the function f. Formally, an $\mathbf{H}$ *quadratic approximation* of the function f is defined as follows.

DEFINITION 1.2. *Suppose* $\mathbb{f}(\mathbf{x})$ *is a second-order continuous function with* $\mathbb{f}(\mathbf{x}_0) = \mathbb{f}_0$ *and* $\frac{\partial \mathbb{f}}{\partial \mathbf{x}}|_{\mathbf{x}_0} = \mathbb{b}$. *An* $\mathbf{H}$ quadratic approximation *of the function* $\mathbb{f}$, *around* $\mathbf{x}_0$, *is defined as*

$$\mathfrak{f}_{\mathbf{H}}(\mathbf{x}_0 + \delta\mathbf{x}) = \mathbb{f}_0 + \delta\mathbf{x}^T\mathbb{b} + \frac{1}{2}\delta\mathbf{x}^T\mathbf{H}\delta\mathbf{x}, \tag{1.7}$$

where $\mathbf{H}$ *is symmetric.*

Note that the Newton approximation and the Gauss–Newton approximation are special quadratic approximations, where $\mathbf{H}$ takes the Hessian matrix or the Gauss–Newton Hessian matrix, respectively. Particularly, for a term $f_i(\theta)$ in (1.1), its Gauss–Newton approximation is

$$\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta_0 + \delta\theta) = f_i + \mathbf{b}_i^T\delta\theta + \frac{1}{2}\delta\theta^T\mathbf{H}_{i,GN}\delta\theta, \tag{1.8}$$

where $f_i = f_i(\theta_0)$, $\mathbf{b}_i = \frac{\partial f_i}{\partial \theta}|_{\theta_0}$, and $\mathbf{H}_{i,GN}$ is defined in (1.5).

With Definitions 1.1 and 1.2, we state the following theorem as the main result in this paper.

THEOREM 1.3. *For the function* $f_i(\theta) \geq 0$ *in the summation in* (1.1), *the Gauss–Newton approximation* (1.8) *around* θ_0 *is the only nonnegative convex* quadratic approximation *that is* **zero-on-**$(n-1)$**-D**.

Theorem 1.3 states that only the Gauss–Newton approximation has the following properties:

- $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) \geq 0$.
- $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta)$ is **zero-on-**$(n-1)$**-D**.
- $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta)$ is convex.

For most parameter estimation problems, the first two properties above are intrinsic with the function $f_i(\theta)$ in (1.1). That is, the Gauss–Newton approximation does not change the "structure" of the objective function (1.1). Consequently,

- at least n independent constraints $\{\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) = 0\}$ are needed to estimate the increment in the Gauss–Newton method.

The convexity property above guarantees that the Gauss–Newton Hessian matrix is positive semidefinite, and, consequently, the Gauss–Newton algorithm finds its downhill way in the iteration; i.e., there exists a neighborhood along the direction of $\widehat{\delta\theta}_{GN}$ so that the objective function takes a value of less than the current objective function.

1.5. Organization. The rest of the paper is organized as follows. Presented in section 2 is a theoretical analysis of the Gauss–Newton Hessian matrix, proving Theorem 1.3. Based on the analysis in section 2, the Gauss–Newton method is extended to the missing-data problem in a low-rank matrix in section 3. Experimental results about the missing-data problem are given in section 4.

2. Analysis. In this section, we first present the Gauss–Newton technique for a general form of objective function. Then, we analyze the properties of the Gauss–Newton Hessian matrix, proving Theorem 1.3.

2.1. A general technique. In some applications, the objective function to be minimized does not explicitly take the form of (1.1). Nevertheless, the objective function is usually nonnegative as it measures the discrepancy of the data points from the model with its minimal value of zero when the data points fit the model exactly. Thus, a more general form of the objective function is used:

$$f(\theta) = \sum f_i(\theta), \tag{2.1}$$

where $f_i(\theta) \geq 0$.

For each $f_i(\theta)$, define a function $g_i(\theta)$ as its square root:

$$g_i(\theta) = \sqrt{f_i(\theta)} \tag{2.2}$$

with its first-order derivative as $\frac{\partial g_i}{\partial \theta} = \frac{\frac{\partial f_i}{\partial \theta}}{2\sqrt{f_i}}$.[8] Then, around θ_0, $f_i(\theta_0 + \delta\theta)$ is approximated as

$$\begin{aligned} f_i(\theta_0 + \delta\theta) &= g_i^2(\theta_0 + \delta\theta) \\ &\approx (g_i + \delta\theta^T \frac{\partial g_i}{\partial \theta})^2 \\ &= f_i + \delta\theta^T \frac{\partial f_i}{\partial \theta} + \frac{1}{4f_i} \delta\theta^T \frac{\partial f_i}{\partial \theta} \left(\frac{\partial f_i}{\partial \theta}\right)^T \delta\theta \end{aligned} \tag{2.3}$$

with its gradient vector and Gauss–Newton Hessian matrix of f_i as

$$\begin{aligned} \mathbf{b}_i &= \frac{\partial f_i}{\partial \theta}, \\ \mathbf{H}_{i,GN} &= \frac{1}{2f_i} \frac{\partial f_i}{\partial \theta} \left(\frac{\partial f_i}{\partial \theta}\right)^T. \end{aligned} \tag{2.4}$$

Similarly, the function f can be approximated as

$$f(\theta + \delta\theta) \approx \sum f_i(\theta) + \delta\theta^T \frac{\partial f_i}{\partial \theta} + \frac{1}{4f_i} \delta\theta^T \frac{\partial f_i}{\partial \theta} \left(\frac{\partial f_i}{\partial \theta}\right)^T \delta\theta. \tag{2.5}$$

The gradient vector and the Gauss–Newton Hessian matrix of f are

$$\begin{aligned} \mathbf{b} &= \sum \frac{\partial f_i}{\partial \theta}, \\ \mathbf{H}_{GN} &= \sum \frac{1}{2f_i} \frac{\partial f_i}{\partial \theta} \left(\frac{\partial f_i}{\partial \theta}\right)^T. \end{aligned} \tag{2.6}$$

It should be pointed out that the Gauss–Newton Hessian matrix $\mathbf{H}_{GN}$ just above is same as that in (1.5), as can be validated by substitution.

[8] For the purpose of concise symbols, f_i is used to denote the function $f_i(\theta)$ here. Otherwise, f_i is also used to denote the value of $f_i(\theta_0)$, for example, in (2.3). Similarly, $\frac{\partial f_i}{\partial \theta}$ denotes the gradient function $\frac{\partial f_i}{\partial \theta}$ or the vector of $\frac{\partial f_i}{\partial \theta}|_{\theta_0}$. This simplification also applies to other functions such as g_i. When no ambiguity is introduced by the symbols such as f_i, no explanation is added below.

