Nonlinear Estimation for Gyroscope Calibration for the Inertial Pseudo Star Reference Unit

by

Marc Wayne McConley

BSE, Mechanical and Aerospace Engineering, Princeton University
Princeton, New Jersey (1992)

Submitted to the Department of Aeronautics and Astronautics in partial fulfillment of the requirements for the degree of Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1994

© Marc Wayne McConley, 1994. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part.

Author ............................................................ Department of Aeronautics and Astronautics

May 24, 1994

Approved by ................................................ James H. Murphy
Charles Stark Draper Laboratory
Thesis Supervisor

Certified by .................................................. Professor Wallace E. VanderVelde
Department of Aeronautics and Astronautics
Thesis Advisor

Accepted by ................................................ Professor Harold Y. Wachman
Chairman, Department Graduate Committee
Nonlinear Estimation for Gyroscope Calibration for the Inertial Pseudo Star Reference Unit

by

Marc Wayne McConley

Submitted to the Department of Aeronautics and Astronautics on May 24, 1994, in partial fulfillment of the requirements for the degree of Master of Science in Aeronautics and Astronautics

Abstract

The gyroscope calibration problem is addressed for the Inertial Pseudo Star Reference Unit. A nonlinear scale factor error is described by two gyro error states (an amplitude and a decay constant) which enter the dynamics in a nonlinear fashion. A comparison is made between the extended Kalman filter (EKF) and the maximum likelihood system identification (MLSI) method for solving the resulting nonlinear estimation problem. The principal advantage of the MLSI over the EKF is improved convergence. By using the MLSI instead of the EKF, a filter designer developing a nonlinear estimation routine for gyroscope calibration would require less trial and error to achieve an acceptable design.

The U-D factorization filter is combined with a separate bias estimation framework, and the results are extended to the nonlinear estimation algorithms under consideration. The separate bias U-D factorization filter requires fewer computations than the standard U-D factorization filter while retaining the advantageous numerical qualities of that approach.

Thesis Supervisor: James H. Murphy
Title: Charles Stark Draper Laboratory

Thesis Advisor: Professor Wallace E. VanderVelde
Title: Department of Aeronautics and Astronautics
Acknowledgments

The author wishes gratefully to acknowledge the substantial assistance offered by the MIT thesis advisor, Professor Wallace VanderVelde. His direction, technical advising, and detailed feedback concerning the work while in progress was extremely helpful. The introduction he provided to literature on the separate bias estimation algorithm and other topics proved especially valuable.

Many people at the Charles Stark Draper Laboratory contributed in important ways toward this project. James Murphy (the Draper thesis supervisor) and Tom Thorvaldsen provided frequent direction, focus, and needed technical experience for the completion of the project. Howie Musoff was available for helpful discussions, particularly where an understanding of the IPSRU hardware was required. Michael Ash provided the initial introduction to, and helpful comments about, the system identification methodology. The author is grateful to Mike Luniewics for performing numerous experimental tests, particularly as the IPSRU program drew to a close. Michael Fikes and Fred Gruman were always available to discuss numerous technical issues on an informal and almost daily basis.

This thesis was typeset by the author using \LaTeX\ and macros maintained by the Student Information Processing Board at MIT. A number of macros defined by Michael Fikes at the Draper Lab also proved useful.

I would like to thank my fiancée, Heather Akin, for her generous encouragement, advice, and support during the time this thesis was in progress. I thank also my family for their faithful encouragement and for their forgiveness as I have experienced so much change in the allocation of my time and energy over the past year.

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship.

Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under Contract F29601-90-C-0017.

Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.
I hereby assign my copyright of this thesis to The Charles Stark Draper Laboratory, Inc., Cambridge Massachusetts.

Marc Wayne McConley
May 24, 1994

Permission is hereby granted by The Charles Stark Draper Laboratory, Inc., to the Massachusetts Institute of Technology to reproduce any or all of this thesis.
Contents

1 Introduction .......................................................... 13

2 IPSRU Dynamic Model .................................................. 17
  2.1 Nonlinear Dynamic Equations ...................................... 17
    2.1.1 Nonlinear Terms ........................................... 18
    2.1.2 Gravity-dependent Terms ................................... 19
  2.2 Linearized Dynamic Equations ...................................... 19

3 Linear Filtering: Applications to IPSRU .......................... 21
  3.1 State-space Description of Linear Dynamics ..................... 21
    3.1.1 IPSRU Gyro Model .......................................... 23
    3.1.2 IPSRU Measurement Model .................................. 24
  3.2 Kalman Filter Equations ......................................... 26
    3.2.1 Initial Conditions .......................................... 26
    3.2.2 Time Propagation Equations ................................ 27
    3.2.3 Measurement Update Equations ................................ 28
  3.3 U-D Factorization Filter ......................................... 29
    3.3.1 Initial Conditions .......................................... 29
    3.3.2 Time Propagation Equations ................................ 30
    3.3.3 Measurement Update Equations ................................ 32
  3.4 Separate Bias Estimation with U-D Factorization Filter .......... 33
    3.4.1 Initial Conditions .......................................... 35
    3.4.2 Time Propagation Equations ................................ 36
    3.4.3 Measurement Update Equations ................................ 37
    3.4.4 Comparison with Standard U-D Factorization Filter .......... 39

4 Nonlinear IPSRU Dynamics ............................................. 41
  4.1 Extended Kalman Filter .......................................... 41
4.1.1 Time Propagation Equations ........................................ 43
4.1.2 Measurement Update Equations .................................... 43
4.2 Maximum Likelihood System Identification .......................... 43
  4.2.1 System Identification Theory ..................................... 45
  4.2.2 Separate Bias System Identification ............................. 47
  4.2.3 Recapitulation of System Identification Algorithm .......... 51

5 Simulation of IPSRU Gyro Calibration ................................. 55
  5.1 IPSRU Test Table Configuration ..................................... 55
    5.1.1 Computation of Direction Cosine Matrices .................. 56
    5.1.2 Computation of Quaternions .................................. 58
  5.2 Description of Simulation ......................................... 59
    5.2.1 Simulated Static Calibration Sequence ....................... 60
    5.2.2 Simulated Dynamic Calibration Sequence ..................... 60
    5.2.3 True Values for Simulation ................................ 62
    5.2.4 Monte Carlo Analysis ....................................... 63
  5.3 Performance Evaluation ............................................ 64
    5.3.1 Static Calibration ........................................... 64
    5.3.2 Dynamic Calibration ......................................... 66
    5.3.3 Maximum Likelihood System Identification: Effect of \( N_{\text{param}} \) .......... 70
  5.4 Convergence Properties ............................................ 73
    5.4.1 Regions of Convergence for Nonlinear Algorithms ........ 75
    5.4.2 Maximum Likelihood System Identification: Effect of \( N_{\text{param}} \) .......... 78
  5.5 Discussion ......................................................... 80

6 Conclusions ...................................................................... 83
  6.1 Suggestions for Future Work ........................................ 84

A Example of Separate Bias Estimation Algorithm for a Linear System 87
  A.1 Initial Conditions .................................................. 87
  A.2 Time Propagation Equations ........................................ 88
  A.3 Measurement Update Equations ..................................... 89
List of Figures

2-1 Transfer function for nonlinear scale factor error. ........................................ 18

5-1 Relation of local level and IPSRU body frames through gimbal angles. .............. 56
5-2 Relation of local level and earth fixed frames through latitude. .................... 57
5-3 Gimbal angles used in the static calibration test sequence. ............................ 60
5-4 Rotation rate sequence for dynamic calibration test sequence, simulated about each of the z-, y-, and z-axes, in order. .......................................................... 61
5-5 Absolute values of mean estimation errors on $A$ and $\alpha$ for MLSI with varying $N_{\text{param}}$. .............................................................. 71
5-6 Standard deviations of estimation errors on $A$ and $\alpha$ for MLSI with varying $N_{\text{param}}$. .............................................................. 71
5-7 Estimation error in $\alpha_z$ as a function of $\alpha_0$ at $A_0 = 50 \text{ ppm}/(\text{deg/s})^2$. .............................................................. 74
5-8 Estimation error in $A_z$ as a function of $\alpha_0$ at $A_0 = 50 \text{ ppm}/(\text{deg/s})^2$. .............................................................. 75
5-9 Regions in $(\alpha_0, A_0)$ space for which EKF and MLSI estimates of $A$ converge to within 10% of the true value. ......................................................... 76
5-10 Regions in $(\alpha_0, A_0)$ space for which EKF and MLSI estimates of $\alpha$ converge to within 10% of the true value. ......................................................... 76
5-11 Regions in $(\alpha_0, A_0)$ space for which EKF and MLSI estimates of both $A$ and $\alpha$ converge to within 10% of the true values. ......................................................... 78
5-12 Regions in $(\alpha_0, N_{\text{param}})$ space for which MLSI estimates of $A$ converge to within 10% of the true values. $A_0 = 20 \text{ ppm}/(\text{deg/s})^2$. ......................................................... 79
5-13 Regions in $(\alpha_0, N_{\text{param}})$ space for which MLSI estimates of $\alpha$ converge to within 10% of the true values. $A_0 = 20 \text{ ppm}/(\text{deg/s})^2$. ......................................................... 79
5-14 Regions in $(\alpha_0, N_{\text{param}})$ space for which MLSI estimates of both $A$ and $\alpha$ converge to within 10% of the true values. $A_0 = 20 \text{ ppm}/(\text{deg/s})^2$. ......................................................... 80
List of Tables

2.1 Input, spin, and output axes in gyro body frame. ........................................... 19

3.1 Comparison of floating point operations for U-D factorization filter with and without separate bias estimation. ............................................. 40

5.1 Inner and middle gimbal angles for each dynamic calibration sequence. ........ 61
5.2 True static gyro errors for IPSRU simulation test. ................................. 62
5.3 True dynamic gyro errors for IPSRU simulation test. ......................... 62
5.4 Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of linear Kalman filter results with $N = 20$. ..... 64
5.5 Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of EKF results with $N = 20$. ........... 65
5.6 Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of MLSI results with $N = 20$. ................. 66
5.7 Static and dynamic estimation errors. Monte Carlo analysis of linear Kalman filter dynamic calibration results with $N = 20$. ........................................ 67
5.8 Static and dynamic estimation errors. Monte Carlo analysis of EKF dynamic calibration results with $N = 20$. ........................................ 68
5.9 Static and dynamic estimation errors. Monte Carlo analysis of MLSI dynamic calibration results with $N = 20$. ........................................ 69
Chapter 1

Introduction

Gyroscopes are important devices for spacecraft attitude control because they provide measurements of angular orientation relative to inertial space. For some applications, high precision attitude control is necessary, and in those cases highly precise gyroscope measurements are also required. Gyroscopes are subject to errors, such as drift biases, scale factor errors, and gravity-sensitive errors in the angular velocity. Hence, calibration is often required in order to estimate these errors for the specific unit in question, so that corrections can be made to any system software which processes the gyro measurements.

Many gyroscope calibration problems are complicated by the presence of nonlinear terms in the dynamic equations for the gyro error rates. We propose two algorithms as solutions to the resulting nonlinear estimation problem — the extended Kalman filter (EKF) and the maximum likelihood system identification (MLSI) method. The thesis compares these two algorithms based on the residual errors in estimating the gyro error parameters. The convergence properties of each method are also investigated. The thesis concludes by evaluating tradeoffs between the EKF and MLSI methods in the context of the gyro calibration problem.

The Inertial Pseudo Star Reference Unit (IPSRU) is considered as a case study in comparing the performance of these two nonlinear estimation algorithms. IPSRU is a high precision pointing system designed to provide jitter suppression for space systems requiring very high precision attitude control. It aids in target tracking applications by generating a collimated beam of light used as a pseudo star to eliminate smearing in a sensor image, as detailed in [8]. The light beam is inertially stabilized by a two-degree-of-freedom gyro. The gyro must be properly calibrated so that IPSRU can provide the precise line of sight information required. The IPSRU hardware and design application concept are described in detail in [8].

IPSRU employs the Kearfott Guidance and Navigation Mod II E/S gyro, the dynamics of which include a nonlinear scale factor error. This is a scale factor error exhibiting exponentially decaying
transient behavior, as detailed in Section 2.1.1 of the thesis. A proper gyroscope calibration requires the estimation of the decay time as well as an amplitude, and these quantities enter the dynamics of the system in a nonlinear fashion. In this thesis we compare the performance of the EKF and MLSI in estimating gyro errors associated with the nonlinear scale factor error.

This thesis evaluates several filtering algorithms and applies them to the IPSRU gyro calibration problem. The most common filter algorithm is the well-known Kalman filter [9]. The Kalman filter is known to be the optimal filter algorithm for linear estimation problems, but several computational approaches exist. Some of these, beginning with Potter (cited in [2]), have been developed to enhance numerical conditioning, through use of some factorized form of the covariance matrix to ensure positive-definiteness. Section 3.3 reviews the U-D factorisation filter, in which the covariance matrix is expressed using diagonal and upper triangular matrix factors.

Another framework which applies especially well to the gyro calibration problem is the separate bias estimation algorithm proposed by Friedland [6]. This algorithm takes advantage of the structure of a dynamic system having a large number of bias states — states which we assume to be constant, though unknown — to be estimated. Most of the states in a gyro calibration (including drift bias, scale factor errors, and coefficients of gravity sensitive errors) are bias states. The separate bias estimation greatly reduces the computation required in the filter algorithm.

During the course of the research represented in this thesis, a separate bias U-D factorization filter (introduced in Section 3.4) is developed [12]. This algorithm combines the reduction in computation time of the separate bias method with the benefits in numerical conditioning associated with U-D factorization. The forms of the equations derived are especially convenient because they allow fairly straightforward extension to include either the EKF or the MLSI algorithms.

Section 4.1 reviews the extended Kalman filter [7]. This filter is commonly used when the dynamics and measurement equations include nonlinear functions of the state variables. The EKF uses a local linearization of the nonlinear functions at the estimated state to approximate the nonlinear dynamics of the system. Caglayan and Lancraft [5] have developed a separated EKF which incorporates separate bias estimation into the EKF, and Section 4.1 extends this to include U-D factorization. While the EKF can yield highly accurate state estimates in a nonlinear system, it may not converge for all choices of initial conditions [1, 17].

Maximum likelihood system identification [11, 1, 17] is an attractive alternative when the system dynamics can be written in terms of linear dynamics involving uncertain parameters, for example, in the state transition matrix. The details of this algorithm are presented in Section 4.2. Incorporation of separate bias estimation and U-D factorization into the MLSI is not so straightforward as in the case of the EKF. Nonetheless, such an integration is possible if we take advantage of a number of approximations proposed by Maybeck [11]. This is thoroughly discussed in Section 4.2.2.

The EKF and MLSI are compared with each other and with the standard Kalman filter using a
simulated IPSRU gyro calibration in Section 5. The comparison includes both a performance test and a convergence test. The performance test is a statistical comparison of the estimation errors resulting from both algorithms for a specific set of initial conditions. The estimation errors from the EKF in the gyro error states contributing to the nonlinear scale factor error are found to be lower than those from the MLSI.

When considering a range of possible initial conditions on these quantities, the MLSI is found to converge (to within 10% of the true values of the relevant gyro error states) more easily than the EKF. By this we mean that the MLSI converges for initial conditions in the vicinity of the true state, whereas the EKF does not necessarily do so. As a result, the filter designer will likely require more trial and error in order to select the right initial conditions for the EKF. While the EKF has the potential to produce lower estimation errors than the MLSI, the convergence properties of the MLSI make that algorithm a very attractive alternative.
Chapter 2

IPSRU Dynamic Model

This chapter gives the dynamic equations associated with the IPSRU gyro system and the relation of gyro error parameters to measured attitude errors. Section 2.1 presents the equations for the full nonlinear system and discusses the physical significance of a number of gyro error terms. Section 2.2 provides linearized equations for the dynamics of the nonlinear self-heating terms.

2.1 Nonlinear Dynamic Equations

When considering gyro dynamics we are primarily concerned with the difference between observed and actual rotation rates about the body axes, which arises as a result of various error terms. For IPSRU, the three components of this gyro error rate (sometimes called the drift rate, \( \omega_B^B \)) are computed as follows [13, 15].

\[
\begin{align*}
\omega_{Bz}^B &= \text{BIAS}_z + ADI_zg_z^B + ADO_zg_z^B + ADS_zg_z^B + ADSS_z(g_z^B)^2 + ADSL_zg_z^Bg_z^B + ADIO_zg_z^Bg_y^B + ADSO_zg_z^Bg_y^B + \text{SP}_z\kappa_z\omega_z^B + \text{SN}_z(1 - \kappa_z)\omega_z^B + \text{SFNL}_zw_z^B + n_z^B \quad (2.1) \\
\omega_{By}^B &= \text{BIAS}_y + ADI_yg_y^B + ADO_yg_y^B + ADS_yg_y^B + ADSS_y(g_y^B)^2 + ADSL_yg_y^Bg_z^B + ADIO_yg_y^Bg_z^B + ADSO_yg_y^Bg_z^B + \text{SP}_y\kappa_y\omega_y^B + \text{SN}_y(1 - \kappa_y)\omega_y^B + \text{SFNL}_yw_y^B + n_y^B \quad (2.2) \\
\omega_{Bx}^B &= \text{BIAS}_x + ADI_xg_x^B + ADO_xg_x^B + ADS_xg_x^B + ADSS_x(g_x^B)^2 + ADSL_xg_x^Bg_z^B + ADIO_xg_x^Bg_z^B + ADSO_xg_x^Bg_z^B + \text{SP}_x\kappa_x\omega_x^B + \text{SN}_x(1 - \kappa_x)\omega_x^B + \text{SFNL}_xw_x^B + n_x^B \quad (2.3)
\end{align*}
\]

where

\[
k_z = \begin{cases} 
1 & \omega_z^B \geq 0 \\
0 & \omega_z^B < 0 
\end{cases} \quad (2.4)
\]

and \( k_y \) and \( k_x \) are defined similarly.

Equations (2.1–2.3) are used to compute the gyro attitude errors about each of the three axes in
the IPSRU body frame. The attitude error, $\varphi$, in the inertial frame is affected by $\omega_B^B$ as follows.

$$\dot{\varphi} = C_B^T \omega_B^B$$  \hspace{1cm} (2.5)

Here $C_B^T$ is the direction cosine matrix between the body frame and the inertial frame.

2.1.1 Nonlinear Terms

The terms involving quantities labeled SFNL in (2.1–2.3) are called nonlinear scale factor errors. Each of these quantities is a self-heating term contributing a scale factor error due to changes in input power to the torquer driving each gyro. This power is proportional to the square of the current, while the current is proportional to the torque input. The torque is directly proportional to the angular rate. There is an exponential decay with time constant $\tau$ such that SFNL is related to the square of the angular rate as shown in Figure 2-1. Defining $\alpha = 1/\tau$, we obtain the following differential equations for SFNL in each of the three body axes.

$$SFNL_x = -\alpha_x SFNL_x + \alpha_x A_x (\omega_x^B)^2$$  \hspace{1cm} (2.6)
$$SFNL_y = -\alpha_y SFNL_y + \alpha_y A_y (\omega_y^B)^2$$  \hspace{1cm} (2.7)
$$SFNL_z = -\alpha_z SFNL_z + \alpha_z A_z (\omega_z^B)^2$$  \hspace{1cm} (2.8)

Note that in the steady state, $SFNL \rightarrow A(\omega^B)^2$.

At this point we remark that for the IPSRU problem, we desire to estimate the amplitude $A$ and the exponential decay constant $\alpha$, in addition to the nonlinear scale factor SFNL. Because the time constant is unknown in addition to $A$ and SFNL (it is also part of the state vector), its estimation is a nonlinear problem. Murphy [13] has investigated the problem of estimating the nonlinear scale factor in a batch process using an approximate model, treating SFNL as an exponentially decaying quantity with amplitude proportional to the value of the most recent change in $(\omega^B)^2$. This thesis will explore the estimation of these parameters in the model specified using two nonlinear recursive methods. The methods to be studied are the extended Kalman filter, and the maximum likelihood system identification method (see Section 4).
2.1.2 Gravity-dependent Terms

The gravity terms in the body frame can be computed by $g^B = C_B^L g^L$, where $g^L$ is the gravitational acceleration vector in the local level frame, and $C_B^L$ is the direction cosine matrix between the local level frame and the body frame. In a test situation, $C_B^L$ is computed from gimbal angles on a test table as discussed in Section 5.1.1.

The coefficients of the gravity terms are defined for each input axis along with its associated output and spin axes. The output axis is determined as follows.

$$x_o = x_t \times x_s$$

(2.9)

For IPSRU’s Mod II E/S gyro, the spin axis corresponding to the $z$- and $y$-input axes is the $z$-axis (in the body frame), while the spin axis for the $z$-input axis is the $y$-axis. Combining these definitions with the above relation for the output axis, we derive the definitions for the input, spin, and output axes corresponding to each axis in the body frame as listed in Table 2.1.

<table>
<thead>
<tr>
<th>Body axis</th>
<th>Input</th>
<th>Spin</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z$</td>
<td>$i_z$</td>
<td>$i_z$</td>
<td>$-i_y$</td>
</tr>
<tr>
<td>$y$</td>
<td>$i_y$</td>
<td>$i_z$</td>
<td>$i_z$</td>
</tr>
<tr>
<td>$z$</td>
<td>$i_z$</td>
<td>$i_y$</td>
<td>$-i_z$</td>
</tr>
</tbody>
</table>

Table 2.1: Input, spin, and output axes in gyro body frame.

