Fault Detection and Identification for Aerospace Systems

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Master of Technology

IN

Faculty of Engineering

BY

Balivada Venkata Ramkiran

Computational Data Science
Indian Institute of Science
Bangalore – 560 012 (INDIA)

August, 2021

Declaration of Originality

I, **Balivada Venkata Ramkiran**, with SR No. **06-18-01-10-51-19-1-17545** hereby declare that the material presented in the thesis titled

Fault Detection and Identification for Aerospace Systems

represents original work carried out by me in the **Department of Computational Data Science** at **Indian Institute of Science** during the years **2020-21**.
With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: 04-06-2021 Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name: Prof. Deepak Subramani Advisor Signature

Acknowledgements

I would like to thank my esteemed supervisor – Dr. Deepak Subramani for his invaluable supervision, support and tutelage during the course of my M.Tech degree. My gratitude extends to the Faculty of IIsc for developing knowledge & understanding to undertake my studies at the Computational Data Science, Indian Institute of Sciences. Additionally, I would like to express gratitude to Mr. Kunni krinshnan (Director, URSC), Mr. Udupa Subrmanyam(DD, URSC) ,Dr. N.K. Philip (DD,URSC), Dr M.S. Shiva (GD,CDDG), Dr. Vinod Kumar (DH, GCDAD), and collegues of CDDG,URSC for their treasured support which was really influential in shaping my experiment methods and critiquing my results. I would like to thank my friends, lab mates, colleagues and research team – Akanksha, Abhishek, Anjali, Chinmay, Devayani, Rathnakar, Rohit, Srikanth, Rahul, Avishek, Karan, Kaushal for a cherished time spent together in the lab, and in social settings. I especially thank Chinmay for his immense support in implementing LNN over satellite data. My appreciation also goes out to my family and friends for their encouragement and support all through my studies.

Abstract

This work report discusses about new methods to identify sensors and actuators faults of a satellite system pertaining to Control system subsystem. We use Kalman filter based data assimilation techniques to identify faults with limited knowledge in system dynamics. Generalised Observer Scheme (GOS) which works based on the observablity of combination of sensors is used to identify sensor faults and new model learning scheme to accommodate unmodelled/changed dynamics is used to identify actuator faults. This work is followed by study of machine learning models for fault detection on data generated by unknown models. Finally the report describes the methods to improve navigation solution of Low Earth Orbit Satellite (LEO) Satellite using only 2-body dynamics model and only Magnetometer and Sun sensor.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter gives various definitions of fault detection and their related work

1.1 Fault Detection & Identification - An introduction

Real-time health monitoring of a satellite system in the space industry has become a necessity with the increasing number of satellites launched per year. Satellite health monitoring typically involves evaluating the various subsystem's health parameters from telemetry data. With the increasing number of satellites, manual health monitoring of the satellites has become very cumbersome and error-prone calling for automation in health monitoring. Fault Detection, Identification, and Isolation (FDI) algorithms are at the forefront to deal with such problems.

FDI algorithms generally deal with some kind of anomaly/Novelty detection on the available data. In the current satellite system, these FDI algorithms should have the methodology to take real-time telemetry data as input and give an accurate prediction of satellite health as soon as possible. Any FDI algorithm generally deals with four major operations namely [22]

1. **Detection** which detects the fault
2. **Isolation** which identifies the variable/part of the system that caused the fault
3. **Identification** to identify the type of fault viz. abrupt/drift/intermittent fault etc
4. **Diagnosis** to find the cause that led to the failure

As such, any satellite subsystem contains typically 30-40 subsystems and it's practically impossible to make FDI algorithms without the consent of the subsystem domain expert. Hence for the current project, we are considering the satellite Attitude & Orbit Control System (**AOCS**) subsystem to test and implement our FDI algorithms.

A brief introduction of **AOCS** subsystem is described below. AOCS is responsible for the maintenance of attitude (orientation) and orbit of the spacecraft. This subsystem measures from the various onboard sensors and also from ground radars to generate the required control signal to actuate wheels and thrusters. AOCS has also got several modes of operation based on the combination of sensors and actuators in use while in operation. The most common mode of satellite operations uses gyros and star-sensor as sensors and momentum exchange devices i.e. reaction wheels as actuators.

In this setup, there are 3 possible places where there can be potential failures namely sensors, actuators, and AOCS computer. In this project, we are aiming to deal with sensor and actuator failure on the most common mode of AOCS operation. The following subsections give a brief introduction to sensors and actuators used in satellite subsystems.

1.1.1 Sensors

Angular rate sensor-Gyros and Star-sensor are the sensors used to measure the orientation of the satellite. Gyros measure the angular rate of the satellite while the Star-Sensor measures the orientation of the satellite concerning the inertial frame. Failure of a gyro/star-sensor can be devastating for satellite health and hence these failures have to be identified immediately before the satellite is lost forever. In this project, a Generalized observer scheme (GOS) is used to identify the sensor failures and this method will be further discussed in the methodology section. Also, methods are developed to identify gyro failures using machine learning (ML) algorithms used in anomaly/novelty detection.

1.1.2 Actuators

Reaction wheels are momentum exchange devices used to control the spacecraft attitude by exchanging angular momentum between the spacecraft and reaction wheels. Failure of reaction wheel in a particular axis could make the spacecraft uncontrollable in that axis and hence attitude control had to be changed to another actuator mode like thrusters immediately. Many algorithms in the FDI domain deal mainly with sensor failure but not actuator failure. Most of the actuator failure algorithms present in the literature have some kind of sensors measuring the performance of the actuator. In the current project, we devised an algorithm that identifies actuator faults via identification of the change in parameters, learnt for an augmented basis to the existing model. More details of this method are discussed in the methodology section.

