Australian Government
Department of Defence
Defence Science and
Technology Group

Development of GPS Receiver Kalman Filter Algorithms for Stationary, Low-Dynamics, and High-Dynamics Applications

Peter W. Sarunic [1]

[1] **Cyber and Electronic Warfare Division**
Defence Science and Technology Group

DST-Group–TR–3260

ABSTRACT

This report presents algorithms that can be utilized in a GPS receiver to convert satellite-to-receiver pseudo-ranges to receiver position estimates. The report discusses a method that is used to determine instantaneous estimates of receiver position and then goes on to develop three Kalman filter based estimators, which use stationary receiver, low dynamics, and high dynamics models for the receiver motion, respectively. These particular dynamic models are utilized because they are commonly used in actual GPS receivers, and cover a wide range of applications. While the standard form of the Kalman filter, of which the three filters just mentioned are examples, is theoretically correct, it can be susceptible to numerical round-off errors, which can in some cases result in poor performance or, in the worst case, filter divergence. This issue, and its solution, is investigated, and another version of the high dynamics filter, which addresses this problem, is presented. Matlab code was developed to test the performance of each of the filters and simulations were performed. The results of the simulations are also presented.

RELEASE LIMITATION

Approved for Public Release

Published by

Cyber and Electronic Warfare Division
Defence Science and Technology Group
PO Box 1500
Edinburgh, South Australia 5111, Australia

Telephone: *1300 333 362*
Facsimile: *(08) 7389 6567*

AR-016-601
June, 2016

APPROVED FOR PUBLIC RELEASE

Development of GPS Receiver Kalman Filter Algorithms for Stationary, Low-Dynamics, and High-Dynamics Applications

Executive Summary

The Global Positioning system (GPS) is the primary source of information for a broad range of positioning, navigation and timing systems. It is an all-weather, satellite-based radio-navigation system which provides world-wide coverage. The objective of this report is to present algorithms used in a central component of the system's receiver position calculation, viz., to convert the satellite-to-receiver pseudo-ranges to receiver position estimates. This report is one outcome of recent efforts to expand our knowledge base for this important component of GPS receiver technology; this increased knowledge will facilitate our capabilities to provide scientifically based advice to the Australian Defence Force.

The report first describes a method for determining instantaneous estimates of receiver position, and then goes on to develop three Kalman filter based receiver position estimators, i.e., a stationary receiver, low dynamics, and high dynamics estimator. As is implied by their names, the three types of filters incorporate dynamic models that are optimized for situations where the receiver is stationary, is subjected to small accelerations, and to large accelerations, respectively. These estimators are designed to optimize performance for commonly occurring applications, as is done in many GPS receivers.

The development of the three types of Kalman filter, as well as the instantaneous estimator is presented in Section 2. Section 3 then presents the results of testing by simulation. It is found that the simulations give indications of performance degradation, resulting from errors associated with numerical round-off, in the case of the high dynamics Kalman filter. This is then further investigated in Section 4 and an alternate form of the high dynamics filter is then developed to overcome the problem. The filter was implemented in Matlab and tested by simulation. The results of the simulations are also in Section 4. Finally, concluding remarks are presented in Section 5.

While this report deals specifically with GPS algorithms, the work covered forms part of a larger effort to develop algorithms for fusing GPS measurements with other sensor data, particularly measurements from inertial navigation systems (INS), to support R&D into multisensor positioning and navigation performance in non-benign environments. The algorithms presented in this report, and software developed for this work, will be required for future research into deep integration of GPS with other sources of position data. The software will also be useful as a component of future modelling software that may need to be developed for performance prediction of current or future systems that incorporate GPS. The ultimate aim is to help inform future capability requirements through the outcomes of this research.

THIS PAGE IS INTENTIONALLY BLANK

Author

Australian Government

Department of Defence

Defence Science and Technology Group

Peter W. Sarunic

CEWD

Peter Sarunic was born in Adelaide, Australia, in 1958. After completing his B. Eng. degree in Electrical Engineering at the University of Adelaide in 1980, he worked as an Electrical/Electronic Engineer in private industry for a period, and then joined the Defence Science and Technology Organisation in 1986. His major areas of work were adaptive tracking, signal processing, and radar systems engineering. In 1996, Peter moved overseas to Canada, where he worked on radar and data fusion problems. In 1998, he returned to DSTO, where he subsequently performed research on multipath track fusion for over-the-horizon radar, multisensor fusion, electronic protection for radar, and control and scheduling of UAVs. While employed at DSTO, Peter completed a B.Sc. degree (mathematical and computer sciences), a M.Eng. degree (electronic engineering) and a PhD. degree (electronic engineering). Peter's current area of research is GPS/INS integration.

THIS PAGE IS INTENTIONALLY BLANK

Contents

Appendices

Figures

THIS PAGE IS INTENTIONALLY BLANK

1 Introduction

The Global Positioning System (GPS) is a satellite-based radio-navigation system that provides world-wide, all-weather, coverage. GPS receivers decode and process messages from in-view satellites to determine the receiver's location as well as the current time; in this report, GPS receiver position determination is considered. To determine the receiver's location, the GPS system uses time of arrival ranging. Each GPS receiver contains an internal clock which it uses to determine the time of arrival of satellite ranging signals; using this information, the receiver calculates the time taken for the signal to travel from the satellite to the receiver. Since the signal travels at the speed of light, c; this time interval can be converted to a distance by simply multiplying by c. The distance calculations are biased by the receiver and satellite clock errors, and are therefore referred to as *pseudo-ranges*. Because of cost and other constraints, the receiver clock is in general much less accurate than the satellite clocks. If the location of four in-view satellites is known, and their ranges to the receiver are measured, the location of the receiver and its clock bias can be computed.

The objective of this report is to present the mathematics used to convert the satellite-to-receiver pseudoranges to receiver position estimates. The report discusses a method that is used to determine instantaneous estimates of receiver position, i.e., estimates based on pseudo-ranges at one time instant, and then goes on to develop three Kalman filter based estimators. Typically the instantaneous estimates are used to initialize a Kalman filter, as Kalman filters require an initial estimate to start their recursions. The three Kalman filter estimators that are presented will be referred to as stationary receiver, low dynamics, and high dynamics filters. As is implied by their names, the three types of filters are optimized for situations where the receiver is stationary, is subjected to small accelerations, and to large accelerations, respectively. This approach is consistent with what can be found in many actual GPS receivers, which allow the user to specify the dynamic level for the given application.

While the standard form of the Kalman filter, of which the three filters just mentioned are examples, is theoretically correct, it can be susceptible to numerical round-off errors. The effects of these errors can be degraded filtering or, in some instances, Kalman filter instability, leading to quite unpredictable behaviour. This issue, and its solution, is investigated, and another version of the high dynamics filter is presented. Matlab code was developed to test the performance of each of the filters and simulations performed. The results of the simulations are also presented.

In Section 2, the development of the three types of Kalman filter, as well as the instantaneous estimator is presented. Section 3 then presents the results of testing by simulation. It is noted that there are some indications of adverse effects due to numerical round-off in the case of the high dynamics Kalman filter. To investigate this issue further, an alternate form of the high dynamics filter is developed in Section 4. The filter was implemented in Matlab and tested by simulation; the results of the simulations are also in Section 4. At the end of the paper, concluding remarks are presented in Section 5.

2 Kalman Filter Development for the Processing of GPS measurements

2.1 Initial Single Point GPS solution

The GPS positioning problem has four unknowns that can be solved using the following equations which use measurements from four satellites [1, p. 145].

$$\begin{aligned}
\tilde{\rho}_1 &= \left[(X_1-x)^2+(Y_1-y)^2+(Z_1-z)^2\right]^{1/2}+ct_r+\chi_1 \\
\tilde{\rho}_2 &= \left[(X_2-x)^2+(Y_2-y)^2+(Z_2-z)^2\right]^{1/2}+ct_r+\chi_2 \\
\tilde{\rho}_3 &= \left[(X_3-x)^2+(Y_3-y)^2+(Z_3-z)^2\right]^{1/2}+ct_r+\chi_3 \\
\tilde{\rho}_4 &= \left[(X_4-x)^2+(Y_4-y)^2+(Z_4-z)^2\right]^{1/2}+ct_r+\chi_4
\end{aligned} \tag{1}$$

where

$$\chi_i = ct_{sv_i}+ct_{a_i}+e_i+m_i+\eta_i, \qquad i=1,..,4$$

and $\tilde{\rho}_i, i = 1,..,4$ are the measured pseudoranges from satellite i to the receiver antenna, X_i, Y_i, Z_i are the earth-centred-earth-fixed (ECEF) position coordinates of satellite i, x, y, z are the ECEF position coordinates of the receiver antenna, t_r is the receiver clock bias, t_{sv_i} is the clock bias of satellite i, t_{a_i} is the atmospheric delay, e_i represents the error in the broadcast ephemeris data, m_i represents the multipath error, η_i represents receiver tracking error noise, and c is the speed of light.

In equations 1, the pseudorange measurements are dependent on the receiver coordinates in a nonlinear manner. While closed form solutions are available, typically the solution is found by first linearizing the measurement equations, which can then be solved iteratively. The method described below relies on Newton's method.

Let us assume that $\chi_i = 0,\ i = 1,..,4$, then the relationships between the pseudoranges and the receiver position are

$$\begin{aligned}
\rho_1 &= \left[(X_1-x)^2+(Y_1-y)^2+(Z_1-z)^2\right]^{1/2}+ct_r \\
\rho_2 &= \left[(X_2-x)^2+(Y_2-y)^2+(Z_2-z)^2\right]^{1/2}+ct_r \\
\rho_3 &= \left[(X_3-x)^2+(Y_3-y)^2+(Z_3-z)^2\right]^{1/2}+ct_r \\
\rho_4 &= \left[(X_4-x)^2+(Y_4-y)^2+(Z_4-z)^2\right]^{1/2}+ct_r
\end{aligned} \tag{2}$$

Note that in the above equation it is effectively assumed that the only source of range bias is the receiver clock bias, which can be calculated and accounted for by solving four simultaneous equations, instead of the minimum of three that would be required if there was no range bias at all.

Defining the vector $\mathbf{x} = (x, y, z, ct_r)$ and linearizing Equations 2 results in

$$\begin{bmatrix} \rho_1(\mathbf{x}) \\ \rho_2(\mathbf{x}) \\ \rho_3(\mathbf{x}) \\ \rho_4(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \rho_1(\mathbf{x}_0) \\ \rho_2(\mathbf{x}_0) \\ \rho_3(\mathbf{x}_0) \\ \rho_4(\mathbf{x}_0) \end{bmatrix} + J \begin{bmatrix} (x - x_0) \\ (y - y_0) \\ (z - z_0) \\ (ct_r - (ct_r)_0) \end{bmatrix} + hot's \tag{3}$$

where $\mathbf{x}_0 = (x_0, y_0, z_0, (ct_r)_0)$ is the point of linearization, $hot's$ represents the higher order terms in the expansion of Equations 2, and

$$J = \begin{bmatrix} \frac{\partial \rho_1}{\partial x} & \frac{\partial \rho_1}{\partial y} & \frac{\partial \rho_1}{\partial z} & \frac{\partial \rho_1}{\partial (ct_r)} \\ \frac{\partial \rho_2}{\partial x} & \frac{\partial \rho_2}{\partial y} & \frac{\partial \rho_2}{\partial z} & \frac{\partial \rho_2}{\partial (ct_r)} \\ \frac{\partial \rho_3}{\partial x} & \frac{\partial \rho_3}{\partial y} & \frac{\partial \rho_3}{\partial z} & \frac{\partial \rho_3}{\partial (ct_r)} \\ \frac{\partial \rho_4}{\partial x} & \frac{\partial \rho_4}{\partial y} & \frac{\partial \rho_4}{\partial z} & \frac{\partial \rho_4}{\partial (ct_r)} \end{bmatrix}_{(x_0, y_0, z_0, (ct_r)_0)} \tag{4}$$

The partial derivatives in Equation 4 can easily be derived as

$$\begin{aligned} \frac{\partial \rho_i}{\partial x} &= \frac{-(X_i - x)}{\left[(X_i - x)^2 + (Y_i - y)^2 + (Z_i - z)^2\right]^{1/2}} \\ \frac{\partial \rho_i}{\partial y} &= \frac{-(Y_i - y)}{\left[(X_i - x)^2 + (Y_i - y)^2 + (Z_i - z)^2\right]^{1/2}} \\ \frac{\partial \rho_i}{\partial z} &= \frac{-(Z_i - z)}{\left[(X_i - x)^2 + (Y_i - y)^2 + (Z_i - z)^2\right]^{1/2}} \\ \frac{\partial \rho_i}{\partial (ct_r)} &= 1 \end{aligned} \tag{5}$$

Now, if we assume that $hot's = 0$ in Equations 3, and $\chi_i = 0,\ i = 1, .., 4$ in Equations 1, we can then form

$$\begin{bmatrix} \rho_1(\mathbf{x}) \\ \rho_2(\mathbf{x}) \\ \rho_3(\mathbf{x}) \\ \rho_4(\mathbf{x}) \end{bmatrix} - \begin{bmatrix} \tilde{\rho}_1 \\ \tilde{\rho}_2 \\ \tilde{\rho}_3 \\ \tilde{\rho}_4 \end{bmatrix} = \begin{bmatrix} \rho_1(\mathbf{x}_0) \\ \rho_2(\mathbf{x}_0) \\ \rho_3(\mathbf{x}_0) \\ \rho_4(\mathbf{x}_0) \end{bmatrix} - \begin{bmatrix} \tilde{\rho}_1 \\ \tilde{\rho}_2 \\ \tilde{\rho}_3 \\ \tilde{\rho}_4 \end{bmatrix} + J \begin{bmatrix} (x - x_0) \\ (y - y_0) \\ (z - z_0) \\ (ct_r - (ct_r)_0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Let

$$\boldsymbol{\varrho}(\mathbf{x}) = \begin{bmatrix} \rho_1(\mathbf{x}) \\ \rho_2(\mathbf{x}) \\ \rho_3(\mathbf{x}) \\ \rho_4(\mathbf{x}) \end{bmatrix} - \begin{bmatrix} \tilde{\rho}_1 \\ \tilde{\rho}_2 \\ \tilde{\rho}_3 \\ \tilde{\rho}_4 \end{bmatrix}$$

then we have

$$\begin{aligned}\boldsymbol{\varrho}(\mathbf{x}) &= \boldsymbol{\varrho}(\mathbf{x}_0) + J\begin{bmatrix}(x-x_0)\\(y-y_0)\\(z-z_0)\\(ct_r-(ct_r)_0)\end{bmatrix}\\ &= \begin{bmatrix}0\\0\\0\\0\end{bmatrix}\end{aligned}$$

Rearranging the above equation we have

$$\boldsymbol{\varrho}(\mathbf{x}_0) + J\begin{bmatrix}x\\y\\z\\ct_r\end{bmatrix} - J\begin{bmatrix}x_0\\y_0\\z_0\\(ct_r)_0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

Multiplying by J^{-1} gives

$$J^{-1}\boldsymbol{\varrho}(\mathbf{x}_0) + \begin{bmatrix}x\\y\\z\\ct_r\end{bmatrix} - \begin{bmatrix}x_0\\y_0\\z_0\\(ct_r)_0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

and rearranging again gives

$$\begin{bmatrix}x\\y\\z\\ct_r\end{bmatrix} = \begin{bmatrix}x_0\\y_0\\z_0\\(ct_r)_0\end{bmatrix} - J^{-1}\boldsymbol{\varrho}(\mathbf{x}_0)$$

which is now in a suitable form for applying Newton's method (also known as the Newton-Raphson method) by just replacing $\mathbf{x} = (x, y, z, ct_r)$ with $\mathbf{x}_{j+1} = \left(x_{j+1}, y_{j+1}, z_{j+1}, (ct_r)_{j+1}\right)$ and $\mathbf{x}_0 = (x_0, y_0, z_0, (ct_r)_0)$ with $\mathbf{x}_j = \left(x_j, y_j, z_j, (ct_r)_j\right)$, $j = 0, 1, .., N$ resulting in

$$\mathbf{x}_{j+1} = \mathbf{x}_j - J^{-1}\boldsymbol{\varrho}(\mathbf{x}_j) \tag{6}$$

We simply start with an initial guess for $\mathbf{x}_0 = (x_0, y_0, z_0, (ct_r)_0)$ and iterate till convergence is reached. A simple test for convergence is $\|\mathbf{x}_{j+1} - \mathbf{x}_j\| < \epsilon$, where ϵ is set to a small positive value.

If measurements from more than four satellites are available, then J^{-1} in Equation 6 can be replaced with $\left(J^T J\right)^{-1} J^T$ to give the least squares solution, resulting in

$$\mathbf{x}_{j+1} = \mathbf{x}_j - \left(J^T J\right)^{-1} J^T \boldsymbol{\varrho}(\mathbf{x}_j) \tag{7}$$

Equation 7 is referred to as the Gauss-Newton method. Note that Equation 7 assumes that the pseudorange measurement errors have identical variances.

2.2 Receiver Clock Bias Dynamic Model

One of the components of the Kalman filter models that are developed in this report is the receiver clock bias model. The state space model used for the receiver clock bias is that described on page 152 of [1]. The discrete time state transition equation is

$$\begin{bmatrix} t_r(k+1) \\ \dot{t}_r(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} t_r(k) \\ \dot{t}_r(k) \end{bmatrix} + \begin{bmatrix} \omega_{d\phi}(k) \\ \omega_{df}(k) \end{bmatrix}$$

where T is the sampling period, and k is the time index.