2.2. Properties of the Gauss–Newton matrix. In this section, we prove Theorem 1.3.

2.2.1. Positive semidefiniteness of $\mathbf{H}_{GN}$. The following theorem states a nice property of the Gauss–Newton Hessian matrix.

THEOREM 2.1. *The Gauss–Newton Hessian matrix in* (2.6) *is positive semidefinite, i.e.,* $\mathbf{H}_{GN} \succeq 0$.

This positive semidefiniteness property is well known; see [14, 15, 28, 29]. It is included here for completeness.

From (1.3), the estimated Gauss–Newton increment $\widehat{\delta\theta}_{GN}$ is $\widehat{\delta\theta}_{GN} = -\mathbf{H}_{GN}^{-1}\mathbf{b}$, and

$$\mathbf{b}^T\widehat{\delta\theta}_{GN} = -\mathbf{b}^T\mathbf{H}_{GN}^{-1}\mathbf{b} \le 0. \tag{2.7}$$

Property (2.7) and Theorem 2.1 state that the Gauss–Newton algorithm finds its downhill way[9] in the iteration; i.e., there exists a neighborhood along the direction of $\widehat{\delta\theta}_{GN}$ so that the objective function takes a value of less than the current objective function.

2.2.2. Nonnegativeness of the approximated function. It is usually assumed that $f_i(\theta) \ge 0$. However, chances are that this nonnegativeness condition is violated in the Newton approximation. Such a violation definitely occurs when one or more eigenvalues of the Hessian matrix are negative. Otherwise, the nonnegativeness of the approximated function of f_i cannot be guaranteed even if the Hessian matrix is positive definite.

Consider a 2D example:

$$f(x, y) = \log(1 + x^2 + 9y^2) + \log(1 + 9x^2 + y^2). \tag{2.8}$$

Select the point of $[0.3, 0.1]^T$*, where the neighborhood is approximated by the Newton approximation as*

$$0.7644 + \delta\mathbf{x}^T \begin{bmatrix} 3.4755 \\ 1.6353 \end{bmatrix} + \frac{1}{2}\delta\mathbf{x}^T \begin{bmatrix} 2.5232 & -1.1017 \\ -1.1017 & 1.2616 \end{bmatrix} \delta\mathbf{x}.$$

The Hessian matrix above has two eigenvalues of 3.1619 *and* 0.6229*, and the approximated function above takes its minimum of* -7.9965 *at* $\widehat{\delta\mathbf{x}_N} = \begin{bmatrix} -3.1410 \\ -4.0391 \end{bmatrix}$.

From the example above, it can be seen that the nonnegativeness of the function f_i may be violated in its Newton approximation. What happens to the Gauss–Newton approximation? Before formally proceeding to the Gauss–Newton approximation, we first give some definitions and cite the Schur complement.

DEFINITION 2.2. *Give a matrix* $\mathbf{M}$ *with blocks as* $\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix}$*, where* $\mathbf{A}$ *and* $\mathbf{C}$ *are symmetric. If* $\det(\mathbf{A}) \ne 0$*, the matrix*

$$\mathbf{S} = \mathbf{C} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B} \tag{2.9}$$

is called the Schur complement [5] *of* $\mathbf{A}$ *in* $\mathbf{M}$.

[9] In the Gauss–Newton method, it does not necessarily hold that $f(\theta_0 + \widehat{\delta\theta}_{GN}) < f(\theta_0)$. However, $\exists\, c > 0\ \forall\, 0 < \lambda < 1$, it holds that $f(\theta_0 + \lambda c\widehat{\delta\theta}_{GN}) < f(\theta_0)$; i.e., along the estimated direction, there exists a neighborhood where the function f takes a value of less than $f(\theta_0)$.

Below is a theorem related to the Schur complement [5].

LEMMA 2.3. *If* $\mathbf{A} \succ 0$, *then* $\mathbf{M} \succeq 0$ *if and only if* $\mathbf{S} \succeq 0$.

See the proof of Lemma 2.3 in [5].

THEOREM 2.4. *For a function* $f(\mathbf{x})$ *with* $f(\mathbf{x}_0) = f_0 > 0$ *and* $\frac{\partial f}{\partial \mathbf{x}}|_{\mathbf{x}_0} = \mathbf{b}$, *its* $\mathbf{H}$ quadratic approximation *around* $\mathbf{x}_0$ *is nonnegative if and only if*

$$\mathbf{H} \succeq \frac{\mathbf{b}\mathbf{b}^T}{2f_0}. \tag{2.10}$$

Proof. In a matrix form, the $\mathbf{H}$ *quadratic approximation* $\mathfrak{f}_{\mathbf{H}}(\mathbf{x})$ is represented as

$$\mathfrak{f}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{2}\tilde{\mathbf{x}}^T \begin{bmatrix} 2f_0 & \mathbf{b}^T \\ \mathbf{b} & \mathbf{H} \end{bmatrix} \tilde{\mathbf{x}} \tag{2.11}$$

with the extended $\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}$.

By Lemma 2.3, $\begin{bmatrix} 2f_0 & \mathbf{b}^T \\ \mathbf{b} & \mathbf{H} \end{bmatrix} \succeq 0$ if and only if $\mathbf{H} \succeq \frac{\mathbf{b}\mathbf{b}^T}{2f_0}$. ∎

Theorem 2.4 states that the Gauss–Newton approximation retains the nonnegativeness property of function f_i.

2.2.3. Uniqueness of the Gauss–Newton approximation. In order to prove Theorem 1.3, we need the following lemma.

LEMMA 2.5. *For an* $n \times n$ *symmetric positive semidefinite matrix* $\mathbf{M}$, *if* $\mathbf{x}^T\mathbf{M}\mathbf{x} = 0$ *holds on an* $(n-1)$*-dimensional affine subspace that does not contain the origin, then* $\mathbf{M} = \mathbf{0}$.

Proof. Suppose the $(n-1)$-dimensional affine subspace is represented as $\mathbb{A} = \{\mathbf{x}_0 + \sum_{i=1}^{n-1} k_i\mathbf{x}_i\}$, where $\{\mathbf{x}_i | 1 \le i \le n-1\}$ is a set of linearly independent vectors and $\mathbf{x}_0 \notin span(\{\mathbf{x}_i | 1 \le i \le n-1\})$.[10] $\{\mathbf{x}_i | 0 \le i \le n-1\}$ is a set of basis for the whole n-D space. Without loss of generality, suppose that $\mathbf{x}_0 \perp \mathbf{x}_i$ for $1 \le i \le n-1$.[11]

Assumption. Assume $\mathbf{M}$ has a positive eigenvalue $\lambda > 0$ and the associated unit eigenvector is $\mathbf{u}$.