1.2 Navigation Solution - An introduction

In this report, another application namely "Navigation solution using Sun-sensor and Magnetometer" is presented applying the similar methodology used to identify actuator failure. Navigation for small satellites using Sun-sensor (SS) and magnetometer (Mag) is an attractive proposition as both these sensors have low cost, weight and power consumption when compared to Global Positioning Sensors (GPS)

Magnetometer and sun sensor measures the vector quantities magnetic field and sun vector respectively, in the body frame. Magnetic field measurements in the body frame depend on the current position (i.e., orbit) and satellite orientation concerning the inertial frame. The sun vector measurements depend on the orientation and universal time. Notably, the sun vector is available for measurement only during the non-eclipse duration of any given orbit, which depends on the orbit shape and semi-major axis. The estimation problem under consideration is to compute the 7 orbital parameters - position (3), velocity (3), sidereal angle (1), and 4 attitude parameters - quaternions, from the noisy magnetic field vector and sun vector measurements.

This report is organised such that each section contains two parts where part A gives details about the implementation of the FDI algorithm for AOCS while part B contains the topics related to the Navigation solution.

1.3 Related Work

1.3.1 FDI - Related Work

FDI is a well estsblished field of study and according to the survey papers [11], [10] and [22] and book [14] FDI algorithms can be classified into three major categories.

1. **Model Based** - The essence of this concept is the comparison of the actual outputs of the monitored system with the outputs obtained from a (redundant. I.e. not physical) analytical mathematical model. It involves two stages: residual generation and residual evaluation. Parity equations, 'State observers', Parameter estimation, Nonlinear models(NN) are some of the model-based methods.

2. **Signal Based** - Signal based methods exploit only available experimental (historical) data. These methods are generally applied to a window of data. PCA, Spectrum Analysis, Dynamic Time Warping (DTW), Self Organising Maps (SOM) are some of the Signal based methods.

3. **Knowledge-Based (AI-ML)** - The consistency between the observed behaviour of the

operating system and the knowledge base (trained model) is then checked, leading to a fault diagnosis decision with the aid of a classifier. The model of the original system is trained from historical data.

In the above-mentioned methods, we primarily concentrate on the model-based methods for the implementation of FDI algorithms for satellite Attitude and the Orbit control system. In model-based FDI, measurements from the real system are compared against the output of an accurate mathematical model. If the error between the actual measurement and model output is above a threshold a fault is detected.

A brief review of Satellite health monitoring and its large scale implementation was described in [6]. Author of [6] also describes the practical implementation architecture design of the FDI system. Authors of [13] describe the detailed implementation of the GOS algorithm for sensor FDI under favourable observability conditions among the various sensors. But the same algorithm fails in the case of an actuator fault detection. [28] describes a parity space-based algorithm for satellite FDI, but by the very design of this algorithm does not work actuator faults.

In this project to implement sensor FDI GOS algorithm was implemented for satellite attitude control system as mentioned in [13]. But for implementing actuator FDI a novel Bayesian learning-based algorithm is implemented. The motivation for this approach is found in [1] and [4] where the authors talk about adaptive updating of the process covariance matrix for dynamical systems with partially known dynamics.

FDI for unknown models can be majorly classified into two different ways. One way would be using model learning algorithms such as LSTM network ([40]), Lagrange Neural Network(LNN) [8], ARIMA type of algorithm. Other methods include extracting features from the time series data and further use these features for anomaly detection. Former methods generally learn about the model that generates the time series data and when faulty time series data is fed into the learnt model, the prediction from the model would be different from the actual measurement. This difference can be used to identify a faulty response. In the later algorithm anomalies are detected using anomaly detection algorithms [25] such as Infinite Gaussian Mixture Model(IGMM) [29],[30], Isolation Forest(IF) [19], one-class SVM (OC-SVM)[20], Self Organising Map (SOM) [2],[7] etc. For these anomaly detection algorithms to work properly we need to extract features that are feasible for anomaly detection.

Several feature learning algorithms are tried to extract features from the time series data. Features could be derived by computing various statistical & frequency measures of time series

data in a specific time window as mentioned in [39],[38],[9], [18] & [37]. Other feature extraction methods include using Auto Encoder-Decoder (AED) network [17] with augmented loss functions for disentanglement [24],[21] of learnt features. We can also use Variational Autoencoder (VAE) as mentioned in [27] to learn the probability distribution of a particular feature from a time series data and we could define a fault based on the magnitude of the probability of a particular time series. All these methods were tried and accuracies of different methods were established in this thesis. Furthermore, several supervised learning algorithms such as Decision trees (DTrees), Random forests(RForest), Adaboost, Artificial Neural Networks (ANN), Support Vector Machines (SVM) etc are also implemented to establish benchmark accuracy on the available data

Anatomy of FDI algorithms can be seen in the Fig. 1.1. All the methods are described will be further discussed in the methodology section of this thesis.