The covariance matrix associated with the *disctrete-time* process noise vector $\begin{bmatrix} \omega_{d\phi}(k) & \omega_{df}(k) \end{bmatrix}^T$ is

$$Q_{d_t}(k) = \begin{bmatrix} S_\phi T + \frac{T^3}{3} S_f & \frac{T^2}{2} S_f \\ \frac{T^2}{2} S_f & S_f T \end{bmatrix}$$

where S_ϕ and S_f are the power spectral densities of ω_ϕ and ω_f (the *continuous-time* process noises) respectively. An example value of the discrete time process noise covariance matrix, scaled to metres, is shown on page 153 of [1]. It is

$$Q_d(k) = c^2 Q_{d_t}(k) = \begin{bmatrix} 0.0114 & 0.0019 \\ 0.0019 & 0.0039 \end{bmatrix} \tag{8}$$

2.3 Plant and Measurement Equations for a Stationary Receiver

Before going further, a comment on notation is required. Many of the equations described in this report contain matrices and vectors whose elements are functions of time (represented by the time index k). To shorten the equations somewhat, a shorthand notation is used where appropriate; viz., consider an $m \times n$ matrix A, with elements $a_{ij}(k)$, $i = 1, .., m$, $j = 1, .., n$, then

$$\begin{bmatrix} a_{11} & . & . & . & a_{1n} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{m1} & . & . & . & a_{mn} \end{bmatrix}_k \equiv \begin{bmatrix} a_{11}(k) & . & . & . & a_{1n}(k) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{m1}(k) & . & . & . & a_{mn}(k) \end{bmatrix}$$

As can be found on page 105 of [1] (although full details are not given there), the discrete time state transition equation that is used for the stationary receiver case is

$$\mathbf{x}(k+1) = F(k)\,\mathbf{x}(k) + \mathbf{v}(k)$$

where

$$\mathbf{x}(k) \triangleq [x, y, z, r_{t_r}, \dot{r}_{t_r}]_k^T, \quad r_{t_r}(k) \triangleq c t_r(k) \tag{9}$$

$F(k)$ is the state transition matrix, $\mathbf{v}(k)$, $k = 0, 1, ...$, is a sequence of five dimensional zero mean white Gaussian process noise, and

$$F(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

The associated process noise covariance matrix (with the clock bias scaled to metres) that is shown on page 105 of [1] as an example is

$$Q_v(k) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.07 & 0.04 \\ 0 & 0 & 0 & 0.04 & 0.08 \end{bmatrix}$$

Note that the first three elements in the leading diagonal of the above matrix are zero; this is because the model assumes that the receiver is stationary.

The above covariance matrix is somewhat different to what would result from Equation 8, i.e.:

$$Q_v(k) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0114 & 0.0019 \\ 0 & 0 & 0 & 0.0019 & 0.0039 \end{bmatrix} \tag{11}$$

This latter covariance matrix is used in the simulations. The most appropriate values for the four bottom right hand elements of the matrix, of course, depend on the details of the particular receiver that is being modelled.

It is convenient to write the measurement equation in component form. In this form it is

$$\begin{aligned} \rho_i(k+1) &= \left[(X_i(k+1) - x(k+1))^2 + (Y_i(k+1) - y(k+1))^2 + (Z_i(k+1) - z(k+1))^2\right]^{1/2} \\ &\quad + r_{t_r}(k+1) \\ \text{for } i &= 1, .., N_s \end{aligned}$$

Note that the measurement vector at time k is

$$\rho(k) = \begin{bmatrix} \rho_1(k) & \rho_2(k) & . & . & \rho_{N_s}(k) \end{bmatrix}^T \tag{12}$$

For small increments in $\Delta\mathbf{x}$ we can linearize as follows. Let $\Delta\mathbf{x}(k+1) = \mathbf{x}(k+1) - \mathbf{x}(k)$, then in component form we can write

$$\Delta\rho_i(k+1) = H_i(k+1)\,\Delta\mathbf{x}(k+1) + v_i(k+1)$$

or in vector form it is

$$\Delta\rho(k+1) = H(k+1)\,\Delta\mathbf{x}(k+1) + \mathbf{v}(k+1)$$

where

$$H\left(k+1\right)=\left[\begin{array}{ccccc}\frac{\partial\rho_1}{\partial x} & \frac{\partial\rho_1}{\partial y} & \frac{\partial\rho_1}{\partial z} & \frac{\partial\rho_1}{\partial r_{t_r}} & 0\\ \frac{\partial\rho_2}{\partial x} & \frac{\partial\rho_2}{\partial y} & \frac{\partial\rho_2}{\partial z} & \frac{\partial\rho_2}{\partial r_{t_r}} & 0\\ . & . & . & . & .\\ . & . & . & . & .\\ \frac{\partial\rho_{N_s}}{\partial x} & \frac{\partial\rho_{N_s}}{\partial y} & \frac{\partial\rho_{N_s}}{\partial z} & \frac{\partial\rho_{N_s}}{\partial r_{t_r}} & 0\end{array}\right]_{k+1} \tag{13}$$

is the $5\times N_s$ dimensional measurement matrix, and N_s is the number of satellite-to-receiver pseudorange measurements at time k.

The partial derivatives in the above matrix are

$$\begin{aligned}\frac{\partial\rho_i}{\partial x} &= \frac{-\left(X_i-x\right)}{\left[\left(X_i-x\right)^2+\left(Y_i-y\right)^2+\left(Z_i-z\right)^2\right]^{1/2}}\\ \frac{\partial\rho_i}{\partial y} &= \frac{-\left(Y_i-y\right)}{\left[\left(X_i-x\right)^2+\left(Y_i-y\right)^2+\left(Z_i-z\right)^2\right]^{1/2}}\\ \frac{\partial\rho_i}{\partial z} &= \frac{-\left(Z_i-z\right)}{\left[\left(X_i-x\right)^2+\left(Y_i-y\right)^2+\left(Z_i-z\right)^2\right]^{1/2}}\\ \frac{\partial\rho_i}{\partial r_{t_r}} &= 1 \qquad\qquad \text{where } i=1,..,N_s\end{aligned}$$

The corresponding measurement noise covariance matrix is

$$R\left(k\right)=\left[\begin{array}{ccccc}\sigma^2_{r_1} & 0 & . & . & 0\\ 0 & \sigma^2_{r_2} & . & . & 0\\ . & . & . & . & .\\ . & . & . & . & .\\ 0 & 0 & . & . & \sigma^2_{r_{Ns}}\end{array}\right]_k \tag{14}$$

where $\sigma^2_{r_i}\left(k\right),\ i=1,..,N_s$ is measurement noise variance for the measurement from satellite i at time k.

2.4 Plant and Measurement Equations for a Low Dynamics Receiver

A short description of the continuous time low dynamics receiver model can be found on page 243 of [1] (although full details are not given there). We need to use a discrete time model; a good approximation to the corresponding discrete time model is as follows [2, Section 6.3.2]. In this model the receiver's acceleration is modelled using piecewise constant white acceleration noise. The discrete-time state transition equation is

$$\mathbf{x}\left(k+1\right)=F\left(k\right)\mathbf{x}\left(k\right)+\Gamma\left(k\right)\mathbf{v}\left(k\right)$$

where $\Gamma(k)$ is noise gain at time k, $\mathbf{x}(k) \triangleq [x, \dot{x}, y, \dot{y}, z, \dot{z}, r_{t_r}, \dot{r}_{t_r}]_k^T$, $r_{t_r}(k) \triangleq ct_r(k)$, and

$$F(k) = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

$$\Gamma(k) = \begin{bmatrix} 0.5T^2 & 0 & 0 & 0 & 0 \\ T & 0 & 0 & 0 & 0 \\ 0 & 0.5T^2 & 0 & 0 & 0 \\ 0 & T & 0 & 0 & 0 \\ 0 & 0 & 0.5T^2 & 0 & 0 \\ 0 & 0 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{16}$$

$$\mathbf{v}(k) = \begin{bmatrix} \ddot{x} & \ddot{y} & \ddot{z} & c\omega_{d\phi} & c\omega_{df} \end{bmatrix}_k^T \tag{17}$$

Let us now determine the process noise covariance matrix associated with this model. First let us consider the covariance matrix associated with $\mathbf{v}(k)$, i.e., $Q_v(k) = E\left\{\mathbf{v}(k)\,\mathbf{v}(k)^T\right\}$. We have

$$Q_v(k) = \begin{bmatrix} \sigma_{\ddot{x}}^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\ddot{y}}^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\ddot{z}}^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{r_\phi}^2 & \sigma_{r_\phi}\sigma_{r_f} \\ 0 & 0 & 0 & \sigma_{r_f}\sigma_{r_\phi} & \sigma_{r_f}^2 \end{bmatrix} \tag{18}$$

where $\sigma_{\ddot{x}}$, $\sigma_{\ddot{y}}$ and $\sigma_{\ddot{z}}$ are the standard deviations of the x, y, and z components of the acceleration noise, respectively, σ_{r_ϕ} and σ_{r_f} are the standard deviation of the clock bias process noise due to the phase error (scaled to metres), and that due to frequency error, respectively, and $\sigma_{r_\phi}\sigma_{r_f} = \sigma_{r_f}\sigma_{r_\phi}$ are their covariances. Note that the components of the acceleration noise are assumed to be independent of each other and the clock bias noises.

Now consider the process noise when multiplied by the gain matrix $\Gamma(k)$, i.e., $Q_{\Gamma v}(k) = E\left\{[\Gamma(k)\,\mathbf{v}(k)]\,[\Gamma(k)\,\mathbf{v}(k)]^T\right\}$. The resulting process noise covariance matrix can be shown to be

$$Q_{\Gamma v}(k) = \Gamma(k)\,Q_v(k)\,\Gamma(k)^T \tag{19}$$

The measurement equation is the same as in Section 2.3, but with the measurement matrix now being

$$H(k+1) = \begin{bmatrix} \frac{\partial \rho_1}{\partial x} & 0 & \frac{\partial \rho_1}{\partial y} & 0 & \frac{\partial \rho_1}{\partial z} & 0 & \frac{\partial \rho_1}{\partial r_{t_r}} & 0 \\ \frac{\partial \rho_2}{\partial x} & 0 & \frac{\partial \rho_2}{\partial y} & 0 & \frac{\partial \rho_2}{\partial z} & 0 & \frac{\partial \rho_2}{\partial r_{t_r}} & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ \frac{\partial \rho_{N_s}}{\partial x} & 0 & \frac{\partial \rho_{N_s}}{\partial y} & 0 & \frac{\partial \rho_{N_s}}{\partial z} & 0 & \frac{\partial \rho_{N_s}}{\partial r_{t_r}} & 0 \end{bmatrix}_{k+1} \tag{20}$$

and the corresponding measurement noise covariance matrix is as in Equation 14.

2.5 Plant and Measurement Equations for a High Dynamics Receiver

A short description of the continuous time high dynamics receiver model can be found on page 244 of [1] (although full details are not given there). We need to use a discrete time model; for this we will use the Wiener process acceleration model described in Section 6.2.3 of [2]; this model is also sometimes called the white noise jerk model. Note that this is a discretized continuous time model, as opposed to the direct discrete time model of Section 6.3.3 of [2]. Which of these two models to use is a matter of choice; both are an approximation to the actual continuous time dynamics of the receiver.

The discrete-time state transition equation is

$$\mathbf{x}(k+1) = F(k)\,\mathbf{x}(k) + \mathbf{v}(k)$$

where

$$\mathbf{x}(k) \triangleq [x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}, r_{t_r}, \dot{r}_{t_r}]_k^T, \quad r_{t_r}(k) \triangleq c t_r(k) \tag{21}$$

and

$$F(k) = \begin{bmatrix} 1 & T & \frac{1}{2}T^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & \frac{1}{2}T^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T & \frac{1}{2}T^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{22}$$

With regard to the process noise model, let us first consider the x component for the continuous time system. The changes in acceleration are modelled by a continuous time zero-mean white noise as follows

$$\dddot{x}(t) = \tilde{v}_x(t)$$

Note that the acceleration is a Wiener process, and its derivative, the jerk, is represented by a white noise model. The changes in acceleration over a sampling period T are of

the order of $\sqrt{\tilde{q}_x T}$, where $\tilde{q}_x$ is the power spectral density of the continuous time process noise $\tilde{v}_x(t)$. The same can be done for the y and z components. Considering all three components as well as the receiver clock error noise model of Section 2.2, we have the following discrete time process noise covariance matrix (i.e., $Q_v(k) \triangleq E\left\{\mathbf{v}(k)\,\mathbf{v}(k)^T\right\}$)

$$Q_v(k) = \begin{bmatrix} & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & Q_x & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & & & & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & & Q_y & & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & & & & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & Q_z & & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{r_\phi} & \sigma_{r_\phi}\sigma_{r_f} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{r_f}\sigma_{r_\phi} & \sigma^2_{r_f} \end{bmatrix} \tag{23}$$

where

$$Q_x = \begin{bmatrix} \frac{1}{20}T^5 & \frac{1}{8}T^4 & \frac{1}{6}T^3 \\ \frac{1}{8}T^4 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \tilde{q}_x$$

$$Q_y = \begin{bmatrix} \frac{1}{20}T^5 & \frac{1}{8}T^4 & \frac{1}{6}T^3 \\ \frac{1}{8}T^4 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \tilde{q}_y$$

$$Q_z = \begin{bmatrix} \frac{1}{20}T^5 & \frac{1}{8}T^4 & \frac{1}{6}T^3 \\ \frac{1}{8}T^4 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix} \tilde{q}_z$$

and $\tilde{q}_x, \tilde{q}_y$ and $\tilde{q}_z$ are the power spectral densities of the x, y and z components of the continuous time jerk noise, i.e., $\tilde{v}_x(t), \tilde{v}_y(t)$ and $\tilde{v}_t(t)$,respectively. Note that the components of the jerk noise are assumed to be independent of each other and the clock bias noises.

The measurement equation is the same as in Section 2.3, but with the measurement matrix now being

$$H(k+1) = \begin{bmatrix} \frac{\partial\rho_1}{\partial x} & 0 & 0 & \frac{\partial\rho_1}{\partial y} & 0 & 0 & \frac{\partial\rho_1}{\partial z} & 0 & 0 & \frac{\partial\rho_1}{\partial r_{t_r}} & 0 \\ \frac{\partial\rho_2}{\partial x} & 0 & 0 & \frac{\partial\rho_2}{\partial y} & 0 & 0 & \frac{\partial\rho_2}{\partial z} & 0 & 0 & \frac{\partial\rho_2}{\partial r_{t_r}} & 0 \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ \frac{\partial\rho_{N_s}}{\partial x} & 0 & 0 & \frac{\partial\rho_{N_s}}{\partial y} & 0 & 0 & \frac{\partial\rho_{N_s}}{\partial z} & 0 & 0 & \frac{\partial\rho_{N_s}}{\partial r_{t_r}} & 0 \end{bmatrix}_{k+1} \tag{24}$$

and the corresponding measurement noise covariance matrix is as in Equation 14.

2.6 Extended Kalman Filter Equations - General Case

An extended Kalman filter [2, Section 10.3] is used as the estimation algorithm in this work. The algorithm is summarized in the sequel; before summarizing the algorithm, some

definitions will first be given, as follows. Let

$$\hat{\mathbf{x}}(j|k) \triangleq E\left[\mathbf{x}(j)|\mathbf{Z}^k\right]$$

where

$$\mathbf{Z}^k \triangleq \{\mathbf{z}(j),\ j \leq k\}$$

denotes the sequence of observations available at time k, and

$$E\left[\mathbf{x}(j)|\mathbf{Z}^k\right]$$

is the conditional expectation of $\mathbf{x}(j)$ at time j given $\mathbf{Z}^k$.

If $j = k$, then $\hat{\mathbf{x}}(j|k)$ is the *estimate* of the state (also called the *filtered* value); if $j = k+1$ then $\hat{\mathbf{x}}(j|k)$ is the *predicted* value (one-step) of the state. The state estimation error at time k is defined as

$$\tilde{\mathbf{x}}(k|k) \triangleq \mathbf{x}(k) - \hat{\mathbf{x}}(k|k)$$

The state prediction error at time k is defined as

$$\tilde{\mathbf{x}}(k+1|k) \triangleq \mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1|k)$$

The state estimation covariance matrix (i.e., the covariance associated with the estimate $\hat{\mathbf{x}}(k|k)$) at time k is

$$P(k|k) \triangleq E\left[\tilde{\mathbf{x}}(k|k)\,\tilde{\mathbf{x}}(k|k)^T|\mathbf{Z}^k\right]$$

The state prediction covariance matrix (i.e., the covariance associated with the prediction $\hat{\mathbf{x}}(k+1|k)$) at time k is

$$P(k+1|k) \triangleq E\left[\tilde{\mathbf{x}}(k+1|k)\,\tilde{\mathbf{x}}(k+1|k)^T|\mathbf{Z}^k\right]$$

The predicted measurement (one-step) is

$$\hat{\mathbf{z}}(k+1|k) \triangleq E\left[\mathbf{z}(k+1)|\mathbf{Z}^k\right]$$

The measurement prediction error (also called the innovation or residual) is defined as

$$\nu(k+1) \triangleq \tilde{\mathbf{z}}(k+1|k) \triangleq \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$$

The measurement prediction covariance matrix or innovation covariance matrix is

$$S(k+1) \triangleq E\left[\tilde{\mathbf{z}}(k+1|k)\,\tilde{\mathbf{z}}(k+1|k)^T|\mathbf{Z}^k\right]$$

The Kalman filter gain is

$$W(k+1) \triangleq P(k+1|k)\,H(k+1)^T\,S(k+1)^{-1}$$

Now consider the nonlinear system with dynamics

$$\mathbf{x}(k+1) = f[k, \mathbf{x}(k), \mathbf{u}(k)] + \mathbf{v}(k)$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ is a known input, $\mathbf{v}$ is the process noise, which is assumed to be additive, zero mean, and white, and f is a vector-valued and possibly time varying non-linear function.

Let the measurement equation be

$$\mathbf{z}(k) = h[k, \mathbf{x}(k)] + \mathbf{w}(k)$$

where $\mathbf{w}$ is the measurement noise, which is additive, zero mean, white, and uncorrelated with the process noise, and the function h is also vector valued and can be time varying.

