The University of Sheffield

ACS6124 Multisensor Assessment

Leander Stephen Desouza

Registration Number: 230118120

Department of Automatic Control Systems Engineering

Contents

List of Figures

1 Task 1 - System Design for Aircraft Climb

1.1 Jacobian Derivation for the Non-Linear System

The system dynamics equations for designing a Kalman Filter for Aircraft climb are usually dictated by the following three equations:

1. Kinematic Equation:

$$\boxed{\dot{x} = f(x(t), c(t), t) + g(x(t), c(t), \omega(t))} \tag{1}$$

2. Observation Equation:

$$\boxed{d(t) = h(x(t), t)} \tag{2}$$

3. Measurement Model:

$$\boxed{d_m(t_k) = d(t_k) + \nu(t_k), k = 1, 2, ...} \tag{3}$$

Here, x is the state variable, c is the input variable, d is the output variable with d_m as the measured outputs, ω is the system noise and ν is the output measurement noise.

First, we linearize the system by calculating the Jacobians for the state transitional model (f), process noise model (g), and the observational model (h).

$$F = \frac{\partial f}{\partial x}, G = \frac{\partial g}{\partial \omega}, H = \frac{\partial h}{\partial x}$$

Here, r is the IMU angular rate, ω_r is the process noise in the IMU yaw, g_{grav} is the acceleration due to gravity, and u and w are the ground earth speed components, and ϕ and θ are their respective angles. The following are the requested elements of the

linearized Jacobian matrix of the following:

$$\boxed{F(5,4) => -r_m = -r + \omega_r}$$

$$\boxed{F(6,7) => -g_{grav}\cos\theta\sin\phi = -9.81\cos\theta\sin\phi}$$

$$\boxed{G(4,6) => -v}$$

$$\boxed{G(9,5) => -\frac{\sin\phi}{\cos\theta}}$$

$$\boxed{H(7,7) => 1}$$

$$\boxed{H(11,4) => \frac{-w}{u^2(\frac{w^2}{u^2}+1)}}$$

1.2 Kalman Filter - Initial State Estimation

The following states' initial values and standard deviations have been mentioned in the instruction manual:

$$\sigma_{A_x} = \sigma_{A_y} = \sigma_{A_z} = 0.01m/s^2$$

$$\sigma_p = \sigma_q = \sigma_r = 0.01 * (\pi/180)rad$$

$$\sigma_{x_E} = \sigma_{y_E} = 5m, \sigma_{z_E} = 10m$$

$$\sigma_u = \sigma_v = \sigma_w = \sigma_{V_{tas}} = 0.1m/s$$

$$\sigma_\phi = \sigma_\theta = \sigma_\psi = \sigma_\alpha = \sigma_\beta = 0.1 * (\pi/180)rad$$

$$[x_E, y_E, z_E] = d_k(1, 1:3), [\phi, \theta, \psi] = d_k(1, 7:9)$$

$$u = v = V_{wxE} = V_{wyE} = V_{wzE} = 0$$

Before generating the estimated states using KF, we proceed to estimate the true wind speed V_{tas} as the mean of all the measurements, which comes out to 55 m/s, and the rest of the components of the wind speed were set to zero; this is quickly replaced by the stabilized value that is 92m/s as seen from 1. The standard deviations of the initial state vector were assumed to be low (0.5) for x_E, y_E, z_E, ϕ, θ and ψ because the initial estimates were taken from the dataset and were known prior. The deviations for the rest of the states were kept at a relatively larger value (100). The estimation for yaw converged

to 6.3rad, which was different from the initial estimate.

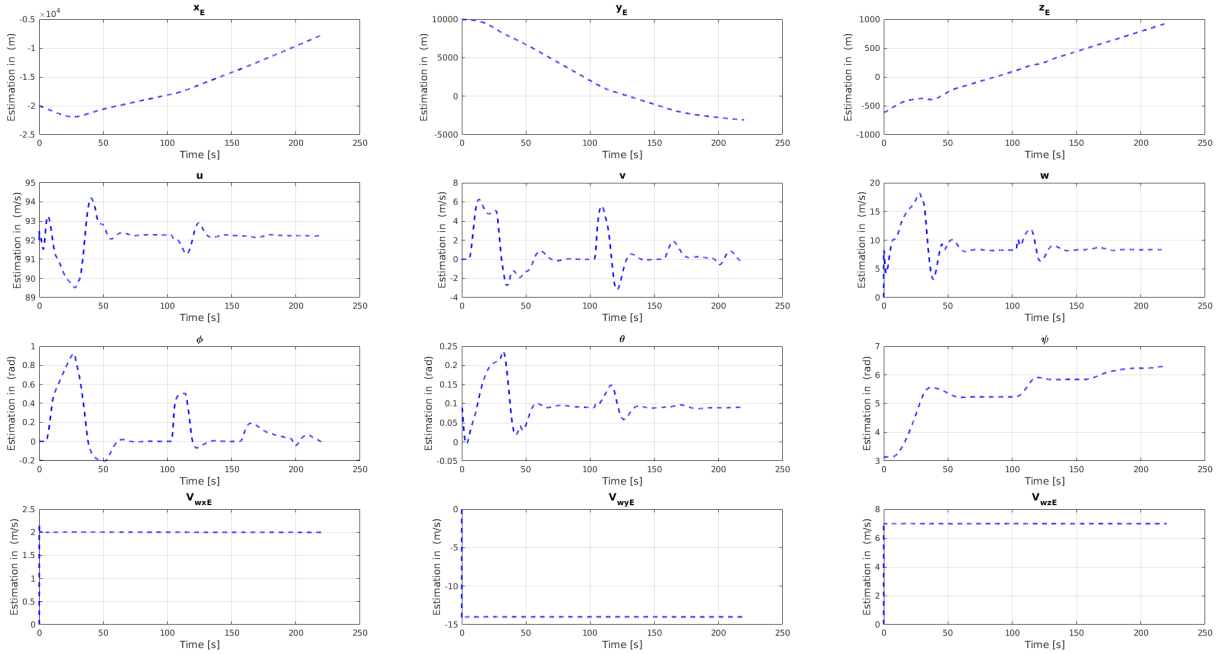

Figure 1: State Estimation Vector after Kalman Filter

2 Task 2: Sensor Biases Removal using Filtering Techniques

2.1 Comparison of datasets with and without Biases

After superimposing the biased dataset of the stated task and the former task, there is a clear indication of some states being unchanged and some with a shift due to added bias. The figure 2 indicates that affected sensors must be the ones measuring the earth's speed in all directions (u, v, and w), the angle of rotation in pitch (θ), and wind speeds in the earth's frame at x (V_{wxE}) and z (V_{wzE}) axes.