First, we prove $\mathbf{u} \in span(\{\mathbf{x}_i | 1 \le i \le n-1\})$. Because $\{\mathbf{x}_i | 0 \le i \le n-1\}$ is a set of basis for the whole n-D space, $\mathbf{u} = \sum_{i=0}^{n-1} c_i\mathbf{x}_i$ holds for some $\{c_i\}$. If $c_0 \ne 0$, define $\mathbf{u}_0 \triangleq \frac{\mathbf{u}}{c_0} = \mathbf{x}_0 + \sum_{i=1}^{n-1} \frac{c_i}{c_0}\mathbf{x}_i \in \mathbb{A}$. From $\mathbf{u}_0 \in \mathbb{A}$,

$$\mathbf{u}_0^T\mathbf{M}\mathbf{u}_0 = 0. \tag{2.12}$$

On the other hand, from the assumption that $\mathbf{u}$ is the eigenvector associated with the positive eigenvalue λ,

$$\mathbf{u}_0^T\mathbf{M}\mathbf{u}_0 = \frac{\lambda}{c_0^2} > 0. \tag{2.13}$$

Equations (2.12) and (2.13) contradict with each other, implying that $c_0 = 0$.

[10] If $\mathbf{x}_0 \in span(\{\mathbf{x}_i | 1 \le i \le n-1\})$, the $(n-1)$-dimensional affine subspace degenerates as an $(n-1)$-dimensional linear subspace, containing the origin.

[11] If not, one can construct a new $\mathbf{x}_0$ by the Gram–Schmidt orthonormalization, satisfying $\mathbf{x}_0 \perp \mathbf{x}_i$ for $1 \le i \le n-1$.

Consequently, $\mathbf{u} \perp \mathbf{x}_0$. Construct a vector $\mathbf{u}' = k\mathbf{u} + \mathbf{x}_0 \in \mathbb{A}$ for $k \neq 0$:

$$\mathbf{u}'^T \mathbf{M} \mathbf{u}' = 0. \tag{2.14}$$

On the other hand, because $\mathbf{u} \perp \mathbf{x}_0$ and $\mathbf{u}$ is an eigenvector of $\mathbf{M}$,

$$\mathbf{u}'^T \mathbf{M} \mathbf{u}' = k^2 \mathbf{u}^T \mathbf{M} \mathbf{u} + \mathbf{x}_0^T \mathbf{M} \mathbf{x}_0 = \lambda k^2 + \mathbf{x}_0^T \mathbf{M} \mathbf{x}_0 \geq \lambda k^2 > 0, \tag{2.15}$$

where the positive semidefiniteness property of $\mathbf{M}$ is used.

Equations (2.14) and (2.15) contradict with each other, implying the Assumption does not hold. That is, $\mathbf{M}$ does not have any positive eigenvalue. Jointly with $\mathbf{M} \succeq 0$, we have $\mathbf{M} = 0$. ∎

Proof of Theorem 1.3. Suppose the $\mathbf{H}$ quadratic approximation $\mathfrak{f}_{\mathbf{H}}(\theta)$ is nonnegative and **zero-on-**$(n-1)$**-D**.

From Theorem 2.4,

$$\mathbf{H} \succeq \mathbf{H}_{i,GN} \tag{2.16}$$

because the $\mathbf{H}$ quadratic approximation is nonnegative. $\mathfrak{f}_{\mathbf{H}}(\theta)$ is decomposed into

$$\begin{aligned} \mathfrak{f}_{\mathbf{H}}(\theta) &= f_i + \theta^T \mathbf{b}_i + \frac{1}{2}\theta^T \mathbf{H} \theta \\ &= f_i + \theta^T \mathbf{b}_i + \frac{1}{2}\theta^T \mathbf{H}_{i,GN} \theta + \frac{1}{2}\theta^T (\mathbf{H} - \mathbf{H}_{i,GN})\theta \\ &= \mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) + \frac{1}{2}\theta^T (\mathbf{H} - \mathbf{H}_{i,GN})\theta. \end{aligned} \tag{2.17}$$

In (2.17),

$$\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) \geq 0 \tag{2.18}$$

because the Gauss–Newton approximation is nonnegative. Thus, combining (2.16) and (2.18), $\mathfrak{f}_{\mathbf{H}}(\theta) = 0$ if and only if $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) = 0$ and $\theta^T(\mathbf{H} - \mathbf{H}_{i,GN})\theta = 0$.

In another form, $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta)$ is

$$\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) = \frac{1}{4f_i}(2f_i + \theta^T \mathbf{b}_i)^2. \tag{2.19}$$

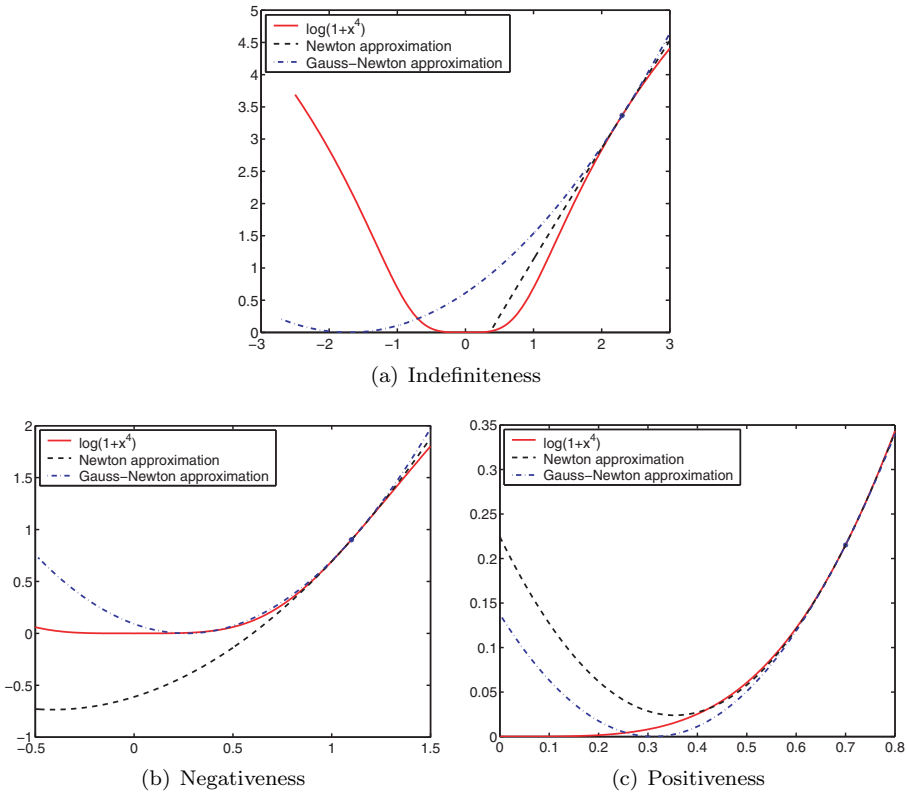
Consequently, $\mathfrak{f}_{\mathbf{H}_{i,GN}}(\theta) = 0$ if and only if $2f_i + \theta^T \mathbf{b}_i = 0$; i.e., $\theta - \frac{2f_i}{||\mathbf{b}_i^2||^2}\mathbf{b}_i \in \mathbf{b}_i^{\perp}$, where $\mathbf{b}_i^{\perp}$ denotes the $(n-1)$-dimensional subspace that is orthogonal to $\mathbf{b}_i$. The equality in (2.18) holds if and only if $\theta \in \frac{2f_i}{||\mathbf{b}_i^2||^2}\mathbf{b}_i + \mathbf{b}_i^{\perp}$.

