High Order Algorithms in Robust Least-Squares Estimation with SDD Information Matrix: Redesign, Simplification and Unification

Alexander Stotsky

Abstract—This paper describes new high order algorithms in the least-squares problem with harmonic regressor and SDD (Strictly Diagonally Dominant) information matrix. Estimation accuracy and the number of steps to achieve this accuracy are controllable in these algorithms. Simplified forms of the high order matrix inversion algorithms and the high order algorithms of direct calculation of the parameter vector are found. The algorithms are presented as recursive procedures driven by estimation errors multiplied by the gain matrices, which can be seen as preconditioners. A simple and recursive (with respect to order) algorithm for update of the gain matrix, which is associated with Neumann series is found. It is shown that the limiting form of the algorithm (algorithm of infinite order) provides perfect estimation. A new form of the gain matrix is also a basis for unification method of high order algorithms. New combined and fast convergent high order algorithms of recursive matrix inversion and algorithms of direct calculation of the parameter vector are presented. The stability of algorithms is proved and explicit transient bound on estimation error is calculated. New algorithms are simple, fast and robust with respect to round-off error accumulation.

Keywords: Least-Squares Estimation, Oscillating Signals, Harmonic Regressor, Strictly Diagonally Dominant Matrix, High Order Algorithms, Recursive Matrix Inversion, Preconditioning, Neumann Series, SDD Solver

I. INTRODUCTION

Recursive least-squares (RLS) algorithms are widely used in many applications such as adaptive control, signal processing, system identification and others [1]. The parameter vector is updated recursively in each step of RLS algorithm, using information available in the previous step as follows:

\[ \theta_i = \theta_{i-1} + \frac{\Gamma_{i-1} \varphi_i}{\lambda_0 + \varphi_i \Gamma_{i-1} \varphi_i} (y_i - \theta_{i-1} \varphi_i) \]  (1)

where \( \theta_i \) is the parameter vector, \( \varphi_i \) is the regressor vector, \( y_i \) is measured signal, \( \Gamma_{i-1} \) is a gain matrix and \( \lambda_0 \) is a forgetting factor, \( i = 1, 2, ... \). Equation (1) shows clearly that round-off errors which are accumulated in step \( i - 1 \) propagate to step \( i \). This propagation has a direct impact on the estimation performance and even on system stability (see the papers [2] - [4] and references therein for quantification of the performance deterioration in the presence of round-off errors).

RLS algorithm (1) is a recursive realization of the solution of the following equation:

\[ A_i \theta_i = b_i \]  (2)

where a symmetric matrix \( A_i \) is called an information matrix, and \( b_i \) is the vector that contains measured signal. To tackle a round-off error propagation problem the equation (2) is solved with respect to \( \theta_i \) in each step \( i \). Two groups of methods are used to solve linear algebraic equation (2): exact methods and recursive methods.

Exact methods represent a finite number of calculations to find the parameter vector. Gaussian elimination method, LU decomposition, square root method and others are the best known exact methods for calculation of the parameter vector [5]-[8]. Exact methods do not give an exact solution of the algebraic equations due to round-off errors, which are always present in a finite precision implementation environment, and become computationally expensive for large scale systems. Moreover, estimation of the accuracy of the calculated parameter vector is quite difficult for exact methods. Nevertheless, equation (2) can be solved using exact methods in each step with some errors, and those errors do not propagate to the next step.

Recursive methods, such as Jacobi and Gauss-Seidel iterative methods provide a solution with controllable accuracy. However, 1) recursive methods may require a large number of iterations to achieve high accuracy, 2) the stability of the methods depends on the initial errors, 3) and finally these methods are computationally expensive. In other words, the recursive methods are not directly applicable in practice due to the potential stability problems and computational complexity.

Performance of the recursive methods may be improved by taking into account the properties of information matrix \( A_i \). This matrix is a strictly diagonally dominant (SDD) and positive definite matrix [9], [10], if the regressor \( \varphi_i \) is a harmonic regressor [11], which consists of trigonometric functions (sines and cosines) at different frequencies. Notice that numerical methods for many physical systems, machine learning, random processes, computer vision, image processing, network analysis, and computational biology are also a source of linear systems of equations with SDD matrices, see [7], [12] and references therein.

Notice also that Jacobi and Gauss-Seidel methods converge, if information matrix is an SDD matrix, but the convergence may be slow.