This can happen for several reasons. The magnetometer measuring the earth's field is prone to the differences in the aircraft's electronics and magnetic fields at the equator and poles. Turbulence and wind gusts can cause the anemometer to start reading from a bias. Gyroscopes in IMUs are also susceptible to drifts over time, as well as the sensor's natural ageing.

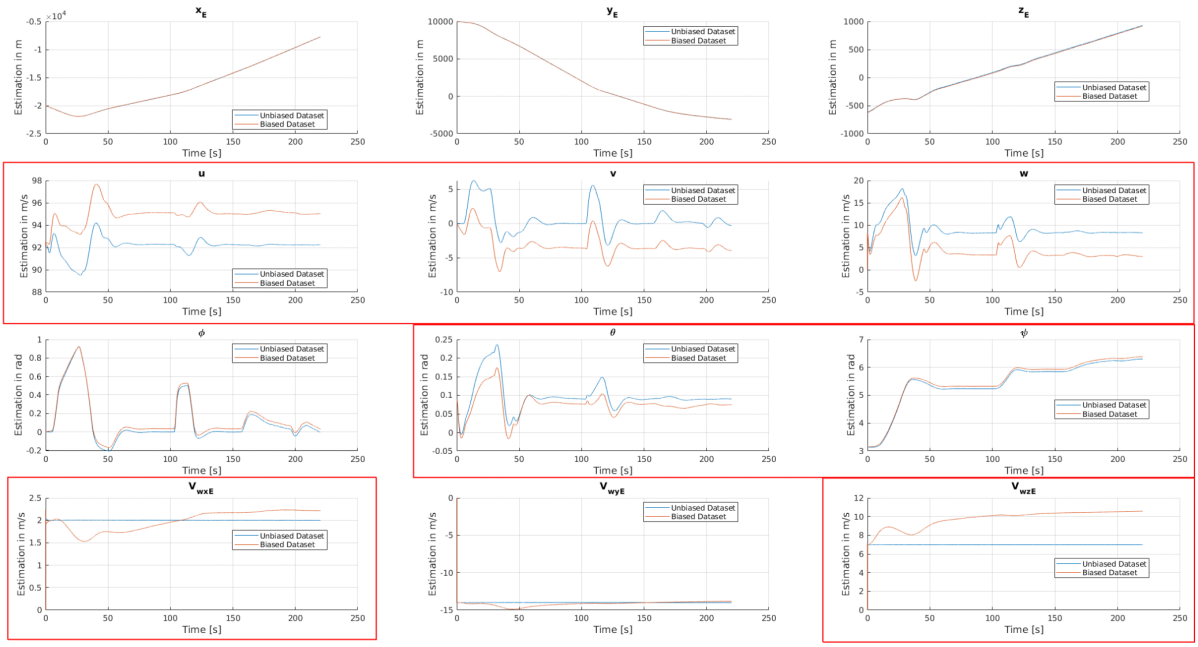

Figure 2: Biased and Unbiased State Estimation after KF

2.2 Biased Sensor Mitigation

The standardized procedure to mitigate bias is to introduce bias terms and inherently capture it. This is done by introducing these values into the state vector and setting its derivative to zero.

$$x_vector = [x_E, y_E, z_E, u, v, w, \phi, \theta, \psi, b_{A_x}, b_{A_y}, b_{A_z}, b_p, b_q, b_r, V_{wxE}, V_{wyE}, V_{wzE}]] \qquad bias_differential = z$$

The F, G, and H matrices must now be recomputed in the linearization process and effectively substituted in the Kalman Filtering algorithm. A very natural change is observed in the form of spikes in the beginning as the bias begins to converge. Hence, we skip about to about 5 seconds in the beginning to observe a gradual settling and dampening process. As seen in 3, all the biases tend to flatline after the initial spike at the shifted time period. Thus, the biases have been mitigated.