2.2 Linearized Dynamic Equations

In this section we consider the gyro equations in terms of perturbation quantities of the states, $x$, about some nominal point, $x_o$. Note that all of the terms in (2.1–2.3) are already linear in the states. Performing the linearization of (2.6–2.8), we obtain the following.

$$SFNL_z = -\alpha_z x_0 SFNL_z + \alpha_z (\omega_z^B)^2 A_z + \left[-SFNL_z x_0 + A_x (\omega_y^B)^2\right] \alpha_z$$

(2.10)

$$SFNL_y = -\alpha_y x_0 SFNL_y + \alpha_y (\omega_y^B)^2 A_y + \left[-SFNL_y x_0 + A_y (\omega_y^B)^2\right] \alpha_y$$

(2.11)

$$SFNL_z = -\alpha_z x_0 SFNL_z + \alpha_z (\omega_z^B)^2 A_z + \left[-SFNL_z x_0 + A_z (\omega_z^B)^2\right] \alpha_z$$

(2.12)

For the case $x_0 = 0$, we note that the SFNL terms become constant bias states (states having a time derivative equal to zero) in like manner to the other terms in (2.1–2.3).
Chapter 3

Linear Filtering: Applications to IPSRU

This chapter describes the dynamics associated with the IPSRU gyro system and reviews linear filtering theory, applying the results to the IPSRU gyro calibration problem. Section 3.1 presents the system model as described by Musoff [15, 16] and Murphy [13]. Section 3.2 reviews the general Kalman filter equations and derives the relevant matrices for the IPSRU gyro system. Section 3.3 describes the implementation of the linear Kalman filter using the U-D factorization filter [10].

Because we are interested in estimating a number of bias terms which have zero derivatives according to the state equation (3.1), it will also be beneficial to consider the method of separate bias estimation [5, 6]. Section 3.4 reviews this method in combination with the U-D factorization filter implementation of Section 3.3.

3.1 State-space Description of Linear Dynamics

Recall that for a standard linear (time varying) dynamic system with a state variable $x$, a measurement vector $z$, and process and measurement noise terms $w$ and $v$, respectively, the general form of the dynamics is described by the following [7]:

$$\dot{x}(t) = A(t)x(t) + L(t)w(t) \quad (3.1)$$
$$z(t) = H(t)x(t) + v(t) \quad (3.2)$$

The particular Kalman filter algorithm best suited for computer implementation is the discrete Kalman filter developed in [7]. If we adopt the notation $x_k = x(t_k)$ (and similarly for the other variables), where $\{t_k\}$ is a discrete time sequence over which the state and other variables are...
computed, then by integrating (3.1) we obtain [7]

\[ x_{k+1} = \Phi(t_{k+1}, t_k)x_k + \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau)L(\tau)w(\tau) d\tau \]  

(3.3)

The state transition matrix, \( \Phi(t_{k+1}, t_k) \), satisfies

\[ \frac{d}{dt} \Phi(t, \tau) = A(t)\Phi(t, \tau) \]  

(3.4)

with the initial condition \( \Phi(\tau, \tau) = I \).

We shall adopt the following notation to represent the discrete system dynamics.

\[ A_k = \Phi(t_{k+1}, t_k) \]  

(3.5)

\[ L_k w_k = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau)L(\tau)w(\tau) d\tau \]  

(3.6)

If the measurement, \( z \), and measurement noise, \( v \), are also treated as discrete quantities, then the general discrete form of the dynamics is the following.

\[ x_{k+1} = A_k x_k + L_k w_k \]  

(3.7)

\[ z_k = H_k x_k + v_k \]  

(3.8)

If we assume \( \Delta t = t_{k+1} - t_k \) is small in the sense that the relevant matrix and vector quantities can be assumed constant over that interval, then we can compute the limiting value for the state transition matrix.

\[ A_k = I + A(t_k)\Delta t \]  

(3.9)

In this thesis, we use varying definitions for the state vector appropriate to the methods being used. Thus it is helpful now to introduce variables corresponding to the various components of the full state vector. We introduce the attitude errors, \( \xi_1 \), nonlinear scale factors, \( \xi_2 \), gyro bias states, \( \xi_3 \), transfer function amplitudes for SFNL, \( \xi_4 \), and the SFNL exponential decay constants, \( \xi_5 \). The forms of these vectors are expressed below.

\[ \xi_1 = \varphi \]  

(3.10)

\[ \xi_2^T = [ \text{SFNL}_z \quad \text{SFNL}_y \quad \text{SFNL}_x ] \]  

(3.11)

\[ \xi_3^T = [ \xi_{3z}^T \quad \xi_{3y}^T \quad \xi_{3z}^T ] \]  

(3.12)

\[ \xi_4^T = [ A_z \quad A_y \quad A_x ] \]  

(3.13)

\[ \xi_5^T = [ \alpha_z \quad \alpha_y \quad \alpha_z ] \]  

(3.14)
In (3.12),
\[
\xi_{3z}^T = [\text{BIAS}_z, \text{ADI}_z, \text{ADO}_z, \text{ADS}_z, \text{ADSS}_z, \text{ADSI}_z, \text{ADIO}_z, \text{ADSO}_z, \text{SP}_z, \text{SN}_z] \tag{3.15}
\]
\[
\xi_{3y}^T = [\text{BIAS}_y, \text{ADI}_y, \text{ADO}_y, \text{ADS}_y, \text{ADSS}_y, \text{ADSI}_y, \text{ADIO}_y, \text{ADSO}_y, \text{SP}_y, \text{SN}_y] \tag{3.16}
\]
\[
\xi_{3x}^T = [\text{BIAS}_x, \text{ADI}_x, \text{ADO}_x, \text{ADS}_x, \text{ADSS}_x, \text{ADSI}_x, \text{ADIO}_x, \text{ADSO}_x, \text{SP}_x, \text{SN}_x] \tag{3.17}
\]

### 3.1.1 IPSRU Gyro Model

Let us recall the dynamic equations (2.1–2.3) where, in the linearized case, SFNL is treated as a constant bias state. We define

\[
x^T = [\xi_1^T, \xi_3^T] \tag{3.18}
\]

\[
w = n^I = C_B^I n^B \tag{3.19}
\]

where (3.18) makes use of the definitions in (3.10–3.14). We shall not consider the SFNL terms in the linearized analysis, because they appear in the dynamics only as terms having a linear dependence on the linear scale factor errors. In the linear analysis, therefore, we shall estimate these linear scale factors only, assuming SFNL ≡ 0 on each axis.

The system matrices are defined as follows.

\[
A_k = \begin{bmatrix} I & A_{13} \Delta t \\ 0 & I \end{bmatrix} \tag{3.20}
\]

\[
A_{13} = C_B^I \begin{bmatrix} a_x^T \\ a_y^T \end{bmatrix} \tag{3.21}
\]

\[
L_k = \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{3.22}
\]

where

\[
a_x^T = [1, g_y^B, g_y^B, (g_y^B)^2, g_y^B g_y^B, g_y^B g_y^B, g_y^B g_y^B, k_x \omega_x^B, (1 - k_x) \omega_x^B] \tag{3.23}
\]

\[
a_y^T = [1, g_y^B, g_y^B, (g_y^B)^2, g_y^B g_y^B, g_y^B g_y^B, g_y^B g_y^B, k_y \omega_y^B, (1 - k_y) \omega_y^B] \tag{3.24}
\]

\[
a_z^T = [1, g_z^B, g_z^B, (g_z^B)^2, g_z^B g_z^B, g_z^B g_z^B, g_z^B g_z^B, k_z \omega_z^B, (1 - k_z) \omega_z^B] \tag{3.25}
\]

Note that $A$ is time varying if the body is rotating relative to an inertial frame.
3.1.2 IPSRU Measurement Model

In this section we describe two different measurement models for IPSRU. The simpler model, which we use in experimental testing, states simply that the orientation of the gyro body in reference to inertial space is computed using test table gimbal angles and known earth rotation rates, as outlined in Section 5.1.1. The IPSRU gyros indicate the small angular displacements about each of the body axes. Using these displacements we generate a quaternion from the gyro body reference frame (denoted by \( B \)) and a computed inertial frame (denoted by \( C \)). This quaternion, denoted by \( q_B^I \), is initialized at the initial value of \( q_B^I \), the quaternion from the gyro body frame to the true inertial frame \( I \). Note that in the absence of gyro errors we have \( q_B^i = q_B^C \). The computation of \( q_B^C \) is discussed in more detail in Section 5.1.2.

We can compute the small angle error in the inertial frame using the quaternions \( q_B^C \) and \( q_B^I \), both of which are available from the test table software (see Section 5.1). To do this, we form the quaternion product \( q_I = q_B^C q_B^I = q_B^C [q_B^I]^T \) as follows.

\[
q_I = q_B^C q_B^I = q_B^C [q_B^I]^T = \left[ 1 - \frac{1}{4} \begin{bmatrix} \Delta \theta^I \end{bmatrix}^T \begin{bmatrix} \Delta \theta^I \end{bmatrix} \right] \frac{1}{2} \Delta \theta^I \tag{3.26}
\]

We identify the angular displacement in the inertial frame with twice the vector component of the quaternion product \( q_I^C \), and this becomes the measurement of interest.

\[
z_I^I = \Delta \theta^I \tag{3.27}
\]

The question arises of how to incorporate the measurement noise into the measurement model. In Section 3.3.3 we see that it is convenient to assume that the noises associated with each element of the measurement vector are mutually independent, such that the covariance of the discrete measurement noise is a diagonal matrix. Since \( q_B^C \) is computed based on measurements of angular displacements in the body frame, it is more natural to express the measurement in terms of those directly measured quantities. Hence, we write

\[
z_B^I = C_I^B z_I = C_I^B \Delta \theta^I \tag{3.28}
\]

where \( C_I^B \) is the direction cosine matrix representation of \( q_I^B \).

To determine the geometry matrix associated with this measurement, we remark that \( \Delta \theta^I \) is equivalent to \( \varphi \) from (3.10). Therefore,

\[
z_B^I = \begin{bmatrix} C_I^B & 0 \end{bmatrix} x + v_B^I \tag{3.29}
\]

where \( v_B^I \) is a discrete white noise having a diagonal covariance matrix. In other words, the mea-
suremenet geometry matrix is given by the following.

\[
H = \begin{bmatrix}
C^B_I & 0
\end{bmatrix}
\]  

(3.30)

An alternative measurement, which is available for IPSRU in an actual operational situation, is obtained using star sensor line of sight (LOS) information [16]. A star sensor aboard a spacecraft identifies a star within its field of view. The star sensor software then computes a unit vector, in the star sensor reference frame, from the spacecraft to that star. This computation is performed based on the focal plane image location of the star, as explained by Musoff [16]. We obtain the desired measurement by comparing this information with the known LOS vector to the star provided by a star catalog. The star catalog vector (denoted by \( u^S_B \)) is provided in an earth centered inertial frame. We must relate this to the star sensor frame through a direction cosine matrix given by \( C^{SS}_C = C^{SS}_B C^B_C \), where \( C^{SS}_B \) is the (fixed) direction cosine matrix between the star sensor and gyro body frames and \( C^B_C \) is the direction cosine matrix between the gyro body frame and a "computed" inertial frame based on gyroscope readings. The measurement vector is the difference between the LOS unit vector based on focal plane measurements of the observed star \( u^S_S \) and the LOS vector provided by a star catalog. Musoff [16] presents this vector difference in the star sensor frame as follows.

\[
m = u^{SS}_S - u^S_S = C^{SS}_B C^B_C u^I_S - u^{SS}_S
\]  

(3.31)

Note that the direction cosine matrix \( C^B_C \) and the measured LOS vector \( u^S_S \) are computed each time a focal plane measurement is made. The star LOS vector, \( u^I_S \), is recalculated each time a new star is observed.

If we add a measurement noise \( v^{SS} \) to this measurement, the measurement vector, \( z \), becomes

\[
z = m + v = Hx + v = C^{SS}_B C^B_C u^I_S - u^{SS}_S + v^{SS}
\]  

(3.32)

and it remains to derive the matrix \( H \) such that \( Hx = m = C^{SS}_B C^B_C u^I_S - u^{SS}_S \) to first order. Musoff [16] has shown that the LOS unit vector difference can also be written as

\[
m = -\Delta \theta^{SS} \times u^{SS}_S
\]

\[
= [C^{SS}_B C^B_C u^I_S] \times [C^B_C \Delta \theta^I]
\]

\[
= C^{SS}_B C^B_C [u^I_S \times \Delta \theta^I]
\]  

(3.33)

where \( C^B_C \) is the direction cosine matrix between the gyro body frame and the true inertial frame. In a real operational simulation, \( C^B_C \) would be approximated by \( C^B_C \). In (3.33), \( \Delta \theta^I = \varphi \) from (3.10),
which implies

\[ m = \begin{bmatrix} C_B^{S} C_I^{B} [u_S^{I} x] & 0 \end{bmatrix} x \]  

(3.34)

where \([u_S^{I} x]\) is the cross product matrix corresponding to the vector \(u_S^{I}\). Hence,

\[ H = \begin{bmatrix} C_B^{S} C_I^{B} [u_S^{I} x] & 0 \end{bmatrix} \]  

(3.35)

It is shown in Section 3.2.3 that if \( C_B^{S} \) is a fixed orthogonal matrix, then we can eliminate it from (3.35), so that measurements are assumed to be taken directly in the body frame rather than the star sensor frame, with no effect on the results.

In future discussion it will be convenient to use the partition of the measurement geometry matrix which multiplies only the attitude errors in the state vector. We shall denote this matrix \( H_1 \). In the case of direct angular displacement measurements on the test table, \( H_1 = C_B^{I} \). If we use star sensor measurements, then \( H_1 = C_B^{S} C_I^{B} [u_S^{I} x] \) for measurements taken in the star sensor frame or \( H_1 = C_I^{B} [u_S^{I} x] \) for measurements taken in the body frame. In the linear filter case we are considering in this section, therefore, (3.30) and (3.35) reduce to

\[ H = \begin{bmatrix} H_1 & 0 \end{bmatrix} \]  

(3.36)

The numerical results in this thesis are based upon simulated test table measurements only. We would expect calibrations based on star sensor measurements to behave similarly, except that longer tests involving more maneuvers would be required in order to achieve full observability because each star sensor measurement actually provides information about only two independent axes.

### 3.2 Kalman Filter Equations

In this section the standard linear Kalman filter implementation is laid out for the IPSRU. To implement the filter, we must evaluate the initial conditions of the system as well as the state transition and discrete noise covariance matrices required in processing the time propagation and measurement update stages of the filter algorithm. The derivation of the Kalman filter is treated in detail by Gelb [7]; only the results and application to the IPSRU gyro calibration problem are presented here. The application of nonlinear filtering methods to IPSRU is discussed in Section 4.

#### 3.2.1 Initial Conditions

First it is necessary to specify the initial conditions for the experiment. To implement the Kalman filter, we require the initial state estimate, \( \hat{x}_0 \), and an initial covariance matrix, \( P_0 \). Both of these
are defined below in terms of expected values of functions of the initial state, \( x_0 \) [7].

\[
\begin{align*}
\dot{x}_0 & = E\{x(0)\} = E\{x_0\} \\
P_0 & = E\{[x_0 - \dot{x}_0][x_0 - \dot{x}_0]^T\}
\end{align*}
\tag{3.37}
\tag{3.38}
\]

Most of the states in the IPSRU gyro calibration are actually error parameters of the gyro, such as biases, scale factors, and so on. These quantities have an expected value of zero, but in reality they may take on nonzero values. The angle errors are assumed to be zero initially. Thus, the initial condition on the state estimate is simply

\[
\dot{x}_0 = 0
\tag{3.39}
\]

Of course, the actual initial state, \( x_0 \), depends on the particular unit to be calibrated; in computer simulations it can be set to various values for comparative testing. Since we assume that the gyro error terms are all constants, this is the vector that we are trying to estimate by means of the Kalman filter.

The initial covariance is also unknown, but it is necessary to assume a \( P_0 \) for the Kalman filter. Part of the experiment involves selecting an appropriate \( P_0 \) matrix such that good estimation results are obtained. Ideally, this should reflect the true covariances in the system.

### 3.2.2 Time Propagation Equations

If we assume that \( E\{w_k\} = 0 \) (where \( w_k \) represents a discrete process noise), then the time propagation of the state estimate is accomplished by taking expected values of both sides of (3.7) to obtain

\[
\dot{x}_{k+1}^- = A_k \dot{x}_k^+
\tag{3.40}
\]

The superscripts indicate that \( \dot{x}_{k+1}^- = \dot{x}(t_{k+1}^-) \), where \( t_{k+1}^- \) represents the time just before the estimate is updated to account for measurement data (Section 3.2.3). Similarly, \( \dot{x}_k^+ = \dot{x}(t_k^+) \) is evaluated just after the measurement update at time \( t_k \).

To update the covariance matrix, we require statistical information about the discrete process noise, \( w \). One common noise model is discrete white noise [7]. If \( w \) is a discrete white noise sequence, then

\[
E\{w_k w_j^T\} = Q_k \delta_{kj}
\tag{3.41}
\]

where \( \delta_{kj} \) is the Kronecker delta and \( Q_k \) is the covariance matrix for the discrete process noise at time \( t_k \). Since the covariance matrix of the state estimation error is defined as

\[
P_{k+1}^- = E\{[x_{k+1} - \dot{x}_{k+1}^-][x_{k+1} - \dot{x}_{k+1}^-]^T\} = E\{\dot{x}_{k+1}^- \dot{x}_{k+1}^-^T\}
\tag{3.42}
\]
we can substitute (3.7) and (3.41) to obtain [7]

\[
P_{k+1}^+ = A_k E \left\{ \hat{x}_k^+ \hat{x}_k^{+T} \right\} A_k^T + L_k E \left\{ w_k w_k^T \right\} L_k^T = A_k P_k^+ A_k^T + L_k Q_k L_k^T
\] (3.43)

The covariance matrix for the discrete process noise is given by

\[
Q_k = Q_f^I = E \left\{ n_k^B n_k^{B*} \right\} = C_f^B E \left\{ n_k^B n_k^{B*} \right\} C_f^B = C_f^B Q_B^B C_f^B
\] (3.44)

For IPSRU gyro calibration, \(Q_B^B\) has the form indicated in Section 5.2.3, which, being a scalar multiple of the identity matrix, is actually invariant under the transformation (3.44) to the inertial frame.

To implement this in a simulation, we use the definition (3.44) of \(Q\) in (3.43) while computing the process noise sequence by

\[
w_k = \sqrt{Q_k} \zeta_k, \quad \zeta_k \sim N(0, I)
\] (3.45)

where the matrix square root operation is defined such that \(\sqrt{M} \sqrt{M}^T = M\). This can be computed for a positive semi-definite \(M\) using the U-D factorization \(M = U D U^T\), where, in indicial notation, \(D_{ij} = D_{ii} \delta_{ij}\). The matrix square root is then simply \(\sqrt{M} = U \sqrt{D}\), where \((\sqrt{D})_{ij} = \sqrt{D_{ii}} \delta_{ij}\).

### 3.2.3 Measurement Update Equations

To process Kalman filter updates based on measurements, it is necessary to characterize the statistical properties of the discrete measurement noise, as we did with the discrete process noise. We use the discrete Kalman filter formulation for the measurements and model the measurement noise as discrete white noise, the statistical properties for which have the following form.

\[
E\{v_k\} = 0 \quad (3.46)
\]

\[
E\{v_k v_k^T\} = R_k \quad (3.47)
\]

Gelb derives the measurement update equations for the discrete Kalman filter as follows [7].

\[
K_k = P_k^- H_k^T [H_k P_k^- H_k^T + R_k]^{-1}
\] (3.48)

\[
\hat{x}_k^+ = \hat{x}_k^- + K_k [z_k - H_k \hat{x}_k^-]
\] (3.49)

\[
P_k^+ = [I - K_k H_k] P_k^-
\] (3.50)

The superscripts are defined in a manner consistent with Section 3.2.2. In simulating the Kalman filter, the measurement, \(z\), is obtained from (3.29) or (3.32), using the following for the discrete
measurement noise.

\[ v_k = \sqrt{R_k} \zeta_k \quad , \quad \zeta_k \sim N(0, I) \]  (3.51)

The measurement geometry matrix, \( H \), is computed from (3.36).

At this point we remark that, if \( C_{SS} \) is a constant, orthogonal matrix (meaning \( [C_B]^{-1} = [C_B]^T = C_{SS} \)), then the star sensor frame differs from the body frame only by this (constant) rotation, and we may eliminate this from the equations (3.48–3.50). In the body frame,

\[ H_k^B = C_{SS}^B H = \left[ C_B^P [u_S^I \times] \mid 0 \right] \]  (3.52)
\[ v_k^B = C_{SS}^B v_k \]  (3.53)
\[ R_k^B = E \left\{ v_k^B v_k^B^T \right\} = C_{SS}^B R_k C_{SS}^B \]  (3.54)
\[ z_k^B = C_{SS}^B z_k = [C_B^C - C_B^P] u_S^I + v_k^B \]  (3.55)

Substituting these relations into (3.48–3.50) yields

\[ K_k^B = K_k C_{SS}^B P_k^{-1} H_k^B H_k^B^T \left[ H_k^B P_k^{-1} H_k^B^T + R_k^B \right]^{-1} \]  (3.56)
\[ \hat{x}_k^+ = \hat{x}_k^- + K_k^B [z_k^B - H_k^B \hat{x}_k^-] \]  (3.57)
\[ P_k^+ = [I - K_k^B H_k^B] P_k^- \]  (3.58)

### 3.3 U-D Factorization Filter

An efficient implementation of the Kalman filter arises from the decomposition of the covariance matrix into upper triangular and diagonal components, \( U \) and \( D \), respectively, such that \( U \) has diagonal entries equal to unity and \( P = UDUT \). The U-D factorization filter [3, 10, 18] is superior to the traditional Kalman filter implementation in that numerical conditioning is improved [3, 10] and positive-definiteness of the error covariance, \( P \), is ensured. Furthermore, the U-D filter is more efficient for computer implementation in terms of memory usage, although it generally requires a higher operation count which makes the standard Kalman filter more attractive on grounds of computation speed. The U-D filter is faster in general than other factorization schemes, however, and the improved numerical properties justify the increased computational burden.