The following inequality, which plays a key role in the algorithm design is valid for SDD matrices:

\[ \| I - D_i^{-1} A_i \| < 1 \]  (3)

where \( A_i \) is an SDD matrix, \( D_i \) is a diagonal matrix that contains diagonal elements of \( A_i \), \( I \) is the identity matrix and
the norm is defined as the maximum row sum matrix norm. This property can be used for performance improvement of the recursive algorithms, which in turn can be divided into two groups: recursive inversion of matrix $A_i$ and direct recursive estimation of the parameter vector $\theta_i$.

Recursive inversion algorithms use the diagonal matrix $D_i^{-1}$ as a starting point for recursive inversion, where the convergence is guaranteed by inequality (3). Recursive inversion algorithms can be classified with respect to order. The second order algorithm is known as Hotelling-Bodewig [13] or Newton-Schulz algorithm [14], and it is described in many books and papers, see for example [15] - [17]. The third order algorithm is described in [18] - [20] and finally sixth and seventh order algorithms are proposed in [18] and [13] respectively. Moreover, some initial form of high order algorithms was outlined in [21] using a high order error model.

Recursive algorithms of direct estimation of parameter vector are also based on the property (3), where the matrix $I - D_i^{-1}A_i$ is directly associated with the error model. Such algorithms can also be classified with respect to order. The first order algorithm was described in [9], and initial form of high order algorithms was proposed in [22]. The first order algorithm is also known as Jacobi method [14] presented in the Richardson form [23].

Since the convergence of recursive inversion algorithms and algorithms of direct calculation of the parameter vector is based on the same inequality (3) these algorithms can be combined. A combination of the second order matrix inversion algorithm and high order algorithm of direct calculation of the parameter vector is presented in [22].

Application of high order algorithms to robust least-squares estimation has a high potential since both an accuracy and the number of steps to achieve this accuracy are controllable. This paper is dedicated towards further development of recursive high order algorithms. The contributions of this paper can be summarized as follows:

1) Simplified forms for both the high order matrix inversion algorithms and the high order algorithms of direct calculation of the parameter vector are found. The algorithms are presented as recursive procedures driven by estimation errors multiplied by the gain matrices. A simple and recursive (with respect to order) algorithm for update of the gain matrices is found, where infinite order of the algorithm (the limiting form) provides perfect estimation. The gain matrix is associated with Neumann series $\sum_{d=0}^{n-1} F_0 d$, which converges to $(D_i^{-1}A_i)^{-1}$ as the order $n \to \infty$, providing information about inverse of the information matrix, where $F_0 = I - D_i^{-1}A_i$. The convergence of this matrix series is guaranteed again by inequality (3). This form of the gain matrix provides also unification method for high order algorithms, facilitates implementation and development of combined algorithms.

2) New combined and fast convergent high order algorithms of recursive matrix inversion and algorithms of direct calculation of the parameter vector is the main contribution of this paper. The stability of algorithms is proved and explicit transient bound on estimation error is calculated.

II. Problem Statement

Suppose that a measured oscillating signal $y_k$ can be presented in the following form

$$y_i = \varphi_i^T \theta_i + \xi_i$$

(4)

where $\varphi_i$ is the harmonic regressor and $\theta_i$ is the vector of constant unknown parameters are defined as follows:

$$\varphi_i^T = \begin{bmatrix} \cos(q_1 i) \\ \sin(q_1 i) \\ \cos(q_2 i) \\ \sin(q_2 i) \\ \cdots \\ \cos(q_n i) \\ \sin(q_n i) \end{bmatrix}$$

(5)

$$\theta_i^T = \begin{bmatrix} \theta_{0i} \\ \theta_{1i} \\ \theta_{2i} \\ \theta_{3i} \\ \cdots \\ \theta_{(2r-1)i} \\ \theta_{(2r)i} \end{bmatrix}$$

(6)

where $i = 1, 2, \ldots$ is the step number, $q_p$, $p = 1, 2, \ldots, r$ are the frequencies and $\xi_i$ is a zero mean white Gaussian noise. The model of the signal (4) is presented in the following form

$$\hat{y}_i = \varphi_i^T \theta_i$$

(7)

with adjustable parameters

$$\theta_i^T = \begin{bmatrix} \theta_{0i} \\ \theta_{1i} \\ \theta_{2i} \\ \theta_{3i} \\ \cdots \\ \theta_{(2r-1)i} \\ \theta_{(2r)i} \end{bmatrix}$$

(8)