Thus, from Definition 1.1, the nonnegative $\mathbf{H}$ *quadratic approximation* in (2.17) is **zero-on-**$(n-1)$**-D** if and only if $\theta^T(\mathbf{H} - \mathbf{H}_{i,GN})\theta = 0$ holds on the $(n-1)$-dimensional affine subspace $\frac{2f_i}{||\mathbf{b}_i^2||^2}\mathbf{b}_i + \mathbf{b}_i^{\perp}$.

Furthermore, from Lemma 2.5, the **zero-on-**$(n-1)$**-D** condition reduces to $\mathbf{H} - \mathbf{H}_{i,GN} = \mathbf{0}$, i.e., $\mathbf{H} = \mathbf{H}_{i,GN}$. ∎

2.3. Graphical explanation. In this section, we graphically explain what Theorem 1.3 means. Consider a 1-D function f:

$$f(x) = \log(1 + x^4). \tag{2.20}$$



(a) Indefiniteness

(b) Negativeness

(c) Positiveness

FIG. 2.1. *Explanation of Theorem* 1.3: *Gauss–Newton approximation vs. Newton approximation.*

Figure 2.1(a) shows a typical case where the Hessian matrix is not positive semidefinite. In this 1-D example, $f''(x_0) < 0$, and therefore the Newton approximation is not convex. In Figure 2.1(b), though the Newton approximation is convex, its nonnegativeness is not guaranteed, violating the condition $f(x) \geq 0$. Similarly, the minimum of the Newton approximation in Figure 2.1(c) is greater than zero, not in compliance with the fact that the penalty in the objective function (1.1) is zero when the model fits the data point exactly.

In comparison, the Gauss–Newton approximation complies with all three requirements: convex, nonnegative, and with a minimum of zero. In the n-D cases, the Gauss–Newton approximation satisfies the three requirements, with an additional requirement: it takes the minimum of zero on an $(n-1)$-dimensional affine subspace, in accordance with the fact that each term of the objective function in (1.1) is **zero-on-$(n-1)$-D** in many parameter estimation problems.

3. Application. The Gauss–Newton method has been successfully applied to computer vision tasks, including optical flow [26, 1], tracking [4, 18], parametric and layered motion estimation [2], mosaic construction [34], medical image registration [13], and active appearance models [27]. In this section, one more application is presented: the missing-data problem in a low-rank matrix. However, its objective function cannot be categorized as the form of (1.1) or (2.1), different from the applications mentioned above. Nevertheless, it will be shown how to extend the Gauss–Newton

Rank r approximation of matrix $\mathbf{M}$*, with missing data, is to find* $\mathbf{R}$ *and* $\mathbf{S}$ *such that minimize* $\mathfrak{f}(\mathbf{R}, \mathbf{S})$*:*

$$\mathfrak{f}(\mathbf{R}, \mathbf{S}) = ||(\mathbf{R}\mathbf{S}^T - \mathbf{M}) \odot \mathcal{M}||^2_{Frobenius}, \tag{3.1}$$

where $\mathbf{R} \in R^{m,r}$*,* $\mathbf{S} \in R^{n,r}$*,* $\mathbf{M} \in R^{m,n}$*, the mask matrix* $\mathcal{M} \in R^{m,n}$ *has entries of only* 1 *for observed data or* 0 *for missing data, and* $\odot$ *denotes the componentwise product.*

FIG. 3.1. *The missing-data problem.*

method to the missing-data problem in section 3.2. It should be stressed that this extension is based on the analysis in section 2.

3.1. Missing data in a low-rank matrix. Low-rank matrix approximation has applications in many fields, such as 3-D reconstruction from an image sequence [35, 30, 12, 10] and 2-D filter design [25, 21]. The singular value decomposition (SVD) [16] plays a fundamental role in computing a low-rank matrix approximation. However, the SVD does not work when there is missing data in the matrix. Formally, the missing-data problem is shown in Figure 3.1 [33, 6, 10].

The missing-data problem has its application in many fields, such as structure from motion, collaborative filtering, system identification, global positioning, and remote sensing, as listed in [9]. Recently, much interest in this problem has been roused in the community of compressive sampling (or compressed sensing); see [7, 8, 9, 37]. In [7, 8, 9, 37], the estimation of missing data is cast as a problem of the recovery of sparse signal by the technique of convex programming. Generally, such methods only provide an approximate solution of minimizing (3.1), which can be used as an ideal initial solution for the Newton-type methods. Here, we will concentrate on the Newton-type methods for the missing-data problem.

In [10], it was pointed out that the objective function (3.1) can take only $\mathbf{R}$ or $\mathbf{S}$ as its variables; precisely, the objective function is defined on subspaces. Thus, the objective function (3.1) was redefined [10] with $\mathbf{N} \in R^{m,r}$ as its variables:

$$f(\mathbf{N}) = \sum \mathbf{m}_i^T (\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i. \tag{3.2}$$

The symbols used in (3.2) are explained below.

Suppose $\tilde{\mathbf{m}}_i$ is the ith column of $\mathbf{M}$ and $\mathbf{m}_i$ is the vector, comprising the observed entries of $\tilde{\mathbf{m}}_i$. That is, $\mathbf{m}_i = \mathbf{L}_i\tilde{\mathbf{m}}_i$, where $\mathbf{L}_i$ is a selection matrix, produced from the $m \times m$ identity matrix $\mathbf{I}_m$. If the jth component of $\tilde{\mathbf{m}}_i$ is missing, delete the jth row of $\mathbf{I}_m$. $\mathbf{P}_i$ is the projection matrix on the subspace $span(\mathbf{N}_i)$: $\mathbf{P}_i \triangleq \mathbf{N}_i(\mathbf{N}_i^T\mathbf{N}_i)^{-1}\mathbf{N}_i^T$ with $\mathbf{N}_i = \mathbf{L}_i\mathbf{N}$.

In [10], the objective function is minimized by employing the LM method, where the full Hessian matrix is used. Redefine the variable of the matrix $\mathbf{N}$ in (3.2) as the column-first vectorization form of $\theta \triangleq vec(\mathbf{N})$. By using the second-order Taylor expansion,

$$f(\theta_0 + \delta\theta) = f(\mathbf{N}_0 + \delta\mathbf{N}) \approx f_0 + \mathbf{b}^T\delta\theta + \frac{1}{2}\delta\theta^T\mathbf{H}_N\delta\theta \tag{3.3}$$

is the starting point of the algorithms in [10]. See the explicit formula of $\mathbf{b}$ and $\mathbf{H}_N$ in [10] and in Appendix A.