#### 3.3.1 Initial Conditions

To represent the initial conditions in the U-D factorization framework, we note that it is necessary to decompose \( P_0 \) into its upper triangular and diagonal components, so that \( P_0 = U_0 D_0 U_0^T \). Maybeck [10] offers an efficient algorithm for carrying out the U-D factorization of the covariance
matrix. Writing the matrices \( U = (u_{ij}) \) and \( D = (d_{ij} \delta_{ij}) \) using indicial notation, we compute, for \( j = n, n-1, \ldots, 1, \)

\[
d_{jj} = p_{jj} - \sum_{k=j+1}^{n} d_{kk} u_{jk}^2
\]

(3.59)

\[
u_{ij} = \begin{cases} 
0 & i > j \\
1 & i = j \\
[p_{ij} - \sum_{k=j+1}^{n} d_{kk} u_{ik} u_{jk}] / d_{jj} & \text{for } i = j - 1, j - 2, \ldots, 1 
\end{cases}
\]

(3.60)

We shall see from the time propagation (Section 3.3.2) and measurement update (Section 3.3.3) equations that we need only apply (3.59–3.60) once, at the initial condition. For subsequent computations, we always continue updating \( U \) and \( D \) individually, so it is not necessary to recompute them from \( P \) each time. On the other hand, \( P \) can be computed at any stage we like. The algorithm for computing \( P \) from \( U \) and \( D \) requires much less computation. In indicial notation,

\[
p_{ij} = \begin{cases} 
u_{ij} d_{jj} + \sum_{k=j+1}^{n} u_{ik} d_{kk} u_{jk} & i < j \\
d_{jj} + \sum_{k=j+1}^{n} d_{kk} u_{jk}^2 & i = j \\
p_{ji} & i > j
\end{cases}
\]

(3.61)

### 3.3.2 Time Propagation Equations

To represent the time propagation (3.43) of the covariance matrix in the U-D factorization scheme, we begin by building the following matrices [11].

\[
Y_k = \begin{bmatrix} A_k U_k^+ & L_k \end{bmatrix}
\]

(3.62)

\[
\tilde{D}_k = \begin{bmatrix} D_k^+ & Q_k \end{bmatrix}
\]

(3.63)

It has been pointed out [10, 18] that, using (3.43), we obtain

\[
Y_k \tilde{D}_k Y_k^T = A_k U_k^+ D_k^+ [U_k^+]^T A_k^T + L_k Q_k L_k^T = P_{k+1}^{-1}
\]

(3.64)

Thornton and Bierman [18] have developed an efficient means of propagating the covariance matrix using weighted Gram-Schmidt orthogonalization and factorization. Defining vectors \( w_j \) as

\[
\begin{bmatrix} w_1 & \cdots & w_n \end{bmatrix} = Y_k^T
\]

(3.65)
we compute a $\tilde{D}_k$-orthogonal set of vectors, $\{v_j\}$ as follows, for $j = n, \ldots, 1$.

\begin{equation}
    v_j = w_j - \sum_{l=j+1}^{n} \frac{w_l^T \tilde{D}_k v_l}{v_l^T \tilde{D}_k v_l} v_j
\end{equation}

The $U$ and $D$ components of the covariance matrix are then updated by the following [18]: for

\begin{equation}
    d_{jj}(t_{k+1}^-) = v_j^T \tilde{D}_k v_j
\end{equation}

and, for $i = 1, 2, \ldots, j - 1$,

\begin{equation}
    u_{ij}(t_{k+1}^-) = \frac{w_i^T \tilde{D}_k v_j}{d_{jj}(t_{k+1}^-)}
\end{equation}

The best method for computing the vectors in (3.66) depends on the particular computing environment. In a MATLAB\textsuperscript{1} script, highly efficient matrix computations can often be achieved using MATLAB's built-in functions for sparse matrices. In this case, we seek to minimize the amount of interpreted (M-file) code in the program. Notice that (3.66) is equivalent to

\begin{equation}
    v_j = \left[ I - \sum_{l=j+1}^{n} \frac{v_l v_l^T \tilde{D}_k}{v_l^T \tilde{D}_k v_l} \right] w_j
\end{equation}

This suggests an algorithm for computing the $v_j$ beginning with $v_n = w_n$ and $N_n = I$ and performing the following iteration for $j = n - 1, \ldots, 1$.

\begin{equation}
    N_j = N_{j+1} - \frac{v_{j+1} v_{j+1}^T \tilde{D}_k}{v_{j+1}^T \tilde{D}_k v_{j+1}}
\end{equation}

\begin{equation}
    v_j = N_j w_j
\end{equation}

On the other hand, Maybeck [10] gives a more efficient algorithm, for implementation in a computer language such as C, which computes $U$ and $D$ directly without the intermediate step of determining the $v_j$ vectors. For $j = n, n - 1, \ldots, 1$, perform the following computations.

\begin{equation}
    c = \tilde{D}_k w_j
\end{equation}

\begin{equation}
    d_{jj}(t_{k+1}^-) = w_j^T c
\end{equation}

\begin{equation}
    d = c/d_{jj}(t_{k+1}^-)
\end{equation}

At each value of $j$, perform, for $i = 1, 2, \ldots, j - 1$,

\begin{equation}
    u_{ij}(t_{k+1}^-) = w_i^T d
\end{equation}

\textsuperscript{1}MATLAB is a trademark of The MathWorks, Inc.
3.3.3 Measurement Update Equations

It has been shown [7] that, in updating the state estimate and covariance due to several measurements, the measurements can be processed one at a time. If the covariance matrix, $R_k$, for the discrete measurement noise is diagonal, this amounts simply to taking $h_k^T$ to be a row of $H_k$, $R_k$ to be the corresponding diagonal entry of $R_k$, and $z_k$ the corresponding element of $z_k$. Then (3.48–3.50) reduce to

$K_k = \frac{P_k^- h_k}{h_k^T P_k^- h_k + R_k}$  \hspace{1cm} (3.77)

$P_k^+ = [I - K_k h_k^T] P_k^-$ \hspace{1cm} (3.78)

$\hat{x}_k^+ = \hat{x}_k^- + K_k [z_k - h_k^T \hat{x}_k^-]$ \hspace{1cm} (3.79)

It has been shown that if these computations are performed individually for each measurement, the result agrees with the full application of (3.48–3.50).

To compute these, we follow Maybeck [10] and define

$$f = [U_k^-]^T h_k$$ \hspace{1cm} (3.80)

$$v = D_k^- f$$ \hspace{1cm} (3.81)

Then, defining $a_0 = R_k$, we perform the following computations for $j = 1, 2, \ldots, n$.

$$a_j = a_{j-1} + f_j v_j$$ \hspace{1cm} (3.82)

$$d_{jj}(t_k^+) = d_{jj}(t_k^-) a_{j-1}/a_j$$ \hspace{1cm} (3.83)

$$q_j = v_j$$ \hspace{1cm} (3.84)

$$p = -f_j / a_{j-1}$$ \hspace{1cm} (3.85)

Then for $i = 1, 2, \ldots, j - 1$ at each $j$, compute,

$$\begin{align*}
    u_{ij}(t_k^+) &= u_{ij}(t_k^-) + q_ip \\
    q_i &= q_i + u_{ij}(t_k^-) v_j
\end{align*}$$ \hspace{1cm} (3.86)

(3.87)

Finally, the denominator term from (3.77) must be incorporated, so the vector $K_k$ is computed from the vector $q$ of the $q_j$ as follows.

$$K_k = q/a_n$$ \hspace{1cm} (3.88)
Note from (3.80–3.82) that
\[
a_n = h_k^TP_k^{-1}h_k + R_k
\] (3.89)

### 3.4 Separate Bias Estimation with U-D Factorization Filter

Recall that in the gyro model for IPSRU, we have a large number of terms which we assume constant, yet which we must estimate. Friedland [6] proposes an efficient method for dealing with this special case, known as separate bias estimation. His paper provides both continuous and discrete algorithms, but neither of these makes use of any factorized form of the covariance matrices. His use of the inverse of the covariance matrix on the bias states does not lend itself directly to factorization. Caglayan and Lancraft [5] later provide a nonlinear extension, including a discrete formulation, but again without making use of any factorized form. Maybeck [10], Bierman [3], and Thornton and Bierman [18], on the other hand, give algorithms for the U-D factorization filter, but these are not applied to the separate bias estimation problem. Bierman [4] provides some discussion on a separate bias formulation for the square root information filter (SRIF), but his treatment of the bias states in this case differs substantially from that of Friedland [6], and it is not clear how to apply it to the U-D factorization or to extend his method to the nonlinear case. There is also some treatment of separate bias processing of the time propagation only in the U-D factorization. An extension of Bierman’s discussion to include measurement updates for the separate bias U-D factorization could possibly be accomplished in a manner analogous to his treatment of measurements in the SRIF, but this is not done. Moreover, it is desirable instead to follow Friedland’s [6] implementation of the separate bias estimation more closely, and to apply the U-D factorization within that framework, because then the extension to the separated EKF with U-D factorization is readily obtained by reference to Caglayan and Lancraft [5]. This section reviews the method of separate bias estimation proposed by Friedland and suggests an implementation which allows us to combine separate bias estimation with the U-D factorization filter, thereby retaining the desirable numerical properties associated with the factorization.

We begin by lumping together into a separate bias vector, \(b\), all of the bias states, which are assumed to be constants to be estimated. Consequently, we include in the dynamic state vector, \(x\), only those state variables which vary with time, and we rewrite (3.1–3.2) as follows.

\[
\begin{align*}
x_{k+1} &= A_kx_k + B_kb_k + L_kw_k \quad (3.90) \\
b_{k+1} &= b_k \quad (3.91) \\
z_k &= H_kx_k + C_kb_k + v_k \quad (3.92)
\end{align*}
\]

Consider a partitioning of the standard Kalman filter state vector such that the dynamic states
and the bias terms are ordered in the full state vector, \( \tilde{x} \), as follows.

\[
\tilde{x} = \begin{bmatrix} x^T & b^T \end{bmatrix}^T
\] (3.93)

The covariance matrix, \( \bar{P} \), is partitioned as

\[
\bar{P} = \begin{bmatrix} \bar{P}_{xx} & \bar{P}_{xb} \\ \bar{P}_{bx} & \bar{P}_{bb} \end{bmatrix}
\] (3.94)

Friedland [6] proposes that we compute a “bias free” estimate, \( \hat{x} \), which he shows relates to the true state estimate through

\[
\hat{x} = \tilde{x} + V\hat{b}
\] (3.95)

We shall see below how \( V \) is determined and how it is used in forming the cross-covariance matrix, \( \bar{P}_{zb} \). The idea of introducing the “bias free” estimate is to estimate the state assuming \( b = 0 \) and then to estimate the bias concurrently, but separately.

In the “bias free” estimation, we define the following individual covariance matrices.

\[
P = E \left\{ (\tilde{x} - (x - Vb))(\tilde{x} - (x - Vb))^T \right\}
\] (3.96)

\[
M = E \left\{ (\hat{b} - b)(\hat{b} - b)^T \right\}
\] (3.97)

These are the covariance matrices which are propagated through the separate bias estimation algorithm. To relate these to \( \bar{P} \), we first employ the result of Friedland [6] that

\[
\bar{P}_{xb} = VM
\] (3.98)

Moreover, it is clear that

\[
\bar{P}_{zb} = E \left\{ (\hat{b} - b)(\hat{b} - b)^T \right\} = M
\] (3.99)

Finally, we compute [4]

\[
\bar{P}_{zz} = E \left\{ (\tilde{x} - x)(\tilde{x} - x)^T \right\} = E \left\{ (\tilde{x} + V\hat{b} - x)(\tilde{x} + V\hat{b} - x)^T \right\}
\]

\[
= E \left\{ (\tilde{x} - (x - Vb) + V(\hat{b} - b))(\tilde{x} - (x - Vb) + V(\hat{b} - b))^T \right\}
\]

\[
= E \left\{ (\tilde{x} - (x - Vb))(\tilde{x} - (x - Vb))^T \right\} + V E \left\{ (\hat{b} - b)(\tilde{x} - (x - Vb))^T \right\}
\]

\[
+ E \left\{ (\tilde{x} - (x - Vb))(\hat{b} - b)^T \right\} V^T + V E \left\{ (\hat{b} - b)(\hat{b} - b)^T \right\} V^T
\]

\[
= P + VMV^T
\] (3.100)
where the error in the bias estimate is uncorrelated with that of the bias free state estimate.

While Friedland [6] provides a discrete formulation for the separate bias estimation algorithm, it is not immediately clear from this how to incorporate U-D factorization of the covariance matrices, $P$ and $M$. In the sections which follow, we shall make use of continuous propagation and update equations (including appropriate Ricatti equations) for the separate bias method in order to motivate development of an alternative discrete algorithm for implementing the separate bias U-D factorization filter. Friedland [6] writes the pertinent continuous equations as follows.

\[
\dot{x} = A\hat{x} + PH^T R^{-1} [z - H\hat{x}] \\
\dot{P} = AP + PA^T + LQL^T - PH^T R^{-1} HP \\
\dot{V} = AV + B - PH^T R^{-1} [HV + C] \\
\dot{b} = M[HV + C]^T R^{-1} [z - H\hat{x} - [HV + C]b] \\
\dot{M} = -M[HV + C]^T R^{-1} [HV + C]M
\]

Note in particular from Friedland's Ricatti equations that the measurement updates are done by computing the bias free innovation $z - H\hat{x}$ to update the bias free estimate, and then using that innovation as a measurement for the bias estimate update.

Appendix A provides analytical verification that the separate bias U-D factorization filter yields a solution identical to that of the standard Kalman filter.

3.4.1 Initial Conditions

The quantities in (3.90-3.92) are easily obtained from the quantities determined for the standard Kalman filter in Section 3.1. Thus, taking

\[ x = \xi_1 \ , \ b = \xi_3 \] 

from (3.18), we conclude

\[ A_k = I \]
\[ B_k = A_{13} \Delta t \]
\[ H_k = H_1 \]
\[ C_k = 0 \]
\[ L_k = I \]
and \( w, v, \) and \( z \) are computed as before, from (3.29), (3.32), (3.45), and (3.51). The discrete noise covariances \( Q \) and \( R \) are the same as those defined in Sections 3.2.2 and 3.2.3. Recall the definitions of the matrices \( A_{13} \) and \( H_1 \) from (3.21) and from Section 3.1.2, respectively.

To compute initial conditions for these covariance matrices, we note first that we determined that the initial covariance matrix was diagonal for IPSRU, so \( V_0 = 0 \). This also gives us \( P_0 = \hat{P}_{zz0} \) and \( M_0 = \hat{P}_{bb0} \). We can factorize these block diagonal elements as \( P = U_x D_x U_x^T \) and \( M = U_b D_b U_b^T \), setting \( U_{zz0} = I \), \( D_{zz0} = P_0 \), \( U_{bb0} = I \), and \( D_{bb0} = M_0 \).

3.4.2 Time Propagation Equations

In this section we process the time propagation for the estimates and covariances of the state and bias, as well as for the cross-covariance \( V \). Because the time derivative of \( b \) in (3.91) is identically zero, however, its estimate, \( \hat{b} \), and error covariance, \( M \), remain unchanged during the time propagation stage. Thus, in the notation of the previous sections,

\[
\hat{b}_{k+1}^- = \hat{b}_k^+ = \hat{b}_k \\
M_{k+1}^- = M_k^+ = M_k
\]  

(3.112) (3.113)

Note that (3.101–3.102) are identical to the corresponding equations in the standard Kalman filter [7]. To propagate \( \hat{x} \) and \( V \), therefore, we compute the state transition matrix using (3.9) and apply (3.90) to obtain

\[
\hat{x}_{k+1}^- = A_k \hat{x}_k^+ + B_k \hat{b}_k \\
\hat{x}_{k+1}^- + V_{k+1}^- \hat{b}_k = A_k \left[ \hat{x}_k + V_k^+ \hat{b}_k \right] + B_k \hat{b}_k \\
\hat{x}_{k+1}^- = A_k \hat{x}_k^+ \\
V_{k+1}^- = A_k V_k^+ + B_k
\]

(3.114) (3.115) (3.116) (3.117)

Finally, we note that (3.102) is identical to the Ricatti equation for the state error covariance in the standard Kalman filter [7], so we can propagate \( P \) using (3.43).

\[
P_{k+1}^- = A_k P_k^+ A_k^T + L_k Q_k L_k^T
\]  

(3.118)

To evaluate the propagation using the U-D factorization of \( P \), we simply substitute \( U_x \) for \( U \) and \( D_x \) for \( D \) in Section 3.3.2 and carry out the computations listed there.
3.4.3 Measurement Update Equations

While Friedland [6] gives a discrete formulation of the separate bias estimation, his equations apply to the case in which all measurements are processed at once. To continue using the U-D factorization filter as we have implemented it here, we need the corresponding formulation for the case in which measurements are processed one at a time. This is most readily obtained by analogy using (3.101–3.105) and the results already obtained for the standard U-D factorization filter. We first derive the measurement update equations for the full vector measurement, then explain how to perform the updates for one measurement at a time.

In Section 3.4.2 we saw that the bias free state estimate and error covariance are to be treated in the same manner as the full state estimate and error covariance of the standard Kalman filter. Continuing in this manner, we define the innovation \( r_k = z_k - H_k \hat{x}_k^- \) and apply the standard Kalman filter update equations [7].

\[
K_k = \frac{P_k^- H_k^T [H_k P_k^- H_k^T + R_k]}{1} (3.119)
\]
\[
P_k^+ = [I - K_k H_k] P_k^- (3.120)
\]
\[
\hat{x}_k^+ = \hat{x}_k^- + K_k r_k (3.121)
\]

We update \( V \) by considering the form of the measurement update term in (3.103). The forms of (3.103) and (3.101) suggest that \( V \) should be updated in a manner similar to that for the update of \( \hat{x} \). Defining \( G_k = H_k V_k^- + C_k \), we substitute \(-G\) for \( r \) and \( V \) for \( \hat{x} \) in (3.121) to obtain

\[
V_k^+ = V_k^- - KG_k (3.122)
\]

Caglayan and Lancraft [5] point out that \( r_k \) can be interpreted as a measurement corresponding to the measurement geometry matrix \( G_k \), to be used in updating the bias estimate, and this is also suggested by (3.104). To determine the bias update equations in the discrete case, we will follow the procedure used by Gelb\(^2\) [7] for deriving the measurement update equations for the standard Kalman filter.

We begin, following Gelb [7], by proposing the form of the estimator for \( b \) and determining the conditions under which it is unbiased. Defining \( \delta b = \hat{b} - b \) and \( \delta \hat{x} = \hat{x} - (x - V b) \),

\[
\begin{align*}
\delta \hat{b}_k &= S_k' \delta \hat{b}_{k-1} + S_k r_k = S_k' \delta \hat{b}_{k-1} + S_k [s_k - H_k \hat{x}_k^-] \\
b_k + \delta \hat{b}_k &= S_k' b_k + S_k' \delta \hat{b}_{k-1} + S_k [H_k x_k + C_k b_k + v_k - H_k (x_k - V_k b_k + \delta \hat{x}_k^-)] (3.124)
\end{align*}
\]

\[
\delta \hat{b}_k = [S_k' + S_k G_k - I] b_k + S_k' \delta \hat{b}_{k-1} + S_k [v_k - H_k \delta \hat{x}_k^-] (3.125)
\]

For an unbiased estimator, given zero mean measurement noise, $E\{\delta \hat{b}\} = 0$, and $E\{\delta \hat{x}\} = 0$, we require

$$S'_k = I - S_k G_k$$

(3.126)

To find the minimum variance unbiased estimator, apply (3.125) and (3.126) and find the covariance matrix for $\delta \hat{b}_k$.

$$M_k = E\left\{\delta \hat{b}_k \delta \hat{b}_k^T\right\} =$$

$$= E\left\{(I - S_k G_k) \delta \hat{b}_{k-1} + S_k \left(v_k - H_k \delta \hat{x}_k^-\right)\right\}\left[(I - S_k G_k) \delta \hat{b}_{k-1} + S_k \left(v_k - H_k \delta \hat{x}_k^-\right)^T\right]$$

$$= [I - S_k G_k] E\left\{\delta \hat{b}_{k-1} \delta \hat{b}_{k-1}^T\right\} [I - S_k G_k]^T + [I - S_k G_k] E\left\{\delta \hat{b}_{k-1} \left[v_k - H_k \delta \hat{x}_k^-\right]^T\right\} S_k^T$$

$$+ S_k E\left\{\left[v_k - H_k \delta \hat{x}_k^-\right] \delta \hat{b}_{k-1}^T\right\} [I - S_k G_k] + S_k E\left\{\left[v_k - H_k \delta \hat{x}_k^-\right]^T\right\} S_k^T$$

$$= [I - S_k G_k] M_{k-1} [I - S_k G_k]^T + S_k E\left\{v_k v_k^T\right\} S_k^T + S_k H_k E\left\{\hat{x}_k^- \hat{x}_k^-^T\right\} H_k^T S_k^T$$

$$M_k = [I - S_k G_k] M_{k-1} [I - S_k G_k]^T + S_k \left[H_k P_k^{-1} H_k^T + R_k\right] S_k^T$$

(3.127)

We note that (3.127) is analogous to (4.2-12) from Gelb [7]. At this point, Gelb [7] computes an expression for the Kalman gain matrix that yields the minimum variance estimator. Applying that result to (3.127) yields

$$S_k = M_{k-1} G_k^T \left[G_k M_{k-1} G_k^T + H_k P_k^{-1} H_k^T + R_k\right]^{-1}$$

(3.128)

$$M_k = [I - S_k G_k] M_{k-1}$$

(3.129)

$$\hat{b}_k = \hat{b}_{k-1} + S_k \left[r_k - G_k \hat{b}_{k-1}\right]$$

(3.130)

Now we consider the procedure for processing each measurement one at a time. For a given scalar element $z_k$ of the measurement vector, we extract the corresponding row vector, $h_k^T$, of the measurement geometry matrix $H_k$. The corresponding innovation sequence is computed as follows.

$$r_k = z_k - h_k^T \hat{x}_k^-$$

(3.131)

Substituting $U_x$ for $U$, $D_x$ for $D$, and $\hat{x}$ for $\hat{x}$, we carry out the computations of Section 3.3.3. In so doing, we update $\hat{x}$ and P, and obtain the vector $K_k$, from the following.

$$K_k = \frac{P_k^{-1} h_k}{h_k^T P_k^{-1} h_k + R_k}$$

(3.132)

$$P_k^+ = \left[I - K_k h_k^T\right] P_k^-$$

(3.133)

$$\hat{x}_k^+ = \hat{x}_k^- + K_k r_k$$

(3.134)

To update V, compute the vector $g_k^T = h_k^T V_k^- + c_k^T$ where $c_k^T$ is the corresponding row of $C_k$. 