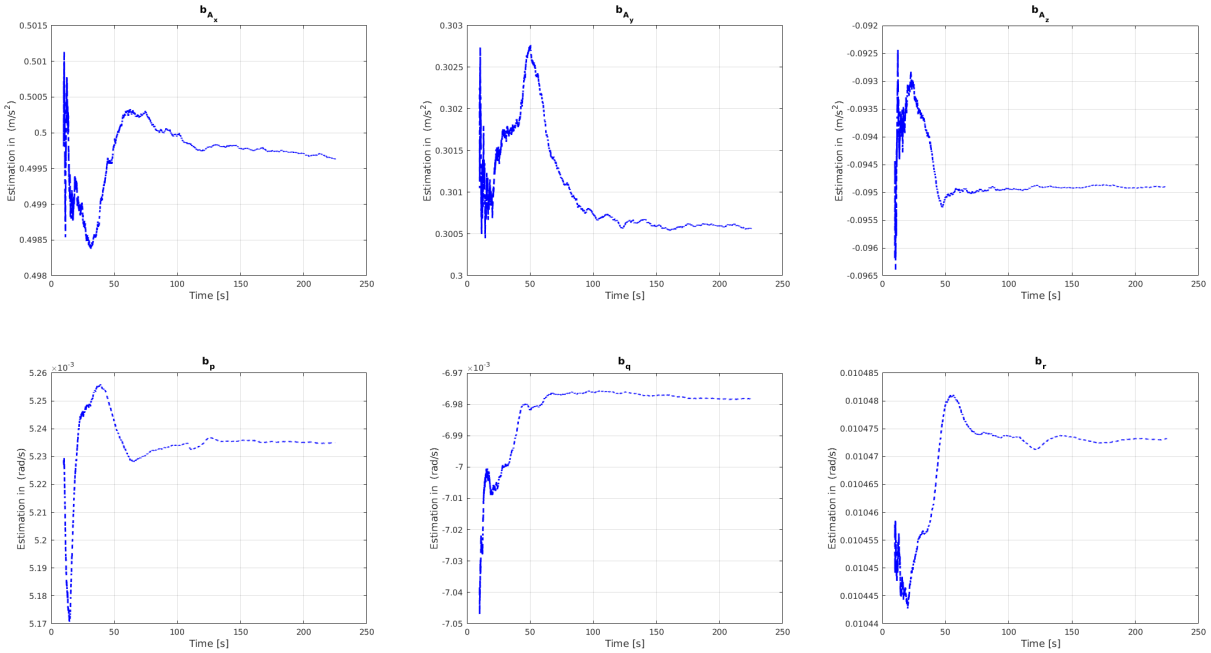

Figure 3: Bias Convergence after State Vector Modification

From the graphs, the biases converge to the following values:

$$b_{A_x} = 0.4996$$

$$b_{A_y} = 0.3005$$

$$b_{A_z} = -0.0948$$

$$b_p = 0.00523$$

$$b_q = -0.00697$$

$$b_r = 0.01047$$

To analyze the effectiveness of mitigation on the system states, we plot the nominal data, biased data and the filtered biased data on a single graph. This is represented in the figure 4.

As observed, the filtered bias data almost exactly overlaps with the nominal data from Task 1. This indicates an obvious smooth mitigation of biases in the state estimation vector.

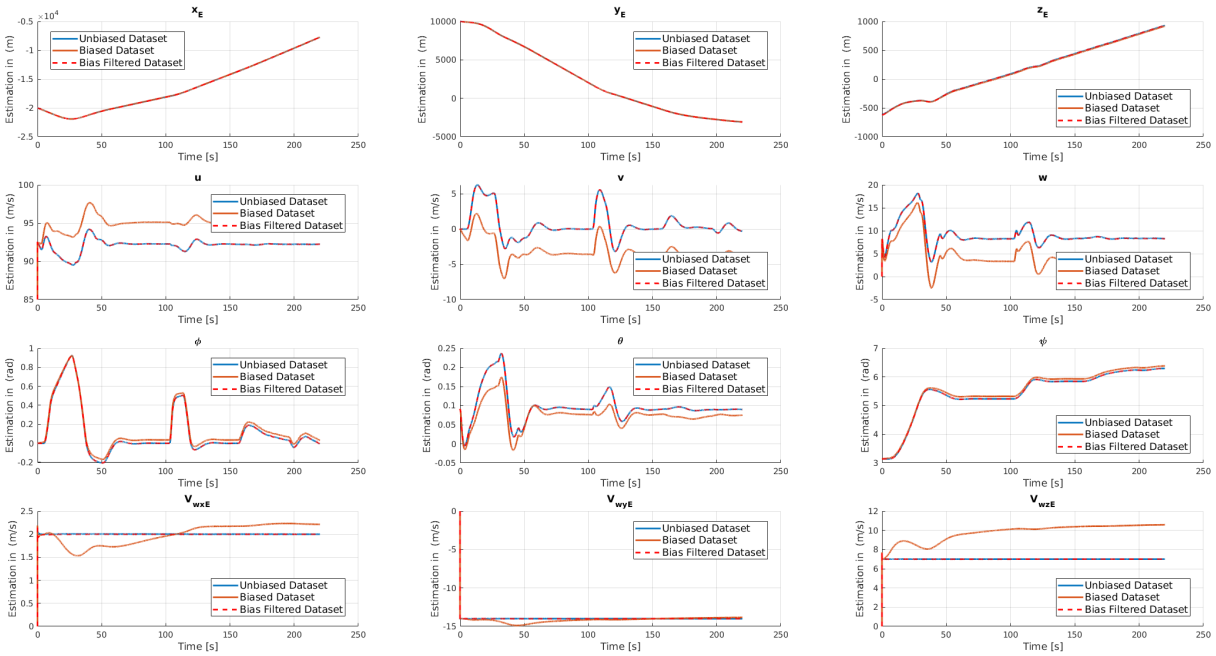

Figure 4: State Estimation after Bias Mitigation

3 Task 3: Sensor Fault Detection and Diagnosis

3.1 Sensor comparison with devoid and fault presence

A structured comparison is made between the current dataset and the mitigated bias dataset from the previous task. The state estimation of the ground earth speeds, angle of rotation and the wind measurement sensors seem to have encountered various faults in its prediction. This can be clearly observed in figure 5.

The influence of these faults leads to a non-zero mean of the innovation data of the states, as seen in figure 6. This leads to a difference in the prediction and the true data, leading to incorrect state estimations. Overall, the innovation of the system is calculated using the following equation:

$$innov(k,:) = z_k(k,:) - z_{k_{m1}}$$

Here *innov* represents the innovation, z_k is the observed measurement, and z_{km1} is the predicted measurement calculated at time k.

