Nikeet Pandit

214118806

Advanced Optimal Estimation Theory and Applications

LE/GS/ESS 5430 3.0

Fall 2021

Professor Wang

Project: Kalman Filter Design and Implementation (associated with specific applications)

Table of Contents

Introduction

The Kalman filter allows for real-time and sequential optimal-unbiased state estimation, provided its stochastic assumptions are met. Statistically, the Kalman filter provides the "best" state estimate. Best in this case refers to an estimate which is unbiased and has the minimum variance when compared to any other estimator. The filter estimates by optimally weighting new measurements and the predicted state, determined by a mathematical model, by taking into account model and measurement uncertainty. In addition to an optimal estimate, the estimate uncertainty is also provided. In this report, the state estimation problem will be applied to the process of orbit determination for GRACE-FO (Gravity Recovery and Climate Experiment – Follow On). In orbital determination, the state vector to be uncovered is the position of velocity of a satellite in a given coordinate system.

The dynamics governing orbit motion is always non-linear, therefore the EKF (extended Kalman) filter is used in this application which linearizes the model about the predicted state estimate. In this application, the measurement model is taken as linear, which is usually not the case because the state vector (position and velocity) cannot normally be measured directly. This has been done to simplify the modelling process. However, if either the measurement model or process model is nonlinear, the EKF must be used. As a result, the work presented in this report can naturally be extended for a nonlinear measurement model as well.

Methodology

Orbital Dynamics

In orbital dynamics, the primary force governing motion of a satellite orbiting earth is its gravitational pull. In the simplified case, one can model the orbital dynamics as the two-body problem described by Newton's laws.

$$\ddot{r} = -\frac{\mu}{|r^3|}\,\vec{r}$$

Where G is the universal constant of gravitation, and $\vec{r}$ is the vector joining the Earth to the satellite. Also,

$\vec{r} = [x\ y\ z]^T$ In the approximated case, where the field is radial, the solutions to the equation are summarized in Kepler's laws [1]. Satellites that are very high in orbit can be approximated quite

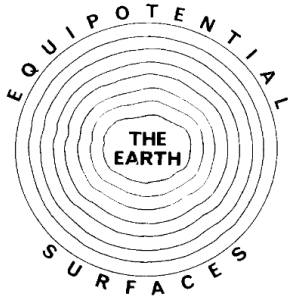


Figure 1 Departure of gravity field from radiality

well by using this simplified case, however, satellites flying in LEO (Low-Earth Orbit), such as GRACE which fly at an altitude of 500km is not approximated well by a central field and require additional perturbing terms to the model. This is attributed to the fact that mass within Earth is not uniformly or symmetrically distributed, however at higher and higher altitudes these higher frequency effects will be smoothed. This can be seen visually above, as given in [2]

Specifically, there is irregularity in attraction due to the flattening of Earth and depending on the coordinate system referred to, centrifugal and Coriolis forces are added as well. Even after all of these effects are accounted for, there is still irregular distribution of mass within the Earth which will also perturb orbits more significantly at lower altitudes due to the inverse squared dimensioning factor of the gravity force functional.

However, for this application only the flattening of the Earth will be considered, and an assumption is made for a symmetric distribution of mass with respect to its spin axis. This is done to simplify the modelling of this effect on the satellite motion and is modelled as the gradient of gravity potential expressed through spherical harmonic expansion. Only the J2 term will be included, as it significantly larger than the other terms. In ECI coordinates, it is given as:

$$a_{J2_x} = \frac{-3J_2\mu R_e^2}{2r^5}\left(1 - \frac{5z^2}{r^2}\right)x$$

$$a_{J2_y} = \frac{-3J_2\mu R_e^2}{2r^5}\left(1 - \frac{5z^2}{r^2}\right)y$$

$$a_{J2_z} = \frac{-3J_2\mu R_e^2}{2r^5}\left(1 - \frac{5z^2}{r^2}\right)z$$

Non-gravitational perturbing to the satellite orbit effects includes the air drag, electromagnetic forces, solar radiation, and relativistic effects [2]. While LEO satellites are significantly affected by drag, it is very complex to model drag accurately when considering all the uncertainties in: (1) Energy exchange between neutral particles and satellite geometry and its material; (2) State and altitude determination; (3) The actual atmospheric models themselves [3]. As a result, the drag component, among other non-gravitational affects, in addition to the inaccuracies of the gravitational component modelling itself (for example not including higher order zonal coefficients) will be included as process noise in the dynamic model.

Dynamic Model

First, the equations which will model the dynamic system and subsequently determine the sate space representation of the dynamic system must be determined, as discussed in the previous section. The state space representation of a linear-time invariant system is (unforced):

$$\dot{x} = Fx(t) + w(t)$$

$$z(t) = Hx(t) + v_k$$

where x(t) is the system state vector, w(t), is process noise, v_k is measurement uncertainty, and F, and H are matrices that arise in the modelling formulation [4].