The Extended Kalman filter is a suboptimal recursive algorithm for the above system, as follows. First, we start with the initial estimate $\hat{\mathbf{x}}(0|0)$ of $\mathbf{x}(0)$ and the associated initial covariance $P(0|0)$, both assumed to be available. Then, for estimation of the state of the system, starting with the state estimate $\hat{\mathbf{x}}(k|k)$ at t_k we have

State Prediction:

$$\hat{\mathbf{x}}(k+1|k) = f[k, \hat{\mathbf{x}}(k|k), \mathbf{u}(k)]$$

Measurement Prediction:

$$\hat{\mathbf{z}}(k+1|k) = h[k+1, \hat{\mathbf{x}}(k+1|k)]$$

Measurement Residual:

$$\boldsymbol{\nu}(k+1) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)$$

Updated State Estimate:

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + W(k+1)\boldsymbol{\nu}(k+1)$$

For state covariance computation, starting with the state covariance $P(k|k)$at t_k we have

Evaluation of Jacobians:

$$\begin{aligned} F(k) &= \left.\frac{\partial f(k)}{\partial x}\right|_{x=\hat{x}(k|k)} \\ H(k+1) &= \left.\frac{\partial h(k+1)}{\partial x}\right|_{x=\hat{x}(k+1|k)} \end{aligned}$$

State Prediction Covariance:

$$P(k+1|k) = F(k) P(k|k) F^T(k) + Q(k)$$

Residual Covariance:

$$S(k+1) = H(k+1) P(k+1|k) H(k+1)^T + R(k+1)$$

Filter Gain:

$$W(k+1) = P(k+1|k) H(k+1)^T S(k+1)^{-1}$$

Updated State Covariance:

$$P(k+1|k+1) = P(k+1|k) - W(k+1) S(k+1) W(k+1)^T$$

2.7 Extended Kalman Filter Equations for a Stationary Receiver

The extended Kalman filter for the stationary receiver case described in Section 2.3 will now be described.

We start with the initial estimate $\hat{\mathbf{x}}(0|0)$ of $\mathbf{x}(0)$ which is determined using the equations presented in Section 2.1. Note that the initial estimate doesn't give any information about the rate of change of receiver clock bias - this needs to be guessed. Our initial guess is that it's zero. We also need to calculate the covariance, $P(0|0)$, of the initial estimate $\hat{\mathbf{x}}(0|0)$, which can be determined as follows. Let P_A be the covariance associated with the estimate of (x, y, z, ct_r) obtained using Equation 7. Referring to Equation 4.11 in section 4.1.1 of [1] we have

$$P_A = \left(J(0)^T R(0)^{-1} J(0) \right)^{-1} \tag{25}$$

where

$$J(0) = \left[\begin{array}{cccc} \frac{\partial \rho_1}{\partial x} & \frac{\partial \rho_1}{\partial y} & \frac{\partial \rho_1}{\partial z} & \frac{\partial \rho_1}{\partial r_{t_r}} \\ \frac{\partial \rho_2}{\partial x} & \frac{\partial \rho_2}{\partial y} & \frac{\partial \rho_2}{\partial z} & \frac{\partial \rho_2}{\partial r_{t_r}} \\ . & . & . & . \\ . & . & . & . \\ \frac{\partial \rho_{N_s}}{\partial x} & \frac{\partial \rho_{N_s}}{\partial y} & \frac{\partial \rho_{N_s}}{\partial z} & \frac{\partial \rho_{N_s}}{\partial r_{t_r}} \end{array} \right] \Bigg|_{\mathbf{x}=\hat{\mathbf{x}}(0|0)} \tag{26}$$

$$R(0) = \left[\begin{array}{ccccc} \sigma_{r_1}^2 & 0 & . & . & 0 \\ 0 & \sigma_{r_2}^2 & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & \sigma_{r_{Ns}}^2 \end{array} \right] \Bigg|_{x=\hat{x}(0|0)} \tag{27}$$

Then

$$P(0|0) = \left[\begin{array}{ccccc} P_{A_{11}} & P_{A_{12}} & P_{A_{13}} & P_{A_{14}} & 0 \\ P_{A_{21}} & P_{A_{22}} & P_{A_{23}} & P_{A_{24}} & 0 \\ P_{A_{31}} & P_{A_{32}} & P_{A_{33}} & P_{A_{34}} & 0 \\ P_{A_{41}} & P_{A_{42}} & P_{A_{43}} & P_{A_{44}} & 0 \\ 0 & 0 & 0 & 0 & 2Q_{d_{22}} \end{array} \right]$$

Note that in the above matrix the covariances of the fifth column and fifth row can't be determined from the measurements made on the initial startup, hence, as a reasonable guess, they are all set to zero, and the variance in the bottom right hand corner equal to $2Q_{d_{22}}$, i.e., twice the variance of element 22 of the discrete time receiver clock process noise covariance matrix, scaled to metres, in Equation 8.

Then, for estimation of the state of the system, starting with the state estimate $\hat{\mathbf{x}}(k|k)$ at t_k we have

State Prediction:

$$\hat{\mathbf{x}}(k+1|k) = F(k)\,\hat{\mathbf{x}}(k|k) \tag{28}$$

Measurement Prediction:

$$\hat{\rho}_i\left(k+1|k\right)=\left[\left(X_i\left(k+1\right)-\hat{x}\left(k+1|k\right)\right)^2+\left(Y_i\left(k+1\right)-\hat{y}\left(k+1|k\right)\right)^2+\left(Z_i\left(k+1\right)-\hat{z}\left(k+1|k\right)\right)^2\right]^{1/2} \tag{29}$$

$$+\hat{r}_{t_r}\left(k+1|k\right)$$

$$\text{for } i=1,..,N_s$$

Measurement Residual:

$$\nu\left(k+1\right)=\rho\left(k+1\right)-\hat{\rho}\left(k+1|k\right) \tag{30}$$

Updated State Estimate:

$$\hat{\mathbf{x}}\left(k+1|k+1\right)=\hat{\mathbf{x}}\left(k+1|k\right)+W\left(k+1\right)\nu\left(k+1\right) \tag{31}$$

For state covariance computation, starting with the state covariance $P\left(k|k\right)$at t_k we have

Evaluation of Jacobians:

$$H\left(k+1\right)=\left[\begin{array}{ccccc}\frac{\partial\rho_1}{\partial x} & \frac{\partial\rho_1}{\partial y} & \frac{\partial\rho_1}{\partial z} & \frac{\partial\rho_1}{\partial r_{t_r}} & 0\\ \frac{\partial\rho_2}{\partial x} & \frac{\partial\rho_2}{\partial y} & \frac{\partial\rho_2}{\partial z} & \frac{\partial\rho_2}{\partial r_{t_r}} & 0\\ . & . & . & . & .\\ . & . & . & . & .\\ \frac{\partial\rho_{N_s}}{\partial x} & \frac{\partial\rho_{N_s}}{\partial y} & \frac{\partial\rho_{N_s}}{\partial z} & \frac{\partial\rho_{N_s}}{\partial r_{t_r}} & 0\end{array}\right]\Bigg|_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)} \tag{32}$$

State Prediction Covariance:

$$P\left(k+1|k\right)=F\left(k\right)P\left(k|k\right)F^T\left(k\right)+Q_{\Gamma v}\left(k\right) \tag{33}$$

Residual Covariance:

$$S\left(k+1\right)=H\left(k+1\right)P\left(k+1|k\right)H\left(k+1\right)^T+R\left(k+1\right) \tag{34}$$

Filter Gain:

$$W\left(k+1\right)=P\left(k+1|k\right)H\left(k+1\right)^TS\left(k+1\right)^{-1} \tag{35}$$

Updated State Covariance:

$$P\left(k+1|k+1\right)=P\left(k+1|k\right)-W\left(k+1\right)S\left(k+1\right)W\left(k+1\right)^T \tag{36}$$

where $\mathbf{x}\left(k\right)$, $F\left(k\right)$, $Q_{\Gamma v}\left(k\right)$, $\rho\left(k\right)$ and $R\left(k\right)$ are defined in Equations 9, 10, 11, 12 and 14, respectively.

2.8 Extended Kalman Filter Equations for a Low Dynamics Receiver

The extended Kalman filter for the low dynamics receiver case will now be described.

Again, we start with the initial estimate $\hat{\mathbf{x}}(0|0)$ of $\mathbf{x}(0)$ which is determined using the equations presented in Section 2.1. Since the the initial estimate doesn't give any information about the rate of change of receiver clock bias, this needs to be guessed. Our initial guess is that it's zero. We also need to calculate the covariance, $P(0|0)$, of the initial estimate $\hat{\mathbf{x}}(0|0)$, which can be determined as follows. As in Section 2.7, Equation 25 is used to determine P_A, with $J(0)$ and $R(0)$ defined as in Equations 26 and 27, respectively. Note that P_A is the covariance associated with the estimate of (x, y, z, ct_r) obtained using Equation 7.

Then

$$P(0|0) = \begin{bmatrix} P_{A_{11}} & 0 & P_{A_{12}} & 0 & P_{A_{13}} & 0 & P_{A_{14}} & 0 \\ 0 & \sigma^2_{\dot{x}}(0) & 0 & 0 & 0 & 0 & 0 & 0 \\ P_{A_{21}} & 0 & P_{A_{22}} & 0 & P_{A_{23}} & 0 & P_{A_{24}} & 0 \\ 0 & 0 & 0 & \sigma^2_{\dot{y}}(0) & 0 & 0 & 0 & 0 \\ P_{A_{31}} & 0 & P_{A_{32}} & 0 & P_{A_{33}} & 0 & P_{A_{34}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{\dot{z}}(0) & 0 & 0 \\ P_{A_{41}} & 0 & P_{A_{42}} & 0 & P_{A_{43}} & 0 & P_{A_{44}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2Q_{d_{22}} \end{bmatrix}$$

Note that in the above matrix the covariances (i.e., the off-diagonal terms) of the second, fourth, sixth and eight row as well as the second, fourth, sixth and eighth column can't be determined from the measurements made on the initial startup, hence, as a reasonable assumption, they are all set to zero. The variances of the x, y and z components of the initial velocity estimate are assumed to be $\sigma^2_{\dot{x}}(0)$, $\sigma^2_{\dot{y}}(0)$ and $\sigma^2_{\dot{z}}(0)$ respectively, and the bottom right hand element of $P(0|0)$ is set to $2Q_{d_{22}}$, i.e., twice the variance of element 22 of the discrete time receiver clock process noise covariance matrix, scaled to metres, in Equation 8.

Then for estimation of the state of the system, starting with the state estimate $\hat{\mathbf{x}}(k|k)$ at t_k and for state covariance computation, starting with the state covariance $P(k|k)$ at t_k we use Equations 28 to 31 and 33 to 36, where $\mathbf{x}(k)$, $F(k)$, $\Gamma(k)$, $Q_{\Gamma v}(k)$, $\rho(k)$ and $R(k)$ are defined in Equations 2.4, 15, 16, 19, 12 and 14, respectively and $H(k+1)$ is given by

$$H(k+1) = \left[\begin{array}{cccccccc} \frac{\partial \rho_1}{\partial x} & 0 & \frac{\partial \rho_1}{\partial y} & 0 & \frac{\partial \rho_1}{\partial z} & 0 & \frac{\partial \rho_1}{\partial r_{t_r}} & 0 \\ \frac{\partial \rho_2}{\partial x} & 0 & \frac{\partial \rho_2}{\partial y} & 0 & \frac{\partial \rho_2}{\partial z} & 0 & \frac{\partial \rho_2}{\partial r_{t_r}} & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ \frac{\partial \rho_{N_s}}{\partial x} & 0 & \frac{\partial \rho_{N_s}}{\partial y} & 0 & \frac{\partial \rho_{N_s}}{\partial z} & 0 & \frac{\partial \rho_{N_s}}{\partial r_{t_r}} & 0 \end{array} \right] \Bigg|_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)} \tag{37}$$

2.9 Extended Kalman Filter Equations for a High Dynamics Receiver

The extended Kalman filter for the high dynamics receiver case will now be described. Again, we start with the initial estimate $\hat{\mathbf{x}}(0|0)$ of $\mathbf{x}(0)$ which is determined using the equations presented in Section 2.1. Since the the initial estimate doesn't give any information about the rate of change of receiver clock bias, this needs to be guessed. Our initial guess is that it's zero. We also need to calculate the covariance, $P(0|0)$, of the initial estimate $\hat{\mathbf{x}}(0|0)$, which can be determined as follows. As in Section 2.7, Equation 25 is used to determine P_A, with $J(0)$ and $R(0)$ defined as in Equations 26 and 27, respectively. Note that P_A is the covariance associated with the estimate of (x, y, z, ct_r) obtained using Equation 7. Then

$$P(0|0) = \begin{bmatrix} P_{A_{11}} & 0 & 0 & P_{A_{12}} & 0 & 0 & P_{A_{13}} & 0 & 0 & P_{A_{14}} & 0 \\ 0 & \sigma_{\dot{x}}^2(0) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\ddot{x}}^2(0) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_{A_{21}} & 0 & 0 & P_{A_{22}} & 0 & 0 & P_{A_{23}} & 0 & 0 & P_{A_{24}} & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{y}}^2(0) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\ddot{y}}^2(0) & 0 & 0 & 0 & 0 & 0 \\ P_{A_{31}} & 0 & 0 & P_{A_{32}} & 0 & 0 & P_{A_{33}} & 0 & 0 & P_{A_{34}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{\dot{z}}^2(0) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{\ddot{z}}^2(0) & 0 & 0 \\ P_{A_{41}} & 0 & 0 & P_{A_{42}} & 0 & 0 & P_{A_{43}} & 0 & 0 & P_{A_{44}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2Q_{d_{22}} \end{bmatrix}$$

In the above matrix, the covariances (i.e., the off-diagonal terms) of the second, third, fifth, sixth, eight, ninth and eleventh row as well as the second, third, fifth, sixth, eight, ninth and eleventh column can't be determined from the measurements made on the initial startup, hence, as a reasonable assumption, they are all set to zero. The variances of the x, y and z components of the initial velocity estimate are assumed to be $\sigma_{\dot{x}}^2(0)$, $\sigma_{\dot{y}}^2(0)$ and $\sigma_{\dot{z}}^2(0)$ respectively, the variances of the x, y and z components of the initial acceleration estimate are assumed to be $\sigma_{\ddot{x}}^2(0)$, $\sigma_{\ddot{y}}^2(0)$ and $\sigma_{\ddot{z}}^2(0)$ respectively, and the bottom right hand element of $P(0|0)$ is set to $2Q_{d_{22}}$, i.e., twice the variance of element 22 of the discrete time receiver clock process noise covariance matrix, scaled to metres, in Equation 8.

Then for estimation of the state of the system, starting with the state estimate $\hat{\mathbf{x}}(k|k)$ at t_k and for state covariance computation, starting with the state covariance $P(k|k)$ at t_k we use Equations 28 to 31 and 33 to 36, where $\mathbf{x}(k)$, $F(k)$, $Q_v(k)$, $\rho(k)$ and $R(k)$ are defined in Equations 21, 22, 23, 12 and 14, respectively and $H(k+1)$ is given by

$$H(k+1) = \left[\begin{array}{ccccccccccc} \frac{\partial \rho_1}{\partial x} & 0 & 0 & \frac{\partial \rho_1}{\partial y} & 0 & 0 & \frac{\partial \rho_1}{\partial z} & 0 & 0 & \frac{\partial \rho_1}{\partial r_{t_r}} & 0 \\ \frac{\partial \rho_2}{\partial x} & 0 & 0 & \frac{\partial \rho_2}{\partial y} & 0 & 0 & \frac{\partial \rho_2}{\partial z} & 0 & 0 & \frac{\partial \rho_2}{\partial r_{t_r}} & 0 \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ \frac{\partial \rho_{N_s}}{\partial x} & 0 & 0 & \frac{\partial \rho_{N_s}}{\partial y} & 0 & 0 & \frac{\partial \rho_{N_s}}{\partial z} & 0 & 0 & \frac{\partial \rho_{N_s}}{\partial r_{t_r}} & 0 \end{array} \right] \Bigg|_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)} \tag{38}$$

3 Testing of Kalman Filter Algorithms

In order to test the algorithms developed in the previous sections, simulations were written using the Matlab programming language. The aim of the simulations was to test the instantaneous and Kalman filter estimators for stability and in general determine if they are performing as expected. Simpifications were made in the scenarios considered, to the extent that was possible, while still achieving the aims of the testing. Modelling of the rotation of the earth, and movement of the satellites was not required for this stage of the testing and hence was not implemented in the simulations, i.e., it was assumed that the receiver was in an arbitrary inertial reference frame and the satellites were stationary in this frame, with a predefined geometric configuration relative to initial receiver position. The receiver-satellite geometry was made consistent with what could be expected for an actual situation. In the testing of the Kalman filter using a stationary receiver model, the receiver was kept stationary, whereas, for the other Kalman filters, the receiver was in motion. Note that further testing with more sophisticated scenarios, utilizing actual satellite trajectories would be desirable to fully test the filters which were developed.

3.1 Kalman Filter with Stationary Receiver Model

The Kalman filter using the stationary receiver model, which was described in Section 2.7, was coded in Matlab and tested by simulation. The details of the simulations are as follows.

The Kalman filter update rate was set to $T = 1$ sec. The number of updates that the Kalman filter was run for was 300 to determine short term performance, and then 3600 to determine performance over a longer period of time (1 hour). The latter served as a more extended test to determine if there are any issues associated with filter divergence due to numerical round-off errors, which is a common problem in Kalman filter implementations. The "actual" receiver range measurement error standard deviation was set to $\sigma_{r_a} = 5$m. The receiver range measurement error standard deviation as modelled by the Kalman filter was set to $\sigma_{r_m} = 5$m. Six satellites were modelled in the simulations; their positions were $\mathbf{x}_{s_1} = (0.9390, -1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_2} = (1.7648, -0.6423, 1.8781) \times 10^7$m, $\mathbf{x}_{s_3} = (1.7648, 0.6423, 1.8781) \times 10^7$m, $\mathbf{x}_{s_4} = (0.9390, 1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_5} = (0.9390, -1.6265, -1.8781) \times 10^7$m, $\mathbf{x}_{s_6} = (0.9390, 1.6265, -1.8781) \times 10^7$m. The receiver position was $\mathbf{x}_r = \big(6.371 \times 10^6, 100, 150\big)$m.

Figures 1 and 2 show the results for the case of 300 updates. Figure 1 shows the errors in the instantaneous position estimates; the instantaneous estimates were calculated using the Gauss-Newton method as described in Section 2.1. Figure 2 shows the errors in the Kalman filter estimates. Note that the error is defined to be the distance between the estimated position and the actual position. Looking at Figure 2, we see that the Kalman filter is very quickly reducing the position estimate errors to well below that of the instantaneous estimates.