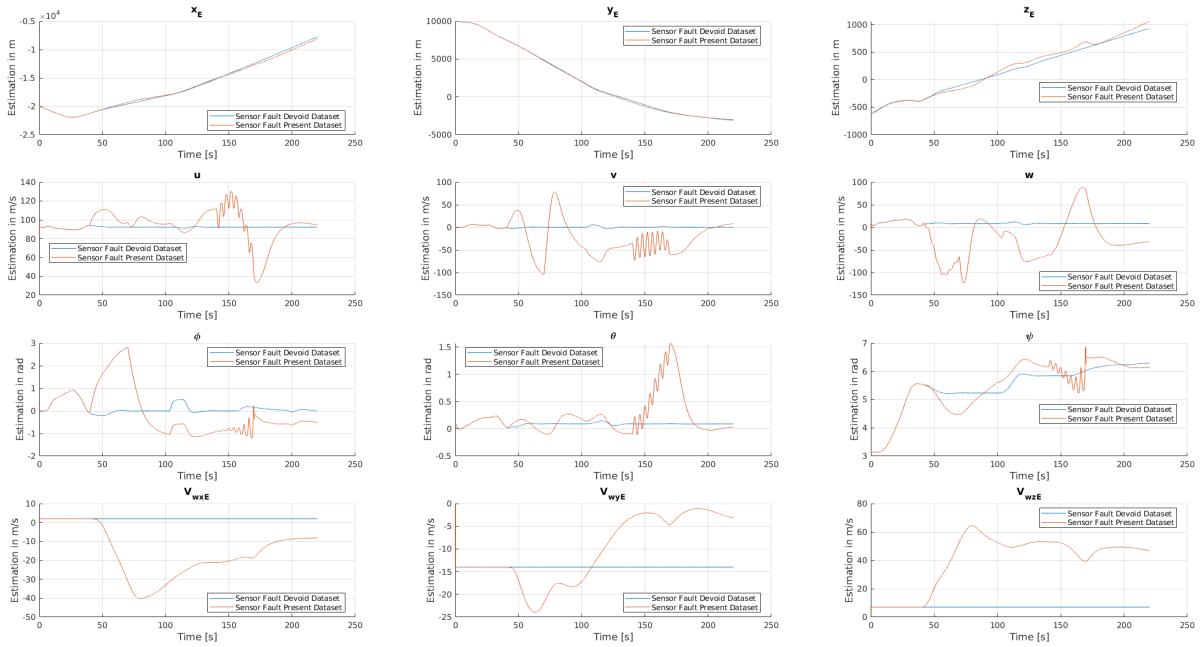

Figure 5: State Estimation of Devoid and Fault presence in feature vector

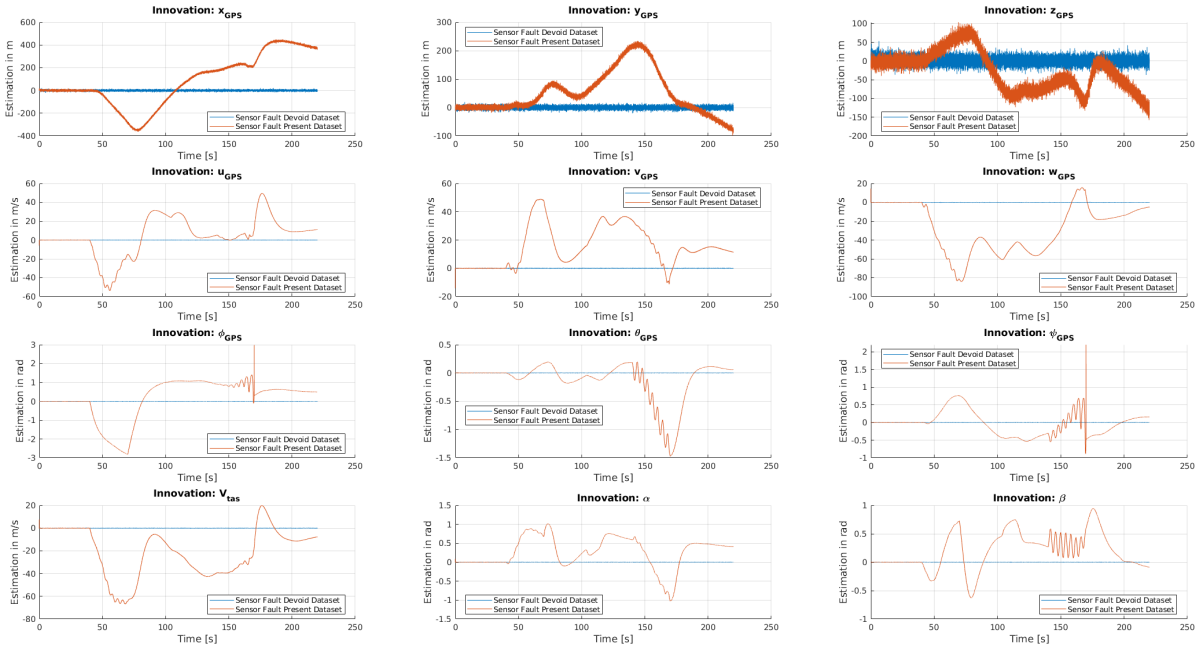

Figure 6: Innovation comparison of Fault and Faultless data

3.2 Removal of Fault using modifications in Kalman Filter

We now proceed to add further noise terms in the process vector. The additional noise terms are incorporated to motivate the absorbing of all the faults present in the sensor.

$$\omega = [\omega_{A_x}, \omega_{A_y}, \omega_{A_z}, \omega_p, \omega_q, \omega_r, \omega_{bA_x}, \omega_{bA_y}, \omega_{bA_z}, \omega_{b_p}, \omega_{b_q}, \omega_{b_r}]^T$$

The bias update for each of the IMU noise gets updated through a random walk model as follows:

$$
\begin{aligned}
b_{A_x}(t_k) &= b_{A_x}(t_k) + \omega_{b_{A_x}}(t_k) \\
b_{A_y}(t_k) &= b_{A_y}(t_k) + \omega_{b_{A_y}}(t_k) \\
b_{A_z}(t_k) &= b_{A_z}(t_k) + \omega_{b_{A_z}}(t_k) \\
b_p(t_k) &= b_p(t_k) + \omega_{b_p}(t_k) \\
b_q(t_k) &= b_q(t_k) + \omega_{b_q}(t_k) \\
b_r(t_k) &= b_r(t_k) + \omega_{b_r}(t_k)
\end{aligned}
$$

For the linearization process, we transform these equations back to continuous time by using the fact that:

$$
\begin{aligned}
\dot{b_{A_x}} &= \frac{1}{T_s}\omega_{b_{A_x}} \\
\dot{b_{A_y}} &= \frac{1}{T_s}\omega_{b_{A_y}} \\
\dot{b_{A_z}} &= \frac{1}{T_s}\omega_{b_{A_z}} \\
\dot{b_p} &= \frac{1}{T_s}\omega_{b_p} \\
\dot{b_q} &= \frac{1}{T_s}\omega_{b_q} \\
\dot{b_r} &= \frac{1}{T_s}\omega_{b_r}
\end{aligned}
$$

Since the process vector is only affected, only the G matrix needs modifications in the Kalman Filter estimations. After the estimation process, we can see that the graph in Figure 7 converges to the state estimations of the previous task.