The states will be position and velocity in ECI given as

$$x(t) = [\boldsymbol{r}\ \boldsymbol{v}]^T$$

Where **r** (position vector), **v** (velocity vector), **a** (acceleration vector) are a function of x, y, and z. The dynamic model then is described as:

The dynamic model will be given as a non-linear function of x(t)

$$\dot{x}(t) = f\big(x(t)\big) + w$$

Written in matrix forum, and compiling equations describe above, the dynamic model is given as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \dfrac{-3J_2\mu R_e^2}{2r^5}\left(1-\dfrac{5z^2}{r^2}\right)x - \dfrac{\mu}{\sqrt{x^2+y^2+z^2}}x \\ \dfrac{-3J_2\mu R_e^2}{2r^5}\left(1-\dfrac{5z^2}{r^2}\right)y - \dfrac{\mu}{\sqrt{x^2+y^2+z^2}}y \\ \dfrac{-3J_2\mu R_e^2}{2r^5}\left(1-\dfrac{5z^2}{r^2}\right)z - \dfrac{\mu}{\sqrt{x^2+y^2+z^2}}z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

Process noise is included in the model for the acceleration terms only because inaccuracies in determination of the acceleration propagates to the position and velocity determination [5]. As mentioned, the drag which is significant to LEO satellites will not be modelled and will be included as process noise, or model inaccuracies

The process noise matrix Q will then be given by

$$\mathrm{Q} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \sigma_a^2$$

where σ_a^2 is the uncertainty of the dynamic model. Due to the fact that the model is missing many terms as indicated, the variance for the acceleration in process noise is given as 1e-8 $(m/s^2)^2$. Process noise will remain constant for this model.

Measurement Model

Total State Observation

GRACE-FO is fitted with a GNSS receiver assembly that provides full state vector observation. Although the state vector observation is very accurate (within 10 cm), for this project measurement noise will be artificially corrupted with significantly more errors to see the effect of the Kalman filter implemented. The measurement model can be seen below

$$\begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + v_k$$

Or, more compactly:

$$z_k = Hx_k + v_k$$

State Prediction

Solving the differential modelling equation for position of velocity is done numerically, calculated using the RK4 expansion method. The method was chosen because of its simplicity and stable solution. The implemented method was referenced from [6] and the implemented software is found in the appendix.

Extended Kalman Filter

If either the measurement model or the dynamic model (or both) is non-linear, the extended Kalman filter should be used. In this case, only the dynamic model is nonlinear while the measurement model is linear. The non-linear model can be approximated by Taylor series expansion evaluated at the predicted state estimate. All stochastic assumptions about the linear filter are used in the extended Kalman filter [7]. Specifically, process and measurement noises are zero mean gaussian processes, and process and measurement noises are uncorrelated. Conceptually, the Kalman filter determines a weighting factor which updates the estimate based on the new measurement information based on the measurement covariance and the propagated covariance from the previous state. To take advantage of the law of propagation, the aforementioned linearization procedure is performed about the current state estimate.

The state prediction equation is given by the RK4 numerical integration of the dynamic model about the current state estimate:

$$\hat{x}_{k+1} = \varphi\,(\hat{x}_k, k)$$

The covariance prediction is given by:

$$P_{k+1/k} = \Phi_{k+1/k} P_k \Phi_{k+1/k}^T + LQ_k L^T$$

where:

$$\Phi_{k+1/k} = I + F\Delta T + \frac{(F\Delta T)^2}{2!} + \frac{(F\Delta T)^3}{3!} + \cdots$$

$$F = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{\partial f1}{\partial x} & \frac{\partial f1}{\partial y} & \frac{\partial f1}{\partial z} & 0 & 0 & 0 \\ \frac{\partial f2}{\partial x} & \frac{\partial f2}{\partial y} & \frac{\partial f2}{\partial z} & 0 & 0 & 0 \\ \frac{\partial f3}{\partial x} & \frac{\partial f3}{\partial y} & \frac{\partial f3}{\partial z} & 0 & 0 & 0 \end{bmatrix}$$

and F is a function of time evaluated at the current state estimate. The final result of the partial derivatives is appended at the end and is evaluated symbolically using MATLAB. The result of the symbolic computation is then appended where current values of the estimated state are evaluated and then the transition matrix $\Phi_{k+1/k}$ may be solved for.

L is the Jacobian of the partial derivative of the function with respect to the noise vector and is given by

$$L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The Kalman gain equation is given by:

$$K_{k+1} = P_{k+1/k}\, C_{k+1}^T (C_{k+1} P_{k+1/k} C_{k+1}^T + R_{k+1})^{-1}$$

The C_{k+1} is the Jacobian of the H matrix with respect to the state. Since all of the states are directly observable, the resultant matrix is simply a 6 by 6 identity matrix.

The state update equation is given by:

$$\hat{x}_{k+1} = \hat{x}_{k+1/k} + K_{k+1}[z_{k+1} - H(\hat{x}_{k+1/k})]$$

The covariance update equation is given by (in the numerically stable form):

$$P_{k+1} = (I - K_{k+1}C_{k+1})P_{k+1/k}(I - K_{k+1}C_{k+1})^T + K_{k+1}R_{k+1}{K_{k+1}}^T$$

Implementation

Filtering Algorithm

Step 1: Initialization
The initial state $\mathbf{x_0}$ with covariance $\mathbf{P_0}$ is provided. This initial state is a guess and is subjected to very high uncertainty.