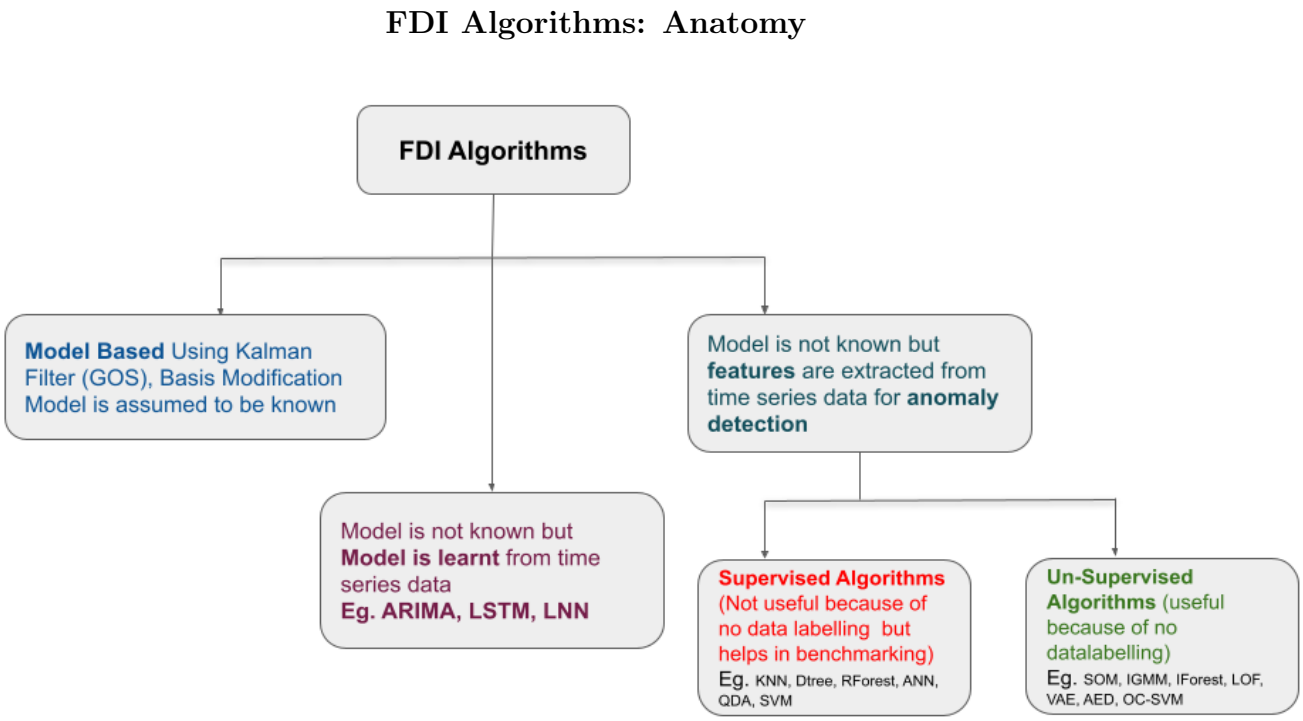

Figure 1.1: **Anatomy FDI algorithms used during different simuations i.e model, model-learning & time-series features**

1.3.2 Navigation Solution - Related Work

For work on navigation solution, the following references were studied. Magnetometer and Sun sensor-based attitude only determination is solved in [3] but the orbital parameters are not computed in this paper. Gyro, Magnetometer and Sun sensor-based attitude only determination is solved in [26]. and [35]. In [35] attitude determination is done considering the albedo model (Sun + Earth radiation together) which improves the accuracy of our estimation but again the orbital parameters are not solved in this paper. Magnetometer based Inertial navigation solution is solved in [36] and since only magnetometer measurement is used for estimation there are considerable orientation errors in estimation. SS and Mag based state vector estimation is solved in [12] eliminating attitude estimation and full navigation solution is computed in [15]. Most of the papers above involved computing partial derivatives of the magnetic field model which if self is a fit on spherical harmonic.

Chapter 2

Model Formulation

2.1 Models in FDI

In the current report FDI is applied on AOCS subsystem of a satellite in use. In the subsequent section we discuss models of various components used in AOCS subsystem. These are the same models used for simulating various nominal/failure scenarios used by the FDI algorithms.

2.1.1 Forward Model - Satellite Attitude Dynamics

Attitude Dynamics of the satellite aptly captures the dynamics of angular motion of the satellite under the application of various onboard actuators like wheels and thrusters. Six states represent Attitude dynamics namely three Euler angles and three-body rates of the satellite. The equations of motion are given in eq[2.1] & eq[2.2]

$$\dot{\vec{\theta}} = \omega \tag{2.1}$$

$$\dot{\vec{\omega}} = -\omega \times I\omega + T_c \tag{2.2}$$

where $\vec{\theta} = [\theta_x, \theta_y, \theta_z]$ & $\vec{\omega} = [\omega_x, \omega_y, \omega_z]$,
$T_c = [T_{cx}, T_{cy}, T_{cz}]$

$$I = \begin{bmatrix} I_{xx}, I_{xy}, I_{zx} \\ I_{xy}, I_{yy}, I_{yz} \\ I_{zx}, I_{yz}, I_{zz} \end{bmatrix}$$

Where

$\theta_x, \theta_y, \theta_z$ are Euler angles
$\omega_x, \omega_y, \omega_z$ are angular rates
I is the Inertia Matrix
T_c is the applied torque vector

2.1.2 Measurement Model - Gyro/Star-sensor Measurement Model

As mentioned above Gyros measure the satellite body angular rates while star-sensor measures the quaternions (i.e equivalent Euler angles) in an inertial frame. In the current problem formulation, Euler angles are assumed to be available for measurement. Hence the onboard sensors directly measure the states in the dynamics with additive Gaussian noise in each axis. The sensor model mentioned is given in eq[2.3] & eq[2.4]

$$\vec{\theta}_{SS} = \vec{\theta}_{dyn} + \vec{\theta}_{noise} \tag{2.3}$$

$$\vec{\omega}_{Gyro} = \vec{\omega}_{dyn} + \vec{\omega}_{noise} \tag{2.4}$$

Where
$\vec{\theta}_{SS}$ is the angles measured by Star-sensor
$\vec{\omega}_{Gyro}$ is the body rates measured by Gyros
$\vec{\theta}_{noise}$ is the noise vector added to angles
$\vec{\omega}_{noise}$ is the noise vector added to body angular rates

Along with the above-mentioned sensors, there are also other sensors like tachometers to measure the speed of the reaction wheel while in use. These tachometer measurements are monitored manually to assess the health of the actuator while in use.

Reference to the above mentioned models are available in [33].

2.1.3 Actuators Model

AOCS considered here has two different types of actuators namely reaction wheels (RW) and thrusters(Thr).