Least squares solution for estimation of the parameter vector can be written as follows [1]:

$$\theta_i = A_i^{-1} b_i$$

(9)

$$A_i = \lambda_0 \begin{bmatrix} \sum_{j=1}^{i-1} \varphi_j \varphi_j^T \\ \varphi_i \varphi_i^T \end{bmatrix}$$

(10)

$$b_i = \lambda_0 \begin{bmatrix} \sum_{j=1}^{i-1} \varphi_j y_j \\ \varphi_i y_i \end{bmatrix}$$

(11)

where the matrix $A_i$ is information matrix, and $0 < \lambda_0 < 1$ is a forgetting factor, $i = 2, 3, \ldots$ The matrix $A_i$ is an SDD matrix for a sufficiently large $i$ and forgetting factor $\lambda_0$ which is close to one [24]. This matrix property is used in the next Sections for the algorithm design and analysis. Notice that positive definiteness of information matrix for system with harmonic regressor can also be shown using partitioning method for insufficently large window size [25].

III. Recursive Algorithms of Direct Calculation of the Parameter Vector

A. First Order Algorithm

Least squares problem is a problem of calculation of the parameter vector $\theta_i$ with high accuracy. First order algorithm of direct calculation of the parameter vector can be described as follows [9]:

$$\dot{\vartheta}_k = \dot{\vartheta}_{k-1} - G_0 (A_i \dot{\vartheta}_{k-1} - b_i)$$

(12)

$$\dot{\vartheta}_k = F_0 \dot{\vartheta}_{k-1}$$

(13)
where \( \tilde{\vartheta}_k = \vartheta_k - \theta_i \) is estimation error, and the error matrix is defined as follows:

\[
F_0 = I - G_0 A_i \tag{14}
\]

and \( G_0 = D_1 \iota \) is a diagonal matrix that contains diagonal elements of \( A_i \), which means that the inverse of the diagonal matrix \( D_i^{-1} \) is used as approximation of \( A_i^{-1} \). The following property of the error matrix \( F_0 \) is established for SDD matrix \( A_i \):

\[
\|F_0\| \leq \kappa < 1 \tag{15}
\]

where the norm is defined as the maximum row sum matrix norm. Inequality (15) guarantees the stability of the error model (13).

Notice that matrix \( G_0 \) is usually called the preconditioner (preconditioning matrix) and matrix \( F_0 \) is called the iteration matrix. Preconditioner is directly associated with estimate of the inverse of \( A_i \). Moreover, algorithm (12) can be seen as Jacobi method, written in the Richardson form [14].

**B. High Order Algorithm: Three Equivalent Forms of the Gain Matrix**

High order algorithms of direct calculation of the parameter vector are described in [22]:

\[
\dot{\vartheta}_k = \dot{\vartheta}_{k-1} - \Gamma_n G_0 \left\{ A_i \dot{\vartheta}_{k-1} - b_i \right\} \tag{16}
\]

\[
\Gamma_n = \sum_{d=0}^{n-1} \frac{n!}{(n-d)! d!} (-1)^{n-d-1} (G_0 A_i)^{n-d-1} \tag{17}
\]

\[
\hat{\vartheta}_k = F_0^n \dot{\vartheta}_{k-1}, \quad \hat{\vartheta}_k = F_0^n k \dot{\vartheta}_0 \tag{18}
\]

where \( n = 1, 2, 3, ... \) is referred as an order of the algorithm, \( k = 1, 2, 3, ... \) and \( \|F_0\| < 1 \).

High order algorithms are tabulated in Table I. The order of the algorithm is chosen according to the accuracy requirements. The order of the algorithm can be easily increased recursively in the algorithm (16), (17) by adding additional terms to the gain matrix. The gain matrix \( \Gamma_n \) of the algorithm of order \( n \), where \( n = 2, 3, 4, ... \) is calculated recursively via the gain matrix \( \Gamma_{n-1} \) as follows:

\[
\Gamma_n = \Gamma_{n-1} + F_0^{n-1} \tag{19}
\]

where \( \Gamma_1 = I \), and \( F_0^{n-1} \) is the gain increment. Identity (19) is proved via explicit evaluation of the difference \( \Gamma_n - \Gamma_{n-1} \), where \( \Gamma_n \) is defined in (17). Gain matrix \( \Gamma_n \) can also be presented as Neumann series as follows:

\[
\Gamma_n = \left( I + F_0 + F_0^2 + F_0^3 + \ldots + F_0^{n-1} \right) \tag{20}
\]

and \( \Gamma_n = \sum_{d=0}^{n-1} F_0^d \).