Step 2: The Time Update (Prediction)
a) Predict trajectory (state vector) at next time step using RK4 to numerically find solution to non-linear dynamic model.

b) Propagate covariance by solving for system description matrix (F), calculating transition matrix, and using error law of propagation (matrix form).

Step 3: Calculate Kalman Gain
a) Check for measurement and measurement covariance

b) Calculate Kalman Gain

Step 4: The Measurement Update (Correction)
a) Update (correct) the state estimate with the innovation

b) Update covariance of corrected state estimate

and continue…

Adding Noise to Data

All POD data is provided for GRACE-FO in the PODAAC (Physical Oceanography Distributed Active Archive Centre). State vector information is provided to an uncertainty to within 10 cm. To see the effect of the Kalman filter, this data is corrupted artificially by normally distributed and zero-mean interference which is scaled by inputted (into the software) standard deviation for the position measurements and velocity measurements, respectively. The standard deviation for the corrupted position measurements is 1200m, and the standard for the velocity measurements is 30m/s which stays constant in all three vector components with no cross-covariance assumed.

It must be noted that all random number generators which are used to corrupt this data is pseudo-random so that results can be recreated and examined. Further, statistical interference causes noise to the random interference because the added noise is only a sample of the population distribution. This means that the errors in the data is not exactly zero mean, nor is it exactly scaled to the standard deviation as inputted into the software. The “true” data is the uncorrupted POD data provided by the PODAAC, the measurements are the corrupted data, and the errors are the corrupted data subtracted by this true data. Below is a histogram of the measurement position errors in the x direction where the total corrupted measurements were arbitrarily selected to be of length 300. The sampled standard deviation and mean of the corrupted position errors in the x direction is 1225m and the mean is 24m. While only the x position errors are shown, the rest of the state components are corrupted in the same way and would show similar normal distribution characteristics that satisfy stochastic assumptions of the Kalman filter.

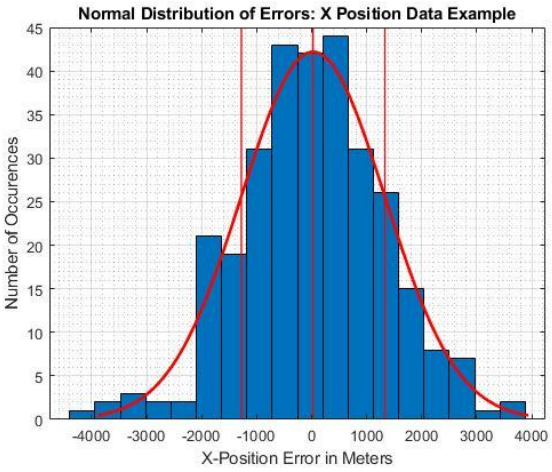


Figure 2 Histogram of X Position Data Errors

Results and Solution

Filtering Solution

In the test solution appended to this report, the filter is run for 80,000 seconds with a step size of 1 second which is also the sampling rate of the POD information for GRACE-FO provided by PODAAC. It must be noted that the software allows for any step size, as selected by the user.

The duration is set to investigate any periodic error residual in the estimation due to the simplified model used in this application. One full orbit for GRACE-FO is completed in approximately 1.6 hours, under which GRACE-FO undergoes significant periodic effects in acceleration such as the drag, solar radiation pressures, other non-gravitational accelerations, and higher order zonal coefficients. Since the model is very simplified and does not include these terms, it then becomes imperative to see if there are any periodic estimation errors due to modelling errors. The initial state is completely corrupted and subsequently its covariance is set very high relative to the measurement uncertainty, which remains constant through the duration of the filter.

To see the effect of the Kalman filter implementation visually, the magnitudes of the measurements, true states, and estimated states are investigated. The magnitude of the measurement residuals is taken as:

$$R_k = \left\|\tilde{l}\right\| - \|l\|$$

where $\tilde{l}$ is the "true" measurements (state vector) given by PODAAC and l is the "measured" measurements, which are artificially corrupted as discussed, and the square brackets signify the norm. Similarly, the magnitude of the estimation residual is taken as:

$$\Delta = \left\|H\tilde{l}\right\| - \|\hat{x}\|$$

where $\hat{x}$ is the estimated state and H is the mapping matrix, as defined above. In both cases the position states and velocity states are separated, and the magnitude is taken for each.