2.1.3.1 Reaction Wheels

RWs are momentum exchange devices and they control the satellite orientation by imparting/absorbing momentum from the spacecraft. Typically there are 4 RWs on satellite mounted in a tetrahedron configuration. For 3 axis attitude control, 3 RWs are sufficient for full contractility but an extra wheel is added for redundancy in case of anyone wheel failure. The torque

generated by RWs are given in Eq[2.5]

$$\vec{T}_{satellite} = W2B \times \vec{T}_{wheels} \tag{2.5}$$

where
$\vec{T}_{satellite}$ is Torque experience in satellite body axis
$\vec{T}_{wheels}$ is Torque generated by RWs
$W2B$ is the wheel to body torque conversion matrix

2.1.3.2 Thrusters

Thrusters are mass expulsion devices used to control both satellite orientation and position. There are 8 thrusters in a typical Indian Remote Sensing (IRS) satellite. Each thruster is mounted in a specific orientation such that all degrees of freedom for satellite control are covered. The torque produced by the thruster is given by Eq[2.6]

$$\vec{T}_{satellite} = \Sigma_{i=1}^{8} r_{cg}^{i} \times \vec{F}_{Thruster}^{i} \tag{2.6}$$

where
$\vec{T}_{satellite}$ is Torque experience in satellite body axis
r_{cg}^{i} is i^{th} thruster position with respect to center of gravity
$\vec{F}_{Thruster}^{i}$ Force vector of i^{th} thruster

Actuator positioning on a satellite is depicted in Fig 2.1

2.1.4 Controller

A Proportional Derivative Controller is used for attitude control of a spacecraft. Actuators and sensors in use can be of any of the above mentioned combination. Control flow diagram of a satellite attitude control is given in Fig 2.2

These models described above are used to simulate data to train Machine learning algorithms.

Satellite Actuator Configuration

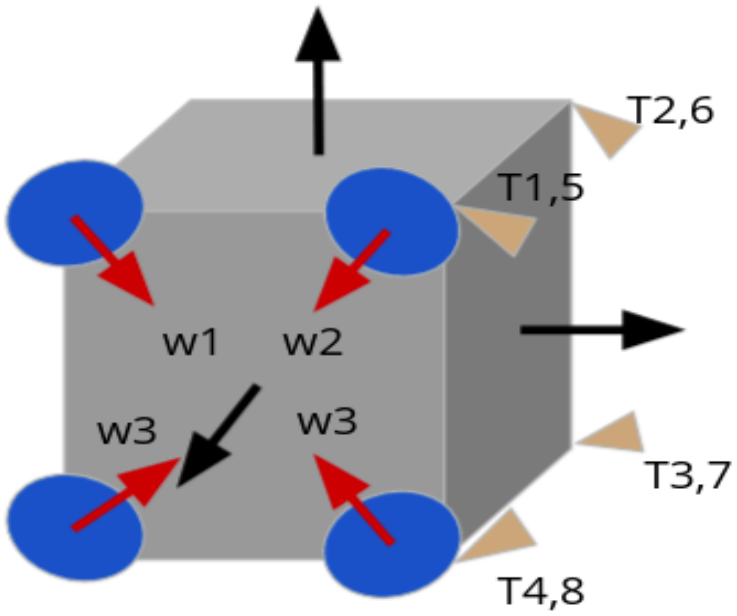

Figure 2.1: **Actuator Configuration of a spacecraft. W here represents wheels, mounted in a tetrahedron configuration and T represents thrusters, also mounted in a tetrahedron configuration**

Control Flow Diagram

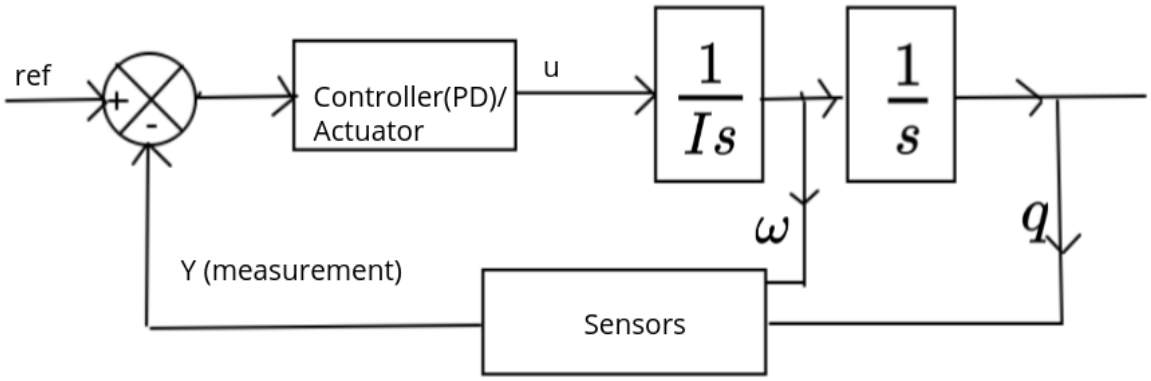

Figure 2.2: **Control flow diagram of a satellite control system with actuators and sensors in loop. q & ω are quaternions and angular rates sensed and this measurement is compared against the reference to generate a controller output (u). This control loop is complete with inclusion of plant dynamics of Satellite with inertia matrix I**

2.1.5 Data for training Machine Learning Models

In this section, we describe the methodology used to generate the data for training various machine learning models. To generate data we use a 3 degree of freedom satellite simulator with the above-described sensors and actuators (This simulator is already validated against actual satellite data). There are two different modes of operation in a satellite attitude control namely wheel mode and thruster mode based on actuator in use while in control. Simulations are done with different initial conditions and with different modes of operation for different time windows.