38
For the scalar measurement under consideration, (3.122) becomes

\[ V_k^+ = V_k^- - K_k g_k^T \]  

(3.135)

Finally, to update \( \breve{b} \) and \( M \) we follow the algorithm of Section 3.3.3, substituting \( U_b \) for \( U \), \( D_b \) for \( D \), \( g \) for \( h \), \( S \) for \( K \), and \( (h^TP^-h + R) \) for \( R \). Recall that the computation of \( h_k^TP_k^-h_k + R_k \) is done automatically in the process of computing the measurement update for the bias-free estimate (3.89). We thereby compute

\[ S_k = \frac{M_k^{-1}g_k}{g_k^TM_k^{-1}g_k + h_k^TP_k^-h_k + R_k} \]  

(3.136)

\[ M_k = (I - S_k g_k^T)M_{k-1} \]  

(3.137)

\[ \breve{b}_k = \breve{b}_{k-1} + S_k \left[ r_k - g_k^T\breve{b}_{k-1} \right] \]  

(3.138)

which is equivalent to (3.128–3.130) for a scalar measurement. We proceed in this manner with (3.131–3.138) sequentially for each measurement. It bears repeating that this assumes the covariance matrix for the discrete measurement noise is diagonal, so that each element of the measurement vector can be treated individually.

Appendix A demonstrates analytically that this algorithm yields results identical to those of the standard Kalman filter.

### 3.4.4 Comparison with Standard U-D Factorization Filter

As shown in Appendix A, the results of the U-D factorization filter with separate bias estimation are equal to those obtained by the standard Kalman filter; they are the optimal estimates and corresponding error covariances. We note, moreover, that the separate bias U-D factorization filter possesses all the desirable numerical qualities of the standard U-D factorization filter. In particular, positive definiteness of the covariance matrices \( P \) and \( M \) is ensured by virtue of their representation in the U-D factorization. Since matrix inversions in the filter algorithm involve only these matrices and the positive definite covariance matrix, \( R \), for the discrete measurement noise, we expect the separate bias U-D factorization filter to have numerical conditioning properties similar to those obtained using the standard U-D factorization filter [10, 4]. Moreover, the separate bias algorithm is preferred because of the significant reduction in computational burden as demonstrated below.

In a memorandum introducing the separate bias U-D factorization filter [12], a comparison between the separate bias U-D factorization filter and the standard U-D factorization filter without separate bias is made by implementing the two filters in MATLAB and comparing the results. MATLAB records the number of floating point operations (FLOPS) required to perform each computation; Table 3.1 lists these for a variety of cases having different values of \( n_x = \text{dim} x \) and
<table>
<thead>
<tr>
<th>Number of states</th>
<th>Propagation FLOPS</th>
<th>Update FLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(per measurement)</td>
</tr>
<tr>
<td></td>
<td>$n_x$</td>
<td>$n_b$</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>24</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>48</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1/2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>48</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 3.1: Comparison of floating point operations for U-D factorization filter with and without separate bias estimation.

$n_b = \dim b$. It is also verified numerically that the results of the two methods are equal, as shown analytically in Appendix A.

In order to make a fair comparison, we use MATLAB’s sparse matrix operations. The primary effect of this is to eliminate unnecessary computations in the filter propagations without separate bias, due to the large 0 and I blocks in the state transition matrix. The operation counts in Table 3.1 represent those obtained using the sparse matrices; if we perform full matrix multiplications, the savings in operation counts are much larger.

As illustrated in Table 3.1, the separate bias U-D factorization filter represents a significant savings in computation time over the standard U-D factorisation filter. The amount of improvement expected using the separate bias formulation increases with the number of bias states relative to the number of dynamic states.
Chapter 4

Nonlinear IPSRU Dynamics

This chapter describes methods for estimating the IPSRU gyro parameters while including the effects of the associated nonlinear dynamics. Two algorithms are applicable to the type of nonlinearity which arises here — the extended Kalman filter [7] (Section 4.1) and the maximum likelihood system identification method [11, 17] (Section 4.2).

4.1 Extended Kalman Filter

A common nonlinear filter is the extended Kalman filter (EKF) [7]. The EKF is implemented by using the full nonlinear differential state equation for propagating and updating the state estimate, while propagating and updating the covariance matrix in the manner of the traditional Kalman filter. Caglayan and Lancraft [5] have developed an algorithm implementing a "separated EKF" which combines the EKF with the separate bias estimation of Friedland [6], discussed in Section 3.4. We shall here consider the latter approach for application to IPSRU.

The full nonlinear differential equation of state and the nonlinear measurement equation are given by the following.

\[
\begin{align*}
\mathbf{x}_{k+1} &= f_k(\mathbf{x}_k, \mathbf{b}_k) + L_k \mathbf{w}_k \\
\mathbf{b}_{k+1} &= \mathbf{b}_k \\
\mathbf{z}_k &= h_k(\mathbf{x}_k, \mathbf{b}_k) + \mathbf{v}_k
\end{align*}
\]

(4.1) (4.2) (4.3)

We must now include in the state and bias vectors the additional elements associated with the nonlinear scale factor which we wish to estimate. Thus,

\[
\mathbf{x}^T = \begin{bmatrix} \xi_1^T & \xi_2^T \end{bmatrix}, \quad \mathbf{b}^T = \begin{bmatrix} \xi_3^T & \xi_4^T & \xi_5^T \end{bmatrix}
\]

(4.4)
Following the derivation of Caglayan and Lancraft [5] (but using the notation of Friedland [6] for naming matrices and our previous indexing notation in order to maintain consistency of notation throughout this text) we compute the following matrix partials and functions corresponding to time $t_k$.

\[
\begin{align*}
A_k &= \frac{\partial f_k(x,b)}{\partial x} = \begin{bmatrix} \dot{x}_k^+ \dot{b}_k^+ \\ \dot{x}_k^- \dot{b}_k^- \end{bmatrix} = \begin{bmatrix} I & A_{12} \Delta t \\ 0 & I + A_{22} \Delta t \end{bmatrix} \\
B_k &= \frac{\partial f_k(x,b)}{\partial b} = \begin{bmatrix} A_{13} \Delta t & 0 \\ 0 & A_{24} \Delta t & A_{25} \Delta t \end{bmatrix} \\
H_k &= \frac{\partial h_k(x,b)}{\partial x} = \begin{bmatrix} H_1 & 0 \end{bmatrix} \\
C_k &= \frac{\partial h_k(x,b)}{\partial b} = 0 \\
u_k &= f_k(\dot{x}_k^+ \dot{b}_k^+) - A_k \dot{\dot{x}}_k^+ - B_k \dot{\dot{b}}_k^+ = \begin{bmatrix} 0 & 0 & 0 & u_x \Delta t \\ 0 & u_y \Delta t \\ 0 & u_z \Delta t \end{bmatrix}^T \\
y_k &= h_k(\dot{x}_k^- \dot{b}_k^-) - H_k \dot{x}_k^- - C_k \dot{b}_k^- = 0
\end{align*}
\]

where

\[
\begin{align*}
A_{12} &= C_B^L \begin{bmatrix} \omega_x^B \\ \omega_y^B \\ \omega_z^B \end{bmatrix} \\
A_{22} &= \begin{bmatrix} -\hat{\alpha}_x \\ -\hat{\alpha}_y \\ -\hat{\alpha}_z \end{bmatrix} \\
A_{24} &= \begin{bmatrix} \hat{\alpha}_x (\omega_x^B)^2 \\ \hat{\alpha}_y (\omega_y^B)^2 \\ \hat{\alpha}_z (\omega_z^B)^2 \end{bmatrix} \\
A_{25} &= \begin{bmatrix} -\text{SFNL}_x + \hat{A}_x (\omega_x^B)^2 \\ -\text{SFNL}_y + \hat{A}_y (\omega_y^B)^2 \\ -\text{SFNL}_z + \hat{A}_z (\omega_z^B)^2 \end{bmatrix} \\
u_i &= -\hat{\alpha}_i \left[ -\text{SFNL}_l + \hat{A}_i (\omega_i^B)^2 \right]
\end{align*}
\]

and $A_{13}$ and $H_1$ are the same as those defined in (3.21) and Section 3.1.2, respectively. Notice that the entries in these matrices depend on the actual estimates (not the bias free estimates) of the states in the nonlinear dynamics of the system.
4.1.1 Time Propagation Equations

Using the above definitions and defining a cross-covariance matrix, $V$, in accordance with (3.95), we obtain the equations for time propagation of the state estimate and of $V$.

\[ \hat{x}_{k+1}^- = A_t \hat{x}_k^+ + B_\hat{b}_k + u_k \] (4.16)

\[ \hat{x}_{k+1}^- + V_{k+1}^- b_k = A_k \left[ \hat{x}_k^+ + V_k^+ \hat{b}_k \right] + B_k \hat{b}_k + u_k \] (4.17)

\[ \hat{b}_{k+1}^- = A_k \hat{b}_k^+ + u_k \] (4.18)

\[ V_{k+1}^- = A_k V_k^+ + B_k \] (4.19)

Notice that (4.19) is identical to (3.117) while (4.18) differs from (3.116) only by the addition of the quantity $u_k$ which, as has been observed by Caglayan and Lancraft [5], depends on both the bias estimate and the state estimate (not bias free). According to Caglayan and Lancraft [5], the covariance propagations are done in the same manner as for the linear case. This last condition also applies to the measurement updates of the covariance matrices, from which we conclude that we may continue to use U-D factorization as before in the EKF.

4.1.2 Measurement Update Equations

For IPSRU we shall use a linear measurement, since we are simply measuring the attitude error about each of the three inertial axes. An examination of the equations of Caglayan and Lancraft [5] will show that the measurement update equations for the separated EKF are the same as those for the linear separate bias estimation (see Section 3.4.3) when the measurement vector is composed of a linear combination of the states (plus a measurement noise). While Caglayan and Lancraft [5] have solved the problem including a nonlinear measurement, we do not require those results here. It is convenient that the IPSRU measurement model is linear, otherwise we would have to evaluate $H_k$ from (4.7) at the estimates $\hat{x}_k^+$ and $\hat{b}_k^+$, update the estimates, and then evaluate $H_k$ at the new, updated estimates, repeating this process for each element in the measurement vector. In the case of a linear measurement model, however, $H_k$ is independent of the current estimates.

4.2 Maximum Likelihood System Identification

So far we have treated the nonlinear dynamics of the IPSRU gyro calibration problem by treating all unknowns to be estimated as states of the system, and using a nonlinear filter when the exponential decay terms appear in the equation of state for the nonlinear scale factors. An alternative representation is to treat $\alpha_x$, $\alpha_y$, and $\alpha_z$ as unknown parameters of a linear system. A procedure for estimating these uncertain parameters is known as system identification [11]. In the method to
be employed in this section, we define a likelihood function, which represents the probability distribution of the states and the measurement history, dependent upon the true values of the uncertain parameters. The parameter estimates are chosen to maximize the value of this likelihood function, and the state estimate and error covariance are evaluated concurrently. This particular approach is known more specifically as maximum likelihood system identification (MLSI) [11] or full information maximum likelihood optimal filtering (FIMLOF) [1, 17].

We assume that the uncertain parameters are approximately constant over an interval of $N_{\text{param}}$ sample periods [11], where $N_{\text{param}}$ is a number selected by the filter designer. We obtain good approximations by making parameter estimate updates only every $N_{\text{param}}$ sample periods, and by using only the measurement history from the most recent $N_{\text{param}}$ periods in each update. This yields substantial reductions in computation time. In Section 5 we compare maximum likelihood system identification with the linear and extended Kalman filters and evaluate its advantages and disadvantages in relation to those other methods.

Maybeck’s treatment of this subject includes a number of approximations which he suggests for the purpose of “attaining online applicability”1 [11]. Some of these approximations are incorporated directly into the summary provided here in the following sections. It turns out that the system identification method using these approximations lends itself nicely to combination with separate bias estimation and U-D factorization, as is demonstrated in Section 4.2.2.

To employ the system identification procedure, we define the state and bias vectors as follows.

$$x^T = \left[ \xi_1^T \xi_2^T \right], \quad b^T = \left[ \xi_3^T \xi_4^T \right]$$ (4.20)

In addition, we define a vector, $a$, containing the unknown parameters; in this case, $a = \xi_5$.

The vectors $z$, $w$, and $v$ are the same as those used in the linear separate bias estimation of Section 3.4. The required system matrices are computed as follows.

$$A_k = \begin{bmatrix} I & A_{12} \Delta t \\ 0 & I + A_{22} \Delta t \end{bmatrix}$$ (4.21)

$$B_k = \begin{bmatrix} A_{13} \Delta t & 0 \\ 0 & A_{24} \Delta t \end{bmatrix}$$ (4.22)

$$H_k = \begin{bmatrix} H_1 \\ 0 \end{bmatrix}$$ (4.23)

$$C_k = 0$$ (4.24)

$$L_k = \begin{bmatrix} I \\ 0 \end{bmatrix}$$ (4.25)

---

1Maybeck, p. 101
Notice that the state transition matrix, \( A_k \), of (4.21) is identical to that used for the EKF (4.5); likewise, the measurement geometry matrix, \( H_k \), of (4.23) is the same as that used for the EKF (4.7).

In this system identification algorithm we will also require the partial derivatives of the state transition matrices with respect to the unknown parameters.

\[
\frac{\partial A_k}{\partial a_i} = \begin{bmatrix}
0 & 0 \\
0 & (\partial A_{22}/\partial a_i) \Delta t
\end{bmatrix}
\]  
\hspace{1cm} (4.26)

\[
\frac{\partial B_k}{\partial a_i} = \begin{bmatrix}
0 & 0 \\
0 & (\partial A_{24}/\partial a_i) \Delta t
\end{bmatrix}
\]  
\hspace{1cm} (4.27)

From (4.12) and (4.13) we compute the necessary partial derivatives, represented in indicial notation as follows.

\[
\left( \frac{\partial A_{22}}{\partial a_p} \right)_{ij} = (- \delta_{ip} \delta_{jp})
\]  
\hspace{1cm} (4.28)

\[
\left( \frac{\partial A_{24}}{\partial a_p} \right)_{ij} = \left( \frac{\omega^p_v}{a} \right)^2 \delta_{ip} \delta_{jp}
\]  
\hspace{1cm} (4.29)

### 4.2.1 System Identification Theory

In this section we follow the discussion of Maybeck [11] detailing the derivation of the required equations for maximum likelihood system identification. In addition to the theory, Maybeck provides some simplifying approximations which reduce both the complexity of the equations and the run time of the algorithm, without significantly compromising accuracy. This discussion introduces those approximations as needed.

We begin with the definition of the following likelihood function [11].

\[
f_{X_k, Z_k|a} = f_{X_k, Z_k, a} \prod_{i=1}^{k} f_{Z_i|Z_{i-1}, a}
\]  
\hspace{1cm} (4.30)

This is the probability density function of the state variable, \( x_k \), and measurement history, \( Z_k \), given the parameter, \( a \). Assuming Gaussian distributions for each of the relevant conditional density functions, we write the following [11].

\[
f_{X_k|Z_k, a}(x_k|z_k, \alpha) = \frac{\exp \left\{ -\frac{1}{2} \left[ x_k - \hat{x}_k^+ \right]^T \left[ P^+_k \right]^{-1} \left[ x_k - \hat{x}_k^+ \right] \right\}}{(2\pi)^{n/2}|P^+_k|^{1/2}}
\]  
\hspace{1cm} (4.31)

\[
f_{Z_i|Z_{i-1}, a}(z_i|z_{i-1}, \alpha) = \frac{\exp \left\{ -\frac{1}{2} \left[ z_i - H_i \hat{x}_i^- \right]^T E_i^{-1} \left[ z_i - H_i \hat{x}_i^- \right] \right\}}{(2\pi)^{m/2}|E_i|^{1/2}}
\]  
\hspace{1cm} (4.32)

where \( E_i = H_i P_i^- H_i^T + R_i, n = \text{dim} x, \text{and} m = \text{dim} z \).
To get the state and parameter estimates, we seek those values which maximize the likelihood function (4.30). It turns out to be easier to work with the natural logarithm [11].

\[
L_k = \ln f_{x_k, z_k | x, z_k | \alpha} (\xi, Z_k | \alpha)
\]

\[
= -\frac{n + km}{2} \ln 2\pi - \frac{1}{2} \ln |P_k^+| - \frac{1}{2} \left[ \xi - \hat{x}_k^+ \right]^T [P_k^+]^{-1} \left[ \xi - \hat{x}_k^+ \right] \\
- \frac{1}{2} \sum_{l=1}^{k} \ln |E_l| - \frac{1}{2} \sum_{l=1}^{k} \left[ \zeta_l - H_l \hat{x}_l^- \right]^T E_l^{-1} \left[ \zeta_l - H_l \hat{x}_l^- \right]
\]  

(4.33)

To find the values of $\xi = x^*(t_k)$ and $\alpha = a^*(t_k)$ which maximize $L_k$, we must take derivatives with respect to these two variables, and set them equal to zero [11]. In the case of the state estimate,

\[
\frac{\partial L_k}{\partial \xi} \bigg|_{\xi = x^*(t_k), \alpha = a^*(t_k)} = - \left[ x^*(t_k) - \hat{x}_k^+ \right] [P_k^+]^{-1} = 0
\]

(4.34)

This yields

\[
x^*(t_k) = \hat{x}_k^+
\]

(4.35)

We compute the optimal state estimates in the same manner as the Kalman filter without system identification, setting $a_k = a^*(t_k)$ at each time interval. Therefore, we may use the same methods for computing the state estimate and covariance that are developed in Section 3, including separate bias estimation and the U-D factorization. It still remains to show that the parameter estimation stage of the algorithm can be included within the framework of the separate bias U-D factorization filter. This is demonstrated in the sequel.

Before considering the derivative of $L_k$ with respect to the unknown parameter, we make use of a simplifying approximation, provided in Maybeck [11], which states that we may consider only the weighted least squares terms in the derivative of the likelihood function. Maybeck points out that the accuracy of the algorithm is not degraded through use of this approximation; in fact, the approximation actually removes a bias associated with some of the neglected terms. Retaining only the weighted least squares terms, we obtain equations of the following form [11] for each element $a_i$ of the parameter vector $a$.

\[
\frac{\partial L_k}{\partial a_i} \bigg|_{\xi = x^*(t_k), \alpha = a^*(t_k)} = \\
\left[ \frac{\partial \hat{x}_k^+}{\partial a_i} \right]^T [P_k^+]^{-1} \left[ x^*(t_k) - \hat{x}_k^+ \right] + \sum_{l=1}^{k} \left[ \frac{\partial \hat{x}_l^-}{\partial a_i} \right]^T H_l^T E_l^{-1} \left[ z_l - H_l \hat{x}_l^- \right] = 0
\]

(4.36)

Maybeck [11] frequently refers to the derivative $[\partial L_k / \partial a]^T$ as the score vector, $s(t_k)$. Substitut-
ing (4.35) into (4.36) yields

\[ s_i(t_k) = \frac{\partial L_k}{\partial a_i} = \sum_{l=1}^{k} \left[ \frac{\partial \hat{x}_i^-}{\partial a_i} \right]^T H_l^T E_i^{-1} r_i = 0 \]  

(4.37)

where \( r_i = e_i - H_i \hat{x}_i^- \). In addition, Maybeck [11] provides the following approximation to the second derivative \( \frac{\partial^2 L_k}{\partial a^2} \).

\[ J_{ij}(t_k) = -\frac{\partial^2 L_k}{\partial a_i \partial a_j} \approx \sum_{l=1}^{k} \left[ \frac{\partial \hat{x}_i^-}{\partial a_i} \right]^T H_l^T E_i^{-1} H_l \left[ \frac{\partial \hat{x}_i^-}{\partial a_j} \right] \]  

(4.38)

We refer to the matrix \( J(t_k) \) as the conditional information matrix[11].