Additionally, the trace of the gain matrix will be plotted to see its dynamic characteristics. The trace for the position states and the velocity trace will be plotted separately to see any differences between the weighting factor for the innovation. First, looking at the Kalman gain for the position and velocity states. It takes approximately 100 seconds and 1000 seconds, respectively, for the

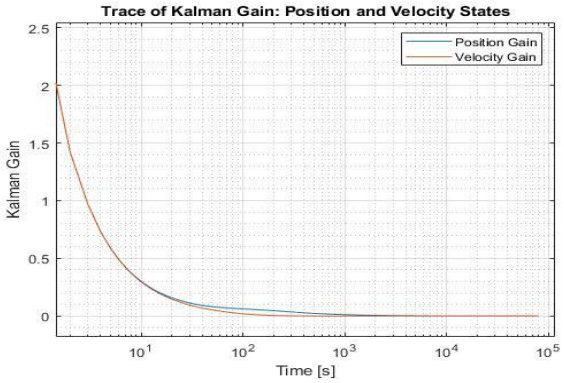


Figure 3 Gain Convergence

velocity and position gain to converge to its value which is approximately zero. The longer time to converge for the position may be attributed two factors (among other unlisted ones). Firstly, the position is subjected to a higher uncertainty than the velocity measurements. Although, due to the relative difference in magnitude of the position measurements in ECI and velocity measurements, perhaps this is less significant. Secondly, since position is a double integral of the dynamic force (acceleration) model, and due to the propagation of error, it could be postulated that this would accumulate errors faster than the velocity state which is only a single integral from the force model. Both of these factors can result in the difference in the time it takes for the gain terms to stabilize. The graph is appended above. It also must be noted that a higher (or dynamic process noise) term could have been tuned to weight more heavily the innovation measurements to see the effect on filter performance.

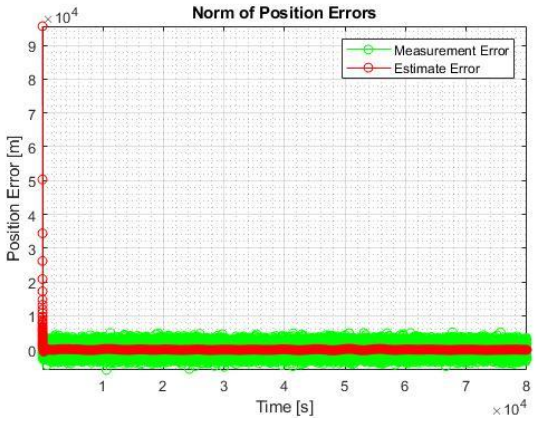


Figure 4 Position State Residuals

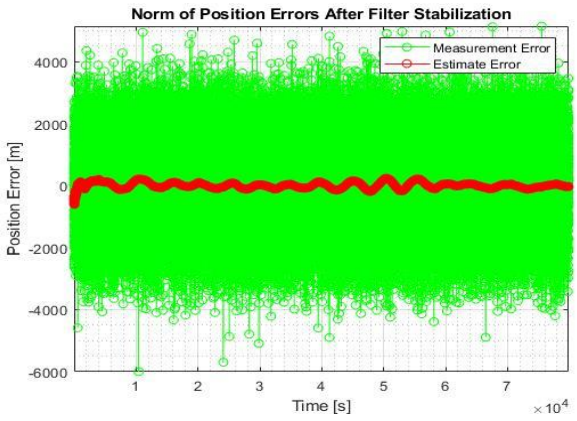


Figure 6 Position State Residuals (after stabilization)

In figure (4) and figure (5), the effect of the erroneous initial condition is apparent where the estimate error is very high before the filter solution starts to converge. This is also apparent in figure (3), where more trust is placed in the innovation, or the new measurements. Thereafter, the filter solution stabilizes closer to the "true" state, and it becomes visually apparent of the effectiveness of incorporating a dynamic model for state estimation when measurements are corrupted with normally distributed uncertainty. Since the estimate errors are large, as the filter is stabilizing, it skews the y-axis of the figure (4) and figure (5) where any visual information about the state estimate becomes obscured. As a result, in figure (6) and figure (7) the same plots will be appended after the filter stabilization. In figure (6) and figure (7) the smoothing effect of the filter against measurement error is very apparent. However, it can also be seen in figure (6) that there is a periodicity in the estimate residual. Upon investigating, the periodic term displays a repition every 1.6 hours, which is also the orbital period of GRACE-FO as previously mentioned.

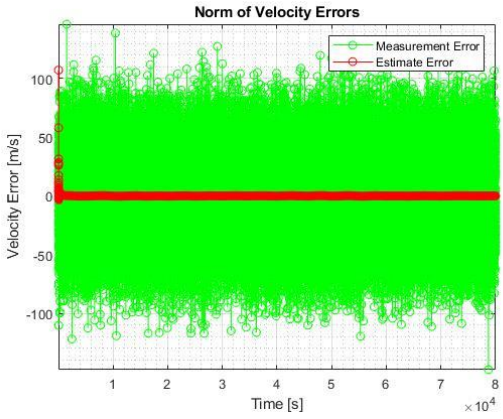


Figure 5 Velocity State Residuals

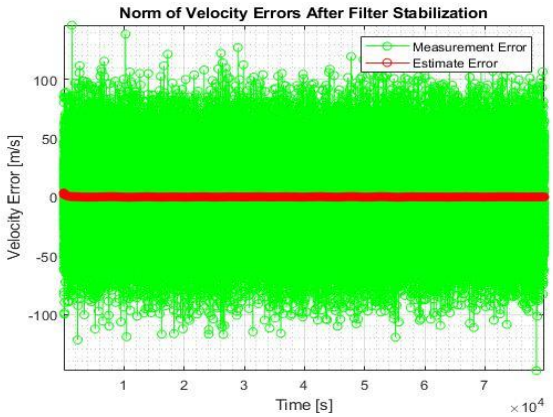


Figure 7 Velocity State Residuals (after stabilization)

This can partially be attributed to the dynamic model which does not include many dominant periodic orbital factors in orbital determination, namely drag and solar radiation pressure. This could be remediated by including higher order non-gravitational terms and higher order-zonal coefficients as well. An alternate approach could be to augment the state to include a sinusoidal residual bias in the state estimation, where the sinusoidal bias is apparent in figure (6). Due to this periodic bias in the state estimation, which is present in both the position and velocity estimates (albeit to a lesser degree), the state estimate does not truly converge and either of the aforementioned solutions must be implemented to construct an unbiased optimal estimate.