10000 simulation cases were generated using the simulation software. About 30% of the simulation cases were simulated with a failure. Failure simulations are done by failing either sensors (gyros) or actuators (wheels/thrusters). 30% still seems a huge failure number usually does not happen in practice but when the failure modes are divided among different actuators and sensors the number of failure cases per sensor/actuator comes down drastically, thus simulating a rare failure scenario. Training data is further subdivided into smaller windows of time series samples usually for 50 seconds duration. This 50 seconds time series is used either to learn dynamics models like LSTM or LNN, or features used in novelty/ fall detection algorithms.

2.2 Models in Navigation solution

In this section, models used (both forward dynamics and measurement model) to find navigation solution of the satellite are discussed in detail.

2.2.1 Dynamics Model

For a navigation system translation and rotational dynamics of the satellite form the forward model. Both these models are highly nonlinear with uncertainties in their respective parameters. Also, there is an extra state in the dynamics model, Sidereal Angle (sa), added to represent for earth's rotation. Since the magnetic field measurements are dependent on the current position of the satellite with respect to the earth fixed frame and since state-vector, orientation is estimated in inertial frame, there should be a transformation between inertial frame and earth fixed frame. This transformation is captured by sidereal angle rotation along the Z-axis (south pole to north pole) of inertial frame. The sidereal angle rotation matrix is given by

$$DCM_{Inertial}^{EarthFixed} = \begin{bmatrix} cos(sa), sin(sa), 0 \\ -sin(sa), cos(sa), 0 \\ 0, 0, 1 \end{bmatrix}$$

Where sa is the sidereal angle

Note inertial frame represented here is Earth centered inertial (ECI) frame also known as J2000 frame. With this representation, equations of forward model is given by

$$\dot{\vec{r}} = v \tag{2.7}$$

$$\dot{\vec{v}} = -\frac{\mu}{||\vec{r}||_2^2}\hat{r} \tag{2.8}$$

$$\dot{sa} = \omega_{er} \tag{2.9}$$

$$\dot{q} = 0 \tag{2.10}$$

Where
$\vec{r}$ is the position vector in ECI
$\vec{v}$ is the velocity vector in ECI
$\vec{sa}$ is the sidereal angle
$q = (q_1, q_2, q_3, q_4)$ is the quaternion representation of satellite in inertial orientation
μ is the earth's gravity constant
ω_{er} is the earth's rotation rate (i.e. 360^o in 24 hrs)

Note there is no body rate representation in the forward model. This is done because typically during the estimation process the body angular rates of the satellite are controlled to zeros degrees/second. If someone wants to estimate the satellite body rates along with quaternions, rotational dynamics of the satellite can also be included along with the actuator torque model in the forward model as presented in [36].

2.2.2 Measurement Model

The measurement model consists of the Geo-Magnetic Field model and Sun Ephemeris model. In the current paper both the models are considered as black-box models i.e the models are available without knowing the internals of the model implementation.

Geo Magnetic field is modelled based on IGRF model which takes state vector in inertial frame (ECI) and sidereal angle as inputs. Sidereal angle is used to transform inertial vector to earth fixed vector. The output of this model is the magnetic field vector in inertial frame. To simulate the magnetic field measurement by magnetometer we need to transform the magnetic field vector in inertial frame to the body frame. This transformation (direction cosine matrix (DCM)) is computed from the quaternions.

Sun sensors on the spacecraft measure the sun vector in body frame. Sun vector variation in inertial frame is seasonal and is modelled using the sun ephemeris model. Output of this model gives sun vector in inertial frame. Transformation similar to magnetic field vector has

to be done on sun vector to simulate the sun vector in body frame.

Equations of the measurement model are given by

$$\vec{B} = DCM(q) * IGRF2015(\vec{r}, \vec{v}, sa) \tag{2.11}$$

$$\vec{S} = DCM(q) * SunModel(time) \tag{2.12}$$

Where

$\vec{B}$ is the magnetic field vector in body frame

$\vec{S}$ is the sun vector in body frame

$DCM(q)$ is the direction cosine matrix from inertial frame to body frame given by

$DCM(q) =$

$$\begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2, & 2(q_1q_2 + q_3q_4), & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4), & q_2^2 - q_1^2 - q_3^2 + q_4^2, & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4), & 2(q_2q_3 - q_1q_4), & q_3^2 - q_1^2 - q_2^2 + q_4^2 \end{bmatrix}$$

$IGRF2015$ is the geo-magnetic field model valid from 2015

$SunModel$ is the sun ephemeris model

During eclipse only eq[2.12] is not available for measurement. These measurement sequence is depicted in Fig 2.3

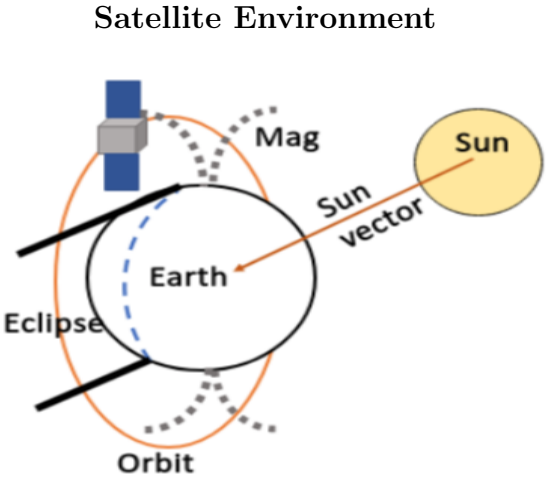

Figure 2.3: **Satellite Environment during an orbit depicting eclipse and non-eclipse durations**

2.2.3 Simulation Considerations for Navigation Solution

A high fidelity orbit model (HFM) constructed for HySIS, ISRO Satellite is available for simulation. This HFM is constructed between state vector and time, from ranging data using

regression algorithm using Fourier power series basis. HFM is considered to be a black box function in the current paper and is used to simulate sensor measurements. The output of HFM, position vector, is used as input to IGRF Model to simulate the magnetic field in the orbit. Quaternion propagation using Runge-Kutta method is used to simulate quaternions.