Figures 3 and 4 show the results for the case of 3600 updates; The primary reason for doing the simulation that resulted in these figures was to test for divergence that may result from numerical round-off errors. As can be seen from the two figures, convergence continued for the duration of the simulation.

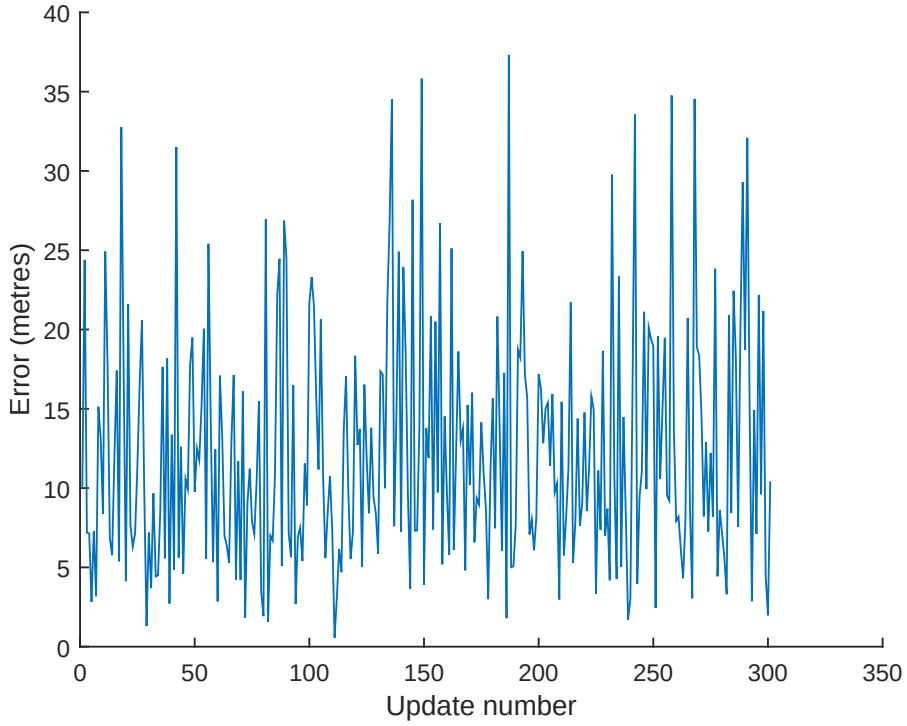

Figure 1: Position estimation error of instantaneous estimates (stationary receiver model, 300 updates).

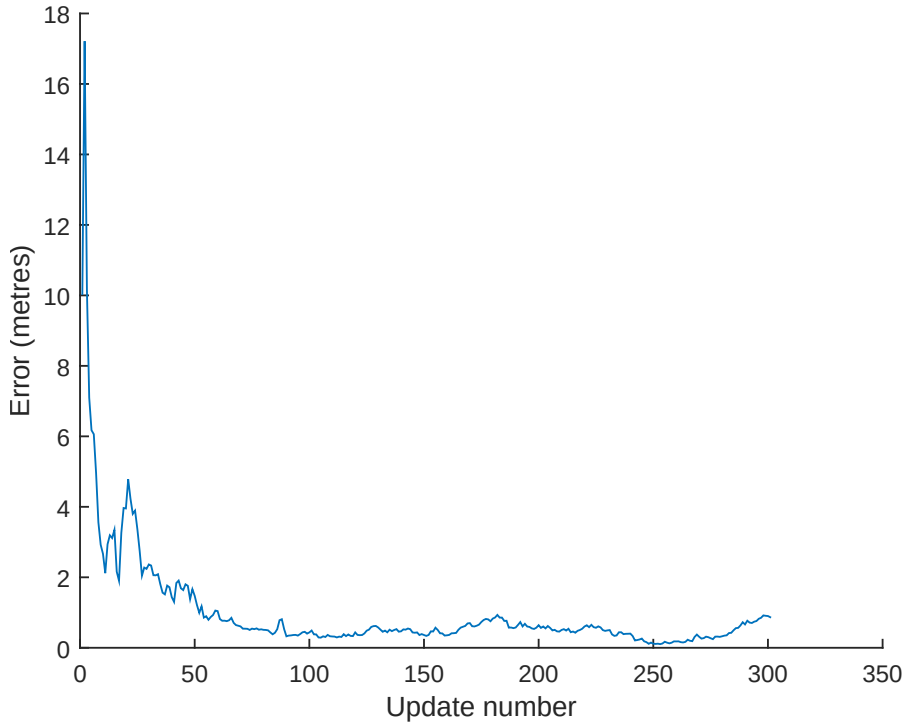

Figure 2: Kalman filter position estimation error (300 updates), using stationary receiver model.

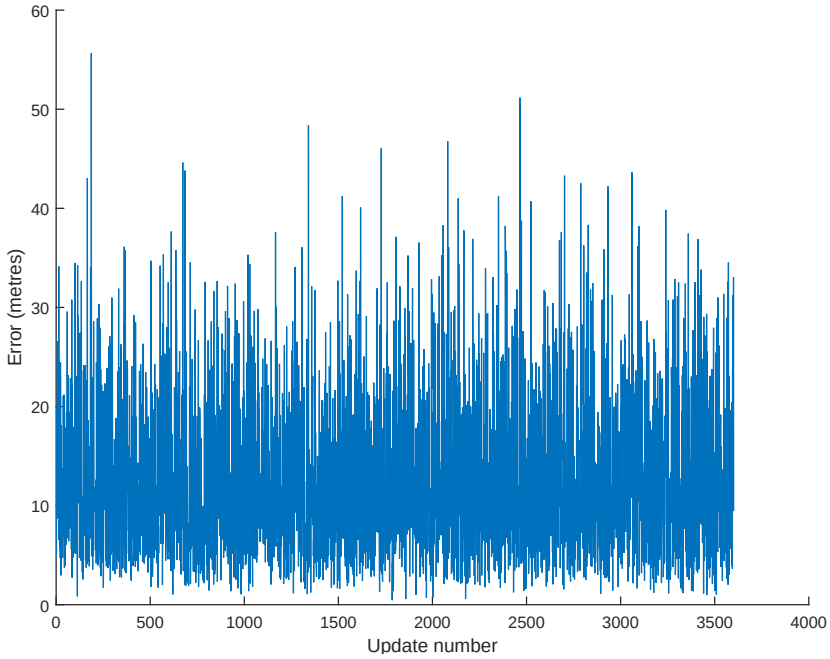

Figure 3: Position estimation error of instantaneous estimates (stationary receiver model, 3600 updates).

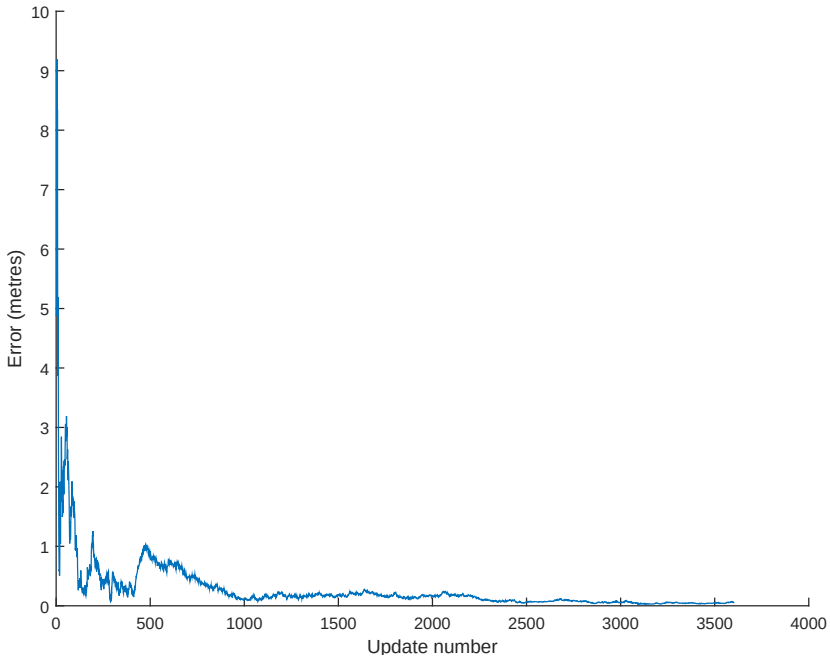

Figure 4: Kalman filter position estimation error (3600 updates), using stationary receiver model.

3.2 Kalman Filter with Low Dynamics Receiver Model

The Kalman filter using the low dynamics receiver model, which was described in Section 2.8, was coded in Matlab and tested by simulation. The details of the simulations are as follows.

The Kalman filter update rate was set to $T = 1$ sec. The number of updates that the Kalman filter was run for was 300 to determine short term performance, and then 3600 to determine performance over a longer period of time (1 hour). The "actual" receiver range measurement error standard deviation was set to $\sigma_{r_a} = 5$m. The receiver range measurement error standard deviation as modelled by the Kalman filter was set to $\sigma_{r_m} = 5$m. The (acceleration) process noise standard deviation in the Kalman filter was set to $\sigma_{\ddot{x}} = \sigma_{\ddot{y}} = \sigma_{\ddot{z}} = 0.2$ m/s^2. Six satellites were modelled in the simulation; their positions were $\mathbf{x}_{s_1} = (0.9390, -1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_2} = (1.7648, -0.6423, 1.8781) \times 10^7$m, $\mathbf{x}_{s_3} = (1.7648, 0.6423, 1.8781) \times 10^7$m, $\mathbf{x}_{s_4} = (0.9390, 1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_5} = (0.9390, -1.6265, -1.8781) \times 10^7$m, $\mathbf{x}_{s_6} = (0.9390, 1.6265, -1.8781) \times 10^7$m. The initial receiver position was $\mathbf{x}_r = \big(6.371 \times 10^6, 100, 150\big)$m; however, in these simulations the receiver was not stationary, but instead had a velocity of $\mathbf{v}_r = (0, 30, 40)$ m/s for the duration of the simulations.

Figures 5 and 6 show the results for the case of 300 updates. Figure 5 shows the errors in the instantaneous position estimates. Figure 6 shows the errors in the Kalman filter estimates. Looking at Figure 6, we see that the Kalman filter is quickly reducing the position estimate errors to below that of the instantaneous estimates. Note, however, that the errors in the position estimates are higher than was the case for the filter using the stationary receiver model. This is to be expected as this filter allows for receiver motion, and hence does not filter the position estimates as heavily. Of course, this filter has the advantage that it can track the position and velocity of a moving receiver, whereas the filter with the stationary receiver model is not designed for a moving receiver, and hence would not be expected to function well for that case.

Figures 7 and 8 show the results for the case of 3600 updates. As can be seen from the two figures, convergence continued for the duration of the simulation.

3.3 Kalman Filter with High Dynamics Receiver Model

The Kalman filter using the high dynamics receiver model, which was described in Section 2.9, was coded in Matlab and tested by simulation. The details of the simulations are as follows.

The Kalman filter update rate was set to $T = 1$ sec. The number of updates that the Kalman filter was run for was 300 to determine short term performance, and then 3600 to determine performance over a longer period of time (1 hour). The "actual" receiver range measurement error standard deviation was set to $\sigma_{r_a} = 5$m. The receiver range measurement error standard deviation as modelled by the Kalman filter was set to $\sigma_{r_m} = 5$m. The power spectral densities, $\tilde{q}_x, \tilde{q}_y$ and $\tilde{q}_z$, of the x, y and z components of the continuous time jerk noise were set to $\tilde{q}_x = \tilde{q}_y = \tilde{q}_z = 0.2$. Six satellites were modelled in the simulation; their positions were $\mathbf{x}_{s_1} = (0.9390, -1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_2} = (1.7648, -0.6423, 1.8781) \times 10^7$m, $\mathbf{x}_{s_3} = (1.7648, 0.6423, 1.8781) \times 10^7$m,

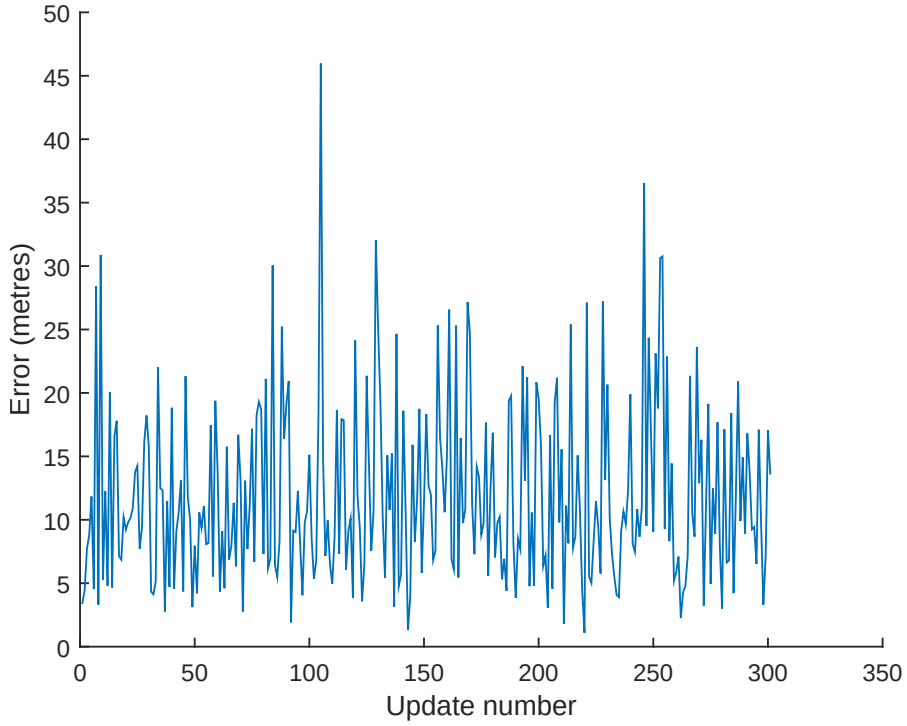

Figure 5: Position estimation error of instantaneous estimates (low dynamics receiver model, 300 updates).

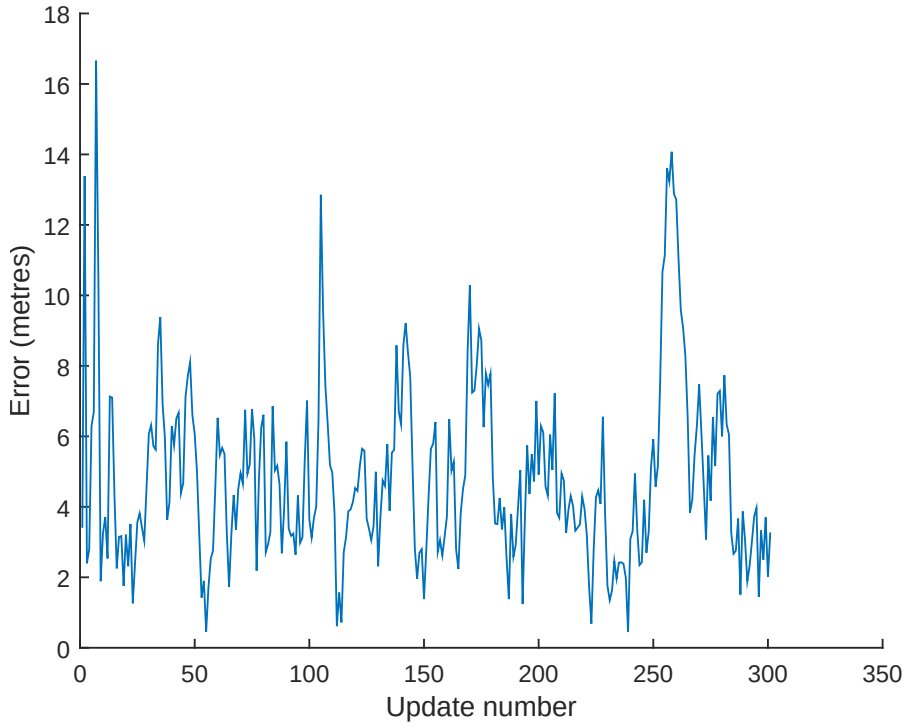

Figure 6: Kalman filter position estimation error (300 updates), using low dynamics receiver model.

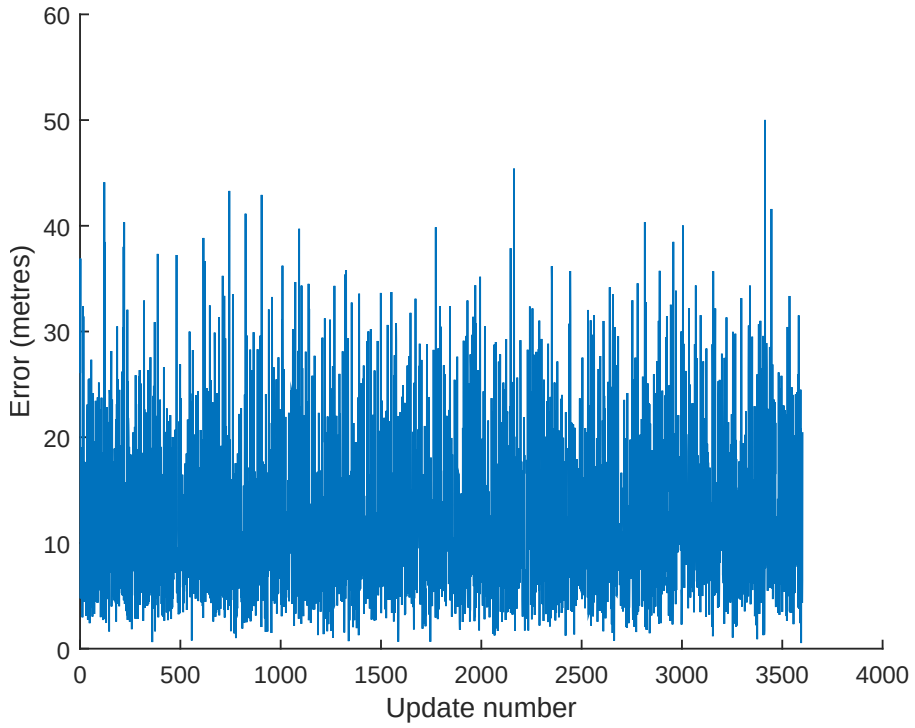

Figure 7: Position estimation error of instantaneous estimates (low dynamics receiver model, 3600 updates).