After fault mitigations, the innovations also prove to have a combined mean of 0. This indicates a successful mitigation of faults in figure 8.

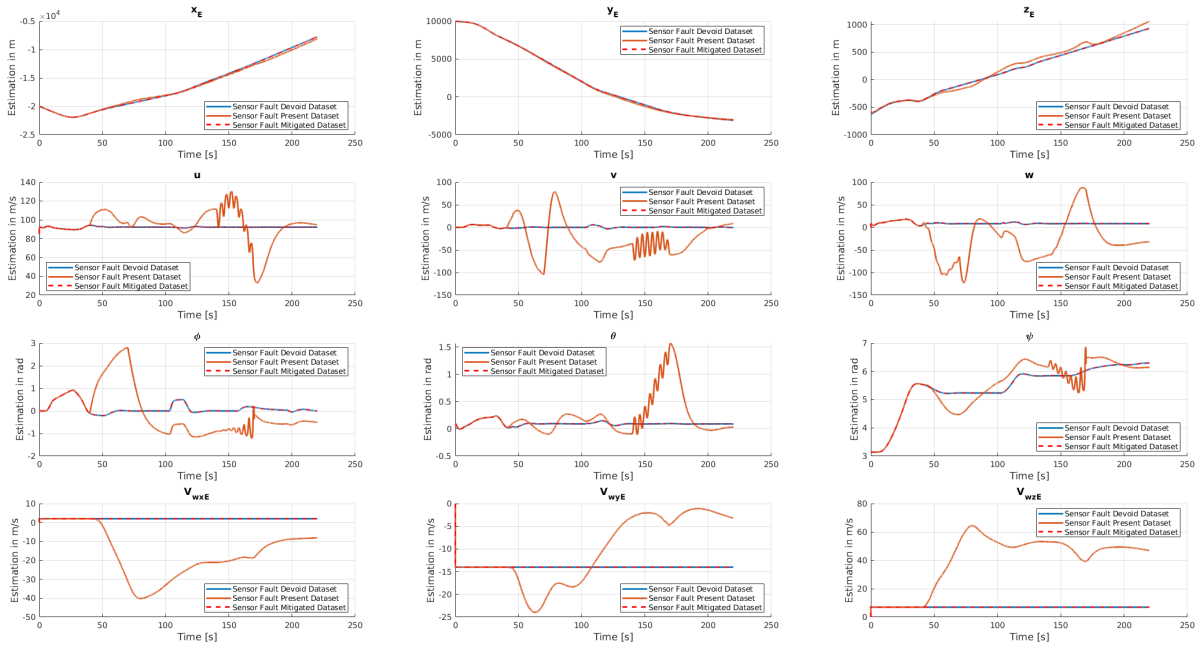

Figure 7: State Estimation after Fault removal

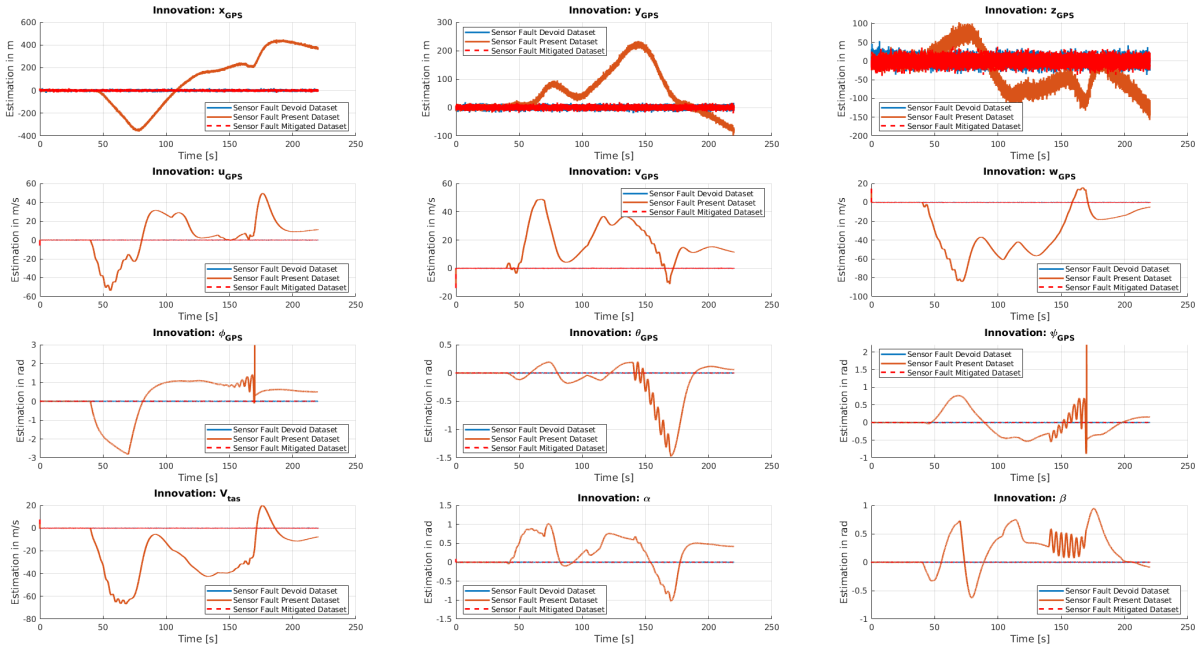

Figure 8: Innovation after Fault removal

3.3 Detection of Fault Onset times through CUSUM

AFter the initial FDD analysis through the Kalman Filter, several observable faults appear in the bias terms that were introduced to counteract the fault introduced. This shows about the nature of various faults that were introduced in the IMU sensors. As seen in the figure 9, there are combinations of step, ramp, and sinusoidal faults.