The optimal parameter estimate, \( \hat{a}^*(t_k) \), is acquired using a Newton-Raphson iteration approach. Given an initial estimate, \( \hat{a}_*(t_k) \), the updated estimate, \( \hat{a}^*(t_k) \), is given by

\[ \hat{a}^*(t_k) = \hat{a}_*(t_k) - \left[ \frac{\partial^2 L_k}{\partial a^2} \right]^{-1} \left[ \frac{\partial L_k}{\partial a} \right]^T = \hat{a}_*(t_k) + J^{-1}(t_k)s(t_k) \]  

(4.39)

where \( s(t_k) \) and \( J(t_k) \) are evaluated at \( \hat{a}_*(t_k) \).

As an alternative to basing the parameter estimates on the entire measurement history, a common approach is to consider only the most recent \( N = N_{\text{param}} \) sample periods. Using this approach, the score vector and conditional information matrix are computed as follows.

\[ s_i(t_k) = \sum_{l=k-N+1}^{k} \left[ \frac{\partial \hat{x}_i^-}{\partial a_i} \right]^T H_l^T E_i^{-1} r_i \]  

(4.40)

\[ J_{ij}(t_k) = \sum_{l=k-N+1}^{k} \left[ \frac{\partial \hat{x}_i^-}{\partial a_i} \right]^T H_l^T E_i^{-1} H_l \left[ \frac{\partial \hat{x}_i^-}{\partial a_j} \right] \]  

(4.41)

If we assume that the unknown parameters are approximately constant over \( N \) sample periods, we achieve a significant reduction in computation by processing parameter estimate updates only after every \( N \) sample periods [11].

### 4.2.2 Separate Bias System Identification

By making use of the approximations provided by Maybeck [11], which are discussed in Section 4.2.1, we may readily incorporate separate bias estimation and U-D factorization into the maximum likelihood system identification algorithm. In order to compute the score and conditional information matrix from (4.40-4.41), we must compute the partial derivatives of the state estimate with respect to the unknown parameters. In the full MLSI algorithm, we also require partial derivatives of the covariance matrix in order to compute the derivatives of the likelihood function. In this implementation, however, we employ Maybeck's [11] approximation which neglects derivatives of the covariance...
matrix, leaving only weighted least squares terms in (4.36).

We note that, just as we partitioned the state vector for the separate bias estimation algorithm, we can also partition the partials of the state vector. However, we are not able to compute the partials of the bias free state estimate independently without also requiring partials of the V matrix. Instead, we compute the partials of the dynamic state estimate and of the bias estimate, without separating out the bias free component of the dynamic state. In this section, we detail the system identification algorithm within the separate bias U-D factorization framework.

Initial Conditions

In addition to the initial conditions on state estimates and covariances for the standard filter algorithm, we require initial conditions on the partial derivatives of the states. Maybeck [11] points out that the initial estimates "are considered to be immutable"2 [11], and, therefore, the partials of the state estimates with respect to each element, $a_i$ of the parameter vector, $a$, are given as follows.

$$
\frac{\partial \hat{x}_0}{\partial a_i} = 0, \quad \frac{\partial \hat{b}_0}{\partial a_i} = 0
$$

(4.42)

Note that this does not apply only at the initial time, $t_0$, however. As part of the system identification algorithm, we choose a value $N_{\text{param}}$ such that we perform parameter updates after every $N = N_{\text{param}}$ sample periods. Those updates, as indicated by (4.40–4.41), depend only on the history for the $N$ most recent intervals. Therefore, if we process a parameter update at time $t_k$, the partials of the state estimates at $t_{k+1-N}$ are also equal to zero.

Because the score and conditional information matrix are expressed as sums involving terms from the $N$ most recent time intervals, they are most effectively computed as running sums. Thus, if we perform a parameter update at time $t_k$, we initialize $s(t_{k-N}) = 0$ and $J(t_{k-N}) = 0$ and update the values each time a measurement update is processed. However, as we shall see later in this section, it may be preferable instead to deal directly with the inverse, $J^{-1}$, of the conditional information matrix. In this case, the designer must select an assumed large initial value for $J^{-1}$.

Finally, we must select an initial condition for the optimal parameter estimate, $\hat{a}^*$.

Time Propagation Equations

To process the time propagation of the state estimates and covariance matrices, we follow exactly the same procedure that we used in the separate bias U-D factorization filter (Section 3.4), since we are now dealing with a linear system with uncertain parameters. To propagate the partials of

---

2Maybeck, p. 84

48
the state estimates, recall,

\[
\hat{x}_{k+1}^- = A_k \hat{x}_k + B_k \hat{b}_k
\]  
\[
\hat{b}_{k+1}^- = \hat{b}_k
\]  

Taking the partials of both sides in (4.43-4.44), we obtain

\[
\frac{\partial \hat{x}_{k+1}^+}{\partial a_i} = A_k \frac{\partial \hat{x}_k^+}{\partial a_i} + B_k \frac{\partial \hat{b}_k^+}{\partial a_i}
\]  
\[
\frac{\partial \hat{b}_{k+1}^+}{\partial a_i} = \frac{\partial \hat{b}_k^+}{\partial a_i}
\]  

Note that to carry out this computation, we require the partials of the matrices \( A_k \) and \( B_k \), evaluated at the current parameter estimate. This is typically done most easily by computing off line the analytical forms of these derivatives and evaluating those functions in the computational algorithm, as in (4.28-4.29).

Measurement Update Equations

In the measurement update stage, we process the estimate and covariance matrix updates as before (Section 3.4), processing each measurement separately from the others in series provided the discrete measurement noise covariance is diagonal. To update the partials of the state estimates, we use

\[
\hat{x}_k^+ = \hat{x}_k^- + V_k^+ \hat{b}_k
\]  
\[
= [\hat{x}_k^- + K_k (z_k - h_k^T \hat{x}_k^-)] + [V_k^- - K_k g_k^T] \left[ \hat{b}_k^- + S_k (z_k - h_k^T \hat{x}_k^- - g_k^T \hat{b}_k^-) \right]
\]  
\[
= \hat{x}_k^- + \left[ K_k (1 - g_k^T S_k) + V_k^- S_k \right] [z_k - h_k^T \hat{x}_k^- - c_k^T \hat{b}_k^-]
\]  
\[
= \hat{x}_k^- + \left[ \frac{h_k^T P_k^- h_k + R_k}{g_k^T M_k^- g_k + h_k^T P_k^- h_k + R_k} \right] K_k + V_k^- S_k \left[ z_k - h_k^T \hat{x}_k^- - c_k^T \hat{b}_k^- \right]
\]  
\[
\hat{b}_k^+ = \hat{b}_k^- + S_k [z_k - h_k^T \hat{x}_k^- - c_k^T \hat{b}_k^-]
\]  

Neglecting the partials of the covariance matrix terms, we take the partials of both sides to obtain

\[
\frac{\partial \hat{x}_k^+}{\partial a_i} = \frac{\partial \hat{x}_k^-}{\partial a_i} + \left[ \frac{h_k^T P_k^- h_k + R_k}{g_k^T M_k^- g_k + h_k^T P_k^- h_k + R_k} \right] K_k + V_k^- S_k \left[ -h_k^T \frac{\partial \hat{x}_k^-}{\partial a_i} - c_k^T \frac{\partial \hat{b}_k^-}{\partial a_i} \right]
\]  
\[
\frac{\partial \hat{b}_k^+}{\partial a_i} = \frac{\partial \hat{b}_k^-}{\partial a_i} + S_k \left[ -h_k^T \frac{\partial \hat{x}_k^-}{\partial a_i} - c_k^T \frac{\partial \hat{b}_k^-}{\partial a_i} \right]
\]

We can write the score and conditional information matrix equations (4.40-4.41) in separate bias
form to get the following update equations.

\[
\begin{align*}
T(t) & \leftarrow T(t) + \frac{\left[ z_t - h^T_t \hat{x}_k - g^T_k \hat{b}_k \right] \left[ h^T_t \frac{\partial \hat{x}_k}{\partial a_i} + c^T_k \frac{\partial \hat{b}_k}{\partial a_i} \right]}{g^T_k M_k g_k + h^T_k P_k h_k + R_k} \\
S(t) & \leftarrow S(t) + \frac{\left[ h^T_k \frac{\partial g^T_k}{\partial a_i} + c^T_k \frac{\partial b^T_k}{\partial a_i} \right] \left[ h^T_k \frac{\partial g^T_k}{\partial a_i} + c^T_k \frac{\partial b^T_k}{\partial a_i} \right]}{g^T_k M_k g_k + h^T_k P_k h_k + R_k}
\end{align*}
\] 

(4.51) 

(4.52)

It has been shown already [7, 10], as was discussed in Section 3.3.3, that serial measurement processing yields the same results for the Kalman filter as processing all measurements simultaneously, which we exploited in order to eliminate the need for matrix inversions in the update algorithm. In the MLSI, this is also the case for the partials of \( \hat{x} \) and \( \hat{b} \), the score, and the conditional information matrix.

Parameter Update and Inversion of Conditional Information Matrix

A complexity introduced at the parameter update stage is the need to compute the inverse of the conditional information matrix in (4.39). This is not a serious problem if the number of uncertain parameters is small, but it is still preferable to avoid this step if possible. Maybeck [11] offers the suggestion of simply using a precomputed \( J^{-1} \) throughout, thereby to avoid updating \( J \) altogether. He argues based on prior results [11] that this does not substantially degrade performance of the estimation algorithm. An alternative approach would be to specify an initial value of \( J^{-1} \), and then to update \( J^{-1} \) directly as measurements are processed, rather than updating \( J \) and inverting it when the parameter updates are made. To see how this can be done, define

\[
\begin{align*}
\frac{\partial r'_k}{\partial a_i} & = \left[ h^T_k \frac{\partial \hat{x}_k}{\partial a_i} + c^T_k \frac{\partial \hat{b}_k}{\partial a_i} \right]^{-1} \\
R'_k & = g^T_k M_k g_k + h^T_k P_k h_k + R_k
\end{align*}
\] 

(4.53) 

(4.54)

(The notation has been chosen for consistency with Section 4.2.3.) Let \( \frac{\partial r'_k}{\partial a_i} \) be regarded as the elements of the gradient \( \nabla a r'_k \), and rewrite (4.52) as follows.

\[
J^+_k = J^-_k + \frac{1}{R'_k} [\nabla a r'_k] [\nabla a r'_k]^T
\] 

(4.55)

Applying the Frobenius matrix inversion lemma [2], this is equivalent to

\[
[J^+_k]^{-1} = [J^-_k]^{-1} - [J^-_k]^{-1} [\nabla a r'_k] [\nabla a r'_k]^T [J^-_k]^{-1} (\nabla a r'_k) + R'_k)^{-1} [\nabla a r'_k]^T [J^-_k]^{-1} 
\] 

(4.56)
This can be computed efficiently as follows.

\[ R_k'' = [\nabla_a r_k]\T [J_k]^{-1} [\nabla_a r_k] + R_k'' \]
\[ n_k = \frac{[J_k]^{-1} [\nabla_a r_k]}{R_k''} \]
\[ [J_k]^{-1} = \left[ I - n_k (\nabla_a r_k)^T \right] [J_k]^{-1} \]

Note that (4.57), (4.58), and (4.59) are analogous to (3.89), (3.77), and (3.78), respectively. From this we conclude that we may update \( J^{-1} \) using the covariance matrix update procedure of the U-D factorization filter.

In order to process updates of \( J^{-1} \) directly, we need an appropriate initialization for \( J^{-1}(t_{k-N}) \). We cannot achieve exactly the value of \( J \) from (4.41) in this manner, which requires the noninvertible initial value \( J(t_{k-N}) = 0 \). Instead, we are forced to start with an approximation of \( J^{-1} \). Maybeck [11] points out that it is possible to use a precomputed value of \( J^{-1} \) without significant performance degradation, so it should also be possible to use a precomputed approximation for the initial \( J^{-1} \).

It has been found based upon preliminary simulation trials for the IPSRU gyro calibration problem that reasonable values for the initial covariance matrix elements relating to the \( \alpha \) terms in an EKF are suitable as initial conditions on \( J^{-1} \) in the MLSI as well.

### 4.2.3 Recapitulation of System Identification Algorithm

We now review the system identification algorithm including separate bias estimation and U-D factorization. It is necessary first to specify initial conditions. For the initial conditions on the state estimates and error covariances, we continue using the same values that we used in the standard Kalman filter algorithm. We also select an initial value of the parameter estimate. In Section 4.2.2 it is argued that the appropriate initial conditions of the score and partials are

\[ s(t_0) = 0 \]
\[ \frac{\partial x_0}{\partial a_i} = 0 \]
\[ \frac{\partial b_0}{\partial a_i} = 0 \]

Finally, we choose an initial value for \( J^{-1}(t_0) \) as discussed in Section 4.2.2.

The format of the system identification algorithm is similar to the standard Kalman filter algorithm, with some steps added in order to perform the parameter updates. At each time \( t_k \), we perform a propagation and a measurement update on the state estimates and error covariances. In system identification, the partial derivatives of the state estimates are also propagated and updated. (Note that in the separate bias framework, it is the partial derivative of the actual state estimate,
rather than the bias free estimate, that is of interest.) Finally, the score vector and conditional information matrix are updated in the measurement update stage.

Unlike the state estimates and covariances, the optimal estimates for the uncertain parameters are computed only at a specified interval of every \( N_{\text{param}} \) time steps, where \( N_{\text{param}} \) is a quantity chosen by the designer representing the number of time steps over which the parameter is expected to remain "essentially constant" [11]. Often parameter updates are also processed at every time step during an initial transient period comprising the first \( N_{\text{trans}} \) time steps to provide early coarse correction for an incorrect initial estimate. The equations needed to carry out each step of the system identification algorithm are listed below.

**Time Propagation Equations**

The equations to be used in the time propagation stage are as follows.

\[
\frac{\partial \hat{x}_{k+1}^-}{\partial a_i} = A_k \frac{\partial \hat{x}_k^+}{\partial a_i} + \frac{\partial A_k}{\partial a_i} \left[ \hat{x}_k^+ + V_k^+ \hat{b}_k^+ \right] + B_k \frac{\partial \hat{b}_k^+}{\partial a_i} + \frac{\partial B_k}{\partial a_i} \hat{b}_k^+ \tag{4.63}
\]

\[
\frac{\partial \hat{b}_{k+1}^-}{\partial a_i} = \frac{\partial \hat{b}_k^+}{\partial a_i} \tag{4.64}
\]

\[
\hat{x}_{k+1}^- = A_k \hat{x}_k^+ \tag{4.65}
\]

\[
P_{k+1}^- = A_k P_k^+ A_k^T + L_k Q_k L_k^T \tag{4.66}
\]

\[
V_{k+1}^- = A_k V_k^+ + B_k \tag{4.67}
\]

\[
\hat{b}_{k+1}^- = \hat{b}_k^+ \tag{4.68}
\]

\[
M_{k+1}^- = M_k^+ \tag{4.69}
\]

Propagation of the U-D factorization of the covariance matrix \( P \) via (4.66) is accomplished by the methods outlined in Section 3.3.2.

**Measurement Update Equations**

If the covariance matrix for the discrete measurement noise is diagonal, the measurement updates are processed one at a time using the following equations.

\[
g_k = [V_k^-]^T h_k + c_k \tag{4.70}
\]

\[
r_k = z_k - h_k^T \hat{x}_k^- \tag{4.71}
\]

\[
R'_k = h_k^T P_k^- h_k + R_k \tag{4.72}
\]

\[
K_k = [P_k^- h_k] / R_k \tag{4.73}
\]

\[
P_k^+ = P_k^- - K_k h_k^T P_k^- \tag{4.74}
\]

\[
\hat{x}_k^+ = \hat{x}_k^- + K_k r_k \tag{4.75}
\]
\[ V^+_k = V^-_k - K_k g^T_k \]  
\[ r'_k = r_k - g^T_k \dot{b}_k \]  
\[ R''_k = g^T_k M_k g_k + R'_k \]  
\[ S_k = [M^-_k g_k] / R''_k \]  
\[ M'^+_k = M^-_k - S_k g_k^T M^-_k \]  
\[ \dot{b}^+_k = \dot{b}^-_k + S_k r'_k \]  
\[ \frac{\partial r'_k}{\partial a_i} = -h^T_k \frac{\partial \dot{x}^-_k}{\partial a_i} - c^T_k \frac{\partial \dot{b}^-_k}{\partial a_i} \]  
\[ \frac{\partial \dot{x}^+_k}{\partial a_i} = \frac{\partial \dot{x}^-_k}{\partial a_i} + \left[ R''^T_k K_k + V^-_k S_k \right] \frac{\partial r'_k}{\partial a_i} \]  
\[ \frac{\partial \dot{b}^+_k}{\partial a_i} = \frac{\partial \dot{b}^-_k}{\partial a_i} + S_k \frac{\partial r'_k}{\partial a_i} \]  
\[ s_i(t_k) \leftarrow s_i(t_k) - \frac{r'_k}{R''_k} \frac{\partial r'_k}{\partial a_i} \]  
\[ J_{ij}(t_k) \leftarrow J_{ij}(t_k) + \frac{1}{R''_k} \left[ \frac{\partial r'_k}{\partial a_i} \left( \frac{\partial r'_k}{\partial a_j} \right) \right] \]  

We remark that (4.82-4.83) update the partial derivative of the state estimate \( \dot{x} \), not the bias free estimate. Updates of the U-D factorization of the covariance matrices \( P \) and \( M \) via (4.74) and (4.80), respectively, are accomplished by the methods outlined in Section 3.3.3. The notation is chosen to clarify the similarity between (4.71-4.75) and (4.77-4.81). This similarity proves especially convenient in developing a computer implementation.

Parameter Update Equations

After the propagations and measurement updates are performed at time \( t_k \), a parameter update is processed if either \( t_k \) falls during the initial transient period \( (k < N_{\text{tran}}) \) or the end of the current parameter estimation interval is reached, at which time \( k \equiv 0 \pmod{N_{\text{param}}} \). The prior optimal estimate, \( \hat{\alpha}_*(t_k) \), is the value of the optimal estimate \( \hat{\alpha}_* \), which was computed at the most recent parameter update. The parameter update at \( t_k \) is accomplished as follows.

\[ \hat{\alpha}^*(t_k) = \hat{\alpha}_*(t_k) + J^{-1}(t_k)s(t_k) \]  

If the initial transient period is completed \( (k \geq N_{\text{tran}}) \), it is also necessary to reset the score, the conditional information matrix, and the partial derivatives of the state estimates to their initial conditions after performing the parameter update. This is done because the likelihood function \( L_k \) is assumed to depend only on the most recent \( N_{\text{param}} \) sample periods, as per (4.40-4.41).
Chapter 5

Simulation of IPSRU Gyro Calibration

This chapter presents numerical results for the filter algorithms developed in the previous chapters based on computer simulations of the IPSRU gyroscope calibration test sequence. Section 5.1 describes the IPSRU test table configuration and provides the necessary quaternions and direction cosine matrices to relate the body frame axes with inertial space through test table gimbal angles. Section 5.2 describes the simulated test sequence and reports the methods used in analyzing the results. In Section 5.3 we present and interpret the results of the simulation and analysis. We also investigate the effect of initial condition specification on convergence of the two nonlinear filters in Section 5.4. Section 5.5 is a discussion of these results and an evaluation of the advantages and disadvantages of both nonlinear methods.

5.1 IPSRU Test Table Configuration

The IPSRU test table configuration is shown in Figure 5-1. The angles $\theta_{IG}$, $\theta_{MG}$, and $\theta_{OG}$ are the inner, middle, and outer gimbal angles, respectively. The local level frame is a right handed coordinate frame with the z-axis pointing up, y east, and z north. This information is used in Sections 5.1.1 and 5.1.2, where a transformation relating the IPSRU body frame to the inertial frame is provided.
5.1.1 Computation of Direction Cosine Matrices

First it is necessary to relate the local level and IPSRU body frames through the direction cosine matrix $C_B^L$, computed as follows.

\[
C_B^L = C_{OG}^L C_{MG}^{OG} C^{IG}_{MG} C_B^{IG}
\] (5.1)

\[
C_B^{IG} = \begin{bmatrix}
0 & 1 & 0 \\
-1 & 0 & 0 \\
0 & 0 & 1
\end{bmatrix}
\] (5.2)

\[
C_{IG}^{MG} = \begin{bmatrix}
1 & 0 & 0 \\
0 & C_{IG} & -S_{IG} \\
0 & S_{IG} & C_{IG}
\end{bmatrix}
\] (5.3)

\[
C_{MG}^{OG} = \begin{bmatrix}
C_{MG} & 0 & -S_{MG} \\
0 & 1 & 0 \\
S_{MG} & 0 & C_{MG}
\end{bmatrix}
\] (5.4)

\[
C_{OG}^L = \begin{bmatrix}
1 & 0 & 0 \\
0 & S_{OG} & -C_{OG} \\
0 & C_{OG} & S_{OG}
\end{bmatrix}
\] (5.5)
Here $C_X$ is shorthand for $\cos \theta_X$ and $S_X$ for $\sin \theta_X$.

To compute the effects of earth rotation, we first transform from the local level frame to an earth fixed frame. Figure 5-2 shows how these frames are related through the latitude, $\beta$. The direction cosine matrix for this transformation is

$$
C^L_B = \begin{bmatrix}
S_{IGS_{MG}} & C_{MG} & -C_{IGS_{MG}} \\
-C_{IGS_{MG}} + S_{IGC_{MG}C_{OG}} & -S_{MG}C_{OG} & -S_{IGS_{MG}} - C_{IGC_{MG}C_{OG}} \\
-C_{IGC_{OG}} - S_{IGC_{MG}S_{OG}} & S_{MG}S_{OG} & -S_{IGC_{OG}} + C_{IGC_{MG}S_{OG}}
\end{bmatrix}
$$

(5.6)

Figure 5-2: Relation of local level and earth fixed frames through latitude.

cosine matrix for this transformation is

$$
C^E_L = \begin{bmatrix}
\sin \beta & 0 & \cos \beta \\
0 & 1 & 0 \\
-\cos \beta & 0 & \sin \beta
\end{bmatrix}
$$

(5.7)

In Cambridge, MA, $\beta = 42.364^\circ$.