**ALGORITHMS FOR DIRECT CALCULATION OF PARAMETERS**

<table>
<thead>
<tr>
<th>ORDER</th>
<th>GAIN MATRIX ( \Gamma_n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( I )</td>
</tr>
<tr>
<td>2</td>
<td>( I + F_0 )</td>
</tr>
<tr>
<td>3</td>
<td>( I + F_0 + F_0^2 )</td>
</tr>
<tr>
<td>4</td>
<td>( I + F_0 + F_0^2 + F_0^3 )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>( n )</td>
<td>( \sum_{d=0}^{n-1} F_0^d \rightarrow (G_0 A_i)^{-1} ) as ( n \rightarrow \infty )</td>
</tr>
</tbody>
</table>

**TABLE I**

A FAMILY OF HIGH ORDER ALGORITHMS

Notice that incremental relation (19) depends on the order only and does not depend on estimation error. This relation allows to increase the order recursively and plays a key role in algorithm (16). This relation has a direct impact on the accuracy of estimation and allows to associate the gain matrix with convergent matrix series. This convergence property results in a new notion of the infinite order algorithm, described in the next Section.

**C. The Infinite Order Algorithm: Limiting Form with \( \Gamma_\infty = \lim_{n\rightarrow \infty} \Gamma_n \)**

The gain matrix \( \Gamma_n = \sum_{d=0}^{n-1} F_0^d \) converges to the matrix \( (G_0 A_i)^{-1} \) as the order \( n \) tends to infinity i.e.,

\[
\Gamma_n = \sum_{d=0}^{n-1} F_0^d \rightarrow (G_0 A_i)^{-1} = A_i^{-1} G_0^{-1} \quad \text{as} \quad n \rightarrow \infty \tag{21}
\]

where \( \|F_0\| < 1 \) and \( F_0 = I - G_0 A_i \). Relation (21) follows directly from the convergence properties of the matrix series, see theorem 5.6.15 in [26]. Notice that the convergence rate of the recursive matrix inversion algorithm based on Neumann series is compared to the convergence rate of the second order algorithm (25)-(26) in Appendix 3 of [9].

Substitution of \( \Gamma_\infty = \lim_{n\rightarrow \infty} \Gamma_n = A_i^{-1} G_0^{-1} \) in (16) results in the following identity \( \dot{\vartheta}_k = \vartheta_i \). Order \( n \) quantifies the proximity between the gain matrix \( \Gamma_n \) and the matrix \( A_i^{-1} G_0^{-1} \), which contains information about the inverse of the information matrix and provides a perfect estimation, where

\[
\Gamma_n = (G_0 A_i)^{-1} - (G_0 A_i)^{-1} (I - G_0 A_i)^n \tag{22}
\]

\[
I - \Gamma_n G_0 A_i = (I - G_0 A_i)^n = F_0^n \tag{23}
\]

Substitution of \( \Gamma_n \) defined in (22) and (23) in (16) results in the error model (18). Finally, the gain matrix \( \Gamma_n \) is presented.

273
in three equivalent forms (17), (19) and (22), where (19) is used in implementation, (22) is useful for stability analysis, and (17) can be seen as intermediate form. Notice that the product \( \Gamma_n G_0 \) in (16) can also be seen as a composite preconditioner, in which \( \Gamma_n \) is a polynomial preconditioner, see [14] for other types of preconditioning. Notice that the dynamics of the tracking error \( \varepsilon_k = A_i \theta_k - b_i \) can be presented as follows:

\[
\varepsilon_k = A_i \theta_k - \Gamma_n G_0 \left( A_i \theta_k - b_i \right) - b_i
\]

\[
(\varepsilon_k - 1) = \left( I - A_i \Gamma_n G_0 \right) \varepsilon_k - 1
\]

and substitution of \( \Gamma_n = A_i^{-1} G_0^{-1} \) in (24) implies also that \( \varepsilon_k = 0 \). Notice also that algorithm (16) has two loops and the estimation error can be minimized by 1) iterating of \( \theta_k \) for a fixed order and minimizing the error \( A_i \theta_k - b_i \), 2) increasing the order, where the gain matrix converges to \( A_i^{-1} G_0^{-1} \).

High accuracy of estimation may be achieved in one step, increasing the order of the algorithm. The performance of the algorithm is illustrated in Figure 1, where the error norm is plotted after the first iteration.