To detect measurement faults, we implement a two-sided cumulative sum (CUSUM)

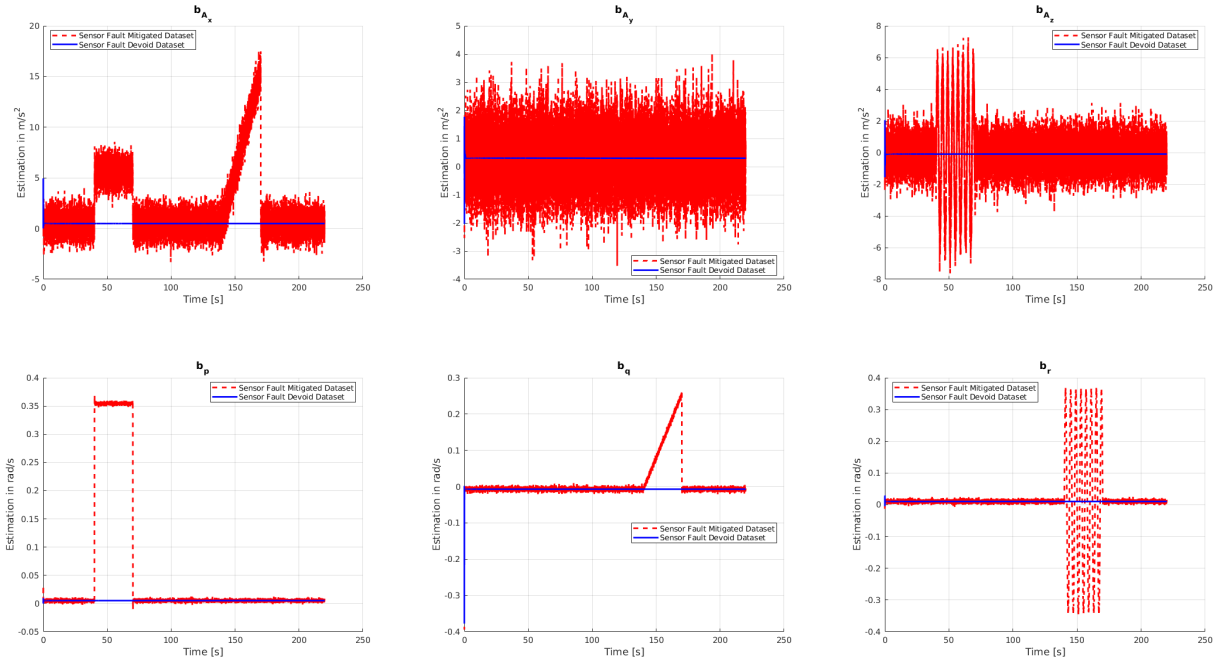

Figure 9: Step, Ramp and Sinusoidal faults observed in bias estimates

approach to detect fault onset times. Here are some of the reasons for this approach:

1. **Reduction of State Estimation Uncertainty**: The biases' structure is usually taken advantage of to achieve a more precise weighted contribution to their corresponding states.

2. **Robustness to false positives/negatives**: The probability of detecting false alarms in the data is reduced as it can effectively differentiate between the two states.

3. **Quick and Time Efficiency**: Since, the CUSUM test is memoryless, it allows for applications in highly dynamic environments and can be applied for situations where data is available at every moment in time.

After applying the CUSUM test on these estimates, we get several fault detections through the thresholds that have been set empirically. However, a lot of false positives tend to show in this process. The following are some of the methods that have been employed to nullify some of the false readings:

1. **Shifting Readings**: The biases' tend to oscillate rapidly in the start and settle at about 5 seconds from its initial state. This helps mitigating false high readings getting through as faults due to the thresholds set.

2. **Tuning Leaks and Thresholds**: Setting leaks and thresholds as a function of the peak of the states, proves to be helpful in the robustness of varying datasets.

3. **Identification of Fault onset times**: Identifying the correct starting point of the fault is crucial, this is done by using the inbuilt *ischange*() function of MATLAB to identify discontinuity in the time instances. Furthermore, basic buffer checks have been made to ensure the continuity of the detected faults.

In the figure 10, we can observe the detection of the faults in the six states. As inferred, there seems to be no fault in the A_y state, whereas clear time instances can be demarcated for the fault times. This values are then plotted over the original bias estimate in 11.