As mentioned, the corruption to the measurements is pseudorandom and can be recreated for experimented reproducibility. However, in real application of this Kalman filter the measurement uncertainty would be dynamic and not constant as presented here. Further, due to the periodic bias as mentioned in the previous paragraph, calculating error statistics on the residual is dependent on external factors. For example, calculating the estimate errors is a function on the length of data that the filter is run for, in addition to changing orbit dynamics. For example, as the altitude of the orbit is further degraded the residuals and the biased periodic component would grow than the dataset which is currently used (Jan 2021), resulting in perhaps larger error than presented here.

In any case, the root mean square deviation (RMSD) is calculated by the following formula

$$RMSD = \frac{\sum(\|x_i - \hat{x}_i\|}{N}$$

where the summation is taken from i = 1 to N. The RMSD is calculated individual for the position in x, y, and z and the velocity in x, y, and z. Again, the square brackets signify the magnitude.

States	RMSE	Unit
Position – Updated Estimate	312	m
Velocity – Updated Estimate	0.4	m/s
Position – Measurements Only	1950	m
Velocity – Measurements Only	50	m/s

The RMSD is to signify the average estimation error against the "true state" for the measurement only solution, and the estimated solution for both the position and velocity state estimation. Despite the periodic bias in the solution and even withstanding the dynamic error characteristics of the filter as described above, it is clear that the Kalman filter reduces error significantly in state estimation than that which could have been achieved from measurement alone.

Another way to visualize the effectiveness of the suboptimal Kalman filter in this application, is demonstrated in figure (8) where the true measurements, the real measurements and the state