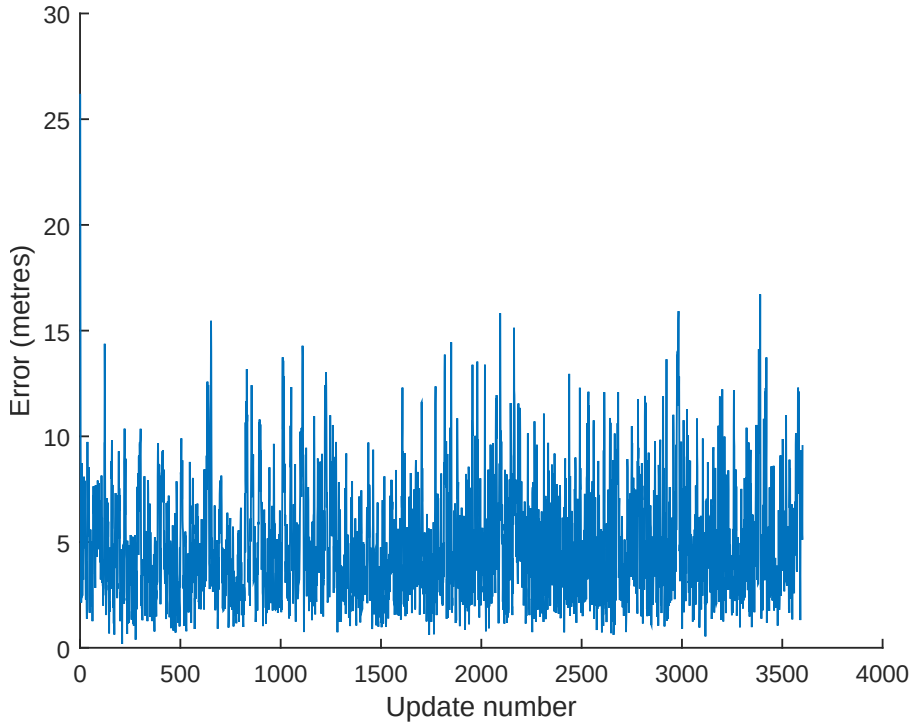

Figure 8: Kalman filter position estimation error (3600 updates), using low dynamics receiver model.

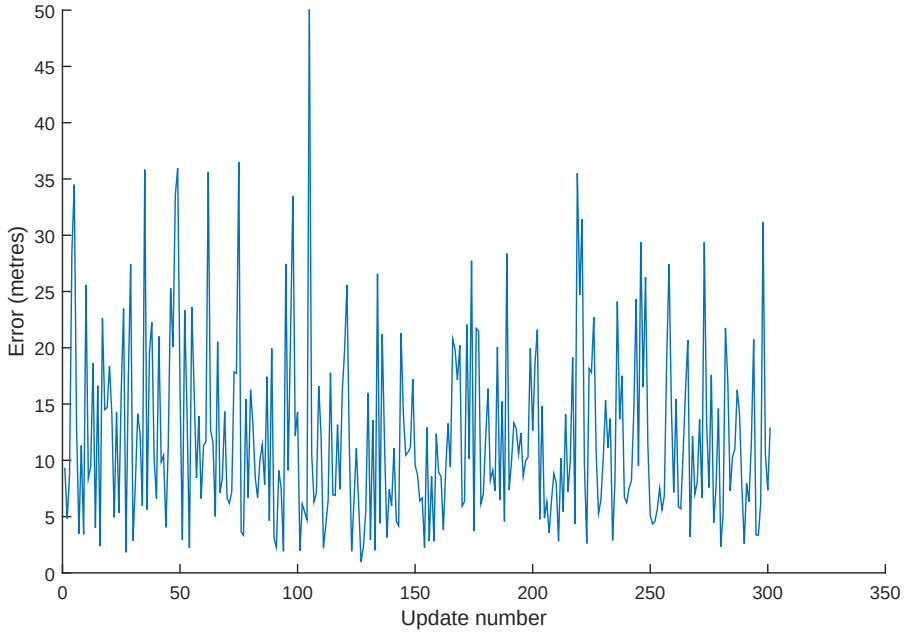

Figure 9: Position estimation error of instantaneous estimates (high dynamics Kalman filter, 300 updates).

$\mathbf{x}_{s_4} = (0.9390, 1.6265, 1.8781) \times 10^7$m, $\mathbf{x}_{s_5} = (0.9390, -1.6265, -1.8781) \times 10^7$m, $\mathbf{x}_{s_6} = (0.9390, 1.6265, -1.8781) \times 10^7$m. The initial receiver position was $\mathbf{x}_r = \big(6.371 \times 10^6, 100, 150\big)$m. The receiver was initially stationary, then, from $t = 101$ s to $t = 200$ s, it experienced an acceleration of $\mathbf{a}_r = (0, 3, 4)$ m/s^2.

Figures 9 and 10 show the results for the case of 300 updates. Figure 9 shows the errors in the instantaneous position estimates. Figure 10 shows the errors in the Kalman filter estimates. Looking at Figure 10, we see that the Kalman filter is quickly reducing the position estimate errors to below that of the instantaneous estimates; however, the errors in the position estimates are higher than was the case for the filters using the stationary receiver and low dynamics models. Also, note that this filter has the advantage that it can track the position, velocity and acceleration of the receiver. Figures 11 and 12 show the velocity and acceleration estimates, respectively, for the case of 300 updates (same simulation as that which produced Figures 9 and 10).

Figures 13 and 14 show the results for the case of 3600 updates. Figure 13 shows the errors in the instantaneous position estimates, and Figure 14 shows the errors in the Kalman filter estimates. Figures 15 and 16 show the velocity and acceleration estimates, respectively, for the case of 3600 updates (same simulation as that which produced Figures 13 and 14). As can be seen from the two figures, convergence continued for the duration of the simulation. While, superficially, the performance of the high dynamics Kalman filter appears correct, a closer look indicates an anomoly. Looking at Figures 12 and 16, we note that the acceleration estimates are very heavily filtered subsequent to about 150 updates. Noting the power spectral densities used in the Kalman filter model for the

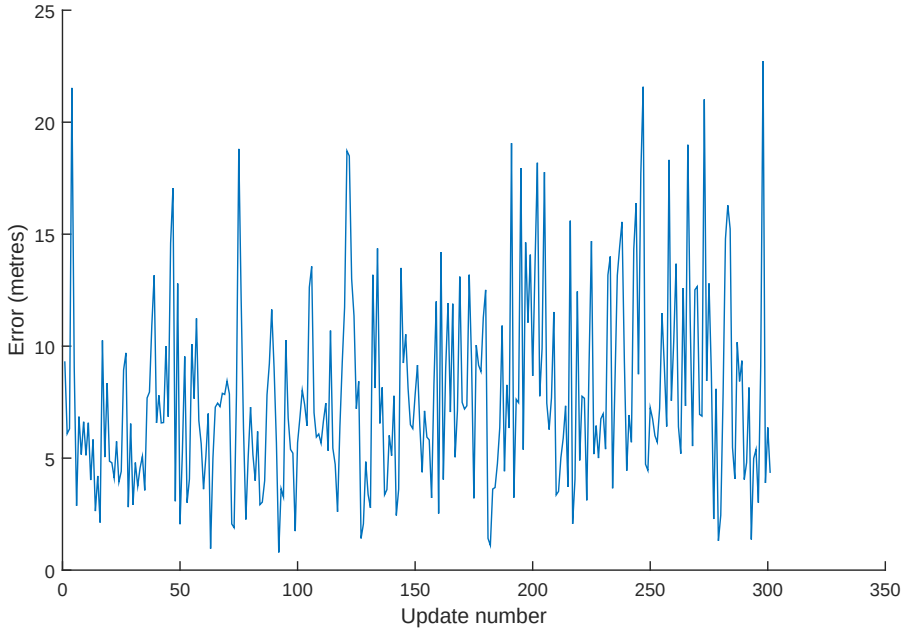

Figure 10: Kalman filter position estimation error (300 updates), using high dynamics receiver model.

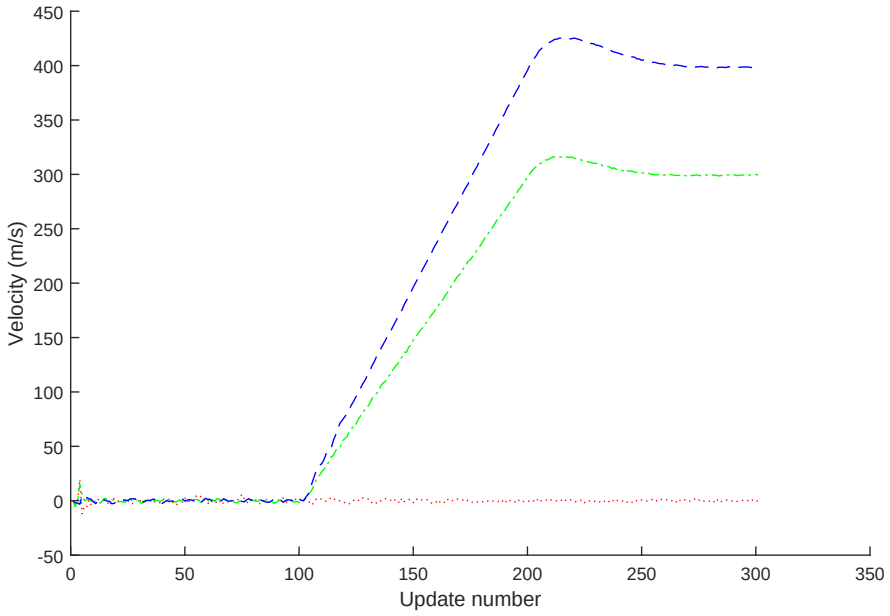

Figure 11: Velocity estimates of high dynamics Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

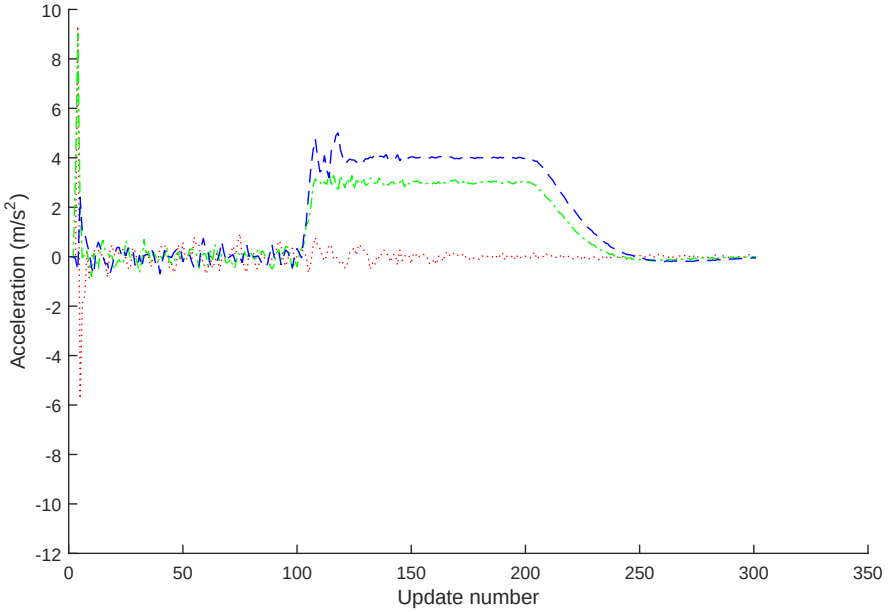

Figure 12: Acceleration estimates of high dynamics Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

continuous-time jerk noise, i.e., $\tilde{q}_x = \tilde{q}_y = \tilde{q}_z = 0.2$, and referring to Equation 6.2.3-6 in [2], we would expect changes of acceleration during a sampling period T to be of the order of $\sqrt{\tilde{q}_x T}$, $\sqrt{\tilde{q}_y T}$ and $\sqrt{\tilde{q}_z T}$ for the x, y and z components respectively, i.e., $\sqrt{0.2} \approx 0.45$ m/s^2. Hence, given noisy measurements, we would intuitively expect that, after a period of convergence, the acceleration estimates of the filter would exhibit acceleration noise of this order. Looking at Figures 12 and 16, we see that the acceleration noise is well below this, indicating that the filter is filtering more heavily than it is designed to do. A possible cause of this is numerical roundoff error. This will be investigated in the following section.

4 Minimizing Round-off Errors

The Kalman filter implementations described up to this point will, from a theoretical standpoint, give correct results based on the models used; however, in practice they can be somewhat sensitive to computer round-off errors. Round-off errors are a side effect of computer arithmetic using a fixed number of bits for representing numbers. In this chapter we will consider an alternative implementation technique that significantly reduces the effects of these errors.

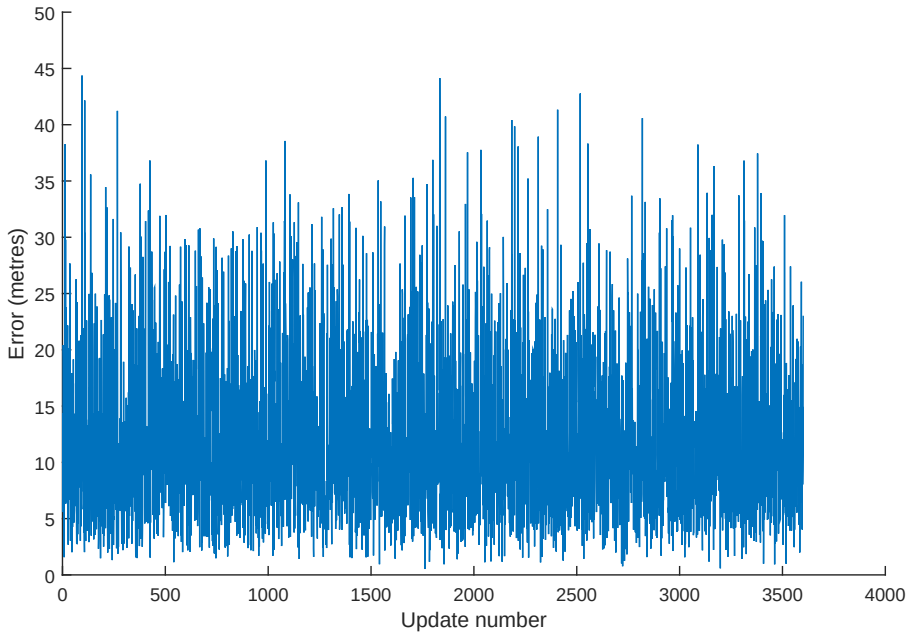

Figure 13: Position estimation error of instantaneous estimates (high dynamics Kalman filter, 3600 updates).

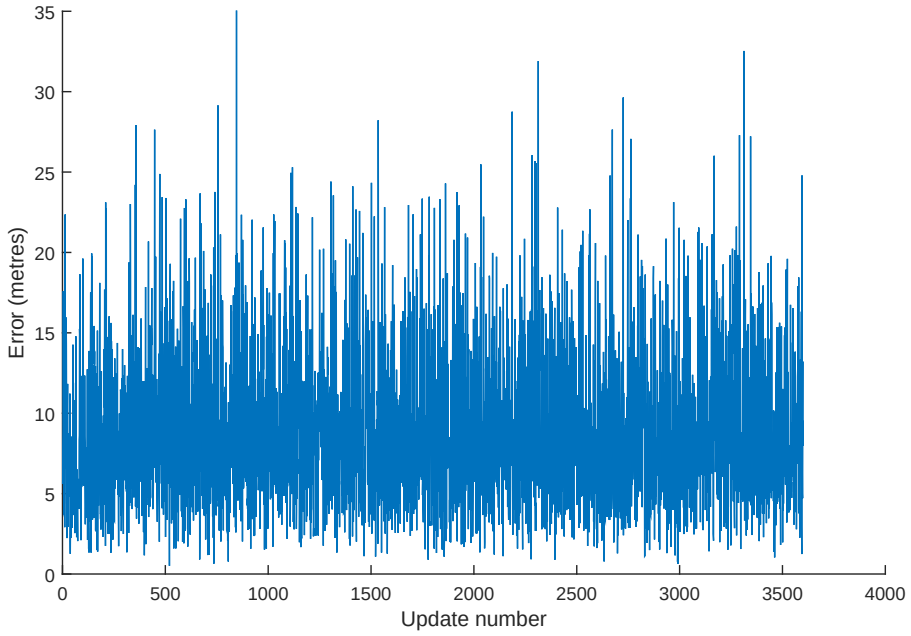

Figure 14: Kalman filter position estimation error (3600 updates), using high dynamics receiver model.