3.2. Gauss–Newton technique for (3.2). It should be emphasized that, though the objective function in (3.2) can be seemingly categorized as the form of (2.1) because $\mathbf{m}_i^T(\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i \geq 0$ where $\mathbf{I}-\mathbf{P}_i$ is a projection matrix, the technique in section 2 cannot be directly employed to minimize (3.2). Based on the theory in section 2, this fact can be explained two ways.

First, there are $\#(\mathbf{m}_i)$ constraints in $\mathbf{m}_i^T(\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i = 0$, where $\#(\mathbf{m}_i)$ denotes the length of the vector $\mathbf{m}_i$. Because $(\mathbf{I}-\mathbf{P}_i)^2 = \mathbf{I}-\mathbf{P}_i$, $\mathbf{m}_i^T(\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i = 0 \Leftrightarrow (\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i = \mathbf{0}$ hints that there are actually $\#(\mathbf{m}_i)$ constraints. Consequently, in order to preserve the "structure" of the original objective function, the associated Gauss–Newton Hessian matrix $\mathbf{H}_i$ should be of rank $\#(\mathbf{m}_i)$, as discussed at the end of section 1.4. However, by the technique in section 2, $\mathbf{H}_i = \frac{\mathbf{b}_i\mathbf{b}_i^T}{2f_i}$ would be of rank one, for the constraint of $\mathbf{m}_i^T(\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i = 0$.

Second, if the technique in section 2 were directly employed to estimate the increment, one needs to guarantee the requirement $n \geq r(m-r)$ so as to estimate the increment during the iteration because there are $r(m-r)$ independent parameters in $\mathbf{N}$. Such a condition obviously contradicts with the fact that it is possible to estimate the missing data when at least $(m+n-r)r$ entries are observed in the matrix $\mathbf{M}$.

Nevertheless, the Gauss–Newton technique can be extended to minimize (3.2). Rewrite (3.2) as

$$f(\theta) = f(\mathbf{N}) = \sum f_i(\mathbf{N}) = \sum \mathbf{m}_i^T(\mathbf{I}-\mathbf{P}_i)(\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i. \tag{3.4}$$

Define the function $\mathbf{g}_i(\theta)$ as (actually as a function vector)

$$\mathbf{g}_i(\theta) = \mathbf{g}_i(\mathbf{N}) = (\mathbf{I}-\mathbf{P}_i)\mathbf{m}_i. \tag{3.5}$$

By the first-order Taylor expansion of

$$\mathbf{g}_i(\theta_0 + \delta\theta) \approx \mathbf{g}_i + \mathbf{B}_i\delta\theta, \tag{3.6}$$

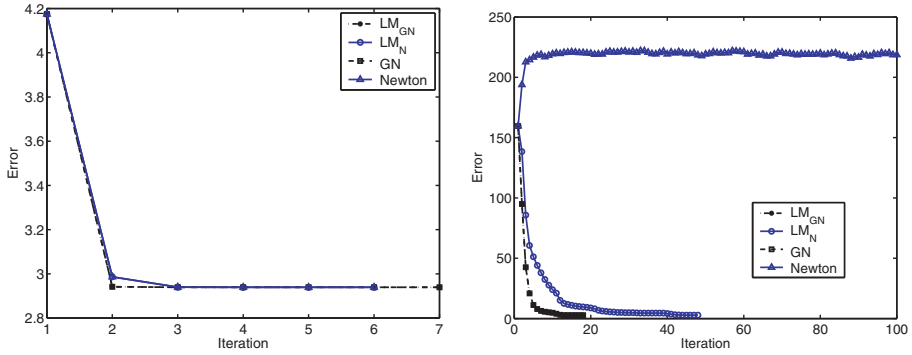
the Gauss–Newton approximation of $f(\theta)$ can be derived as

$$f(\theta_0 + \delta\theta) \approx \sum \mathbf{g}_i^T\mathbf{g}_i + 2\delta\theta^T\mathbf{B}_i^T\mathbf{g}_i + \delta\theta^T\mathbf{B}_i^T\mathbf{B}_i\delta\theta \tag{3.7}$$

with the gradient vector $\mathbf{b} = 2\sum \mathbf{B}_i^T\mathbf{g}_i$ and the Gauss–Newton Hessian matrix $\mathbf{H}_{GN} = 2\sum \mathbf{B}_i^T\mathbf{B}_i$. See the explicit formula of $\mathbf{H}_{GN}$ in Appendix A. Note that $\mathbf{b}$ in (3.3) is the same as that in (3.7), though in different forms.

4. Experiments. We have shown how to extend the Gauss–Newton method to the missing-data problem in a low-rank matrix. In this section, we present empirical results about how the basin of convergence is affected by the Hessian matrix and the Gauss–Newton Hessian matrix. It is done by evaluating the performance of four methods: the Newton method, the Gauss–Newton method, and two variants of the LM method. Here, we use LM_N to denote the LM method where the full Hessian matrix $\mathbf{H}_N$ is used in step 3 in Figure 1.2; we use LM_{GN} for the LM method where the Gauss–Newton Hessian matrix $\mathbf{H}_{GN}$ is used.

4.1. Two illustration examples. In this section and in section 4.2, we mainly use the structure from motion (SfM) problem [35, 12, 10] as an application example for the missing-data problem, where the missing-data pattern is often banded, as shown in Figure 4.2. However, the conclusion should not be restricted to the SfM problem because no specific knowledge about the SfM problem except the rank-four constraint



(a) Example 1 (b) Example 2

FIG. 4.1. *Two examples of the missing data in a low-rank matrix.*

is utilized in simulations. Thus, similar results can be obtained in other applications [9], such as collaborative filtering, system identification, global positioning, and remote sensing.

For the purpose of demonstration, two simulation examples are first presented in Figure 4.1. (The simulation setting is explained in section 4.2.) When the initial point is close to the minimum, both the Newton method and the Gauss–Newton method (denoted by GN in Figure 4.1) converge quickly, as shown in Figure 4.1(a). It takes the Newton method and the Gauss–Newton method 5 and 6 iterations, respectively, to converge. On the other hand, the Newton method does not converge when the initial point is far from the minimum, and the Gauss–Newton method still converges after 17 iterations, as shown in Figure 4.1(b).

One can see that the trust region technique improves the Newton method's performance as far as the basin of convergence is concerned. The LM_N converges in both examples after 5 and 47 iterations, respectively. The LM_{GN} also converges in both examples after 6 and 17 iterations, respectively.