Finally, we must relate this earth fixed frame to a reference inertial frame. If we define $t = 0$ to be the time at which these two frames coincide, then at time $t$ they are related as follows.

$$
C^I_B = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \omega_x t & -\sin \omega_x t \\
0 & \sin \omega_x t & \cos \omega_x t
\end{bmatrix}
$$

(5.8)
where \( \omega_z \) is the earth rotation rate. Combining all of these transformations we compute

\[
C_I^B = C_E^L C_G^E C_B^L = C_E^L C_G^E C_O^G C_M^G C_I^G
\]

(5.9)

We can compute the effects of local gravity on the IPSRU body using (5.8) since we know the gravity in the local level frame to be

\[
g^L = \begin{bmatrix}
-1 \\
0 \\
0
\end{bmatrix}
\]

(5.10)

\[
g = 9.80665 \frac{m}{s^2}
\]

(5.11)

Thus \( g^B = C_B^L g^L = C_B^L T g \) yields the gravitational accelerations applied in the body frame.

5.1.2 Computation of Quaternions

It will often be more efficient in this application to work with the quaternions relating to the direction cosine matrices computed above. By the standard definition of the quaternion [2], we obtain

\[
q^{I_G} = \begin{bmatrix}
\frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}}
\end{bmatrix}^T
\]

(5.12)

\[
q^{M_G} = \begin{bmatrix}
\cos \frac{1}{2} \theta_M & \sin \frac{1}{2} \theta_M & 0 & 0
\end{bmatrix}^T
\]

(5.13)

\[
q^{O_G} = \begin{bmatrix}
\cos \frac{1}{2} \theta_O & \sin \frac{1}{2} \theta_O & 0 & 0
\end{bmatrix}^T
\]

(5.14)

\[
q^{O_G} = \begin{bmatrix}
\cos \frac{1}{2} (\theta_O - \frac{\pi}{2}) & -\sin \frac{1}{2} (\theta_O - \frac{\pi}{2}) & 0 & 0
\end{bmatrix}^T
\]

(5.15)

\[
q^E = \begin{bmatrix}
\cos \frac{1}{2} (\beta - \frac{\pi}{2}) & 0 & -\sin \frac{1}{2} (\beta - \frac{\pi}{2}) & 0
\end{bmatrix}^T
\]

(5.16)

\[
q_I^E = \begin{bmatrix}
\cos \frac{1}{2} \omega_t & \sin \frac{1}{2} \omega_t & 0 & 0
\end{bmatrix}^T
\]

(5.17)

To represent the gyro errors in order to obtain the test measurements described in Section 3.1.2, we must compute the quaternion from the body frame to a computed inertial frame, \( q_B^G \). In an experiment, this quaternion is computed by the system software using the angular displacements in the body frame, \( \Delta \theta_B \), measured directly from the gyros under test.

In a simulation, \( q_B^G \) is computed based on an approximation to first order in \( \Delta \theta_B \). The angular displacements are simulated by combining angular displacements due to the command rate, the gyro error rate, and the earth rotation rate. The quaternion \( q_B^G \) is computed by setting the initial value \( q_B^G(t_0) = q_B^I(t_0) \) and then updating \( q_B^G \) after each sample period in which new angular displacement
information is measured. We represent the rotation in the body frame as follows.

$$\Delta q = \left[ 1 - \frac{1}{4} [\Delta \theta^B]^T [\Delta \theta^B] \right]^T$$

We update $q_B^C$ using this rotation in the body frame as follows.

$$q_B^C \leftarrow q_B^C [\Delta q]$$

The product symbol represents quaternion multiplication [2]. In the absence of gyro errors, we have $q_B^C = q_B^I$, modulo an error due to the use of the small angle approximation.

5.2 Description of Simulation

In the remaining sections of this chapter we consider the results of a simulated gyro calibration test using the linear Kalman filter and the two nonlinear filter algorithms. The simulated test sequence consists of two parts — a static calibration and a dynamic calibration. In an actual test situation, the static calibration is performed first to estimate the static gyro error parameters. These include the drift bias and the gravity-sensitive terms. Then the dynamic calibration is performed, using the estimates from the static calibration as initial conditions on the static terms. The dynamic calibration is used to estimate the gyro scale factors. In the IPSRU model, this includes both a linear scale factor (a constant which multiplies the body rotation rate) and a nonlinear scale factor (an exponentially decaying quantity with amplitude proportional to the square of the rotation rate), as detailed in Section 2.1.

We carry out the simulation as follows. First, we run the static test simulation and estimate the static gyro error terms using three filter algorithms — the Kalman filter, extended Kalman filter, and maximum likelihood system identification method. We compare these three methods for estimation accuracy on the static terms and select the results of the best one. The standard deviations of each of the static gyro error terms are computed at the end of the test by means of a Monte Carlo analysis of the results, as discussed in Section 5.2.4. We use these quantities as the initial standard deviations for the dynamic test, and the initial estimates are set to the true values listed in Section 5.2.3. Finally, we run the dynamic test and estimate the dynamic gyro error terms using each of the three filter algorithms. We compute statistical properties of the results by the Monte Carlo method and compare the performance of each algorithm.

The actual test sequences are described in Sections 5.2.1 and 5.2.2.
5.2.1 Simulated Static Calibration Sequence

The simulation sequence begins with a static calibration intended to ascertain the values of the static gyro errors — the bias and gravity-sensitive terms. In a real experimental test, the static calibration is performed by moving the test table through a sequence of positions, holding at each one for a prescribed interval while collecting angular displacement data. This is easily simulated given the assumed true values for the gyro error states (see Section 5.2.3). Murphy [14] has computed the optimal positions for the static calibration. A test sequence using those table angles is plotted in Figure 5-3. The simulated results are analyzed to determine the standard deviations on each static variable which would result in a real test. These values are used as the initial conditions on the standard deviations of the static variables in the dynamic test described below. Additionally, the true values of the static variables are used for the initial estimates in the dynamic simulation.

5.2.2 Simulated Dynamic Calibration Sequence

The dynamic calibration is simulated using the static calibration results as indicated in Section 5.2.1 and the initial conditions on the dynamic gyro errors specified in Section 5.2.3. The test consists of three sequences of rotations, one each about the z-, y-, and z-axes, in order. The rotation rate is varied as shown in Figure 5-4 to permit observability of both the linear and nonlinear scale factors. Table 5.1 lists the inner and middle gimbal angles for each rotation sequence. The prescribed axis is pointed up, and the outer gimbal is used to produce the desired rotations. Because the dynamic
Body Rotation Rates

Figure 5-4: Rotation rate sequence for dynamic calibration test sequence, simulated about each of the x-, y-, and z-axes, in order.

Table 5.1: Inner and middle gimbal angles for each dynamic calibration sequence.
sequence is designed to estimate the dynamic gyro errors \((SP, SN, A, \alpha)\) about one axis, the estimates and standard deviations of these quantities at the end of the sequence are saved to be used as the initial conditions for the subsequent sequence. Initial conditions on all other parameters remain unchanged. In this way the dynamic gyro errors on all three axes are estimated upon completion of the dynamic calibration.

5.2.3 True Values for Simulation

In this simulation, we select values for the (assumed) true static error states which are close to typical values for the real IPSRU system. Over the course of the research involved in this thesis, experimental data from several static calibrations of the IPSRU gyros were collected and analyzed. True values will, of course, vary among actual units, but some typical values are listed in Table 5.2. These quantities were used as the true static gyro errors in the simulation. The initial estimates for the filter were set equal to zero, with initial standard deviations of 0.1 deg/hr on BIAS for each axis, 0.1 \((\text{deg/hr})/g\) for the \(g\)-sensitive terms, and 0.1 \((\text{deg/hr})/g^2\) for the \(g^2\)-sensitive terms.

In a similar manner, typical true dynamic gyro error states were selected for each axis. Quantities related to the nonlinear scale factor were chosen based on previous results of Murphy [13] and Musoff [15]. The true values assumed are listed in Table 5.3. These quantities were used as the true dynamic gyro errors in the simulations. The initial estimates for SP and SN in the filter were set equal to zero, with initial standard deviations of 100 ppm. For the nonlinear scale factor error term,
the amplitude $A$ was set to an initial estimate value of 50 ppm/(deg/s)$^2$ with a standard deviation of 30 ppm/(deg/s)$^2$. The initial estimate of $\alpha$ was chosen to be 0.02 s$^{-1}$ (corresponding to a 50 s decay constant) with a standard deviation of 0.005 s$^{-1}$. Finally, the true values and initial filter estimates of the attitude error and SFNL states were set to zero, with standard deviations of 1 mR and 100 ppm, respectively.

Reasonable values for the discrete noise covariances are also required. Suitable quantities were selected based on [15]. For a simulation sampling at 6 Hz, the following are typical noise covariances for the gyro calibration problem.

\[
Q_k = \begin{bmatrix} 5.0 \times 10^{-13} \left(\frac{\text{rad}}{\text{s}}\right)^2 \end{bmatrix} \text{I} \tag{5.20}
\]

\[
R_k = \begin{bmatrix} 1.5 \times 10^{-10} \left(\text{rad}\right)^2 \end{bmatrix} \text{I} \tag{5.21}
\]

For the maximum likelihood system identification method, a value for $N_{\text{param}}$, the number of sample intervals between parameter updates, must be specified. The value selected for the analyses in Sections 5.3.1 and 5.3.2 is $N_{\text{param}} = 15$. This value resulted in low estimation errors in both $A$ and $\alpha$ in preliminary dynamic simulations about a single axis. Section 5.3.3 reports the effects of varying $N_{\text{param}}$ on the MLSI results.

### 5.2.4 Monte Carlo Analysis

Having collected the results of several simulations for each method, we analyze the statistical properties of the estimation results using the Monte Carlo analysis. The simulation is run several times using different random sequences for the process and measurement noises. The mean value and standard deviation of each state estimation error at each time interval are estimated based on the simulation results. Hence, given the estimation errors of an element $z_i$ of the state vector at time $t_k$ for $N$ Monte Carlo trials, we estimate the mean and standard deviation associated with that state estimation error as follows.

\[
\hat{\mu}_i^N(t_k) = \frac{\sum_{j=1}^{N} \tilde{z}_i^j(t_k)}{N} \tag{5.22}
\]

\[
\hat{\sigma}_i^N(t_k) = \sqrt{\frac{\sum_{j=1}^{N} \left[\tilde{z}_i^j(t_k)\right]^2 - N \left[\hat{\mu}_i^N(t_k)\right]^2}{N - 1}} \tag{5.23}
\]

where $\hat{\mu}_i^N$ and $\hat{\sigma}_i^N$ denote the statistical quantities after $N$ Monte Carlo trials and $\tilde{z}_i^j$ is the estimation error of $z_i$ corresponding to the $j$th trial. The estimation error is simply the difference between the state estimate and the true state.
5.3 Performance Evaluation

This section evaluates the results of a simulation applied to a sample system according to the procedure described in Section 5.2.

5.3.1 Static Calibration

Table 5.4 shows the results of the static calibration analyzed using the linear Kalman filter. The

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\bar{\mu}^N$</th>
<th>$\sigma^N/\sqrt{N}$</th>
<th>$\sigma^N$</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIAS</td>
<td>-0.0016</td>
<td>0.0016</td>
<td>0.0071</td>
<td>deg/hr</td>
</tr>
<tr>
<td>ADI</td>
<td>-0.0049</td>
<td>0.0011</td>
<td>0.0048</td>
<td>(deg/hr)/g</td>
</tr>
<tr>
<td>ADO</td>
<td>-0.0064</td>
<td>0.0017</td>
<td>0.0035</td>
<td>(deg/hr)/g</td>
</tr>
<tr>
<td>ADS</td>
<td>-0.0008</td>
<td>0.0000</td>
<td>0.0080</td>
<td>(deg/hr/g)</td>
</tr>
<tr>
<td>ADSS</td>
<td>-0.0067</td>
<td>0.0030</td>
<td>0.0149</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td>ADSI</td>
<td>-0.0191</td>
<td>0.0022</td>
<td>0.0158</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td>ADIO</td>
<td>-0.0043</td>
<td>0.0036</td>
<td>0.0046</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td>ADSO</td>
<td>-0.0124</td>
<td>0.0030</td>
<td>0.0035</td>
<td>(deg/hr)/g^2</td>
</tr>
</tbody>
</table>

Table 5.4: Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of linear Kalman filter results with $N = 20$.

The standard deviation of the mean is included to demonstrate that the results of the static calibration processed using the linear Kalman filter exhibit a bias in some of the variables. In particular, the mean values of $\text{BIAS}_y$, $\text{ADI}_y$, $\text{ADO}_x$, $\text{ADSI}_y$, $\text{ADIO}_x$, and $\text{ADSO}_y$ all lie outside the interval $[-3\sigma^N/\sqrt{N}, 3\sigma^N/\sqrt{N}]$. It is to be expected that a bias in the estimates might result due to the nonlinear dynamics (unmodeled in the Kalman filter) which influence the system each time the gimbal angles change to move the body to a new orientation.

To check this, the same calibration was processed using the extended Kalman filter, with the initial conditions on the dynamic terms listed in Section 5.2.3. The results of that test appear in
Table 5.5. In the EKF results, only ADS lies slightly outside the interval $[-3\sigma_N/\sqrt{N}, 3\sigma_N/\sqrt{N}]$.

<table>
<thead>
<tr>
<th>Variable</th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIAS</td>
<td>$\mu_N$</td>
<td>0.0023</td>
<td>-0.0016</td>
<td>0.0022</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0011</td>
<td>0.0015</td>
<td>0.0012</td>
</tr>
<tr>
<td>ADI</td>
<td>$\mu_N$</td>
<td>-0.0014</td>
<td>0.0029</td>
<td>-0.0006</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0010</td>
<td>0.0012</td>
<td>0.0016</td>
</tr>
<tr>
<td>ADO</td>
<td>$\mu_N$</td>
<td>0.0011</td>
<td>-0.0037</td>
<td>0.0020</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0011</td>
<td>0.0017</td>
<td>0.0019</td>
</tr>
<tr>
<td>ADS</td>
<td>$\mu_N$</td>
<td>0.0061</td>
<td>-0.0016</td>
<td>-0.0025</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0018</td>
<td>0.0017</td>
<td>0.0013</td>
</tr>
<tr>
<td>ADSS</td>
<td>$\mu_N$</td>
<td>-0.0024</td>
<td>0.0008</td>
<td>-0.0021</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0029</td>
<td>0.0033</td>
<td>0.0022</td>
</tr>
<tr>
<td>ADSI</td>
<td>$\mu_N$</td>
<td>0.0012</td>
<td>-0.0049</td>
<td>-0.0009</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0042</td>
<td>0.0032</td>
<td>0.0032</td>
</tr>
<tr>
<td>ADIO</td>
<td>$\mu_N$</td>
<td>-0.0029</td>
<td>0.0040</td>
<td>0.0023</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0046</td>
<td>0.0036</td>
<td>0.0028</td>
</tr>
<tr>
<td>ADSO</td>
<td>$\mu_N$</td>
<td>-0.0094</td>
<td>0.0069</td>
<td>-0.0008</td>
</tr>
<tr>
<td></td>
<td>$\sigma_N$</td>
<td>0.0038</td>
<td>0.0038</td>
<td>0.0046</td>
</tr>
</tbody>
</table>

Table 5.5: Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of EKF results with $N = 20$.

and only BIAS, ADI, and ADO lie outside the region $[-2\sigma_N/\sqrt{N}, 2\sigma_N/\sqrt{N}]$. Hence including the nonlinear dynamics in the filter model has improved the estimates of the static gyro error terms. Errors are not completely eliminated, however, primarily because the static test provides little observability on the nonlinear scale factor terms. This results in errors in the estimates of those terms, which consequently influence the static gyro error terms.

For completeness we present in Table 5.6 the results of the static calibration using the maximum likelihood system identification method. The mean values of the variables BIAS, ADI, and ADSS all lie outside the interval $[-3\sigma_N/\sqrt{N}, 3\sigma_N/\sqrt{N}]$, indicating a minor improvement over the Kalman filter for processing the static calibration test. Again, errors are introduced by a lack of observability on the nonlinear terms.

In the static calibration, the EKF is the most likely of these three methods to yield estimation errors within two or three standard deviations of the corresponding means. However, we are primarily interested in the dynamic calibration, in which the nonlinear parameters are more effectively estimated. In Section 5.3.2 we consider dynamic calibration performance of the two nonlinear es-
Table 5.6: Mean, standard deviation of the mean, and standard deviation of static estimation errors. Monte Carlo analysis of MLSI results with $N = 20$.

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\hat{\mu}_N$</th>
<th>$\hat{\sigma}_N/\sqrt{N}$</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIAS</td>
<td>0.0016</td>
<td>0.00051</td>
<td>deg/hr</td>
</tr>
<tr>
<td></td>
<td>0.0016</td>
<td>0.0051</td>
<td></td>
</tr>
<tr>
<td>ADI</td>
<td>-0.0013</td>
<td>0.0052</td>
<td>(deg/hr)/g</td>
</tr>
<tr>
<td></td>
<td>0.0016</td>
<td>0.0012</td>
<td></td>
</tr>
<tr>
<td>ADS</td>
<td>0.0012</td>
<td>0.00054</td>
<td>(deg/hr)/g</td>
</tr>
<tr>
<td></td>
<td>0.0054</td>
<td>0.0013</td>
<td></td>
</tr>
<tr>
<td>ADSS</td>
<td>-0.0004</td>
<td>0.0013</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td></td>
<td>0.0041</td>
<td>0.0015</td>
<td></td>
</tr>
<tr>
<td>ADSI</td>
<td>0.0025</td>
<td>0.00067</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td></td>
<td>0.0056</td>
<td>0.0015</td>
<td></td>
</tr>
<tr>
<td>ADIO</td>
<td>-0.0005</td>
<td>0.0011</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td></td>
<td>0.0027</td>
<td>0.0013</td>
<td></td>
</tr>
<tr>
<td>ADSO</td>
<td>0.0004</td>
<td>0.0005</td>
<td>(deg/hr)/g^2</td>
</tr>
<tr>
<td></td>
<td>0.0024</td>
<td>0.0013</td>
<td></td>
</tr>
</tbody>
</table>

Typical filter run times for the static calibration are as follows: for the linear Kalman filter, 243 seconds CPU time; for the EKF, 409 seconds CPU time; and for the MLSI, 403 seconds CPU time. These times are for a Sun Microsystems Sparc IPX.

5.3.2 Dynamic Calibration

In selecting the initial conditions for the dynamic test simulation, the true values of the static gyro errors are used for their initial state estimates. The initial state covariance was chosen to be a diagonal matrix with standard deviations on the static states equal to the corresponding values of $\hat{\sigma}_N$ from Table 5.5. The initial conditions on the remaining states were chosen according to Section 5.2.3.

The results of the linear Kalman filter applied to the dynamic calibration test appear in Table 5.7. In this example, the nonlinear dynamics are included in the simulation but are not modeled by the
The linear Kalman filter. There is clearly a strong bias in the estimate of the linear scale factor, and this bias is the effect of the unmodeled nonlinear scale factor. Moreover, since the nonlinear scale factor changes over time, other gyro error estimates are affected. In this case there is also a clear bias on the quantities $\text{BIAS}_x$ and $\text{ADI}_x$. Recall from Section 5.2.2 that the estimates of the static gyro errors are ignored because the initial conditions on these terms are reset at the beginning of the sequence on each axis (so we would expect similar results for $\text{BIAS}_z$ and $\text{ADI}_z$, and for $\text{BIAS}_y$ and $\text{ADI}_y$, prior to the corresponding resets). Therefore, the values for the static gyro errors in Table 5.7 should not be understood as the actual optimal estimates. Rather, they merely reflect the unmodeled dynamics. The actual estimates of the static gyro errors are unimportant because we assume these have been measured to sufficient accuracy in the static calibration test.

The estimation errors in $\text{ADO}$, $\text{ADSI}$, $\text{ADIO}$, and $\text{ADSO}$ are all equal to zero to the number of decimal places listed in Table 5.7. This is also true of the results from the EKF and the MLSI. Therefore, results for those quantities are not listed. The estimates in the static variables are initialized to their true values at the beginning of each dynamic calibration sequence. Hence the errors in static variables listed in Table 5.7 arise only from the final dynamic sequence, with the $z$-axis pointed up and with rotations about the $z$-axis only. Therefore, the gravity components in the $z$- and $y$-axes are both equal to zero in this sequence, and any variables which multiply $g_z$ or $g_y$ in the gyro error rate equations are not observable. Note that $\text{ADI}_x$, $\text{ADI}_y$, $\text{ADS}_x$, and $\text{ADSS}_x$ also have zero error, in agreement with the above.