IV. INVERSION ALGORITHMS OF AN SDD MATRIX

A. Second Order Algorithm

The parameter vector can be also calculated via (9) and an estimate of the inverse of the information matrix. Strict diagonal dominance of the information matrix \( A_i \) allows fast and computationally efficient estimation of the inverse of information matrix as follows:

\[
G_k = G_{k-1} + F_{k-1} G_{k-1}
\]

(25)

\[
F_k = I - G_k A_i
\]

(26)

where \( G_k \) is an estimate of the inverse of \( A_i \), and matrix \( F_k \) is associated with the inversion error, \( I \) is the identity matrix, and \( k = 1, 2, \ldots \). Algorithm (25)-(26) is initialized according to (14), where inequality (15) is valid. Matrix \( F_k \) which represents the inversion error can be written as follows:

\[
F_k = I - G_k A_i = F_0^{2k}
\]

(27)

and \( \lim_{k \to \infty} G_k = A_i^{-1} \) due to property (15).

Algorithm (25)-(26) can be derived from the following error model

\[
F_k = F_{k-1}^2
\]

(28)

\[
I - G_k A_i = (I - G_{k-1} A_i)^2
\]

(29)

where (29) is solved with respect to \( G_k \), and it is called the second order algorithm or Newton-Schulz algorithm [15] - [17].

![Fig. 1. The error norm \( \tilde{\theta}_i = \theta_i - \theta_i \) is plotted as function of the order of the algorithm (16) after the first step.](image)

### Table II

<table>
<thead>
<tr>
<th>ORDER</th>
<th>GAIN UPDATE ALGORITHM ( L_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( I )</td>
</tr>
<tr>
<td>3</td>
<td>( I + F_{k-1} )</td>
</tr>
<tr>
<td>4</td>
<td>( I + F_{k-1} + F_{k-1}^2 )</td>
</tr>
<tr>
<td>5</td>
<td>( I + F_{k-1} + F_{k-1}^2 + F_{k-1}^3 )</td>
</tr>
<tr>
<td>( m )</td>
<td>( m )</td>
</tr>
<tr>
<td>( \sum_{d=0}^{m-2} F_{k-1}^d \to (G_{k-1} A_i)^{-1} ) as ( m \to \infty )</td>
<td></td>
</tr>
</tbody>
</table>

B. High Order Matrix Inversion Algorithm

A family of high order matrix inversion algorithms is derived from the following error model:

\[
F_k = F_k^{m_k}
\]

(30)

where \( m = 2, 3, \ldots \) is referred as the order of the algorithm, and \( F_k = F_k^{m_k} \), where \( F_k \) and \( F_0 \) are defined in (26) and (14) respectively, \( k = 1, 2, 3, \ldots \).

Solving (30) with respect to \( G_k \) yields:

\[
G_k = G_{k-1} + F_{k-1} L_m G_{k-1}
\]

(31)

error gain

where \( L_m \) is a gain matrix, which is defined as follows:

\[
L_m = \sum_{d=0}^{m-2} \frac{(m-1)!}{(m-d-1)!} (-1)^{m-d-2} (G_{k-1} A_i)^{m-d-2}
\]

(32)
The gain matrix of order \( m \) is calculated recursively via the gain matrix of order \( m - 1 \) (similar to (19)), where \( m = 3, 4, 5, \ldots \) as follows:

\[
L_m = L_{m-1} + F_{m-1}^{m-2}
\]

where \( L_2 = I \) and \( L_m = \sum_{d=0}^{m-2} F_d^{m-2} \). Notice that the gain matrix \( L_m \) can be presented in the following form

\[
L_m = (G_{k-1}A_i)^{-1} - (G_{k-1}A_i)^{-1} F_{k-1}^{m-1}, \quad m = 2, 3, \ldots
\]

and the error model (30) is obtained after substitution of this gain matrix in (31).

High order algorithms are tabulated in Table II. The performance of high order algorithms is illustrated in Figure 2, where the norm of the error matrix \( F_k \) is plotted for three steps \( k = 0, 1, 2 \) as a function of the order \( m \).

Notice that algorithm (31),(33) is a unified form of high order algorithms described in [13], [21] and in many other papers.