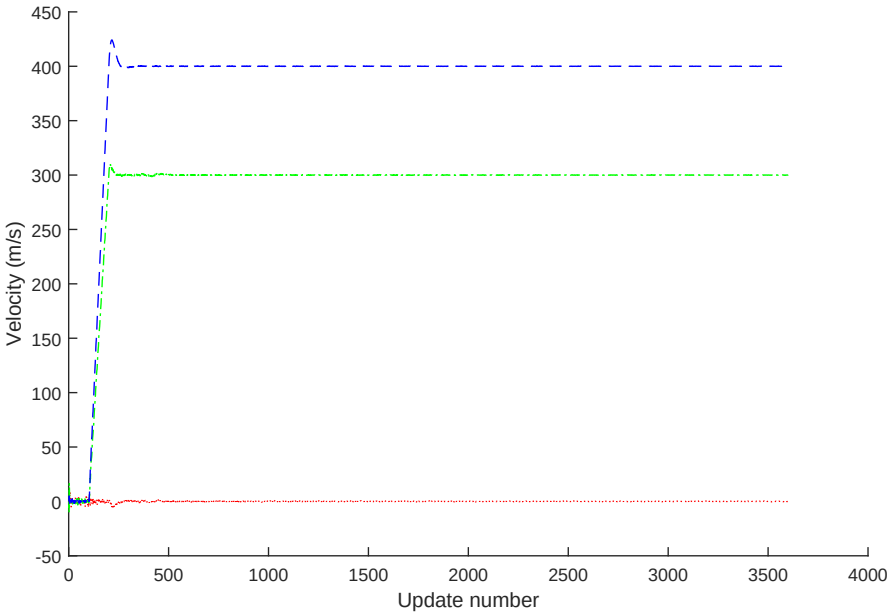

Figure 15: Velocity estimates of high dynamics Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

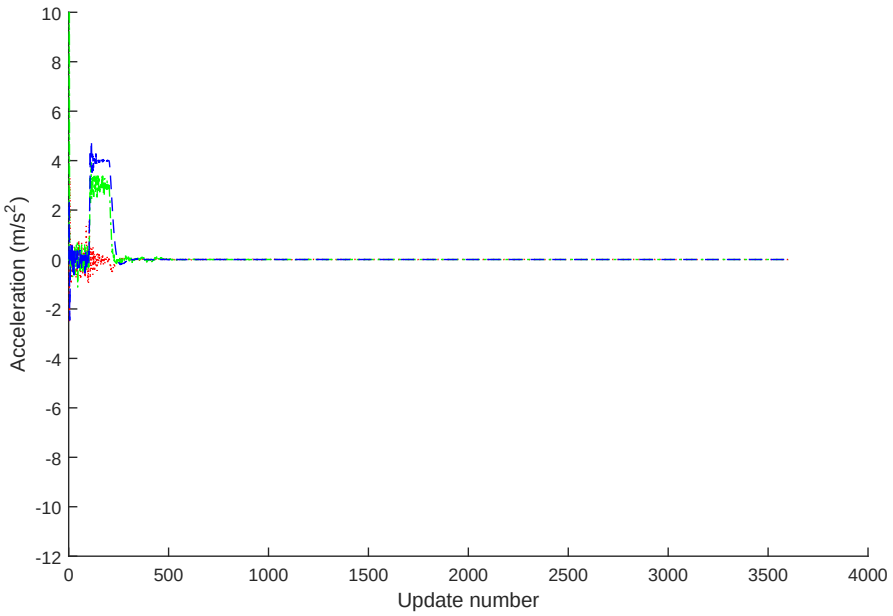

Figure 16: Acceleration estimates of high dynamics Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

4.1 Some Preliminaries

There have been various techniques developed as alternatives to the standard Kalman filter, with the aim of reducing the effects of round-off errors. A description of some of the most important techniques can be found in Chapter 6 of [3]. Amongst these, the most reliable and numerically stable implementations of the Kalman filter are collectively referred to as *square-root filters* [3, Section 6.4]. Square-root filters use a reformulation of the state prediction and state estimate equations such that the dependent variable is a Cholesky factor, or modified Cholesky factor of the covariance matrices $P(k+1|k)$ and $P(k+1|k+1)$. Two of the more favoured implementations of square-root filter are the *Carlson-Schmidt square-root filter* and the *Bierman-Thornton UD filter*. We will concentrate on the Bierman-Thornton UD filter, as it, in particular, has been used successfully on problems with thousands of state variables [3, p. 262].

First, let us summarize what Cholesky and modified Cholesky factors are [3, Section 6.4.3]. The product of a matrix C with its own transpose in the form $CC^T = M$ is called the *symmetric product* of C, and C is called a *Cholesky factor* of M. Note that, strictly speaking, a Cholesky factor is not a matrix square root, although the terms are often used interchangeably. All symmetric nonnegative definite matrices (of which covariance matrices are an example) have Cholesky factors. An upper triangular matrix U is called *unit upper triangular* if its diagonal elements are all 1. Similarly, a lower triangular matrix L is called *unit lower triangular* if all of its diagonal elements are 1. The *modified* Cholesky decomposition of a symmetric positive definite matrix M is a decomposition into products $M = UDU^T$ such that U is unit upper triangular and D is a diagonal matrix. This is also often called *UD decomposition*. The Bierman-Thornton UD filter relies on UD decomposition of the covariance matrices $P(k+1|k)$ and $P(k+1|k+1)$ to achieve superior numerical stability and robustness. The following section describes this filter.

4.2 Bierman-Thornton UD Filtering

For the sake of compactness, we now introduce the following subscript notation. Let $P_{k|k} \triangleq P(k|k)$, $P_{k+1|k} \triangleq P(k+1|k)$, and so on. Now, let $P_{k|k} = U_{k|k}D_{k|k}U_{k|k}^T$, and $P_{k+1|k} = U_{k+1|k}D_{k+1|k}U_{k+1|k}^T$. Consider one cycle of the Kalman-filter covariance update now. The state estimate error covariance matrix at time t_k is $P_{k|k} = U_{k|k}D_{k|k}U_{k|k}^T$. Consider first the temporal update of the Kalman filter. The state prediction covariance for cycle $k+1$ is

$$P_{k+1|k} = F_k P_{k|k} F_k^T + Q_k$$

Now, (from [3, Section 6.5.2.2]), let

$$A = \begin{bmatrix} U_{k|k}^T F_k^T \\ G_k^T \end{bmatrix}$$

$$D_w = \begin{bmatrix} D_{k|k} & 0 \\ 0 & D_{Q_k} \end{bmatrix}$$

and

$$Q_k = G_k D_{Q_k} G_k^T$$

where $G_k D_{Q_k} G_k^T$ is the modified Cholesky decomposition of Q_k. Then

$$\begin{aligned} A^T D_w A &= F_k U_{k|k} D_{k|k} U_{k|k}^T F_k^T + G_k D_{Q_k} G_k^T \\ &= F_k P_{k|k} F_k^T + Q_k \\ &= P_{k+1|k} \\ &= U_{k+1|k} D_{k+1|k} U_{k+1|k}^T \end{aligned}$$

Now, using the modified weighted Gram-Schmidt orthogonalization procedure ([3, p. 272]) with respect to the diagonal *weighting* matrix D_w, we produce a unit lower triangular $n \times n$ matrix L^{-1}, a matrix B, and a diagonal matrix D_β such that

$$A = BL$$

and

$$B^T D_w B = \text{diag}_{1 \le i \le n} \{\beta_i\} = D_\beta$$

hence

$$\begin{aligned} A^T D_w A &= (BL)^T D_w BL \\ &= L^T B^T D_w BL \\ &= L^T D_\beta L \end{aligned}$$

Consequently, the factors

$$\begin{aligned} U_{k+1|k} &= L^T \\ D_{k+1|k} &= D_\beta \end{aligned}$$

are the solutions of the (Thornton) temporal update problem for update k of the UD filter. Note that the code that we used for implementing this was *thornton.m* as supplied in soft-copy form with [3]. It was found in the directory *Chapter_8*.

Now, let us consider the measurement update. The updated state estimate covariance for cycle k is

$$P_{k+1|k+1} = P_{k+1|k} - P_{k+1|k} H_{k+1}^T S_{k+1}^{-1} H_{k+1} P_{k+1|k}$$

where

$$S_{k+1} = H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1}$$

(Equations 2-229 and 2-224 of [4], respectively). Let us now consider the case where the measurement update is a scalar. Then we have

$$P_{k+1|k+1} = P_{k+1|k} - P_{k+1|k} \mathbf{h}_{k+1} \alpha_{k+1}^{-1} \mathbf{h}_{k+1}^T P_{k+1|k}$$

where $\mathbf{h}_{k+1}$ is the vector corresponding to the row of H_{k+1} that applies to the scalar measurement being considered,

$$\alpha_{k+1} = \mathbf{h}_{k+1}^T P_{k+1|k} \mathbf{h}_{k+1} + r_{k+1}$$

and r_{k+1} is the variance of the measurement. Let $P_{k+1|k+1} = U_{k+1|k+1} D_{k+1|k+1} U_{k+1|k+1}^T$ and $P_{k+1|k} = U_{k+1|k} D_{k+1|k} U_{k+1|k}^T$, then we have

$$\begin{aligned} U_{k+1|k+1} D_{k+1|k+1} U_{k+1|k+1}^T &= U_{k+1|k} D_{k+1|k} U_{k+1|k}^T \\ &\quad - U_{k+1|k} D_{k+1|k} U_{k+1|k}^T \mathbf{h}_{k+1} \alpha_{k+1}^{-1} \mathbf{h}_{k+1}^T U_{k+1|k} D_{k+1|k} U_{k+1|k}^T \\ &= U_{k+1|k} \left[D_{k+1|k} - D_{k+1|k} U_{k+1|k}^T \mathbf{h}_{k+1} \alpha_{k+1}^{-1} \mathbf{h}_{k+1}^T U_{k+1|k} D_{k+1|k} \right] U_{k+1|k}^T \end{aligned}$$

Let $\mathbf{v} = D_{k+1|k} U_{k+1|k}^T \mathbf{h}_{k+1}$ then

$$U_{k+1|k+1} D_{k+1|k+1} U_{k+1|k+1}^T = U_{k+1|k} \left[D_{k+1|k} - \mathbf{v} \alpha_{k+1}^{-1} \mathbf{v}^T \right] U_{k+1|k}^T$$

(note that $D_{k+1|k+1} = D_{k+1|k+1}^T$ because $D_{k+1|k+1}$ is a diagonal matrix).

Now perform UD decomposition on $(D_{k+1|k} - \mathbf{v} \alpha_{k+1}^{-1} \mathbf{v}^T)$ to get

$$\begin{aligned} U_{k+1|k+1} D_{k+1|k+1} U_{k+1|k+1}^T &= U_{k+1|k} \left[\bar{U} \bar{D} \bar{U}^T \right] U_{k+1|k}^T \\ &= \left(U_{k+1|k} \bar{U} \right) \bar{D} \left(U_{k+1|k} \bar{U} \right)^T \end{aligned}$$

hence

$$\begin{aligned} U_{k+1|k+1} &= U_{k+1|k} \bar{U} \\ D_{k+1|k+1} &= \bar{D} \end{aligned}$$

The algorithim for the UD decomposition of $\left(D_{k+1|k} - \mathbf{v} \alpha_{k+1}^{-1} \mathbf{v}^T \right)$ to produce $\bar{U} \bar{D} \bar{U}^T$ can be found on page 78 of [5], and the corresponding Matlab code that was written to implement the algorithm is listed in Appendix A.

Now, if the measurement is a vector, and the measurement covariance matrix is diagonal, then the scalar components of the measurement can simply be processed serially as scalar observations with statistically independent measurement errors. This, in fact, is the case for the measurements that we have. If the measurement covariance matrix is not diagonal, then the components cannot be processed serially; however, the measurement vector can be redefined, via a linear transformation, so that the measurement errors of its components are uncorrelated, i.e., the covariance matrix of the redefined measurement vector is diagonal. A procedure for doing this is described in Section 6.4.3.3 of [3].

Now, to start the Kalman filter, we need $U_{0|0}$ and $D_{0|0}$. To obtain these, we simply perform UD decomposition on $P_{0|0}$ as per Section 6.4.3.2 (Table 6.4) of [3]. UD decomposition is also used in one other place in our simulation code, i.e., for computing P_A as per Equation 25. This is done as follows; we have

$$P_A = \left(J(0)^T R(0)^{-1} J(0) \right)^{-1} \tag{39}$$

But $R(0) = \text{diag}\{r_i(0)\}$ is a diagonal matrix, so $R(0)^{-1} = \text{diag}\{1/r_i(0)\}$ and hence computing $\left(J(0)^T R(0)^{-1} J(0) \right)$ is computationally efficient and not very sensitive to round-off errors. However, this is not the case when taking its inverse. Circumvention

of this problem is done as follows. Let $P_I = \left(J(0)^T R(0)^{-1} J(0)\right)$ and now perform UD decomposition on P_I, resulting in $P_I = U_I D_I U_I^T$, hence

$$\begin{aligned} P_A &= \left(U_I D_I U_I^T\right)^{-1} \\ &= \left(U_I^T\right)^{-1} D_I^{-1} U_I^{-1} \\ &= \left(U_I^{-1}\right)^T D_I^{-1} U_I^{-1} \end{aligned}$$

Hence, to obtain P_A, we now only have to invert D, which is a diagonal matrix and U which is a unit upper triangular matrix. This turns out to be less sensitive to numerical round-off errors than direct inversion of P_I. The reader is referred to Section 6.4.3.5 and Tables 6.4, 6.7 and 6.8 of [3] for a description of the MATLAB code for doing this. Note that the functions that we used were *SPDinv.m*, *UD_decomp.m* and *UDinv.m*, as supplied in soft-copy form with [3]. They were found in the directory *TABLE6pt8*.

4.3 Testing of Bierman-Thornton UD Filter with High Dynamics Receiver Model

The Bierman-Thornton implementation of the Kalman filter with the high dynamics receiver model, as described in Section 4.2, was coded in Matlab and tested by simulation. These simulations and tests were done with the purpose of comparing the performance with that of the standard form Kalman Filter (also using the high dynamics receiver model). With the exception of now using the Bierman-Thornton implementation, all other parameters were kept identical to those used for the standard form Kalman Filter simulations.

Figures 17, 18, 19 and 20 show the results for the case of 300 updates. Figure 17 shows the errors in the instantaneous position estimates. Figure 18 shows the errors in the Bierman-Thornton Kalman filter position estimates. Figures 19 and 20 show the velocity and acceleration estimates, respectively, for the case of 300 updates (same simulation as that which produced Figures 17 and 18).

Comparing Figure 18 with Figure 10, we see that the filtered position estimate errors of the Bierman-Thornton Filter appear to be similar to that of the standard form high dynamics Kalman filter. However, if we compare the velocity estimates in Figure 19 with that of Figure 11, we note a considerable difference in performance. Comparing the acceleration estimates of Figure 20 with that of 12, we see an even greater difference in performance.

Figures 21, 22, 23 and 24 show the results for the case of 3600 updates. Figure 21 shows the errors in the instantaneous position estimates. Figure 22 shows the errors in the Bierman-Thornton Kalman filter estimates. Figures 23 and 24 show the velocity and acceleration estimates, respectively, for the case of 3600 updates (same simulation as that which produced Figures 21 and 22). Comparing Figure 22 with Figure 14, we see that the filtered position estimate errors of the Bierman-Thornton Filter appear to be slightly smaller than that of the standard form high dynamics Kalman filter. As was the case for the 300 update simulations, when we compare the velocity estimates in Figure 23 with that of Figure 15, we note a considerable difference in performance. Comparing the

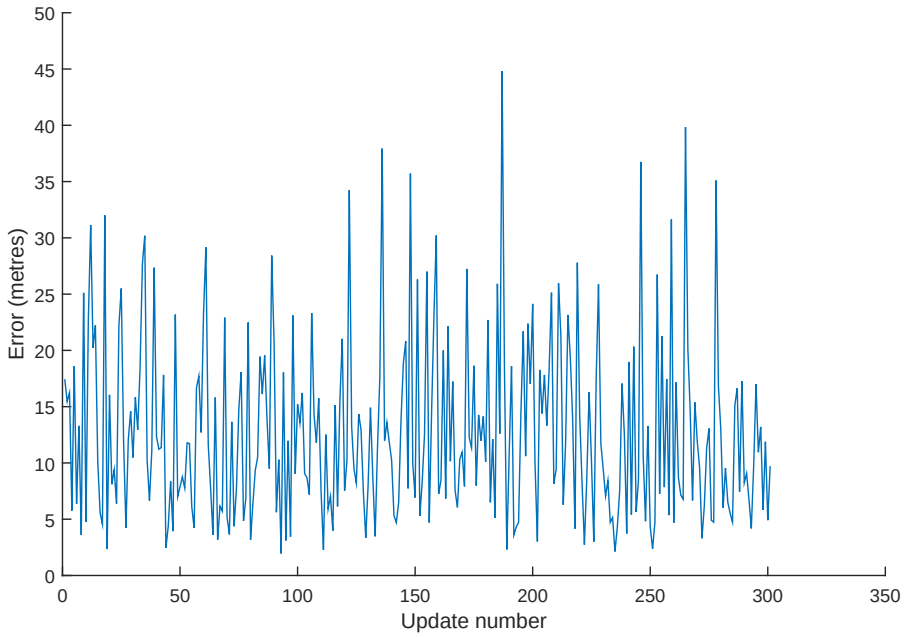

Figure 17: Position estimation error of instantaneous estimates (high dynamics Bierman-Thornton Kalman filter, 300 updates).

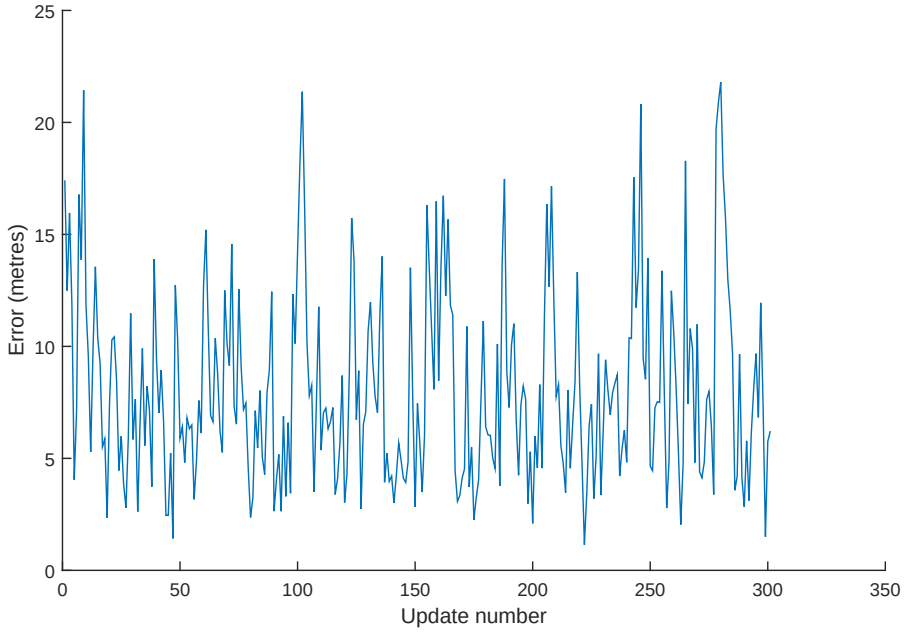

Figure 18: Bierman-Thornton Kalman filter position estimation error (300 updates), using high dynamics receiver model.