<table>
<thead>
<tr>
<th>Variable</th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu^N$</td>
<td>0.0005</td>
<td>-0.0011</td>
<td>-0.0635</td>
<td>deg/hr</td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0003</td>
<td>0.0007</td>
<td>0.0006</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.0016</td>
<td>0.0030</td>
<td>0.0028</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1050</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>-0.0012</td>
<td>0.0014</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0009</td>
<td>0.0009</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.0041</td>
<td>0.0039</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0012</td>
<td>-0.0052</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0047</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.031</td>
<td>-0.0052</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.0024</td>
<td>0.0033</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.0108</td>
<td>0.0149</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>202.0550</td>
<td>255.0209</td>
<td>306.0585</td>
<td>ppm</td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>0.1845</td>
<td>0.2319</td>
<td>0.2336</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>0.2502</td>
<td>0.2342</td>
<td>0.2114</td>
<td>ppm</td>
</tr>
<tr>
<td>$\mu^N$</td>
<td>179.0193</td>
<td>235.7040</td>
<td>278.1124</td>
<td></td>
</tr>
<tr>
<td>$\hat{\sigma}^N/\sqrt{N}$</td>
<td>1.1190</td>
<td>1.0475</td>
<td>0.9455</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.7: Static and dynamic estimation errors. Monte Carlo analysis of linear Kalman filter dynamic calibration results with $N = 20$. 
The primary focus of this simulation is to compare the EKF and the MLSI for estimation of the terms associated with the nonlinear scale factor. The Monte Carlo results of the dynamic calibration are presented in Table 5.8 for the EKF and Table 5.9 for the MLSI.

### Table 5.8: Static and dynamic estimation errors. Monte Carlo analysis of EKF dynamic calibration results with \( N = 20 \).

<table>
<thead>
<tr>
<th>Variable</th>
<th>( \hat{\mu}^N / \sqrt{N} )</th>
<th>( \hat{\sigma}^N / \sqrt{N} )</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIAS</td>
<td>0.0002</td>
<td>0.0003</td>
<td>deg/hr</td>
</tr>
<tr>
<td>ADI</td>
<td>0.0015</td>
<td>0.0024</td>
<td></td>
</tr>
<tr>
<td>ADS</td>
<td>-0.0004</td>
<td>0.0000</td>
<td></td>
</tr>
<tr>
<td>ADSS</td>
<td>0.0011</td>
<td>-0.0012</td>
<td></td>
</tr>
<tr>
<td>SP</td>
<td>5.0103</td>
<td>-0.6457</td>
<td>ppm</td>
</tr>
<tr>
<td>SN</td>
<td>3.3919</td>
<td>1.5244</td>
<td>ppm</td>
</tr>
<tr>
<td>A</td>
<td>-0.5538</td>
<td>-0.3584</td>
<td>ppm/(deg/s)^2</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>0.0585</td>
<td>-0.0212</td>
<td>( \times 10^{-3} ) s^{-1}</td>
</tr>
</tbody>
</table>

First we note that the EKF and MLSI perform about equally with respect to the estimation of the static gyro errors in the dynamic calibration. In both cases, none of the mean estimation errors lies outside the corresponding \([-2\hat{\sigma}^N / \sqrt{N}, 2\hat{\sigma}^N / \sqrt{N}]\) interval. This suggests that if the static gyro parameters are adequately estimated in the static calibration, no bias errors in these quantities arise during the dynamic calibration when either the EKF or MLSI is used as the estimation algorithm.

The EKF estimates of the linear scale factors (SP and SN) are unbiased in the sense that the mean estimation errors lie within the interval \([-2\hat{\sigma}^N / \sqrt{N}, 2\hat{\sigma}^N / \sqrt{N}]\). In the MLSI this is also the case, except for SP\(_z\) and SN\(_x\), which have large estimation errors compared with the standard deviation of the mean, indicating a bias. This, and the quite low standard deviations about the z-axis compared with those about the z- and y-axes in both sets of results, indicate that the ordering of the actual dynamic calibration affects the outcome of the results of a nonlinear filter. This principle was verified using some additional informal simulations in which the ordering of the calibration was...
Table 5.9: Static and dynamic estimation errors. Monte Carlo analysis of MLSI dynamic calibration results with $N = 20$.

Altered and the last rotation axis in the sequence exhibited the lowest standard deviations; those results are not detailed here. The standard deviations of the estimation errors of SP and SN do not depend substantially on whether the EKF or MLSI is used.

A study of the results corresponding to the nonlinear terms reveals that there is a clear negative bias on the estimation error of the amplitude $A$, but that the mean estimation errors for the decay constant $\alpha$ all lie within the interval $[-3\hat{\sigma}^N/\sqrt{N}, 3\hat{\sigma}^N/\sqrt{N}]$. The bias on $A$ is probably related to estimation errors in the other dynamic gyro error states. In particular, a small estimation error in $\alpha$ may be balanced by a large estimation error in $A$ due to the exponential nature of the SFNL dynamics. Additionally, errors in SP and SN may be interpreted in part by the filter as errors in SFNL, especially after a long time when SFNL has decayed to a small value relative to the linear scale factor error. In both of these cases, a fairly large estimation error in $A$ might be balanced by a minute error in one of the other variables, such that the filter sees only small residuals despite the possibly large state estimation errors. This condition is unlikely to arise in a linear filter when the covariances are selected properly, but it is not surprising that it should arise when nonlinear dynamics are involved.
We remark at this point that the bias in \( A \) probably depends to a great extent on the initial conditions supplied to the filter, so that other conditions might produce bias with larger or smaller values or perhaps with the opposite sign. Still, the estimation errors in \( A \) for this simulation are well below 5% of the true values.

In this example, the standard deviations of the estimation errors of \( A \) and \( \alpha \) are significantly lower for the EKF than for the MLSI. On average, the standard deviation on \( A \) using the EKF is 42% of that obtained using the MLSI, and the standard deviation on \( \alpha \) using the EKF is 23% of that obtained using the MLSI. From this we conclude that the EKF yields the lower error estimates for the gyro error terms relating to the nonlinear scale factor under the particular set of conditions for the simulation described here. We reiterate that this comparison is based on an MLSI with \( N_{\text{param}} = 15 \). In Section 5.3.3 we evaluate the effect of varying \( N_{\text{param}} \) on the estimation errors of \( A \) and \( \alpha \), and Section 5.4 investigates the results over a range of possible initial conditions.

Typical filter run times for the dynamic calibration about each axis are as follows: for the linear Kalman filter, 115 seconds CPU time; for the EKF, 203 seconds CPU time; and for the MLSI, 201 seconds CPU time. This represents the amount of time required to run the filters on a Sun Microsystems Sparc IPX, for the dynamic calibration about each axis.

5.3.3 Maximum Likelihood System Identification: Effect of \( N_{\text{param}} \)

The previous sections evaluate the performance of the maximum likelihood system identification method in estimating the nonlinear scale factor terms using the prespecified value \( N_{\text{param}} = 15 \). This number was selected to minimize errors in \( A \) and \( \alpha \) in preliminary simulation tests. In this section, we compare more formally the performance of the MLSI using different values of \( N_{\text{param}} \). It is important to check that such performance is not strongly sensitive to this parameter, as this could impose a severe limitation on the usefulness of the MLSI for design of the estimation algorithm.

To compare the results at different values of \( N_{\text{param}} \), we simply execute the dynamic simulation described in Section 5.2.2 using different values of \( N_{\text{param}} \) in the filter, and we analyze the results using the Monte Carlo method of Section 5.2.4 with \( N = 20 \). We want to compare the standard deviations of the resulting estimation errors in \( A \) and \( \alpha \) as a function of \( N_{\text{param}} \) and to determine whether a bias is introduced as we vary that parameter. The mean and standard deviation of the estimation errors in \( A \) and \( \alpha \) about all three axes in a range of \( N_{\text{param}} \) near 15 are plotted in Figures 5-5 and 5-6, respectively.

In terms of bias (Figure 5-5), the MLSI performance is essentially independent of \( N_{\text{param}} \) for values in the range \( 10 \leq N_{\text{param}} \leq 20 \). Outside this region, the mean estimation errors on \( \alpha \) grow large. This is especially true for \( N_{\text{param}} < 10 \), although there is an increase in the mean estimation error on the y- and z-axes for \( N_{\text{param}} > 20 \) as well. In the latter region, however, Figure 5-6 shows higher standard deviations, so that a greater portion of the mean estimation error is actually due
Biases

\[ \text{mean error} = \frac{\text{absolute values of mean estimation errors on } A \text{ and } \alpha \text{ for MLSI with varying } N_{\text{param}}} \]

Figure 5-5: Absolute values of mean estimation errors on \( A \) and \( \alpha \) for MLSI with varying \( N_{\text{param}} \).

Standard Deviations

\[ \text{standard deviation} = \frac{\text{standard deviations of estimation errors on } A \text{ and } \alpha \text{ for MLSI with varying } N_{\text{param}}} \]

Figure 5-6: Standard deviations of estimation errors on \( A \) and \( \alpha \) for MLSI with varying \( N_{\text{param}} \).
simply to random errors rather than a clear bias in the results. The errors in the amplitude $A$ are less sensitive to $N_{\text{param}}$, although the biases are demonstrably higher for $N_{\text{param}} < 10$.

There is some indication that the optimal choice of $N_{\text{param}}$ is dependent on the decay time $1/\alpha$. For example, the $z$-axis has the longest decay time (80 seconds), and this is also the axis which exhibits the largest mean error in the estimate of $\alpha$ for low $N_{\text{param}}$. Conversely, the $y$-axis, with the shortest decay time of 60 seconds, exhibits the smallest mean estimation error for $\alpha$ at low $N_{\text{param}}$. At high $N_{\text{param}}$, the situation is nearly reversed, although there is considerable fluctuation so that neither the $y$-axis nor the $z$-axis shows unambiguously the largest bias. This condition may be due to the dependence of the nonlinear estimation results on the order of the dynamic calibration sequence. At any rate, the trend seems to imply that high values of $N_{\text{param}}$ are desired for long decay times. This result is to be expected; it suggests that the interval between parameter updates should be related to the fundamental time constant associated with the exponential decay.

It is clear from Figure 5-5 that the biases are lowest in the range $10 < N_{\text{param}} < 20$. It is not surprising that errors are introduced when $N_{\text{param}}$ is set too low. In this case, the interval between parameter updates is too short, and the contribution to the accumulated angular displacements due to process and measurement noise is likely to be far from the mean of zero. When estimating the ordinary states of the system, this does not deteriorate performance because the covariance matrix contains information necessary to determine how much weight to attribute to each measurement based on the discrete noise covariances. In estimating the uncertain parameter $\alpha$ in the MLSI, however, the conditional information matrix $J$ is reinitialized after each parameter update. Too short an interval between parameter updates, then, results in too high a mean value of the noise over that interval, which in turn yields large mean estimation errors appearing as biases.

The reason for deteriorating performance at high $N_{\text{param}}$ is more subtle. At high $N_{\text{param}}$ we have better smoothing of the output signal, so that the effect of noise is de-emphasized and the problem associated with low $N_{\text{param}}$ is less likely to arise. The problem at high $N_{\text{param}}$ arises instead because $\alpha$ is estimated too infrequently. Over the entire interval between parameter updates, the most recent estimate of $\alpha$ is used. We expect this estimate to be less accurate than the next one, that is, that the estimates should improve as more information is gathered in the course of the filtering algorithm. Hence, a poorer estimate of $\alpha$ is being used over a longer interval of time. This is turn degrades the estimates of all the other terms, so that future estimates of $\alpha$ are also affected. By the end of the calibration, the state estimates turn out not to have suffered too greatly by the use of too large a value of $N_{\text{param}}$, but the estimate of $\alpha$ is obviously worse.

Figure 5-6 shows a general trend of the standard deviation of the estimate of $\alpha$ increasing with $N_{\text{param}}$. The arguments above explaining the errors arising when $N_{\text{param}}$ is set too high or too low also apply to the standard deviations. More important in this case, however, is simply that as $N_{\text{param}}$ increases, the number of parameter updates for a given calibration sequence decreases. An
increase in the standard deviation of the estimation error is the typical result of what is effectively a decrease in sample size.

The conclusions in this section concerning performance of the EKF and MLSI are valid for the IPSRU gyro calibration subject to the initial conditions on the estimates listed in Section 5.2.3. Because these are both nonlinear estimation algorithms, the results are dependent upon the assumed initial conditions. Indeed, it has been pointed out that the EKF may not converge [1, 17] for some initial conditions. (This is also true of the MLSI.) Section 5.4 compares the EKF and MLSI on the basis of convergence over a wide range of initial conditions.

5.4 Convergence Properties

Nonlinear optimization algorithms (including recursive filters) frequently fail to converge over some region in the space of possible initial conditions. Divergence, of course, results in unacceptable errors in the outcome of the algorithm. In the gyro calibration problem, it is important to assess the region of convergence of the filter so that we may be confident that the resulting estimates of the gyro error terms are correct. This section compares the convergence properties of the extended Kalman filter and the maximum likelihood system identification method to determine the initial conditions which admit valid gyro error estimates.

For clarity in presentation of the results, we consider the dynamic calibration about one axis only in evaluating the convergence properties of the nonlinear filters. This is a sufficient basis for comparing convergence of the EKF and MLSI in the gyro calibration problem. With the initial conditions selected in Section 5.2.3, we see in Section 5.3 that both algorithms converge to the true solution. In this section, we investigate the effect of varying those initial conditions. In particular, we allow $\dot{\gamma}_z(t_0)$ and $\dot{\gamma}_z(t_0)$ to vary, and we compare the performance of the two filters in a dynamic calibration about the $z$-axis. All other initial conditions, including the initial covariance matrix, are held fixed so that we concentrate on a dimension 2 space of initial conditions, which is easy to visualize (see Section 5.4.1).

We test for convergence empirically as follows. First, we specify an initial condition $\dot{\gamma}_z(t_0)$ (which for brevity we denote $A_0$ throughout this context). We then select a region over which $\dot{\gamma}_z(t_0)$ (or $\alpha_0$) is to vary, and run both filters at several values of $\alpha_0$ in that region. Only one actual simulation of the dynamic calibration is required; we run the filters on the same set of simulation data each time. We could, of course, perform a Monte Carlo analysis, which would have a fine tuning effect on the outcome. The results from a single simulation, however, are sufficient for demonstrating the difference between the two filter algorithms, as the remainder of this section shows.

Having tested a filter at a given $(\alpha_0, A_0)$, we examine the error in the final estimates of $\dot{\gamma}_z$ and $\alpha_z$ as a percentage of the true values, denoted by $\ddot{\gamma}_z$ and $\ddot{\alpha}_z$. We define the estimation errors, $\ddot{\gamma}_z$
and $\tilde{\alpha}_e$, as follows.

\[
\tilde{\alpha}_e = \frac{(100\%) |\hat{\alpha}_e(t_f) - \tilde{\alpha}_e|}{\tilde{\alpha}_e} \tag{5.24}
\]

\[
\bar{\alpha}_e = \frac{(100\%) |\hat{\alpha}_e(t_f) - \bar{\alpha}_e|}{\bar{\alpha}_e} \tag{5.25}
\]

where $t_f$ is the time at the end of the calibration test. In Figure 5-7, $\bar{\alpha}_e$ is presented as a function

![Estimation Error in alpha](image)

Figure 5-7: Estimation error in $\alpha_e$ as a function of $\alpha_0$ at $A_0 = 50$ ppm/(deg/s)$^2$. Of $\alpha_0$ for $A_0 = 50$ ppm/(deg/s)$^2$; results appear for both the EKF and the MLSI.

While the estimation error plot does not provide a rigorous characterisation of filter convergence, there are clear indications that, for example, both filters diverge for $\alpha_0 > 2 \times 10^{-1}$ s$^{-1}$ or so. Moreover, the EKF diverges for $\alpha_0 < 2 \times 10^{-3}$ s$^{-1}$ or so. By contrast, the estimation errors resulting from the MLSI remain less than 10\% for $\alpha_0$ over most of the interval $(1 \times 10^{-4}, 1 \times 10^{-1})$ s$^{-1}$. This is a preliminary indication that the MLSI converges over a wider range of initial conditions than does the EKF, although for certain initial conditions the EKF performs better.

At the value $\alpha_0 = 2 \times 10^{-2}$ s$^{-1}$ used in Section 5.3, the estimation errors in $\alpha_e$ are about the same in both the EKF and MLSI. The Monte Carlo simulation showed a higher error from the MLSI under these conditions, but Figure 5-7 represents the results of a single simulation, in which errors can differ somewhat from the mean estimation errors, according to the standard deviations listed in Tables 5.8 and 5.9. Since we are only interested in convergence issues here, where the estimation errors can become very large, slight deviations from the average in the single simulation under consideration are not of concern.
The estimation errors in $A_e$ for the same set of conditions are plotted in Figure 5-8. The EKF clearly diverges for $\alpha_0 < 2 \times 10^{-3}$ s$^{-1}$; otherwise, the results are hard to interpret. There are no other clear regions where the filters diverge even though some of the errors are as large as 25%. We are really interested in a more practical definition of convergence. In Section 5.4.1, we use the methods outlined here to develop "regions of convergence" for both algorithms, defined arbitrarily as the regions in ($\alpha_0, A_0$) space where the estimation errors are less than 10%. Further discussion of the convergence in $A_e$ is deferred until that section. In this case, the estimation error from the EKF is substantially lower than that from the MLSI at the condition $\alpha_0 = 2 \times 10^{-2}$ s$^{-1}$ used in Section 5.3.

5.4.1 Regions of Convergence for Nonlinear Algorithms

In Figures 5-9 and 5-10 we plot the regions of convergence in $A_e$ and $\alpha_e$, comparing the results from the EKF and the MLSI. The regions which appear in the figures are those over which the estimation errors are less than 10%, as defined in Section 5.4.

Figure 5-9 depicts the 10% convergence regions in $A_e$ for the EKF and the MLSI — the regions bounded by the solid and dotted lines, respectively. We observe that, provided $\alpha_0$ is selected correctly, the MLSI can converge to within 10% of the true value of $A_e$ even for $A_0$ beyond the range of $[-200, 200]$ ppm/(deg/s)$^2$ shown in the graph. The EKF convergence, on the other hand, is limited approximately to the region $[-150, 150]$ ppm/(deg/s)$^2$. Both of these regions are large relative to typical values expected for IPSRU, however. In the vicinity of the true state in this
Figure 5-9: Regions in \((\alpha_0, A_0)\) space for which EKF and MLSI estimates of \(A\) converge to within 10% of the true value.

Figure 5-10: Regions in \((\alpha_0, A_0)\) space for which EKF and MLSI estimates of \(\alpha\) converge to within 10% of the true value.
simulation, the regions of convergence of the two algorithms are certainly different, but it is not clear which would be classified as "better" because both are of roughly the same size. This is in very qualitative terms, of course. The point is that we need to consider the convergence properties in $\alpha_e$ as well before concluding much about the general convergence properties of the two nonlinear filter algorithms.

Figure 5-10 demonstrates a more significant difference in the convergence of $\alpha$ between the EKF and MLSI. The convergence region for the EKF is narrow and irregular, while the MLSI has a nice wide region over which $\alpha$ converges. This is important; it shows that MLSI can estimate $\alpha$ effectively anywhere in the vicinity of the true state. This is not the case for the EKF. Instead, in any given calibration test using the EKF, a great deal of trial and error may be required before a suitable initial condition is determined. In particular, Figure 5-10 shows that, for the particular system under study in this simulation, the EKF estimate of $\alpha$ does not settle to within 10% of the true value even if we select that true value as the initial condition! The state estimate initially wanders from the true value because the initial estimates of the linear scale factor errors are set equal to zero, and the EKF attempts to update the estimates of $A$ and $\alpha$ along with the linear scale factor errors beginning with the first rotation of the dynamic calibration.

We can understand this difference in convergence in $\alpha$ as a result of the way each algorithm estimates that term. The EKF treats $\alpha$ as a state, in the same manner as the amplitude $A$ and the linear scale factor errors. Therefore, initial gyro error rates are interpreted as arising from errors in all of the states (including $\alpha$) according to the elements of the covariance matrix. At the first rotation of the dynamic calibration, the error rate measurements are used to update all of the states, and any errors introduced by this procedure are propagated throughout the remainder of the run. In the MLSI, however, $\alpha$ is treated as an uncertain parameter, which is updated only after every $N_{\text{param}}$ sample periods. Therefore, the first measurements are used to update the remaining states. If $\alpha_0$ is close to the true value of $\alpha$ and $N_{\text{param}}$ is sufficiently large, then the state estimates should be close enough to the true values by the time of the next update in $\alpha$ that the change in its estimate is more commensurate with its actual estimation error rather than an assumed initial variance. We conclude that the MLSI performs better than the EKF for convergence of the estimates of those quantities which are classified as uncertain parameters in the gyro calibration problem.

As a practical matter we may be interested in the sets of initial conditions for which both $A$ and $\alpha$ converge to within 10% of the true values. These are represented simply as the intersections of the regions in Figures 5-9 and 5-10. These intersection regions are plotted in Figure 5-11. Note that convergence in $\alpha$ is not a limiting factor in the MLSI and that the convergence region for that algorithm is identical to the corresponding region for convergence in $A$ alone. In the EKF, this is not the case.

Here the conclusions about convergence are similar to those derived in considering convergence
Convergence Region for both $A$ and $\alpha$

Figure 5-11: Regions in $(\alpha_0, A_0)$ space for which EKF and MLSI estimates of both $A$ and $\alpha$ converge to within 10% of the true values.

In $A$ alone, except that the true state now lies outside the convergence region for the EKF, as it did in the analysis of $\alpha$ convergence. Due to the irregularity of the EKF convergence region, we may not achieve convergence in the neighborhood of the true state. Again, this implies that a good deal of trial and error may be necessary to achieve convergence to the correct gyro error parameters if we use the EKF.

These results are based on the value $N_{\text{param}} = 15$ in the MLSI. Section 5.4.2 investigates the effect of $N_{\text{param}}$ on MLSI convergence.