4.2. Performance evaluation. In this section, we evaluate in two ways how the Gauss–Newton Hessian matrix and the full Hessian matrix affect the optimization processes. First, the "pure" Newton method and the "pure" Gauss–Newton method are compared. Second, the Gauss–Newton Hessian matrix and the full Hessian matrix are investigated in the framework of the LM method by evaluating the LM_{GN} and LM_N methods.

4.2.1. Experimental setting. The experimental setting is as follows.

- 300 3-D feature points are randomly distributed in a $10 \times 10 \times 10$ cubic, and 30 images are produced with a camera rotating around a radius-100 circle. Then, the feature points in 2-D images are translated and scaled so that they lie in the range of $[0 : 256, 0 : 256]$, and i.i.d. (independent and identically distributed) Gaussian noise, with a noise level of 2, is added. This way, a 60×300 data matrix is produced, which is approximately of rank four [35, 12, 10]. At last, about 70% of the 2-D image points are randomly supposed to be "missing," producing the missing-pattern mask.
- In the SfM problem, the missing-pattern mask is usually banded, meaning that a feature point is tracked over only a few consecutive frames. In order to present realistic simulations, banded masks are used in what follows. A

FIG. 4.2. *A banded* 30×300 *mask, where about* 70% *data is missing. A back pixel at* (i, j) *denotes the* j*th feature point appears in the* i*th frame, and a gray pixel denotes a missing feature.*

banded mask is shown in Figure 4.2, where 300 feature points are tracked over 30 frames and about 70% data of the 60×300 matrix is missing.

- In this paper, we are concerned with the *basin of convergence* of different algorithms. Precisely, we are concerned with how the Gauss–Newton Hessian matrix and the full Hessian matrix affect the basin of convergence, and with how the technique of λI in the LM algorithm affects the basin of convergence. Thus, the experimental setting is intentionally kept unchanged except for the variation of the initial point from the ground truth (GT) solution. In simulation, the perturbation from the GT solution can be controlled so that the effect on the basin of convergence can be clearly seen.
- For a rank-four matrix $\mathbf{M} \in R^{60,300}$ with missing data, the parameters to be estimated are $\mathbf{N} \in R^{60,4}$. Suppose the GT solution is $\widetilde{\mathbf{N}}$ with orthonormal columns. The initial point is obtained by adding i.i.d. Gaussian noise to $\widetilde{\mathbf{N}}$. The noise level varies (I) from 0.001 to 0.01 and (II) from 0.05 to 0.5. (Note that the square root of the mean square of the entries in $\widetilde{\mathbf{N}}$ is about 0.1291.) At each noise level, the experiment is repeated 1000 times with different feature points and different noise randomly generated.

In group (I), with a noise level of from 0.001 to 0.01, the initial point is close to the GT solution, and the initial point in group (II) is far from the GT solution. *It should be emphasized that the noise level here measures how far the initial point starts from the GT solution.*

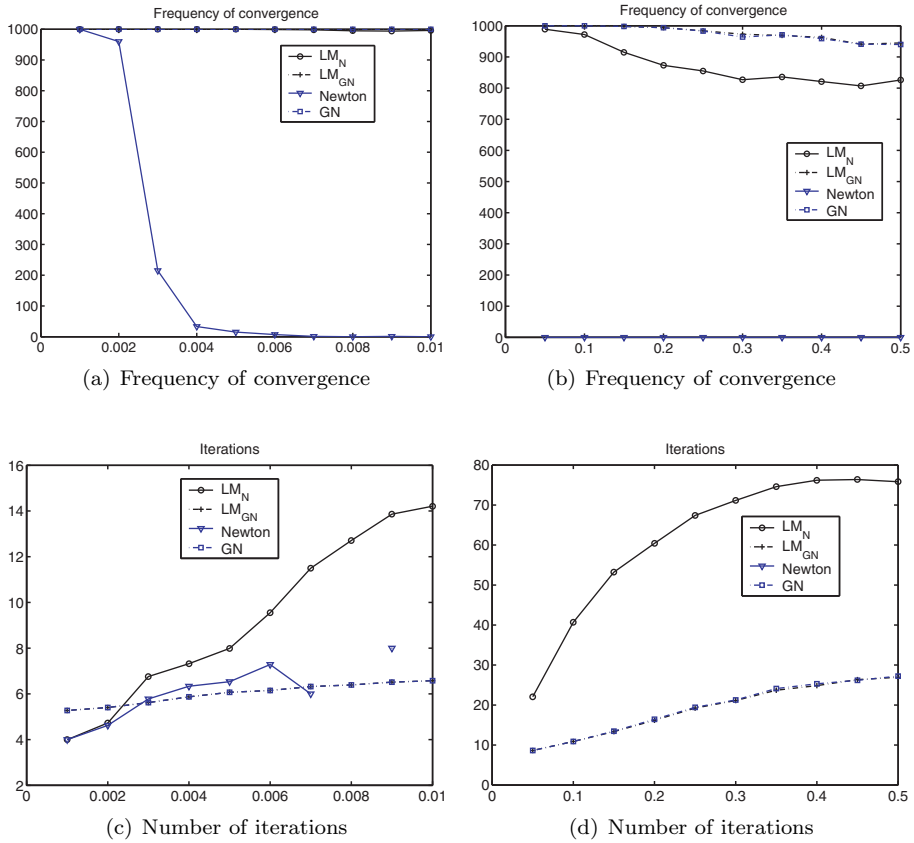
4.2.2. When the initial point is close to the GT solution. As mentioned above, of more concern is the frequency of convergence of four investigated methods, as shown in Figures 4.3(a) and 4.3(b).

One can observe from Figure 4.3(a), where the initial point is close to the GT solution:

- *Newton vs. Gauss–Newton.* The Newton method converges in all $1k$ trials only when the noise level is 0.001, and its performance quickly deteriorates as the noise level increases. In total, the Newton method fails to converge in 7768 trials. On the other hand, the Gauss–Newton method converges to the optimum in all $10k$ trials.
- *LM methods.* By adding the term of λI, the LM_N performs much better than the Newton method. Precisely, the LM_N method does not converge to the minimum in 18 cases, over all $10k$ trials. On the other hand, the extra term of λI has no effect on the Gauss–Newton method; i.e., the LM_{GN} converges in all $10k$ cases.

4.2.3. When the initial point starts far from the GT solution. As the initial point starts far from the GT solution, one can observe from Figure 4.3(b):

- *Newton vs. Gauss–Newton.* The Newton method fails to converge in all $10k$ trials. On the other hand, the Gauss–Newton method performs well even when the noise level is 0.5.



(a) Frequency of convergence (b) Frequency of convergence

(c) Number of iterations (d) Number of iterations

FIG. 4.3. *Performance of four methods. Note that the Newton method converges only in one case when the noise level is* 0.007 *or* 0.009*; and it does not converge in any cases when the noise level is* 0.008 *or* 0.010.