C. The Infinite Order Algorithm, \( m = \infty \)

The gain matrix \( L_m = \sum_{d=0}^{m-2} F_d^{m-1} \) converges to the matrix \((G_{k-1}A_i)^{-1}\) as the order \( m \) tends to infinity i.e.,

\[
L_m = \sum_{d=0}^{m-2} F_d^{m-1} \to (G_{k-1}A_i)^{-1} = A_i^{-1}G_{k-1}^{-1} \quad \text{as} \quad m \to \infty
\]

where \( \|F_{k-1}\| < 1 \). Substitution of \( L_\infty = A_i^{-1}G_{k-1}^{-1} \) in (31) results in the following identity \( G_k = A_i^{-1} \).

V. COMBINED HIGH ORDER ALGORITHMS

A. Description of the Algorithms

High order algorithms described in Section III-B and Section IV-B can be combined as follows:

\[
F_{k-1} = I - G_{k-1}A_i
\]

\[
L_m = L_{m-1} + F_{m-1}^{m-2} = \sum_{d=0}^{m-2} F_d^{m-1}
\]

\[
G_k = G_{k-1} + F_{k-1}L_mG_{k-1}
\]

\[
F_k = I - G_kA_i
\]

\[
\Gamma_n = \Gamma_{n-1} + F_{n-1}^{n-1} = \sum_{d=0}^{n-1} F_d^{n-1}
\]

\[
\tilde{\vartheta}_k = \vartheta_{k-1} - \Gamma_n G_k \{ A_1\vartheta_{k-1} - b_1 \}
\]

\[
\tilde{\vartheta}_k = F_k^{\vartheta} \tilde{\vartheta}_{k-1}
\]

\[
F_k = F_{k-1}^{\vartheta}
\]

where \( G_k \) is an estimate of the inverse of information matrix \( A_i \), \( F_k \) is an estimation error, and \( \tilde{\vartheta}_k \) is an estimate of the parameter vector \( \theta_i \). Algorithms (35)-(40) are initialized according to (14) with two parameters to be chosen. The first one is the order of the matrix inversion algorithm \( m = 2, 3, 4, \ldots \) and the second one is the order of the algorithm of direct calculation of the parameters \( n = 1, 2, 3, \ldots \). These orders have impact on the gain matrices \( L_m \) and \( \Gamma_n \) only. The computational burden of the algorithm may be reduced using the same calculations for \( \Gamma_n \) and \( L_m \).

Notice that the substitution of \( L_\infty = A_i^{-1}G_{k-1}^{-1} \) and \( \Gamma_\infty = A_i^{-1}G_{k-1}^{-1} \) in (37) and (40) implies that \( G_k = A_i^{-1} \) and \( \tilde{\vartheta}_k = \theta_i \). Matrix \( G_k \) can be seen as preconditioner and matrices \( L_m \) and \( \Gamma_n \) are polynomial preconditioners.

The following error model is valid for algorithms (35)-(40):

\[
\tilde{\vartheta}_k = F_k^{\vartheta} \tilde{\vartheta}_{k-1} = F_k^{\vartheta} \hat{\vartheta}_{k-1}
\]

\[
F_k = F_{k-1}^{\vartheta}
\]

where \( \hat{\vartheta}_k = \vartheta_k - \theta_i, k = 1, 2, \ldots \). This model can be written in the following form:

\[
\tilde{\vartheta}_k = F_0 \frac{(m^{k+1} - m)}{(m - 1)} \hat{\vartheta}_0
\]

where \( F_0 = I - G_0A_i \), \( \hat{\vartheta}_0 = \vartheta_0 - \theta_i \) and \( \|F_0\| < 1 \), which guarantees the system stability.

B. Which Order \( n \) or \( m \) ?

Two parameters should be chosen in algorithms (35)-(40): the order \( n \) and the order \( m \). Error model (43) can be used to facilitate the choice of these orders. The norm of the error \( \hat{\vartheta}_1 \) calculated in one step for two cases is plotted in Figure 3. The norm of the second order matrix inversion algorithm \( m = 2 \) is plotted with a red line for the order of the algorithm of direct calculation of the parameter vector which is equal to \( n = 1, 2, 3, 4, 5 \). The same norm for the first order algorithm of direct calculation of the parameter vector \( n = 1 \) is plotted with a blue line for the order of the matrix inversion algorithm which is equal to \( m = 2, 3, 4, 5 \). The Figure shows that higher order of the algorithm of direct calculation of the parameter vector should be chosen that essentially reduces estimation error. Notice that the error model for \( m = 2 \) and
This work was supported by the ÅF. The author is grateful to the anonymous reviewers for their helpful comments.