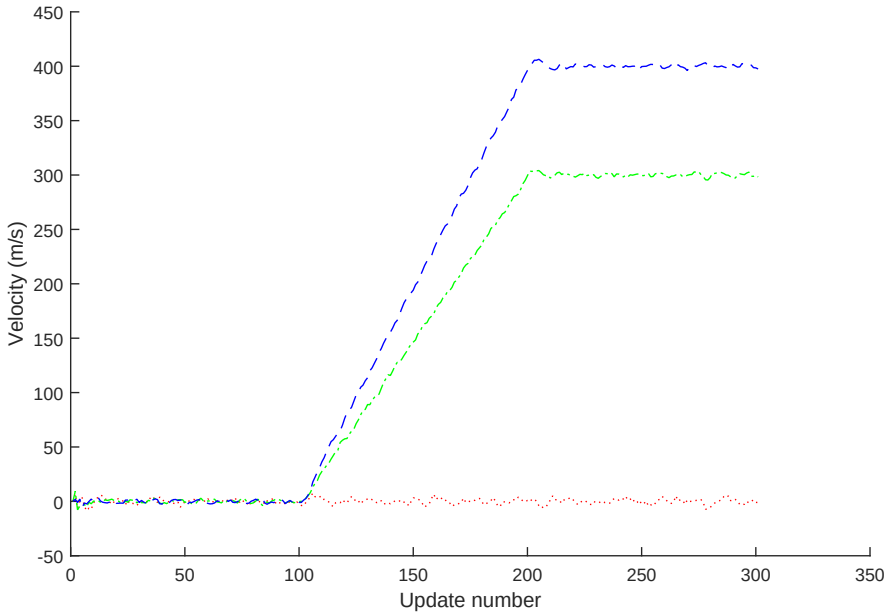

Figure 19: Velocity estimates of high dynamics Bierman-Thornton Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

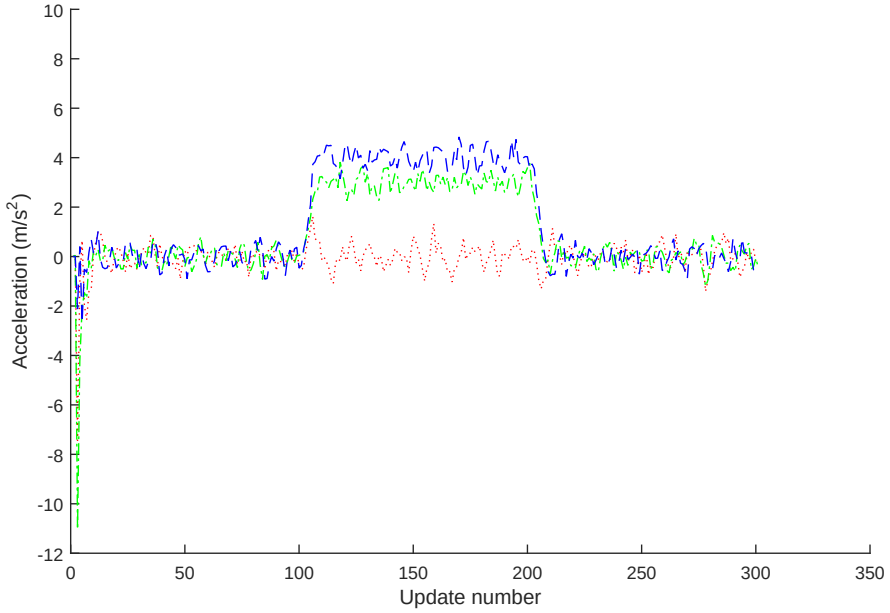

Figure 20: Acceleration estimates of high dynamics Bierman-Thornton Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

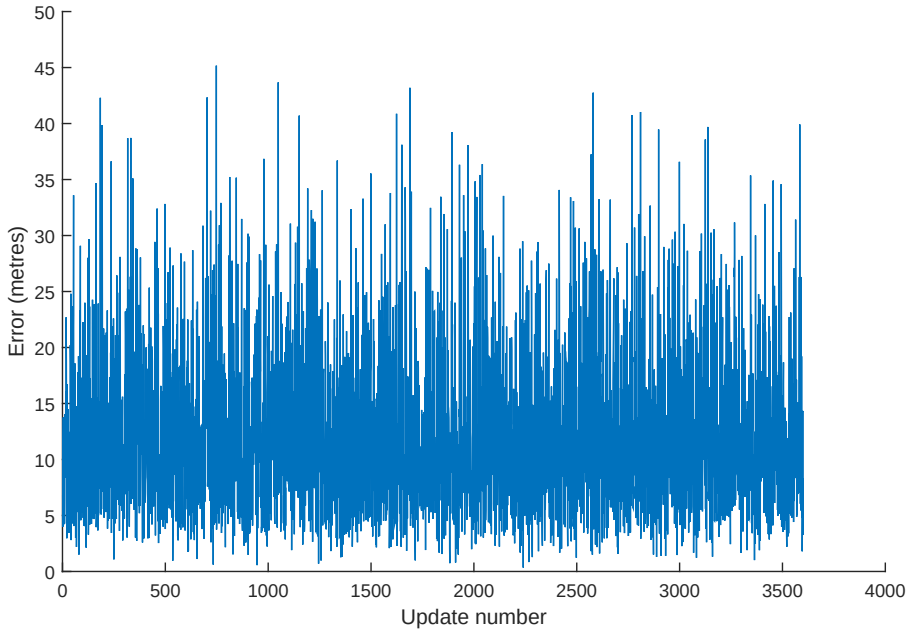

Figure 21: Position estimation error of instantaneous estimates (high dynamics Bierman-Thornton Kalman filter, 3600 updates).

acceleration estimates of Figure 24 with that of 16, we again see an even greater difference in performance.

Now looking at Figures 20 and 24, we note that the variations in the acceleration estimates appear to be consistent with the acceleration noise used in the Kalman filter model, as calculated at the end of Section 3.3. As further confirmation of this consistency, another run of the simulation was performed for the case of 3600 updates, and the standard deviation of the last 3000 acceleration estimates, for the x-component of acceleration, was computed, giving a result of approximately 0.41 m/s^2, which is reasonably close to the value of 0.45 m/s^2 that was calculated at the end of Section 3.3. Based on these results, indications are that the Bierman-Thornton filter is performing correctly, and the standard form Kalman filter, which was tested in Section 3.3, is not. To confirm this, a third form of the Kalman filter (i.e., the Josephson form), was implemented and simulations were run for comparison with the standard form and Bierman-Thornton filters. This is described in the following section.

4.4 Josephson Form Covariance Update

To confirm which of the two high dynamics filters is giving the correct estimates of the receiver's acceleration, the covariance update equation (Equation 36), as used in the standard filter, was replaced with the Josephson form, which is considered to be less sensitive to round-off errors. The Josephson form of the covariance update equation is as follows

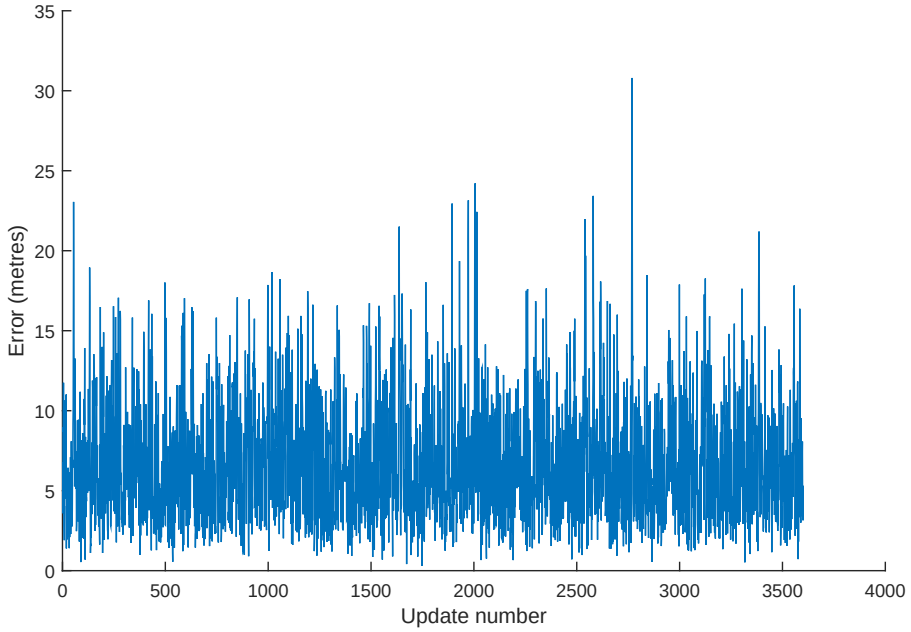

Figure 22: Bierman-Thornton Kalman filter position estimation error (3600 updates), using high dynamics receiver model.

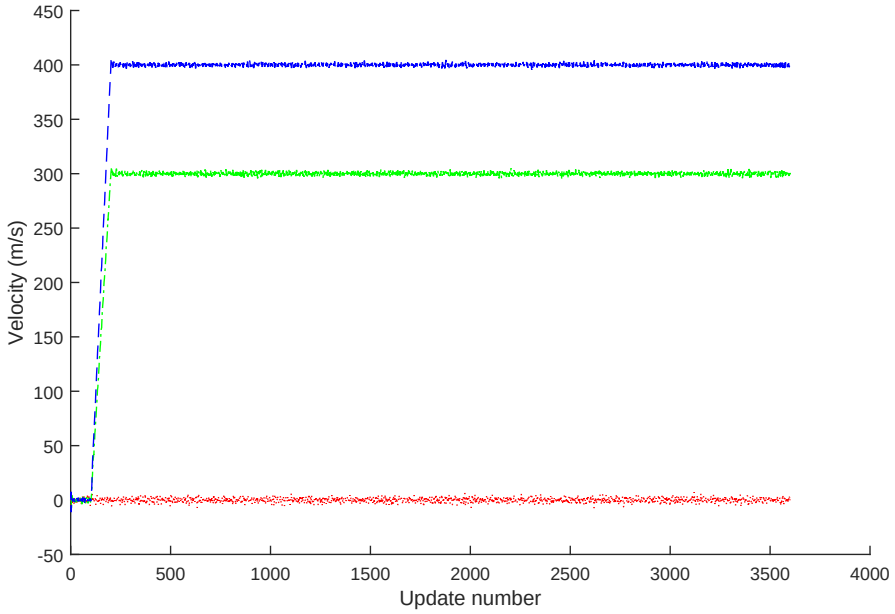

Figure 23: Velocity estimates of high dynamics Bierman-Thornton Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

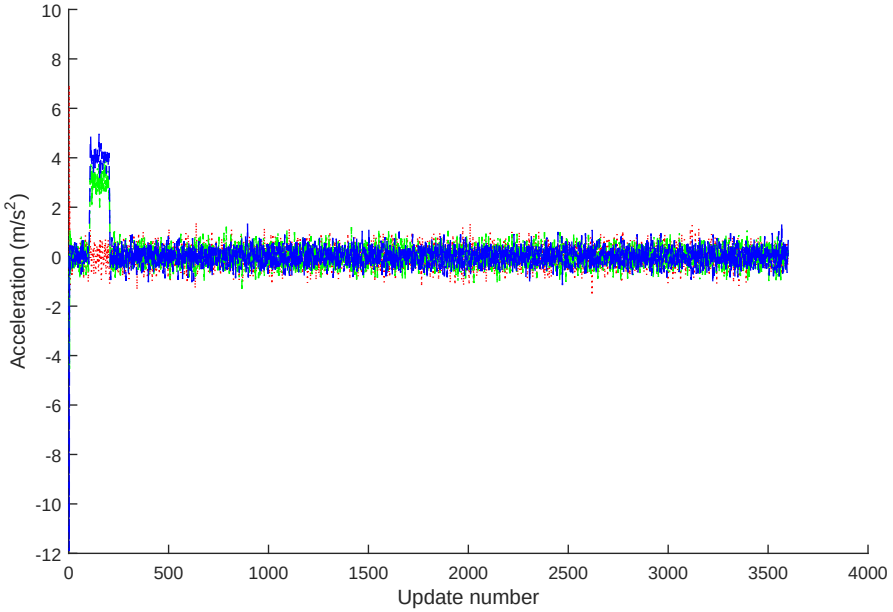

Figure 24: Acceleration estimates of high dynamics Bierman-Thornton Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

[2, p. 294]

$$\begin{aligned} P(k+1|k+1) &= [I - W(k+1)H(k+1)]P(k+1|k)[I - W(k+1)H(k+1)]^T \\ &\quad + W(k+1)R(k+1)W(k+1)^T \end{aligned}$$

The Kalman filter with the high dynamics receiver model, with the Josephson form replacement, was coded in Matlab and tested by simulation. These simulations and tests were done with the purpose of comparing the performance with that of the standard form Kalman Filter (also using the high dynamics receiver model), and the Bierman-Thornton version of the Kalman filter. All other parameters were kept identical to those used for the standard form Kalman Filter and the Bierman-Thornton simulations.

Figures 25, 26, 27 and 28 show the results for the case of 300 updates. Figure 25 shows the errors in the instantaneous position estimates. Figure 26 shows the errors in the Josephson Kalman filter estimates. Figures 27 and 28 show the velocity and acceleration estimates, respectively, for the case of 300 updates (same simulation as that which produced Figures 25 and 26).

Figures 29, 30, 31 and 32 show the results for the case of 3600 updates. Figure 29 shows the errors in the instantaneous position estimates. Figure 30 shows the errors in the Josephson Kalman filter estimates. Figures 31 and 32 show the velocity and acceleration estimates, respectively, for the case of 3600 updates (same simulation as that which produced Figures 29 and 30).

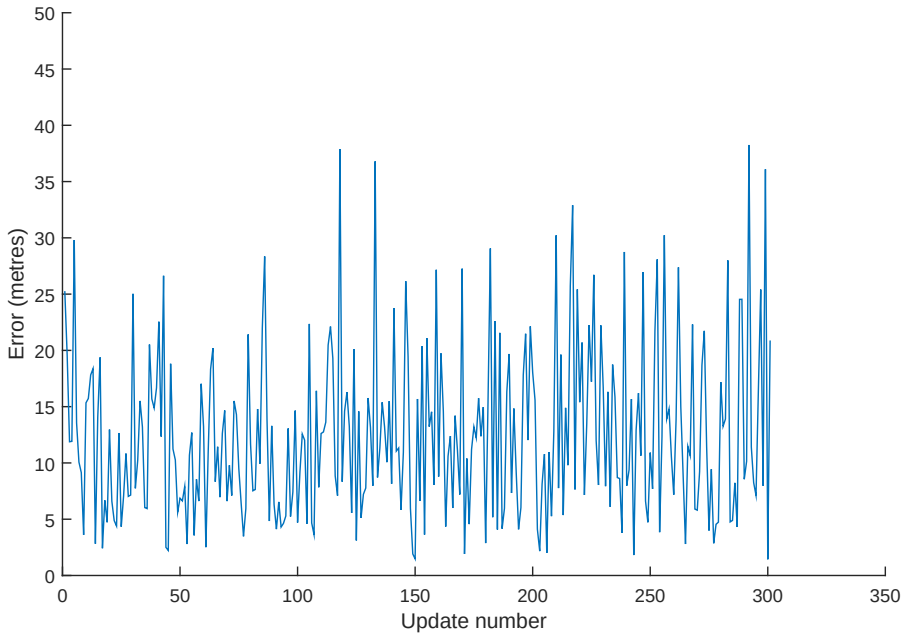

Figure 25: Position estimation error of instantaneous estimates (high dynamics Josephson Kalman filter, 300 updates).

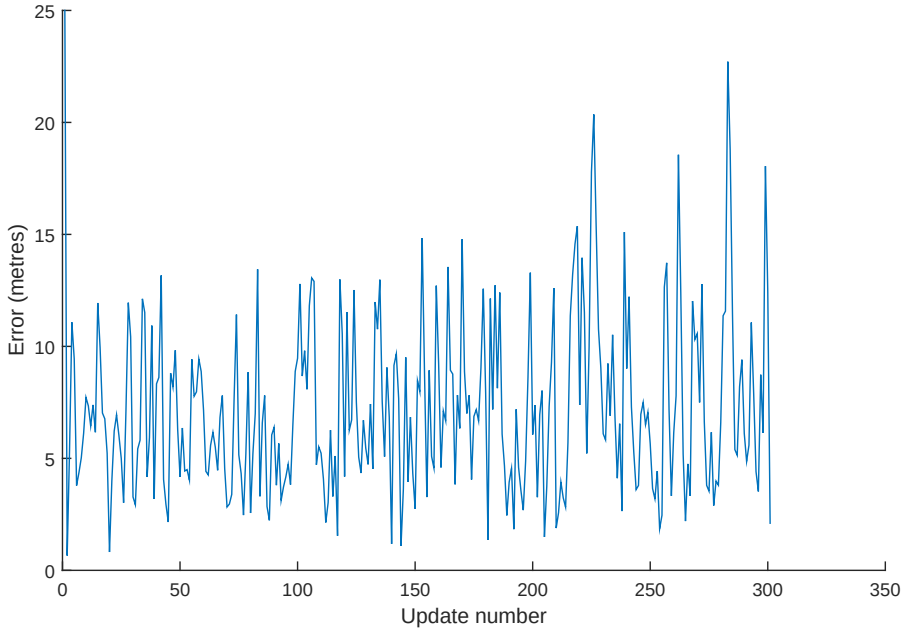

Figure 26: Josephson Kalman filter position estimation error (300 updates), using high dynamics receiver model.