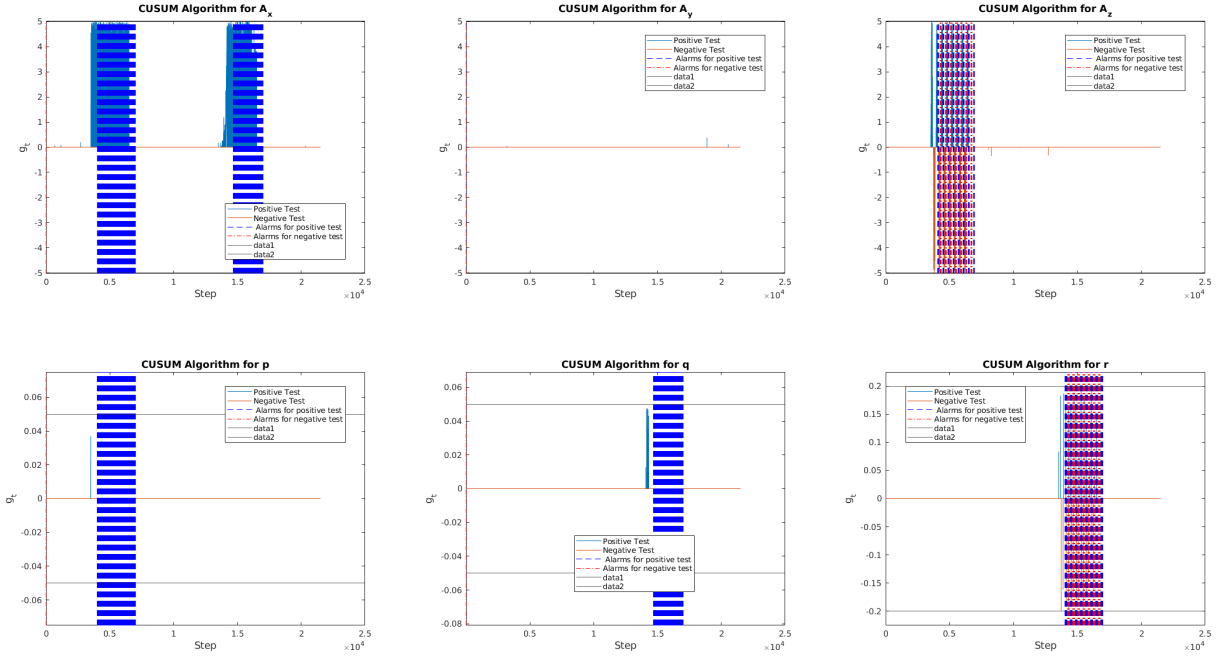

Figure 10: Time estimates of bias faults through CUSUM analysis

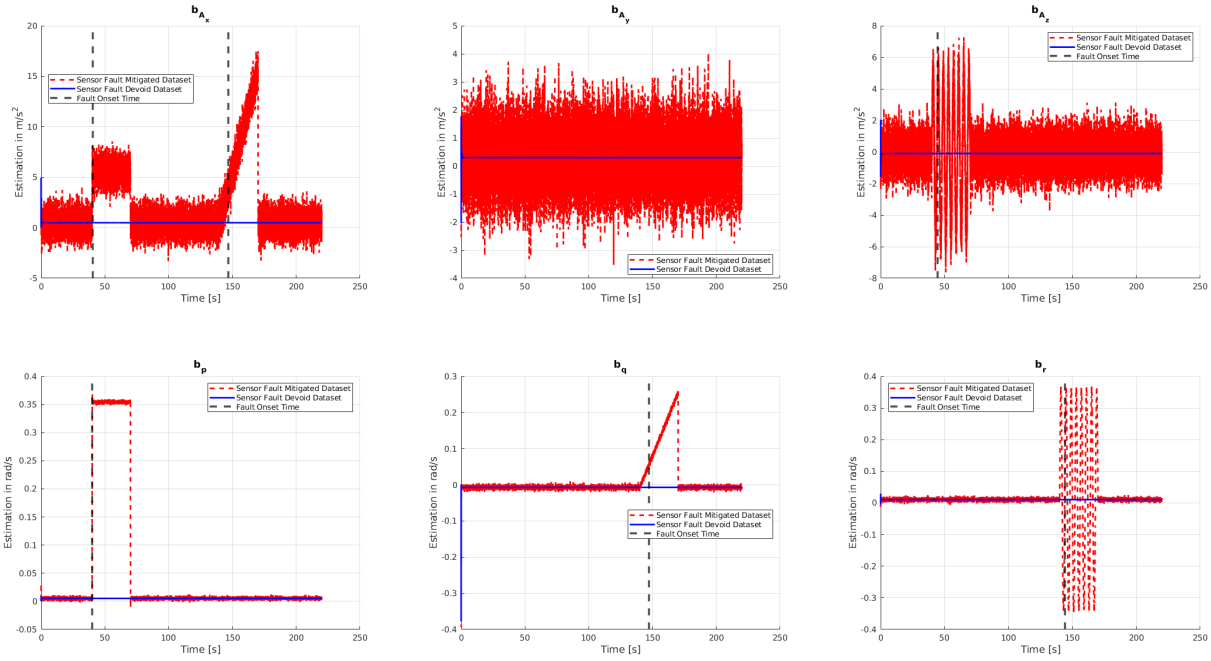

Figure 11: Fault onset times estimated from CUSUM

3.4 Cyber Attack on the sensor measuring Angle of Attack

The corresponding dataset provided indicates tampering with the angle of attack. This is indicated by the spike in the innovation plot of α. To combat this, we measure the standardized innovation against a known variance factor. If the innovation of the AoA exceeds the former threshold, then the measurement is ignored, and the innovation is set to 0. In the following equation, where the standardized innovation is represented, V_e is the overall prediction.

```
innov(k, :) = (z_k(k,:) - z_k_km1) / sqrtm(Ve);


if abs(innov(k, 11)) > var_factor
    z_k(k, 11) = z_k_km1(11); % Discard measurement
    innov(k, 11) = 0;
end
```

We can immediately observe that after this change in the measurement update of the Kalman Filter, the innovation drops to its nominal value. This is seen in the figure 12. The onset time of the fault is observed from the CUSUM test on the angle of attack and is superimposed on the said figure.