- *LM methods.* By adding the term of λI, the LM_N performs much better than the Newton method. On the other hand, the extra term of λI has little effect on the Gauss–Newton method, as already observed in experiments in [1].
- *LM_N vs. Gauss–Newton-based methods.* Two Gauss–Newton-based methods—the Gauss–Newton and the LM_{GN}—perform better than the LM_N method. Precisely, over all $10k$ trials, the Gauss–Newton, the LM_{GN}, and the LM_N methods converge in 9750, 9767, and 8721 cases, respectively.

4.2.4. Computational time. A side topic is the convergence speed:

- One iteration of the LM_N (or the Newton method) costs about three times as much as that of the LM_{GN} method (or the Gauss–Newton method.) This fact is due to the much more complicated computation of the Hessian matrix than that of the Gauss–Newton Hessian matrix. See the computation of the Hessian matrix and the Gauss–Newton Hessian matrix in Appendix A.
- From Figure 4.3(d), one can observe the Gauss–Newton method and the LM_{GN} method need much fewer iterations to converge, compared with the LM_N method, when the initial point starts far from the GT solution.

- On the other hand, when the initial point is near the optimal solution, for instance, when the noise level is 1×10^{-3}, the full Hessian matrix is preferred,[12] as shown in Figure 4.3(c).
- The Gauss–Newton method and the LM_{GN} method need almost the same number of iterations to converge.

4.3. Discussion. From Figure 4.3, one can see that the Gauss–Newton Hessian matrix-based methods perform better on the missing-data problem than those based on the full Hessian matrix, especially when the initial point is far from the optimal solution. A similar conclusion can be made about the problem of fundamental matrix estimation [11].[13] Moreover, it has already been observed in [1] that the Gauss–Newton Hessian matrix is preferred to the full Hessian matrix on image alignment. Actually, this phenomenon should not been restricted to problems where the advantage of the Gauss–Newton Hessian matrix has been demonstrated by experiments. For many parameter estimation problems where each term of the objective function is usually **zero-on-**$(n-1)$**-D**, one can expect the Gauss–Newton Hessian matrix-based methods to perform better than those based on the full Hessian matrix.

The underlying theory of the Newton method is to use the second-order Taylor expansion to approximate the objective function to be minimized; then the increment is estimated by minimizing the approximated quadratic function. This strategy works well when the initial point is close to the solution. However, when located far from the solution, one cannot expect that the solution still fits the approximated quadratic function very well.

Though the Gauss–Newton approximation is also a quadratics approximation of the objective function and the increment in the Gauss–Newton method is similar to that in the Newton method, the Gauss–Newton method has its own distinctive traits. First, the Gauss–Newton method utilizes *global* information of the objective function except for local behavior around the initial value. As discussed in section 2.1, only the first-order derivative of the function $g_i(\theta)$ is employed in the Gauss–Newton Hessian matrix, as in (2.3). Then, the summation of $f_i(\theta)$ is approximately transformed into the quadratic form (2.5) due to the *global* property that each term of the objective function in nonlinear least squares problems is nonnegative; i.e., $f_i(\theta) \geq 0$. Particularly, when each term $f_i(\theta)$ is **zero-on-**$(n-1)$**-D**, the Gauss–Newton approximation (2.4) is the unique quadratic approximation that retains such a structure.

Second, the Gauss–Newton method can be regarded as a general gradient descent method to some degree. The minimizer for the Gauss–Newton approximation (2.3) is

$$\widehat{\delta\theta}_i = -\frac{\partial g_i}{\partial \theta}/g_i, \tag{4.1}$$

which is actually a gradient descent step and whose step length is controlled by the constraint of $f_i(\theta) \geq 0$.

In another form, the minimizer in (4.1) is

$$\widehat{\delta\theta}_i = -\mathbf{H}^{+}_{i,GN}\mathbf{b}_i, \tag{4.2}$$

where $\mathbf{M}^{+}$ denotes the pseudoinverse of $\mathbf{M}$. It can be validated by substitution that (4.2) is equivalent to (4.1).

[12] Actually, the Gauss–Newton method or the LM_{GN} method are preferred when the computational time for each iteration is taken into account.

[13] Simulations have also been conducted on the problem of fundamental matrix estimation. However, they are not included because nothing new is added, except a few plots that are similar to those in Figure 4.3.

The increment in the Gauss–Newton method is

$$\widehat{\delta\theta} = -\mathbf{H}_{GN}^{-1}\mathbf{b}, \tag{4.3}$$

where $\mathbf{H}_{GN} = \sum_i \mathbf{H}_{i,GN}$ and $\mathbf{b} = \sum_i \mathbf{b}_i$. By comparing (4.3) and (4.2), the increment (4.3) in the Gauss–Newton method can be regarded as a general weighted summation of $\{\widehat{\delta\theta}_i\}$.

5. Conclusion. In this paper, we present a theoretical analysis of the advantage of the Gauss–Newton Hessian matrix, explaining the reason why it is preferred in Newton-type methods when the initial point is far from the solution. With this explanation, one can freely apply the Gauss–Newton-based methods to other potential nonlinear least squares problems, as illustrated in the missing-data problem in a low-rank matrix. More importantly, insight can be found about how to construct the Gauss–Newton Hessian matrix, as can be seen in the missing-data problem. A side contribution is a more efficient and more stable algorithm for the missing-data problem.

Appendix A. Gradient, Hessian matrix, and Gauss–Newton Hessian matrix of (3.2). Define $\mathbf{A}_i \triangleq \mathbf{N}_i(\mathbf{N}_i^T\mathbf{N}_i)^{-1}$, $\Phi_i \triangleq \mathbf{I}_r \otimes \mathbf{L}_i^T$, and the permutation matrix $\mathbf{C}_i$, satisfying $vec(\mathbf{N}_i^T) = \mathbf{C}_i vec(\mathbf{N}_i)$. ($\mathbf{L}_i$ and $\mathbf{N}_i$ are defined in section 3.1.)