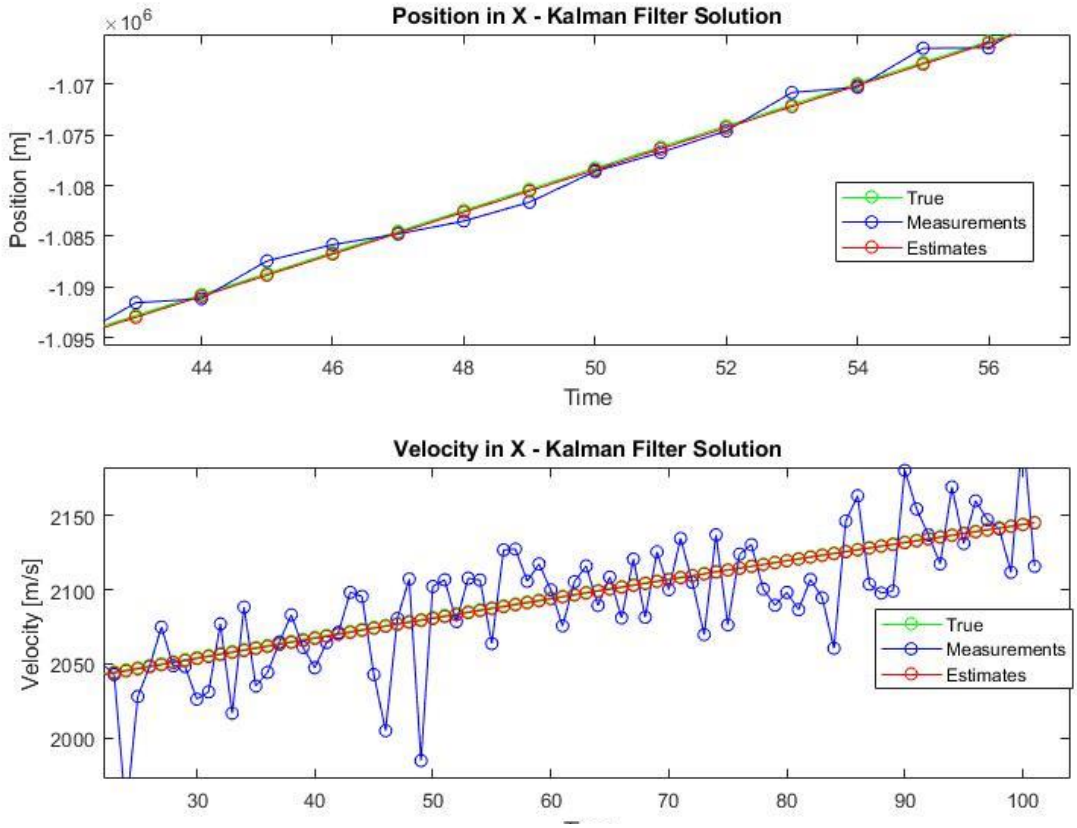


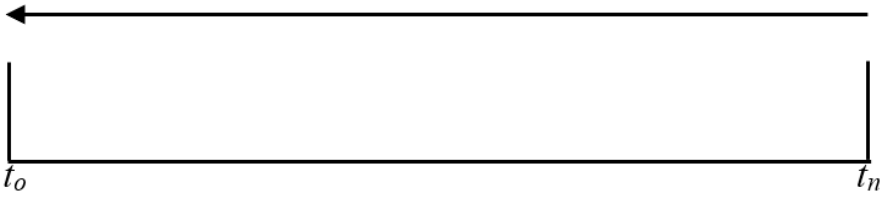
Figure 8 Kalman Filter Solution

estimates are all plotted individually. In figure (8), which is zoomed in a specific window of time after filter stabilization, the effectiveness of the filter can be seen by the scattering of the measurements around the true state and the convergence of the estimate to the true state.

Fixed Point Smoothing

As mentioned, the initial state provided to the Kalman filter is a guess with high uncertainty in this application. This is reflected in the initial covariance provided to the filter, as well as the time it takes for the filter state estimate to stabilize to the true state. Fixed point smoothing is when the optimal estimate for a specific epoch is constructed, using both the forward and backward measurements and solutions. An example of this then could be to optimally estimate the initial state and associated covariance, given the backward solution. To do this, the Kalman filter implementation that is appended in the appendix only needed to be slightly appended to optimally estimate the initial conditions. In the case of the initial conditions, the optimal estimate is only a function of the backward estimate and the initial corrupted state which is treated as a measurement. Further, the RK4 numerical integration for orbit determination then uses a negative step size to iterate backwards. The resultant system description matrix and transition matrix are then also determined in reverse, as it is a function of the predicted state. Fixed point smoothing to determine the initial conditions can be visualized below:

$$\hat{x}(t_o) = F(z(t_o), \hat{x}_b(t_n, t_o)$$



It must be noted as well that the form of the optimal smoother, for any epoch, cannot be a function of both the forward solution and the backward solution due to the double counting of the measurement at the epoch [8].

The results of the fixed point smoothing to recover the initial condition is summarized in the table below.

State Vector		Covariance (Trace)
Initial	[-1664240, 4889607, 2738206, 855, -2939, 5689]^T	450e6
Smoothed	[-1958322, 5751982, 3221560, 1026, -3437, 6713]^T	1740
True	[-1958587, 5751820, 3221439, 1027, -3437, 6713]^T	--

Then the root square deviation is then calculated:

States	RMSE	Unit
Position – Smoothed Initial Condition	330	m
Velocity – Smoothed Initial Condition	0.5	m/s
Position – Initial Condition Only	1031000	m
Velocity – Initial Condition Only	1150	m/s

which shows that the smoothed initial condition is a significant improvement over the offered initial state.

Conclusion

In this project, an extended Kalman filter for orbit determination was presented. In the beginning, a brief discussion on orbital dynamics was presented followed by the dynamic model for orbital propagation was presented. Pitfalls in the dynamic model were highlighted, such as failure to model non-gravitational accelerations such as drag, solar radiation pressure, and higher order zonal coefficients. Improvements to the dynamic model would focus specifically on adding these terms to the dynamic model to ensure a more representative model. The dynamic model was numerically integrated using RK4 due its simple and stable solution. Thereafter, the measurement model was presented, where full state vector observation was used for this application due to the simplicity. It must be noted that this work can be expanded on for different measurement models depending on available measurements, even if this relationship is non-linear by extending the extended Kalman filter methodology presented here. After, the methodology of the filter in the form of the algorithm and mathematical equations governing the EKF was given, followed by a discussion on how noise, which satisfied the stochastic assumptions of the EKF, was added to corrupt the “true” measurements.

Finally, an implementation of the Kalman filter was applied to optimally estimate artificially corrupted POD data (the measurements) for GRACE-FO on January 2021. Although the data was not coming in truly at real time, the algorithm was performed as if the measurements were coming in real time and by extension would be valid for a real-time application. The true data was taken to be the official Jet Propulsion Laboratory (JPL) solutions as provided by the PODAAC database. In the implementation, it was seen that there was a sinusoidal bias in the state estimation which was significant in the position state estimate and less so in the velocity state estimated. This may be attributed to the higher position uncertainty relative to velocity uncertainty, in addition to more errors propagating to the position state determination to the double, as opposed to single, integration of the dynamic model. It was uncovered that this sinusoidal bias had a 1.6-hour periodicity, the same as it takes for GRACE-FO to complete one full orbit. It was postulated that this could be partially attributed to the dynamic model that did not include many and significant LEO non-gravitational terms to the model, such as drag. Recommended remediations for this were to add more terms to the dynamic model to ensure a more representative model and to potential augment the state to include the sinusoidal bias. Since the state did not converge and had this bias, it was also said that this was an unbiased estimate. Withstanding this case, the smoothing effect of the Kalman filter against measurement uncertainty was apparent where the average estimate error was decreased by a factor of 6 and 125 for the position and velocity estimates, respectively, when using the Kalman estimate versus the measurement estimate alone. An application for fixed point smoothing was also presented, where the initial state to the system was optimally smoothed, by optimally combining the initial state, treated as a measurement, and the backward solution. The smoothed initial condition was a significant improvement over the given initial state to the filter.