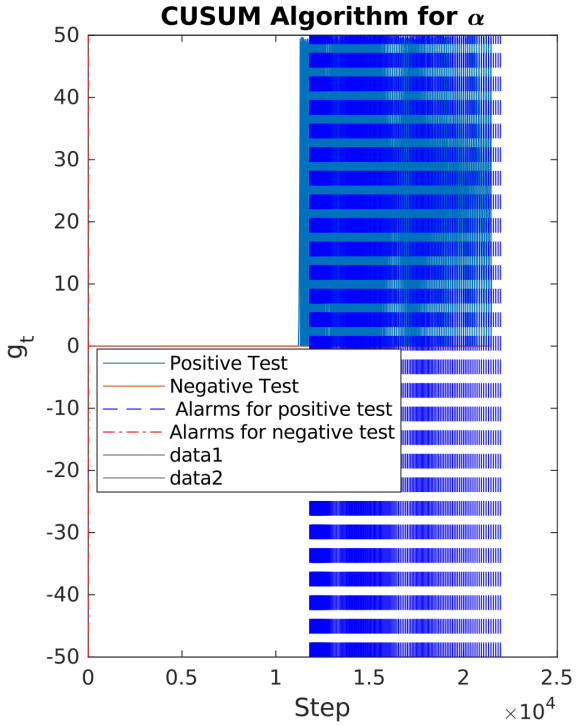

Figure 12: Angle of Attack fault time observed from the CUSUM test

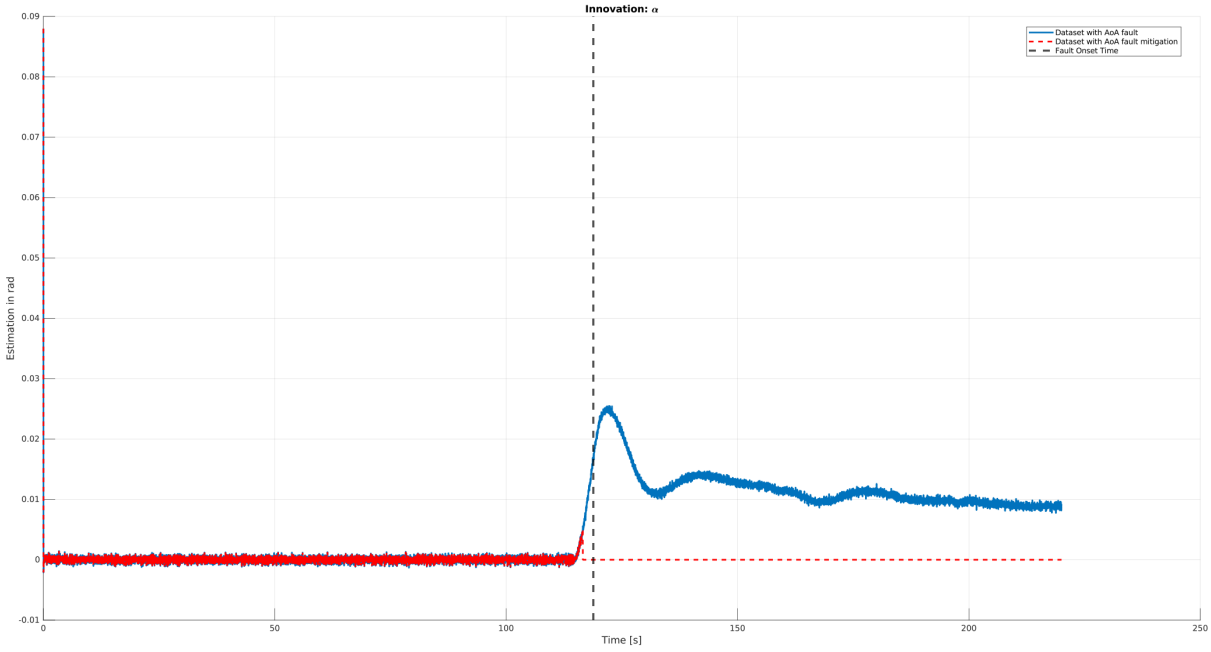

Figure 13: Innovation plot with reset and fault onset time embedded

A good way to check if the fault was mitigated is to analyze what input state variables, the angle of attack depended on.

$$\boxed{\alpha = \arctan(\frac{w}{u})}$$

Therefore, there should be observable changes in the mitigated data and the faulty data in terms of the u and v state variables. As seen in the figure 14, there exists justifiable correction in u, and hence, we have successfully mitigated the cyber attack on the angle of attack sensor.

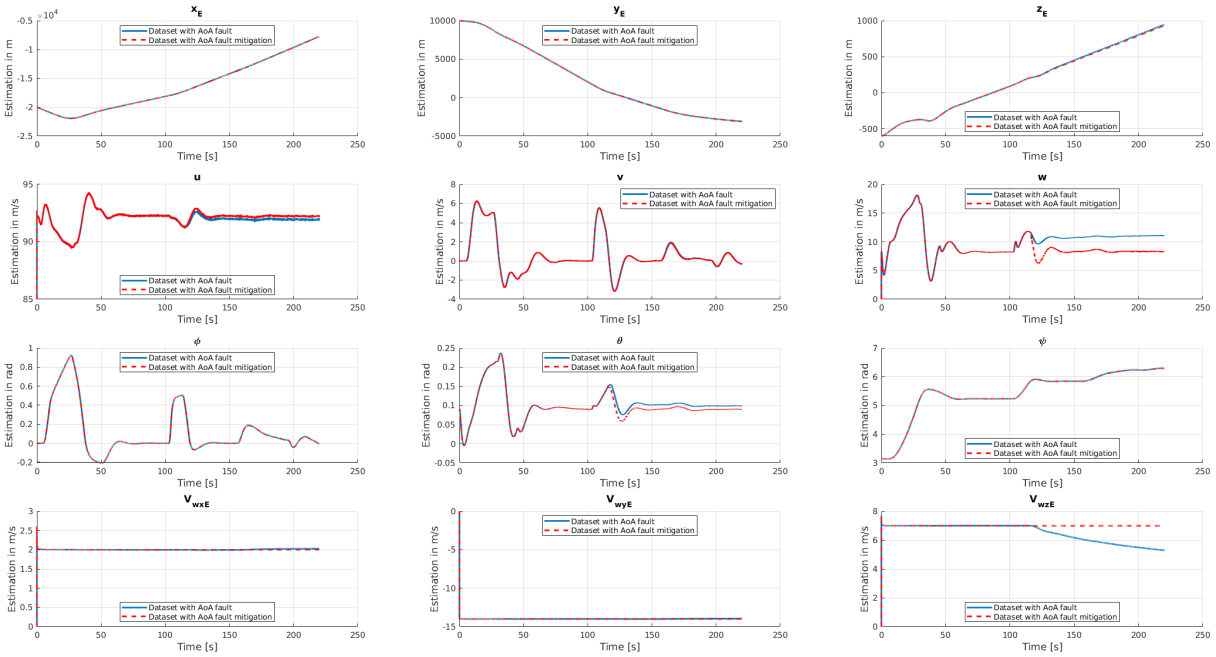

Figure 14: System States after Cyber Attack Mitigation