Dynamics model available for forward propagation used during the prediction step of the Kalman filter is assumed to be inaccurate since J2, J4, .. gravity terms are not considered for estimation in the forward model. This will lead to the errors prediction step which has to be taken care of, for proper estimation. This is done by choosing/estimating the right Process co-variance matrix during the estimation process.

The measurement model is assumed to be accurately described without any errors in this paper. In practice, the magnetometer/sun-sensor measurement is subjected to errors like scale-factor, misalignment and biases. These components can be added as extra states in the dynamics model for estimation if the user wanted to extend this algorithm for the estimation of these parameters.

Chapter 3

Methodology: FDI

3.1 FDI-Satellite Control System-(Model Based)

FDI Satellite control system can be divide into two major sub section namely

1. Sensor FDI
2. Actuator FDI

Sensor FDI uses the GOS algorithm which may not work for an actuator FDI. We have developed a new method to identify actuator FDI. In the following subsections first, we start with sensor FDI using the GOS scheme followed by the actuator FDI.

3.1.1 Sensor FDI

A great deal of algorithms are covered in sensor FDI in survey papers [11], [10] and [22] and book [14]. But almost all the applications deal primarily with sensor fault detection and very few algorithms deal with actuator fault isolation. Here we are presenting the application Generalised Observer scheme applied on the satellite system.

Generalised Observer Scheme (GOS)

The Generalised Observer Scheme is one of the famous methods to detect and isolate and do an error-free control under faulty measurements [13]. In GOS an estimator is dedicated to identifying a sensor fault i.e each sensor has its corresponding paired observer. This observer takes in the measurements from all the sensors except its paired sensor and gives an estimate of the measurement of its paired sensor. This Estimated measurement is then compared against

the actual sensor to conclude the health of this sensor. Thus a GOS is suitable for detecting a single fault at a time. Operational flow chart of GOS scheme [13] is given in Fig. 3.1

GOS Operational architecture [13]

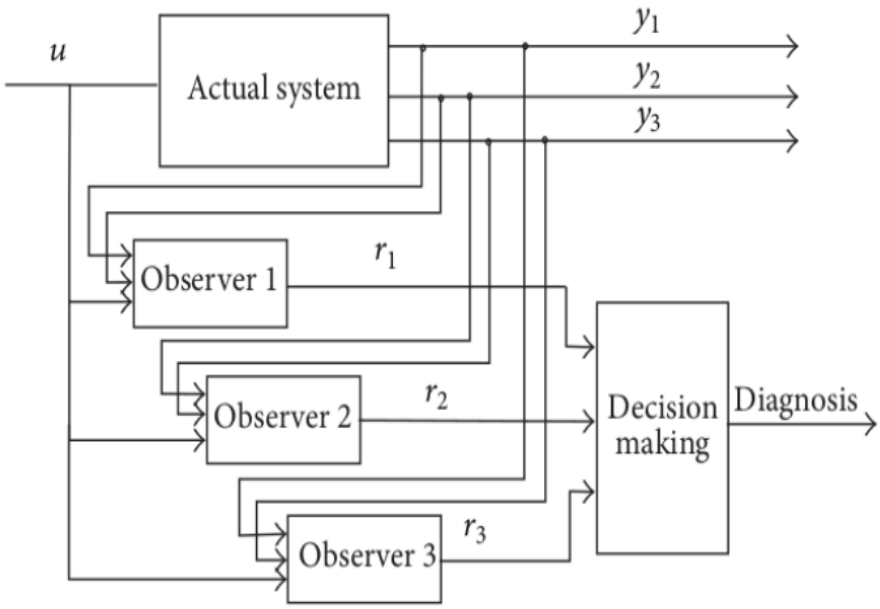

Figure 3.1: **Illustration of GOS for sensor fault detection**

The following operations are done by every observer corresponding to a given sensor

1. ith observer takes in all the measurements except the ith measurement and estimates the state

2. Estimates the measurement of the paired sensor and a residue is computed between actual sensor measurement and estimated measurement

3. Returns a large error when one of the used measurements contains a corrupted measurement in all the other observers except for the paired observer. Measurement error will be less in the paired observer as this observer does not take in corrupted sensor measurement during measurement generation as shown in Fig. 3.1

Note that for this process to be successful, observability condition should hold for the measurement of a given sensor from the remaining sensors. In case of a sensor failure, the estimated state of the observer corresponding to the failed sensor is used as feedback information to the control system as this observer estimates the state without the corrupted sensor.

The above mentioned GOS is used to identify gyro fault in a satellite system. Observability condition holds as star-sensor output is equivalent to the integrated output of gyro rates. Note observability does not hold when an SS fails as gyro rates alone cannot determine the inertial orientation (Initial condition of integration is missing). We have successfully implemented the GOS scheme on AOCS to identify the notorious gyro fault and simulation results are presented in the results section.

But GOS scheme as such can only identify sensor fault detection but not actuator faults as both the model and the actual system take the same control input (a digital signal) without the actual knowledge of actuator performance. Note that sensor measurements among various sensors would remain consistent even though there are actuator faults. Hence a new scheme is developed to identify actuator faults from measurement data using model augmentation techniques.

3.1.2 Actuator FDI

As mentioned above actuator FDI is not possible using the GOS algorithm. To understand the methodology of identifying the actuator fault we will take small detour about the implementation.