The gradient vector is

$$\mathbf{b} = \sum \mathbf{b}_i = -\sum_i \Phi_i \mathbb{b}_i, \tag{A.1}$$

where

$$\mathbb{b}_i = (\mathbf{A}_i^T\mathbf{m}_i \otimes (\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i) + \mathbf{C}_i^T((\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i \otimes \mathbf{A}_i^T\mathbf{m}_i).$$

The Hessian matrix is

$$\mathbf{H}_N = \sum_i \mathbf{H}_i = -2\sum_i \Phi_i \mathbb{H}_i \Phi_i^T, \tag{A.2}$$

where $\mathbb{H}_i$ is defined as

$$\mathbb{H}_i = \mathcal{H}_i + \mathcal{H}_i^T$$

with

$$\begin{aligned}\mathcal{H}_i =& [\mathbf{I}_r \otimes (\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i][\mathbf{m}_i^T(\mathbf{I} - \mathbf{P}_i) \otimes (\mathbf{N}_i^T\mathbf{N}_i)^{-1}]\mathbf{C}_i \\ &- 2[\mathbf{I}_r \otimes (\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i^T][\mathbf{m}_i^T\mathbf{A}_i \otimes \mathbf{A}^T] \\ &- \mathbf{C}_i^T[\mathbf{I} \otimes \mathbf{A}_i^T\mathbf{m}_i][\mathbf{m}_i^T\mathbf{A}_i \otimes (\mathbf{I} - \mathbf{P}_i)].\end{aligned}$$

The Gauss–Newton Hessian matrix is

$$\mathbf{H}_{GN} = 2\sum_i \Phi_i \mathbf{B}_i \mathbf{B}_i^T \Phi_i^T \tag{A.3}$$

with

$$\mathbf{B}_i = (\mathbf{A}_i^T\mathbf{m}_i \otimes (\mathbf{I} - \mathbf{P}_i)) + \mathbf{C}_i^T((\mathbf{I} - \mathbf{P}_i)\mathbf{m}_i \otimes \mathbf{A}_i^T).$$

Acknowledgments. The author would like to thank Prof. David Suter and Prof. Ran Yang for their careful proofreading. He also thanks the reviewers for their insightful comments, which greatly improved this paper.

REFERENCES

[1] S. Baker and I. Matthews, *Lucas–Kanade 20 years on: A unifying framework*, Int. J. Comput. Vision, 56 (2004), pp. 221–255.
[2] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani, *Hierarchical model-based motion estimation*, in Proceedings of the European Conference on Computer Vision, 1992, pp. 237–252.
[3] A. Bjorck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
[4] M. Black and A. Jepson, *Eigen-tracking: Robust matching and tracking of articulated objects using a view-based representation*, Int. J. Comput. Vision, 36 (1998), pp. 101–130.
[5] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, London, 2004.
[6] A. M. Buchanan and A. W. Fitzgibbon, *Damped Newton algorithms for matrix factorization with missing data*, in Proceedings of the IEEE Computer Vision and Pattern Recognition, 2005, pp. 316–322.
[7] J.-F. Cai, E. J. Candes, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM J. Optim., 20 (2008), pp. 1956–1982.
[8] E. J. Candes and T. Tao, *The power of convex relaxation: Near-optimal matrix completion*, IEEE Trans. Inform. Theory, 56 (2010), pp. 2053–2080.
[9] E. J. Candes and Y. Plan, *Matrix completion with noise*, Proc. IEEE, 98 (2010), pp. 925–936.
[10] P. Chen, *Optimization algorithms on subspaces: Revisiting missing data problem in low-rank matrix*, Int. J. Comput. Vision, 80 (2008), pp. 125–142.
[11] P. Chen, *Why not use the LM method for fundamental matrix estimation?*, IET Computer Vision, 4 (2010), pp. 286–294.
[12] P. Chen and D. Suter, *Recovering the missing components in a large noisy low-rank matrix: Application to sfm*, IEEE Trans. Pattern Anal. Mach. Intell., 26 (2004), pp. 1051–1063.
[13] G. E. Christensen and H. J. Johnson, *Image consistent registration*, IEEE Trans. Medical Imaging, 20 (2001), pp. 568–582.
[14] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, 1983. Reprinted as *Classics in Applied Mathematics* 16, SIAM, Philadelphia, 1996.
[15] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., John Wiley & Sons, New York, 1987.
[16] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.
[17] E. Haber, U. M. Ascher, and D. Oldenburg, *On optimization techniques for solving nonlinear inverse problems*, Inverse Problems, 16 (2000), pp. 1263–1280.
[18] G. D. Hager and P. N. Belhumeur, *Efficient region tracking with parametric models of geometry and illumination*, IEEE Trans. Pattern Anal. Mach. Intell., 20 (1998), pp. 1025–1039.
[19] R. I. Hartley, *In defense of the eight-point algorithm*, IEEE Trans. Pattern Anal. Mach. Intell., 19 (1997), pp. 580–593.
[20] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed., Cambridge University Press, London, 2003.
[21] Y. Hua, M. Nikpour, and P. Stoica, *Optimal reduced-rank estimation and filtering*, IEEE Trans. Signal Process., 49 (2001), pp. 457–469.
[22] K. Kanatani, *Statistical bias of conic fitting and renormalization*, IEEE Trans. Pattern Anal. Mach. Intell., 16 (1994), pp. 320–326.
[23] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier, Amsterdam, 1996.
[24] H. C. Longuet-Higgins, *A computer algorithm for reconstructing a scene from two projections*, Nature, 293 (1981), pp. 133–135.
[25] W. S. Lu, S. C. Pei, and P. H. Wang, *Weighted low-rank approximation of general complex matrices and its application in the design of 2-d digital filter*, IEEE Trans. Circuits Syst. I Regul. Pap., 44 (1997), pp. 650–655.

[26] B. Lucas and T. Kanade, *An iterative image registration technique with an application to stereo vision*, in Proceedings of the International Joint Conference on Artificial Intelligence, 1981, pp. 674–679.
[27] I. Matthews and S. Baker, *Active appearance models revisited*, Int. J. Comput. Vision, 60 (2004), pp. 135–164.
[28] J. J. Moré and S. J. Wright, *Optimization software guide*, Frontiers Appl. Math. 14, SIAM, Philadelphia, 1993.
[29] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research, 2nd ed., Springer, New York, 2006.
[30] C. Poelman and T. Kanade, *A paraperspective factorization method for shape and motion recovery*, IEEE Trans. Pattern Anal. Mach. Intell., 19 (1997), pp. 206–219.
[31] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd ed., Cambridge University Press, London, 1992.
[32] P. D. Sampson, *Fitting conic sections to "very scattered" data: An iterative refinement of the Bookstein algorithm*, Comput. Graphics Image Process., 18 (1982), pp. 97–108.
[33] H. Shum, K. Ikeuchi, and R. Reddy, *Principal component analysis with missing data and its applications to polyhedral object modeling*, IEEE Trans. Pattern Anal. Mach. Intell., 17 (1995), pp. 854–867.
[34] H. Shum and R. Szeliski, *Construction of panoramic image mosaics with global and local alignment*, Int. J. Comput. Vision, 16 (2000), pp. 63–84.
[35] C. Tomasi and T. Kanade, *Shape and motion from image streams under orthography: A factorization method*, Int. J. Comput. Vision, 9 (1992), pp. 137–154.
[36] Z. Zhang, *On the optimization criteria used in two-view motion analysis*, IEEE Trans. Pattern Anal. Mach. Intell., 20 (1998), pp. 717–729.
[37] Z. Zhu, A. M.-C. So, and Y. Ye, *Fast and near-optimal matrix completion via randomized basis pursuit*, preprint, 2009.