References

[1] P. Vanicek. *Gravimetric Satellite Geodesy*. Geodesy and Geomatics Engineering UNB. 1973.

[2] P. Vanicek. *Geodesy: The Concepts 2ed*. Elsevier Science Publishers. 1986

[3] S. Behzadpour, Mayer-Gurr, and S. Krauss. "GRACE Follow-On Accelerometer Data Recovery". *In: JGR Solid Earth*. 2021.

[4] A. Gelb. *Applied Optimal Estimation*. The M.I.T. Press. 1974.

[5] R. Ottemark. "Autonomous Satellite Orbit Determination Based on Magnetometer and Sun Sensor Measurements". *Lulea University of Technology*. 2015

[6] W. Hu. *Fundamental Spacecraft Dynamics and Control*. WILEY. 2015.

[7] J Wang. *Lecture 9: The Extended Kalman Filter*. York University. 2021

[8] J. Wang. *Lecture 11: Optimal Linear Smoothing*. York University. 2021

Appendix

*Plotting and error calculations are not included in appendix for brevity.

Main Script

Loading Data and Setting Variables

```
clearvars
close all
load GNSS_StateVector.mat
n = 300; %seconds to run filter for
h = 1; %time step
True_Measurements = GNSS_StateVector(1:n*h+1,:); %Appending extra row (first row is used for
corrupted guess)
```

Corrupting True Measurements

```
%Adding normally distributed noise scaled by variance of Postion and Velocity
stdPos = sqrt(1500000); stdVel = sqrt(1000); varACC = 1e-08;

[k,j] = size(True_Measurements); Corrupted_Measurements = zeros(k,j);

rng(0,'twister'); r = normrnd(0,stdPos,[k*3,1]); %normally distributed array with mean 0 and
standard deviation
rng(1,'twister');  r1 = normrnd(0,stdVel,[k*3,1]); %normally distributed array with mean 0 and
standard deviation

for i = 1:j
    if i <= 3
        if i == 1
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r(1:k);
        elseif i == 2
```

```
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r(k+1:k*2);
        else
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r(k*2+1:k*3);
        end
    else
        if i == 4
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r1(1:k);
        elseif i == 5
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r1(k+1:k*2);
        else
            Corrupted_Measurements(:,i) = True_Measurements(:,i)+ + r1(k*2+1:k*3);
        end
    end
end

R = [eye(3,3)*stdPos^2 zeros(3,3); zeros(3,3) eye(3,3)*stdVel^2]; %Measurement Uncertainty
```

Jacobian Matrix L and C

```
H = eye(6,6); %Jacobian of measurement function h(x) [Linear Relationship]
L = [zeros(3,3) zeros(3,3); zeros(3,3) eye(3,3)];  %Jacobian of non linear function wrt to noise
vector
```

Initial State Set

```
Xo = Corrupted_Measurements(1,:)*0.85;
Po = R*10; %State Uncertainty (Very High)
```

Process Noise Matrix Set

```
Q = [zeros(3,3) zeros(3,3); zeros(3,3) eye(3,3)*varACC];

for i = 1:n
```

```
    if i == 1 %Initialization
        Xk = Xo;
        Pk = Po;
        to = 1;
    end
```

Time Update (Predict State + State Covariance)

```
    %----------- Predict State using RK4 (Numerical Integration of System Dynamics)
    Xk_pred = rk4_prop(Xk,h);
    %----------- State Covariance Prediction
    F = calc_sys_descrip_mat(Xk_pred(1:3));                 %Calculate System Description Matrix
    TransitionMatrix = calc_trans_mat(F,h);                 %Calculate Transition Matrix
    Pk_pred = TransitionMatrix*Pk*TransitionMatrix'+L*Q*L';  %Propagate Covariance
```

Check Measurement

```
    t = to+h; %Update Time
    Z = Corrupted_Measurements(t,:);     %State Vector Measurements from GNSS
```

Calculate Kalman Gain

```
    K = Pk_pred*H'*inv(H*Pk_pred*H'+R);
```

The Measurement Update (Correct)

```
    Xk_updated(i,:) = transpose(Xk_pred' + K*(Z'-H*Xk_pred'));
    Pk_updated = (eye(6,6)-K*H)*Pk_pred*(eye(6,6)-K*H)' + K*R*K';
```

And Continue...

```
    Xk = Xk_updated(i,:);
    Pk = Pk_updated;
    to = t;
```

end

Dynamic Model Function

```
function accel_total = calc_acc(pos_vec)
mu = 3.986004418e14;
Re = 6378.16e3; % m
J2 = 1.082629e-3;
x = pos_vec(1); y = pos_vec(2); z = pos_vec(3);
r = sqrt(x^2+y^2+z^2);
flat_effect_A = 1.5*J2*(Re/r)^2*(1-5*(z/r)^2); %Harmonic A
flat_effect_B = 1.5*J2*(Re/r)^2*(3-5*(z/r)^2); %Harmonic B
xACC = -((mu*x)/r^3)*(1+flat_effect_A);
yACC = -((mu*y)/r^3)*(1+flat_effect_A);
zACC = -((mu*z)/r^3)*(1+flat_effect_B);
accel_total = [xACC yACC zACC];
```