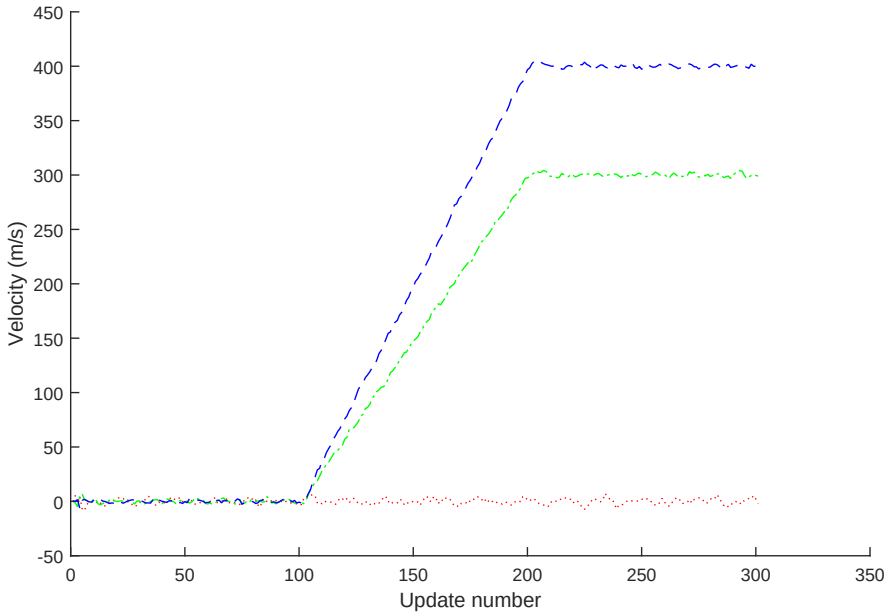

Figure 27: Velocity estimates of high dynamics Josephson Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

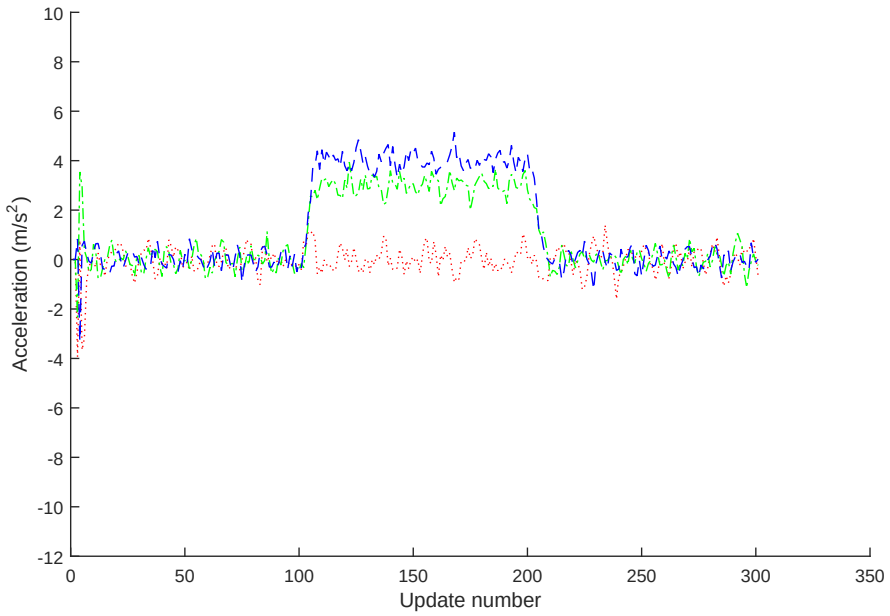

Figure 28: Acceleration estimates of high dynamics Josephson Kalman filter (300 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

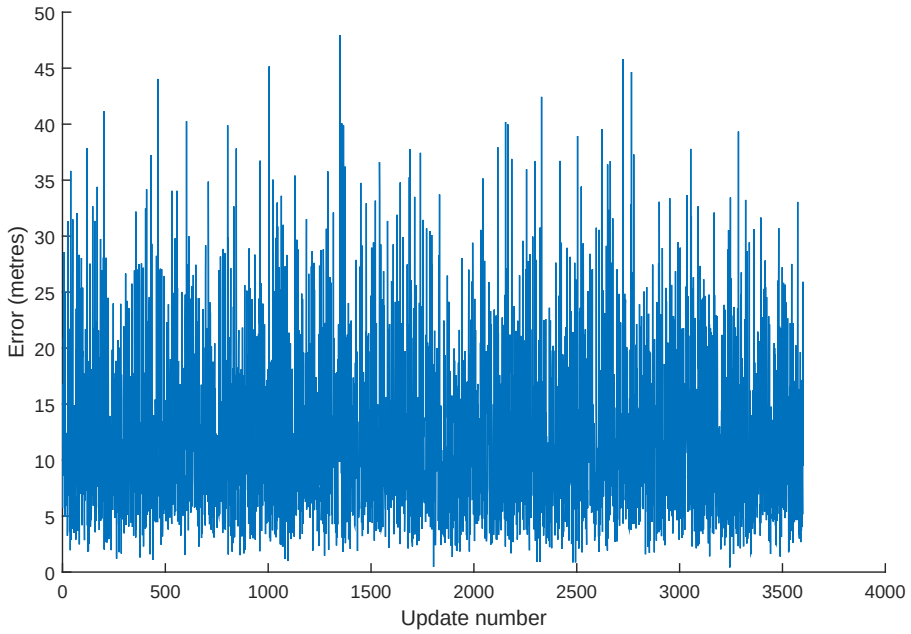

Figure 29: Position estimation error of instantaneous estimates (high dynamics Josephson Kalman filter, 3600 updates).

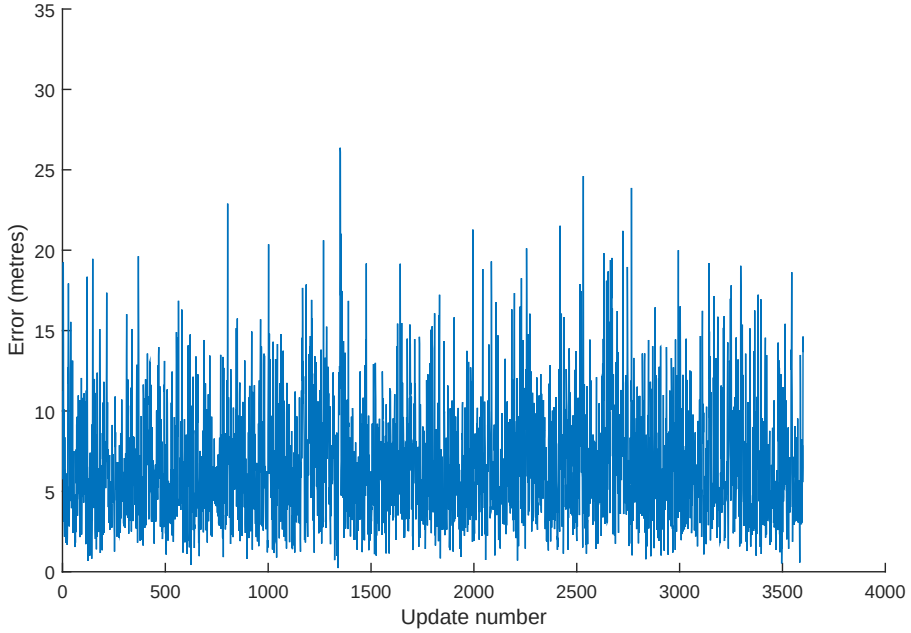

Figure 30: Josephson Kalman filter position estimation error (3600 updates), using high dynamics receiver model.

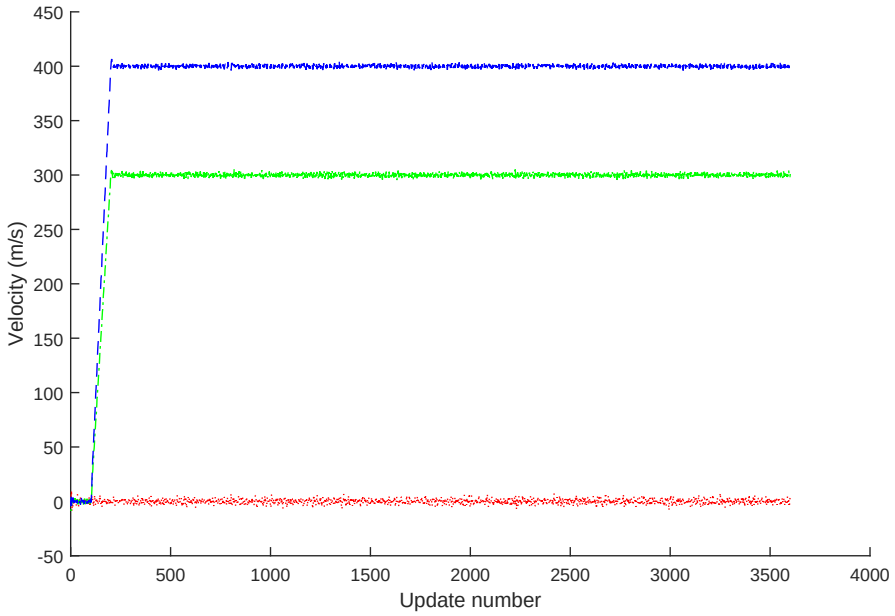

Figure 31: Velocity estimates of high dynamics Josephson Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the velocity, respectively.

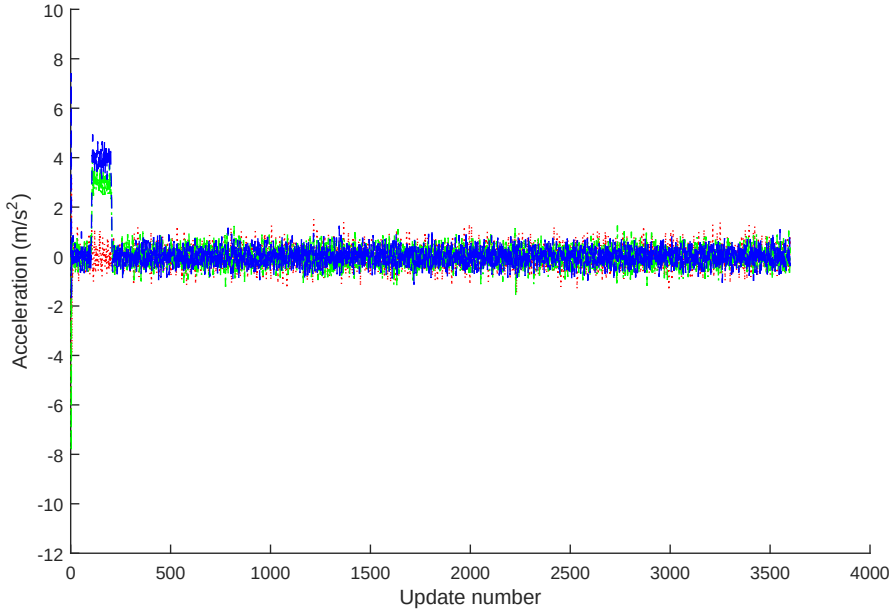

Figure 32: Acceleration estimates of high dynamics Josephson Kalman filter (3600 updates). The red, green and blue plots are the x, y and z components of the acceleration, respectively.

If we now compare the simulation results for Josephson form filter with the standard form filter and the Bierman-Thornton UD filter we find the following.

Firstly, consider the simulations that were run for 300 updates. Comparing Figures 10, 18 and 26, we note that the position estimation error appears to be approximately the same for all three filters. Comparing Figures 11, 19 and 27, we see that the velocity estimates of Bierman-Thornton and Josephson filters are approximately the same. The velocity estimates of standard form filter are different, i.e, they are filtered more heavily. Comparing Figures 12, 20 and 28, we see that the acceleration estimates of Bierman-Thornton and Josephson filters are approximately the same. The acceleration estimates of standard form filter are considerably different, i.e, they are filtered more heavily.

Now let us consider the simulations that were run for 3600 updates. Comparing Figures 14, 22 and 30, we see that the position estimation error appears to be approximately the same for the Bierman-Thornton and Josephson filters. The position estimation error for the standard form filter appears to be slightly greater than for the other two filters. Comparing 15, 23 and 31, we find that the velocity estimates of Bierman-Thornton and Josephson filters are approximately the same. The velocity estimates of standard form filter are different, i.e., they are filtered more heavily. Comparing Figures 16, 24 and 32, we see that the acceleration estimates of Bierman-Thornton and Josephson filters are approximately the same. The acceleration estimates of standard form filter are different, i.e., they are filtered more heavily.

In summary, the Bierman-Thornton and Josephson filters give approximately the same performance, whereas the standard form filter gives substantially different results, confirming that the standard form filter is performing incorrectly. Given that the only difference between the standard form and the Josephson form filter algorithms is the equation for the state estimate covariance update, the result indicates that, for the simulations considered, the primary cause of the incorrect behaviour of the standard form filter is the effects of numerical error on the standard form state estimate covariance update equation i.e., Equation 36.

5 Concluding Remarks

In this report, the mathematics used to convert GPS satellite-to-receiver pseudo-ranges to receiver position estimates is presented. First, the report discusses a method that is used to determine instantaneous estimates of receiver position; it then goes on to develop three Kalman filter based estimators. The three Kalman filter estimators use a stationary receiver, low dynamics, and high dynamics model for the receiver kinematics, respectively. The development of the three types of Kalman filter, as well as the instantaneous estimator is presented in Section 2. Section 3 then presents the results of testing of the filters by simulation. It is found that there are some indications of degraded performance due to numerical round-off in the case of the high dynamics Kalman filter. To investigate this issue further, an alternate form of the high dynamics filter, using modified Cholesky factors of covariance matrices, is developed in Section 4. The filter was implemented in Matlab and tested by simulation. It is found that, for the simulations considered, the alternate form filter overcomes the problems associated with numerical errors. The results of the

simulations are also in Section 4.

References

1. Jay A. Farrell & Matthew Barth, *The Global Positioning System & Inertial Navigation*, McGraw-Hill, 1999.

2. Yaakov Bar-Shalom, Xiao-Rong Li, *Estimation and Tracking: Principles, Techniques, and Software*, Artech House, Inc., 1993.

3. Mohinder S. Grewal, Angus P. Andrews, *Kalman Filtering Theory and Practice using MATLAB*, Third Edition, John Wiley and Sons, Inc., 2008.

4. Yaakov Bar-Shalom, Thomas E. Fortmann, *Tracking and Data Association*, Academic Press, Inc., 1988.

5. Gerald J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, Academic Press, Inc., 1977.

Appendix A Matlab Code for Bierman Measurement Update

The following code was used in the implementation of the Bierman UD measurement update (cf., Section 4.2).

```
function [KalGain,U_post,D_post] = MyBierman(r,h,U_prior,D_prior)
%
% Matlab implementation of the Bierman "square root filtering without square roots",
% as interpreted from "Factorization Methods for Discrete Sequential Estimation",
% Gerard J. Bierman, 1977, Section V.3.
%
% P. W. Sarunic
%
% INPUTS:
%  r         variance of measurement error
%  h         measurement sensitivity (column) vector (note "h" is the same
%            as the (column) vector "a" in Bierman's book, Section V.3).
%  U_prior   unit upper triangular factor of covariance matrix of a priori state uncertain
%  D_prior   diagonal factor of covariance matrix of a priori state uncertainty
%
% OUTPUTS:
%  U_post    upper triangular UD factor of a posteriori state uncertainty covariance
%  D_post    diagonal UD factor of a posteriori state uncertainty covariance
%  KalGain   Kalman filter gain
%

[VecLength, Unused] = size(U_prior);
f = U_prior' * h;
v = D_prior * f;
alpha = zeros(VecLength,1);
alpha(1,1) = r + v(1,1)*f(1,1);
D_post = zeros(VecLength,VecLength);
D_post(1,1) = D_prior(1,1)*r/alpha(1,1);
K = zeros(VecLength,VecLength);
K(1,1) = v(1,1);
lambda = zeros(VecLength,1);
U_post = zeros(VecLength,VecLength);
U_post(:,1) = U_prior(:,1); % Note: I added this myself (it wasn't expicitly
                            % mentioned in Bierman's pseudo-code).
for j = 2:VecLength
    alpha(j,1) = alpha(j-1,1) + v(j,1) * f(j,1);
    D_post(j,j) = D_prior(j,j) * alpha(j-1,1)/alpha(j,1);
    lambda(j,1) = -f(j,1)/alpha(j-1,1);
    U_post(:,j) = U_prior(:,j) + lambda(j,1)*K(:,j-1);
    K(:,j) = K(:,j-1) + v(j,1) * U_prior(:,j);
```

```
    end;
    KalGain = K(:,VecLength)/alpha(VecLength,1);
```

DISTRIBUTION LIST

Development of GPS Receiver Kalman Filter Algorithms for Stationary, Low-Dynamics, and High-Dynamics Applications

Peter W. Sarunic

S&T Program

Chief of Cyber and Electronic Warfare Division	1
Research Leader, Systemic Protection and Effects	1
Head, PNT Technologies and Systems	1
Science Team Leader	1
Author	1

DEFENCE SCIENCE AND TECHNOLOGY GROUP DOCUMENT CONTROL DATA	1. DLM/CAVEAT (OF DOCUMENT)

2. TITLE	3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)
Development of GPS Receiver Kalman Filter Algorithms for Stationary, Low-Dynamics, and High-Dynamics Applications	Document (U) Title (U) Abstract (U)
4. AUTHOR	5. CORPORATE AUTHOR
Peter W. Sarunic	Defence Science and Technology Group PO Box 1500 Edinburgh, South Australia 5111, Australia

6a. DST GROUP NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DST-Group–TR–3260	016-601	Technical Report	June, 2016

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. No. OF PAGES	12. No. OF REFS
			44	5

13. DST Group Publications Repository	14. RELEASE AUTHORITY
`http://dspace.dsto.defence.gov.au/dspace/`	Chief, Cyber and Electronic Warfare Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DST GROUP RESEARCH LIBRARY THESAURUS

Kalman filters, GPS, GNSS, Global Positioning System

19. ABSTRACT

This report presents algorithms that can be utilized in a GPS receiver to convert satellite-to-receiver pseudo-ranges to receiver position estimates. The report discusses a method that is used to determine instantaneous estimates of receiver position and then goes on to develop three Kalman filter based estimators, which use stationary receiver, low dynamics, and high dynamics models for the receiver motion, respectively. These particular dynamic models are utilized because they are commonly used in actual GPS receivers, and cover a wide range of applications. While the standard form of the Kalman filter, of which the three filters just mentioned are examples, is theoretically correct, it can be susceptible to numerical round-off errors, which can in some cases result in poor performance or, in the worst case, filter divergence. This issue, and its solution, is investigated, and another version of the high dynamics filter, which addresses this problem, is presented. Matlab code was developed to test the performance of each of the filters and simulations were performed. The results of the simulations are also presented.