5.4.2 Maximum Likelihood System Identification: Effect of $N_{\text{param}}$

In this section we investigate convergence of the MLSI as $N_{\text{param}}$ is allowed to vary. We set $A_0 = 20 \text{ ppm/(deg/s)}^2$, so that the initial amplitude estimate is equal to the assumed true value from Section 5.2.3. Figures 5-12 and 5-13 show the 10% convergence regions in $A$ and $\alpha$, respectively, for varying $\alpha_0$ and $N_{\text{param}}$.

The previous section concludes that an important advantage of the MLSI over the EKF is convergence for initial conditions on the estimates of $A$ and $\alpha$ near the true values. Hence it is desirable to retain that property over as wide a range of $N_{\text{param}}$ as possible. The vertical dashed lines in Figures 5-12 and 5-13 represent this true state.

Figure 5-12 indicates that the MLSI converges in $A$ for all values of $N_{\text{param}}$ shown. For $\alpha$, on the other hand, Figure 5-13 demonstrates that the true state is outside the 10% convergence region.
Figure 5-12: Regions in $(\alpha_0, N_{\text{param}})$ space for which MLSI estimates of $A$ converge to within 10% of the true values. $A_0 = 20 \text{ ppm}/(\text{deg}/s)^2$.

Figure 5-13: Regions in $(\alpha_0, N_{\text{param}})$ space for which MLSI estimates of $\alpha$ converge to within 10% of the true values. $A_0 = 20 \text{ ppm}/(\text{deg}/s)^2$. 

79
at $N_{\text{param}} = 12$. Moreover, at $N_{\text{param}} = 20$ the true state falls close to the convergence region boundary. Nevertheless, we achieve convergence for the values $13 \leq N_{\text{param}} \leq 19$. We recall from Figure 5-5 that the mean estimation errors in the MLSI are lowest for $10 \leq N_{\text{param}} \leq 20$. Having investigated MLSI convergence, we find that there is still a significant range of values of $N_{\text{param}}$ which admit acceptable MLSI performance.

For completeness, we include Figure 5-14, which shows the values of $\alpha_0$ and $N_{\text{param}}$ for which MLSI converges to 10% in both $A$ and $\alpha$. This information does not provide any significant new insights beyond those revealed in Figure 5-13.

5.5 Discussion

The results of this comparison between the EKF and the MLSI are particularly important from the standpoint of nonlinear filter design for gyroscope calibration. They provide important information which suggests that an engineer attempting to design such a filter should expect less trial and error in achieving an acceptable design by selecting the MLSI as the estimation algorithm.

Before evaluating filter performance, it is helpful to compare the run times of the two nonlinear algorithms. More computations are required than for a standard Kalman filter, and in selecting a nonlinear filter it is important that the computational burden be kept to a minimum. As the algorithms are implemented in this thesis, however, the difference in run time between the EKF and
the MLSI is negligible; the EKF takes only about 1% longer. Hence, the run time is not a deciding factor in choosing between the two methods.

In the Monte Carlo comparison we find that, for a specific set of initial conditions, the EKF can yield lower estimation errors than the MLSI. This result depends strongly on initial conditions, however. In investigating convergence we find the MLSI to be the more desirable of the two algorithms. Specifically, the estimation errors in $A$ and $\alpha$ resulting from the MLSI are less than 10% of the true values when the initial estimates of those quantities are sufficiently close to the true values. This is not the case for the EKF, where initial estimation errors in other terms (most notably the linear scale factor errors) cause the EKF estimates to diverge from the true values. This distinction between EKF and MLSI convergence is most notable in estimating $\alpha$, which the MLSI treats as an uncertain parameter.

If the MLSI is to be considered a useful alternative to the EKF, the designer needs some guidance in selecting an appropriate value of $N_{\text{param}}$. More research could possibly establish a rigorous criterion for selecting $N_{\text{param}}$ in the gyro calibration problem. We expect that a relationship exists between the decay time of the dominant exponential term and $N_{\text{param}}$, which is directly proportional to the time between parameter updates. Hence, approximate knowledge of the decay time should help the designer select $N_{\text{param}}$. There is a wide enough range over which MLSI performance is not highly dependent on $N_{\text{param}}$ so that a reasonable value could be found by trial and error after only a few guesses. Moreover, we achieve good convergence properties for most of the values of $N_{\text{param}}$ within this range.

The convergence issue appears to be the most important one in which there is a substantial difference in performance between the EKF and MLSI. From the standpoint of a filter designer, this influences the amount of trial and error necessary to select initial estimates at which the filter converges. Since the true values of the state variables related to the nonlinear terms are most likely close to the best guess for those quantities, convergence of the MLSI in the neighborhood of that point in the space of possible initial conditions is highly desirable. If, in the case of the EKF, convergence is not achieved there, a trial-and-error search is necessary just to find appropriate initial conditions. This property of convergence near the true state makes the MLSI an attractive alternative to the EKF.
Chapter 6

Conclusions

This thesis provides a number of results pertinent to the IPSRU gyroscope calibration problem. Some of these are theoretical results related to the implementation of the algorithms. Others are comparisons of the EKF and the MLSI derived from numerical simulations. In addition, some topics are suggested for future research which would build on the conclusions presented in this thesis.

An important theoretical result is the incorporation of separate bias estimation into the U-D factorization filter. This is done in such a way that the results are easily extended to the nonlinear case, yielding an implementation of the separated EKF with U-D factorization. Using some of Maybeck's [11] approximations, we also combine these concepts into the MLSI. The U-D factorization generates an algorithm which guarantees positive-definiteness of the covariance matrix, and the separate bias estimation reduces the total number of computations required.

In the numerical simulations, we compare performance of the EKF and the MLSI. In some cases, the EKF may yield lower estimation errors in the terms related to the nonlinear scale factor error, but these results depend greatly on the initial conditions provided to the filter. The MLSI turns out to converge over a wide range of initial conditions about the true values of the nonlinear terms. In the EKF this is not necessarily the case, because we treat all of the nonlinear terms as states, with no provision to hold any of them constant while the remaining estimates approach their true values. This is demonstrated in estimating the decay constant of the nonlinear scale factor error, which the MLSI treats as an uncertain parameter.

Both methods have approximately the same execution time.

These results have implications for design of nonlinear filters for gyroscope calibration. An engineer selecting initial estimates of the nonlinear terms for the MLSI should simply choose values as close as possible to the anticipated true values of those quantities. The MLSI converges on the correct values provided that the initial conditions are sufficiently close to the true state. (Selection of $N_{\text{param}}$ is also a limiting factor, but there is a relatively wide range of possible $N_{\text{param}}$ which yield
converging MLSI designs.) The EKF is not as likely as the MLSI to converge for initial estimates near the true solution. We therefore expect the amount of trial-and-error time in designing an acceptable MLSI algorithm to be significantly less than that for the EKF. The MLSI is a very attractive alternative from a practical standpoint.

There is much that could be done to compare the EKF and MLSI from different points of view and also to improve implementation of the MLSI algorithm for gyro calibration applications. Some suggestions are provided in Section 6.1.

6.1 Suggestions for Future Work

There are opportunities for related future research both in the computational aspects of implementation of the MLSI and in the analysis of simulation results. Though it is not likely that more research of this nature will be applied to IPSRU directly, the methods outlined in this thesis are applicable, with only minor modifications to the system models, for a large class of similar gyro calibration problems. This thesis demonstrates some of the possible advantages of the MLSI over the EKF in gyro calibration, and more research could be applied to answering specific questions of interest which would help in understanding the tradeoffs more rigorously. This work would be of general utility in actual calibration testing of gyroscopes with nonlinear error dynamics such as those appearing in IPSRU's gyro.

In the MLSI implementation used in this thesis, we apply numerous simplifying approximations without evaluating their effect on the estimation results. All of these approximations come from Maybeck [11], who cites available results indicating that most of them do not significantly degrade filter performance. (At least one actually improves performance.) It would be helpful, however, to determine whether the MLSI can be improved by performing parameter updates after every sample period instead of only every $N_{\text{param}}$. In this case, the score and conditional information matrix would still be computed based on the most recent $N_{\text{param}}$ intervals, but more calculation would be required. Terms in the summations for those two quantities would be stored individually rather than as a running sum, increasing the memory requirement somewhat.

Another future research project could compare the MLSI implementation of this thesis with the full MLSI algorithm which computes the score and conditional information matrix based on the full measurement history at each time step, instead of using only the most recent $N_{\text{param}}$. Again, the resulting algorithm would be more expensive computationally, but we would expect to achieve better estimates.

We could also investigate the effects of removing other approximations. The likely result would be a tradeoff between accuracy and computational burden. Also, the separate bias estimation applied to the MLSI would no doubt be harder to formulate without the benefit of some of these
approximations.

We have made no special attempt to control divergence in the MLSI. It may be possible, for example, to compare the residuals at a given parameter update stage with those at a previous update and to use that information to decide whether actually to perform the update. This is not really possible with the EKF, because the decay constant is treated as a state just like any other. For the MLSI, however, more research could develop a criterion for deciding when to make parameter updates so that the convergence properties are improved.

The clearest advantage of the MLSI over the EKF arises in estimating the decay constant, which the MLSI treats as an uncertain parameter. It would be interesting to see if estimation of the amplitude corresponding to the nonlinear scale factor error could also be improved by treating it as a parameter in the MLSI.

There is some evidence that the optimal value of $N_{param}$ should be related to the decay time of the nonlinear term. This should be researched more thoroughly to determine whether such a relationship actually exists and, if so, the exact nature of that relationship. The results contained in this thesis do not indicate whether the optimal $N_{param}$ is influenced by the other states as well.

A more direct comparison of performance between the EKF and the MLSI would emerge from the generation of some comparison region between the two. This would consist of the region in the space of possible initial conditions (like the convergence regions produced in this thesis) in which the estimation errors for the EKF are lower than those for the MLSI. Such a plot should report the comparison results in a statistical sense, possibly by means of a Monte Carlo analysis.

All of these results are applicable, not only to IPSRU, but to the more general class of gyroscope calibration problems in which nonlinear scale factor errors arise. The results of the future research suggested here, in addition to results provided in this thesis, may contribute much toward developing a systematic approach to MLSI design. A more automated gyroscope calibration methodology would be the ultimate goal of this research.
Appendix A

Example of Separate Bias Estimation Algorithm for a Linear System

This appendix goes through the Kalman filter propagation and update equations using both the standard Kalman filter formulation and the separate bias estimation for the first time propagation and measurement update on a system which can be described using the discrete separate bias formulation of (3.90-3.92). Not every step in the analysis will be discussed in detail; only the results of the computations necessary to execute each algorithm will be presented. For more discussion on the computations, refer to Sections 3.2-3.4.

A.1 Initial Conditions

Assume the measurement noise to be a discrete white noise sequence with covariance \( R_k \). The process noise only enters the equations for the states in \( x \), so we have a discrete process noise covariance matrix \( \bar{Q}_k = \text{Diag}[Q_k, 0] \) and a coefficient matrix \( \bar{L}_k = [ L_k^T \ 0 ]^T \). The full measurement geometry matrix is \( \bar{H}_k = [ H_k \ C_k ] \). The remaining initial conditions are denoted as follows.

\[
\begin{align*}
\hat{x}_0 &= \begin{bmatrix} \hat{x}_0 + V_0\hat{b}_0 \\ \hat{b}_0 \end{bmatrix} \\
\bar{P}_0 &= \begin{bmatrix} P_0 + V_0M_0V_0^T & V_0M_0 \\ M_0V_0^T & M_0 \end{bmatrix}
\end{align*}
\] (A.1) (A.2)
A.2 Time Propagation Equations

For the separate bias estimation, perform the propagations as follows.

\[
\begin{align*}
\hat{x}_{k+1}^- &= A_k \hat{x}_k^+ \\
P_{k+1}^- &= A_k P_k^+ A_k^T + L_k Q_k L_k^T \\
V_{k+1}^- &= A_k V_k^+ + B_k \\
\hat{b}_{k+1}^- &= \hat{b}_k^+ \\
M_{k+1}^- &= M_k^+ 
\end{align*}
\]  

(A.3) \hspace{1cm} (A.4) \hspace{1cm} (A.5) \hspace{1cm} (A.6) \hspace{1cm} (A.7)

To incorporate the U-D factorization in this stage, follow Section 3.3.2 to evaluate (A.4) using \(U_x\) and \(D_x\). It has already been shown \([18, 10]\) that this gives the correct U-D factorization of \(P_{k+1}^-\) in (A.4).

To compare with the Kalman filter, compute the following.

\[
\begin{align*}
\hat{x}_{k+1}^- &= \hat{x}_{k+1}^- + V_{k+1}^+ \hat{b}_{k+1}^+ = A_k \hat{x}_k^+ + [A_k V_k^+ + B_k] \hat{b}_k^+ \\
[P_x]_{k+1}^- &= V_{k+1}^- M_{k+1}^- = [A_k V_k^+ + B_k] M_k^+ \\
[P_{xz}]_{k+1}^- &= P_{k+1}^- + V_{k+1}^- M_{k+1}^- V_{k+1}^- \text{T} \\
&= A_k P_k^+ A_k^T + L_k Q_k L_k^T + [A_k V_k^+ + B_k] M_k^+ [A_k V_k^+ + B_k]^T 
\end{align*}
\]  

(A.8) \hspace{1cm} (A.9) \hspace{1cm} (A.10)

For the standard Kalman filter, compute,

\[
\begin{align*}
\bar{A}_k &= \begin{bmatrix} A_k & B_k \\ 0 & I \end{bmatrix} \\
\bar{x}_{k+1}^- &= \bar{A}_k \bar{x}_k^+ = \begin{bmatrix} A_k \bar{x}_k^+ + [A_k V_k^+ + B_k] \bar{b}_k^+ \\ \bar{b}_k^+ \end{bmatrix} \\
\bar{P}_{k+1}^- &= \bar{A}_k P_k^+ \bar{A}_k^T + \bar{L}_k \bar{Q}_k \bar{L}_k^T \\
&= \begin{bmatrix} A_k P_k^+ A_k^T + L_k Q_k L_k^T + [A_k V_k^+ + B_k] M_k^+ [A_k V_k^+ + B_k]^T & [A_k V_k^+ + B_k] M_k^+ \\ [A_k V_k^+ + B_k] M_k^+ [A_k V_k^+ + B_k]^T & M_k^+ \end{bmatrix} 
\end{align*}
\]  

(A.11) \hspace{1cm} (A.12) \hspace{1cm} (A.13)

Thus we verify that the two algorithms yield the same results for the time propagation.
A.3 Measurement Update Equations

For the separate bias estimation, we first define the following.

\[ G_k = H_k V_k + C_k \quad (A.14) \]
\[ D_k = G_k M_k^- G_k^T + H_k P_k^- H_k^T + R_k \quad (A.15) \]
\[ E_k = H_k P_k^- H_k^T + R_k = D_k - G_k M_k^- G_k^T \quad (A.16) \]
\[ F_k = P_k^- H_k^T + V_k^- M_k^- G_k^T \quad (A.17) \]
\[ K_k = P_k^- H_k^T E_k^{-1} \quad (A.18) \]
\[ S_k = M_k^- G_k^T D_k^{-1} \quad (A.19) \]

Perform the updates as follows.

\[ \tilde{x}_k^+ = \tilde{x}_k^- + K_k [z_k - H_k \tilde{x}_k^-] = \tilde{x}_k^- + P_k^- H_k^T E_k^{-1} [z_k - H_k \tilde{x}_k^-] \quad (A.20) \]
\[ P_k^+ = [I - K_k H_k] P_k^- = P_k^- - P_k^- H_k^T E_k^{-1} H_k P_k^- \quad (A.21) \]
\[ V_k^+ = V_k^- - K_k G_k = V_k^- - P_k^- H_k^T E_k^{-1} G_k \quad (A.22) \]
\[ \hat{b}_k^+ = \hat{b}_k^- + S_k [z_k - H_k \tilde{x}_k^- - G_k \hat{b}_k^-] = \hat{b}_k^- + M_k^- G_k^T D_k^{-1} [z_k - H_k \tilde{x}_k^- - G_k \hat{b}_k^-] \quad (A.23) \]
\[ M_k^+ = [I - S_k G_k] M_k^- = M_k^- - M_k^- G_k^T D_k^{-1} G_k M_k^- \quad (A.24) \]

To incorporate the U-D factorization in this stage, we use the forms of the above corresponding to a scalar measurement and, provided \( R_k \) is diagonal, follow the procedure through for each measurement. The first step is to follow Section 3.3.3 to evaluate (A.21) using \( P = U_s D_s U_s^T \). In the process, \( K \) defined in (A.18) and \( E \) from (A.16) are computed. We then update \( \tilde{x} \) and \( V \) and follow Section 3.3.3 again to evaluate (A.24) using \( M = U_s D_s U_s^T \). We thereby obtain \( S \) defined in (A.19) and \( D \) from (A.15), which we use to update \( \hat{b} \). It has already been shown [3, 10] that this gives the correct U-D factorization of \( P_k^+ \) in (A.21) and of \( M_k^+ \) in (A.24).

To compare with the Kalman filter, compute the following.

\[ \dot{x}_k^+ = \dot{x}_k^+ + V_k^+ \hat{b}_k^+ = \dot{x}_k^- + P_k^- H_k^T E_k^{-1} [z_k - H_k \tilde{x}_k^-] + \\
\quad [V_k^- - P_k^- H_k^T E_k^{-1} G_k] \left[ \hat{b}_k^- + M_k^- G_k^T D_k^{-1} \left( z_k - H_k \tilde{x}_k^- - G_k \hat{b}_k^- \right) \right] \\
= \dot{x}_k^- + V_k^- \hat{b}_k^- + F_k D_k^{-1} [z_k - H_k \tilde{x}_k^- - G_k \hat{b}_k^-] \quad (A.25) \]
\[ [P_{z_k}]_k^+ = V_k^+ M_k^- = [V_k^- - P_k^- H_k^T E_k^{-1} G_k] [M_k^- - M_k^- G_k^T D_k^{-1} G_k M_k^-] \\
= V_k^- M_k^- - [V_k^- M_k^- G_k^T + P_k^- H_k^T] D_k^{-1} G_k M_k^- = V_k^- M_k^- - F_k D_k^{-1} G_k M_k^- \quad (A.26) \]
\[ [\hat{P}_{zz}]_k^+ = P_k^+ + V_k^+ M_k^+ V_k^+\text{T} \]
\[ = P_k^+ - P_k^+ H_k^T E_k^{-1} H_k P_k^+ + [V_k^- M_k^- - F_k D_k^{-1} G_k M_k^-] \left[ V_k^T - G_k^T E_k^{-1} H_k P_k^- \right] \]
\[ = P_k^+ + V_k^+ M_k^- V_k^-\text{T} - F_k D_k^{-1} H_k P_k^- - F_k D_k^{-1} G_k M_k^- V_k^-\text{T} \]
\[ = P_k^+ + V_k^+ M_k^- V_k^-\text{T} - F_k D_k^{-1} F_k^T \] (A.27)

For the standard Kalman filter,
\[ \tilde{K}_k = \tilde{P}_k \overline{H}_k \overline{C}_k \]
\[ = \left[ \begin{array}{cc} P_k^+ + V_k^+ M_k^- V_k^-\text{T} & V_k^- M_k^- \\ M_k^- V_k^-\text{T} & M_k^- \end{array} \right] \left[ \begin{array}{c} H_k^T \\ C_k^T \end{array} \right] D_k^{-1} = \left[ \begin{array}{c} F_k D_k^{-1} \\ M_k^- G_k^T D_k^{-1} \end{array} \right] \] (A.28)
\[ \hat{x}_k = \hat{x}_k^- + \tilde{K}_k [z_k - \overline{H}_k \hat{x}_k^-] = \left[ \begin{array}{c} \hat{x}_k^- + V_k^- \tilde{b}_k^- + F_k D_k^{-1} [z_k - H_k \hat{x}_k^- - G_k \tilde{b}_k^-] \\ \tilde{b}_k^- + M_k^- G_k^T D_k^{-1} [z_k - H_k \hat{x}_k^- - G_k \tilde{b}_k^-] \end{array} \right] \] (A.29)
\[ \tilde{P}_k^+ = [I - \tilde{K}_k \overline{H}_k] \tilde{P}_k^- \]
\[ = \left[ \begin{array}{cc} I - F_k D_k^{-1} H_k & -F_k D_k^{-1} C_k \\ -M_k^- G_k^T D_k^{-1} H_k & I - M_k^- G_k^T D_k^{-1} C_k \end{array} \right] \left[ \begin{array}{cc} P_k^- + V_k^- M_k^- V_k^-\text{T} & V_k^- M_k^- \\ M_k^- V_k^-\text{T} & M_k^- \end{array} \right] \]
\[ = \left[ \begin{array}{cc} P_k^+ + V_k^- M_k^- V_k^-\text{T} - F_k D_k^{-1} F_k^T & V_k^- M_k^- - F_k D_k^{-1} G_k M_k^- \\ M_k^- V_k^-\text{T} - M_k^- G_k^T D_k^{-1} F_k^T & M_k^- - M_k^- G_k^T D_k^{-1} G_k M_k^- \end{array} \right] \] (A.30)

Thus we verify that the two algorithms yield the same results for the measurement update. Note that to update the measurements one at a time we simply use the forms of these equations for a scalar measurement. Since we have shown that the two algorithms (with and without separate bias and U-D factorization) yield the same results for the measurement update, and we know [10] that the updates may be performed one at a time in the standard Kalman filter, it follows that the separate bias U-D factorization filter yields the same results for the measurement update as the Kalman filter.
Bibliography