System Description Matrix Function

```
function [F,F_Mat] = calc_sys_descrip_mat(pos_vec)
mu = 3.986004418e14;
Re = 6378.16e3; % m
J2 = 1.082629e-3;
x = pos_vec(1); y = pos_vec(2); z = pos_vec(3);

df1_dx = -(2*mu*(x^2 + y^2 + z^2)^3 - 6*mu*x^2*(x^2 + y^2 + z^2)^2 + 3*J2*Re^2*mu*(x^2 + y^2 +
z^2)^2 + ...
    105*J2*Re^2*mu*x^2*z^2 - 15*J2*Re^2*mu*x^2*(x^2 + y^2 + z^2) - 15*J2*Re^2*mu*z^2*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));
```

```
df1_dy = (6*mu*x*y*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*x*y*z^2 + 15*J2*Re^2*mu*x*y*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df1_dz = (6*mu*x*z*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*x*z^3 + 45*J2*Re^2*mu*x*z*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df2_dx = (6*mu*x*y*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*x*y*z^2 + 15*J2*Re^2*mu*x*y*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df2_dy = -(2*mu*(x^2 + y^2 + z^2)^3 - 6*mu*y^2*(x^2 + y^2 + z^2)^2 + 3*J2*Re^2*mu*(x^2 + y^2 +
z^2)^2 + 105*J2*Re^2*mu*y^2*z^2 ...
    - 15*J2*Re^2*mu*y^2*(x^2 + y^2 + z^2) - 15*J2*Re^2*mu*z^2*(x^2 + y^2 + z^2))/(2*(x^2 + y^2 +
z^2)^(9/2));


df2_dz = (6*mu*y*z*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*y*z^3 + 45*J2*Re^2*mu*y*z*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df3_dx = (6*mu*x*z*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*x*z^3 + 45*J2*Re^2*mu*x*z*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df3_dy = (6*mu*y*z*(x^2 + y^2 + z^2)^2 - 105*J2*Re^2*mu*y*z^3 + 45*J2*Re^2*mu*y*z*(x^2 + y^2 +
z^2))/(2*(x^2 + y^2 + z^2)^(9/2));

df3_dz = -(2*mu*(x^2 + y^2 + z^2)^3 - 6*mu*z^2*(x^2 + y^2 + z^2)^2 + 9*J2*Re^2*mu*(x^2 + y^2 +
z^2)^2 + 105*J2*Re^2*mu*z^4 - 90*J2*Re^2*mu*z^2*(x^2 + y^2 + z^2))/(2*(x^2 + y^2 + z^2)^(9/2));


F_Mat = [df1_dx df1_dy df1_dz; df2_dx df2_dy df2_dz; df3_dx df3_dy df3_dz];

F = [zeros(3,3) eye(3,3); F_Mat zeros(3,3)]; %Final F System Description Matrix
```

```
%% Non Linear Equations for Motions in ECI Frame:
% -- Symbolic Math is Inserted Above
% -- Matrix if a function of time because elements change but symbolically
% it is static and only needs to be evaluated once
%
% syms r x y z J2 Re mu flat_effect
% r = sqrt(x^2+y^2+z^2);
% flat_effect_A = 1.5*J2*(Re/r)^2*(1-5*(z/r)^2); %Harmonic A
% flat_effect_B = 1.5*J2*(Re/r)^2*(3-5*(z/r)^2); %Harmonic B
% xACC = -((mu*x)/r^3)*(1+flat_effect_A);
% yACC = -((mu*y)/r^3)*(1+flat_effect_A);
% zACC = -((mu*z)/r^3)*(1+flat_effect_B);
% F_Time = simplify(jacobian([xACC; yACC; zACC],[x,y,z]));
```

Transition Matrix Function

```
function [TransitionMat] = calc_trans_mat(F,delta_t) %Parses in System Design Matrix
TransitionMat = eye(6,6) + F*delta_t + (F^2*delta_t^2)/factorial(2) +
(F^3*delta_t^3)/factorial(3) + (F^4*delta_t^4)/factorial(4);
```

RK4 Propagation Function

```
%Numerical Integration Method: RK4 to find numerical solution to dynamic equations

function [state_prop] = rk4_prop(state_vec_old,h)
r = state_vec_old(1:3); v = state_vec_old(4:6);
k1 = [v calc_acc(r)]; %[dr dv]
k2 = [v + (k1(4:end)/2)*h calc_acc(r+(k1(1:3)/2)*h)];
k3 = [v + (k2(4:end)/2)*h calc_acc(r+(k2(1:3)/2)*h)];
k4 = [v + k3(4:end)*h calc_acc(r+k3(1:3)*h)];

state_prop = state_vec_old + (1/6)*h*(k1+2*k2+2*k3+k4);
```