Consider a Lorenz oscillator in eq[3.1] to eq[3.3]

$$\frac{dx}{dt} = \sigma(y - x) \tag{3.1}$$

$$\frac{dy}{dt} = x(\rho - z) - y \tag{3.2}$$

$$\frac{dz}{dt} = xy - \beta z \tag{3.3}$$

We know that Lorenz oscillator is very sensitive to the initial conditions. Let the Lorenz oscillator be dynamics of our example system and let measurement is given by eq[3.4]

$$Z = C * X \tag{3.4}$$

where

Z is the measurement vector of size (3,1)

C is a invert-able matrix of size (3,3)

X is the Lorenz oscillator state $[x, y, z]$

Now lets consider that the 'Known model' available for forward propagation be 'Lorenz oscillator without the non-linear terms'. i.e eq[3.5] to eq[3.7]

$$\frac{dx}{dt} = \sigma(y - x) \tag{3.5}$$
$$\frac{dy}{dt} = \rho x - y \tag{3.6}$$
$$\frac{dz}{dt} = -\beta z \tag{3.7}$$

and let all the uncertainties in forward model, be absorbed by the process co-variance matrix Q. Thus our final system model for estimation using simple Euler integration would look like

$$X_{t+1} = (I + A^{'} * dt)X_t \tag{3.8}$$
$$Z = CX_t \tag{3.9}$$

Where
dt is the step size
$A^{'} = \begin{bmatrix} -\sigma, \sigma, 0 \\ \rho, -1, 0 \\ 0, 0, -\beta \end{bmatrix}$
$X = [x, y, z]$
Now while implementing Kalman filter we do the following steps
Also let $A = I + A^{'} * dt$

- **Predict Step**

$$X^{-}_{e,t+1} = AX_{e,t} \tag{3.10}$$
$$P^{-}_{t+1} = A * P_t * A^T + Q \tag{3.11}$$

- **Update Step**

$$v = Z - CX^{-}_{e,t+1} \tag{3.12}$$

$$S = C * P_{t+1}^{-} * C^T + R \tag{3.13}$$

$$K = P_{t+1}^{-} * C^T * S^{-1} \tag{3.14}$$

$$x_{e,t+1} = X_{e,t+1}^{-} + Kv \tag{3.15}$$

$$P_{t+1} = P_{t+1}^{-} - KSK^T \tag{3.16}$$

$$w = x_{e,t+1} - (I + A * dt) * x_{e,t} \tag{3.17}$$

After doing Kalman filter update if we take the difference between new estimate and old prediction, i.e computing w in eq[3.17] at each instant, it should resemble the uncertainty matrix Q. But when the experiment is done with eq[3.8],eq[3.9] on the original dynamics eq[3.1] to eq[3.3], the difference between two subsequent estimates look as shown in Fig. 3.2

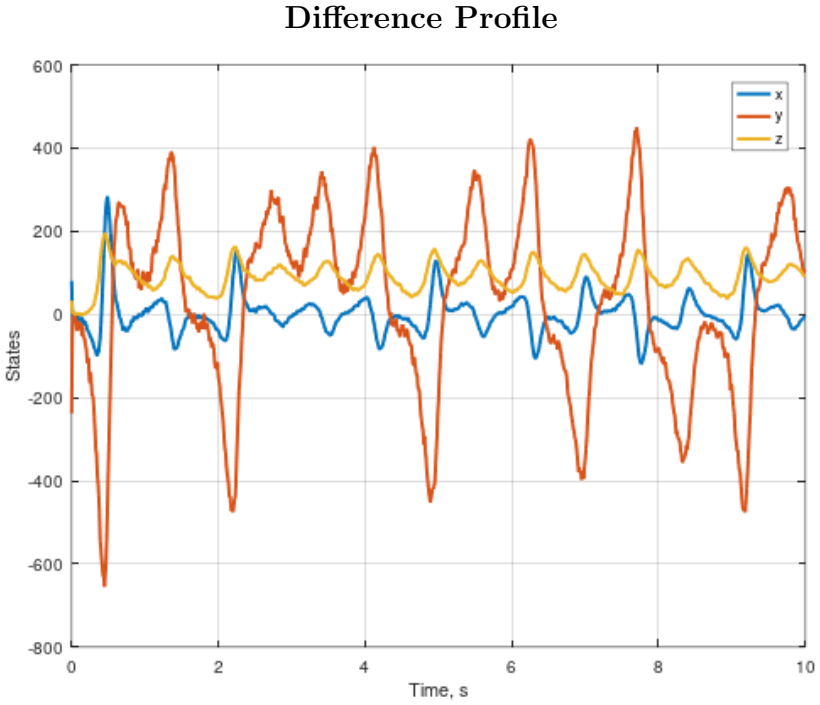

Figure 3.2: **Difference between estimated state and dynamics predicted state for Linearized Lorenz oscillator**

Clearly, this is not a Gaussian distributed random sequence as defined by Q with zero mean and specified variance. Our idea here is to use this data to update the forward dynamics model for better model identification.

Our aim here is to make sure that the difference profile as shown in Fig. 3.2 to be as close to the Gaussian signal as possible. To do so we define an augmentation to the original forward

dynamics as a function of the estimated state itself represented in the eq[3.19]

$$w = x_{e,t+1} - (I + A * dt) * x_{e,t} \tag{3.18}$$

$$\implies w = coeff * basis \tag{3.19}$$

where
$basis = [1, x, y, z, x^2, y^2, z^2, xy, yz, zx]$
$coeff$ matrix has to be learnt is of size (3×10(i.e, length of basis))

With these updates the final Kalman filter forward propagation is represented by eq[3.20]

$$X^-_{e,t+1} = AX_{e,t} + coeff * basis \tag{3.20}$$

$$P^-_{t+1} = (A + dA) * P_t * (A + dA)^T + Q \tag{3.21}$$

where $dA = coeff * \frac{d(basis)}{dX}$
Also, $coeff$ is learnt by applying the Recursive Least Square (RLS) algorithm between the output w from eq[3.17] and input from eq[3.19]. The term $coeff * basis$ actually represents the forward dynamics function augmentation to learn the unmodelled dynamics of the forward model. With these updates, the final RMSE can be seen in Fig. 3.3

Thus we could successfully reduce the RMSE of the estimation by the augmented model to the forward dynamics. But for our requirement, the actual model of the dynamics is well known for a Satellite attitude control system. The idea is to identify actuator fault by redesigning the problem as a parameter variation problem i.e when ever there is a failure in the actuator the forward dynamics of the model changes as actuator failure can be interpreted as a change in control matrix B in eq[3.22]

$$X_{t+1} = AX_t + Bu_t + coeff * basis \tag{3.22}$$

$$Y_t = Cx_t \tag{3.23}$$

Here we are assuming that while determining the coefficients, satellite system has never entered any failure mode. Once coefficients are fully determined, there will be a change in the magnitude of the coefficients (learnt online using RLS algorithm) in case of an actuator failure thus helping us identify the failure immediately.

With the methodology mentioned above, we could successfully identify actuator failure in

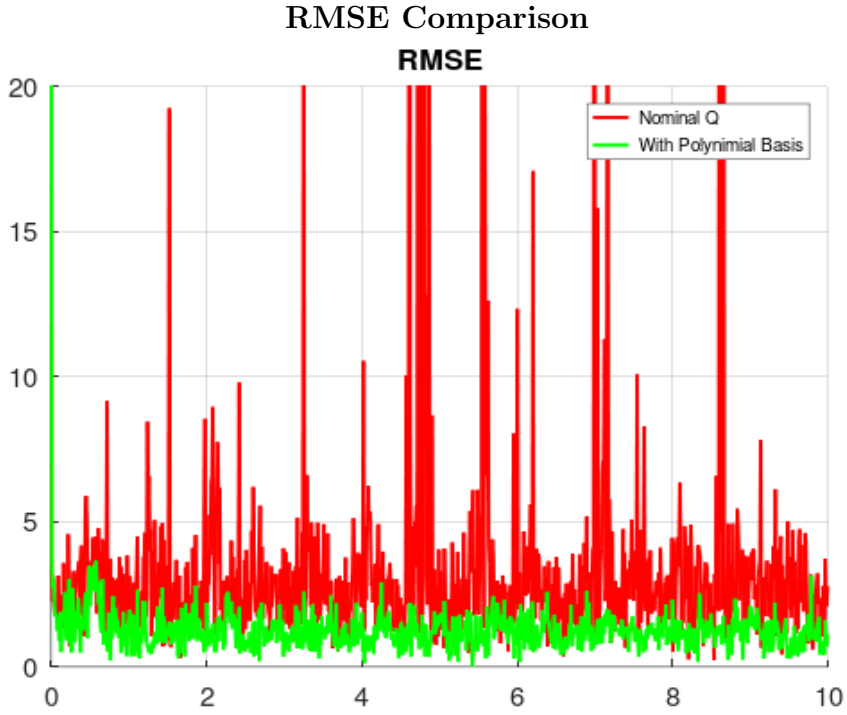

Figure 3.3: **Realised RMSE by different schemes.A clear improvement in RMSE can be seen with the augmented basis method**

real-time by observing coefficients of the augmented function. We could even isolate the failure by observing the coefficient of the basis vector. Results of the actuator failure will be discussed in the results section.

3.2 FDI using Machine Learning Techniques - A Preamble

Until now we have seen methods to identify faults using already known dynamics models. These methods are useful only when the forward models are known. But this is not the case in many other subsystems in satellite. In such cases it is only natural to either learn the model that is generating the time series or to learn the features from the time series data, to identify the faults from data. Faults can be classified as unknown patterns observed in time series and we aim to identify the unknown patterns on real-time telemetry data. When using these methods unlike in model-based methods we are only going to identify the fault but not the cause of faults. Since data from other subsystems are not available, for testing these algorithms we use available AOCS data for fault detection.

In the subsequent sections, we describe various methodologies used in FDI. Initially, we

start with model learning algorithms like LSTM/LNN which learn the underlying time series model. Any fault can be identified by computing the deviation of the prediction generated by this model against the actual measured data.

The second type of algorithms typically includes identifying features from the time series data and using this feature to identify faults. In these methods initially, we start with a supervised learning set-up where we can establish the benchmark accuracy to identify faults. Note that a supervised learning setup is not useful in practice especially in satellite systems since there is no method to mark data that is faulty or not without the help of a subsystem engineer. Once benchmarking for accuracy has been done using supervised learning algorithms we can move on to unsupervised learning algorithms to identify faults. In the unsupervised learning setup, faults are identified as anomalies and will be further discussed in the subsequent sections.

3.3 FDI Using Learnt Models

In this section, the aim is to identify the model that generates the time series using LSTM or LNN. In this method we try to learn the model from the available time-series data. This section is further divided into two parts where part 1 describes model learning using the LSTM network and part 2 describe the same using LNN.

3.3.1 LSTM Network

In an LSTM network, time-series data of nominal simulations are used to train the network. Idea is that once the network is trained on nominal data, the network now will be representative of the dynamics of the system. Once the dynamics of the system are known we can use this model to predict future measurements. We identify a fault by measuring the difference between the predicted measurement against the actual observation. If this difference is greater than a prefixed threshold a fault is identified. We can think of this difference as the usual anomaly score.

While different kinds of network models can be used to learn the dynamics from the data the following model shown in Fig 3.4 seems to outperform most of the models (many to one, many to many, we here use one to many) that were tried. In this model input to hidden data is also learnt from the observed data. Also, only one measurement is used to generate all the necessary States for the subsequent time as shown in Fig 3.4.

3.3.2 LNN Network

Lagrange Neural network (LNN) as mentioned in [8] is a physics-based neural network where the dynamics of the system is learnt by learning the Lagrangian. Lagrangian is the difference be-

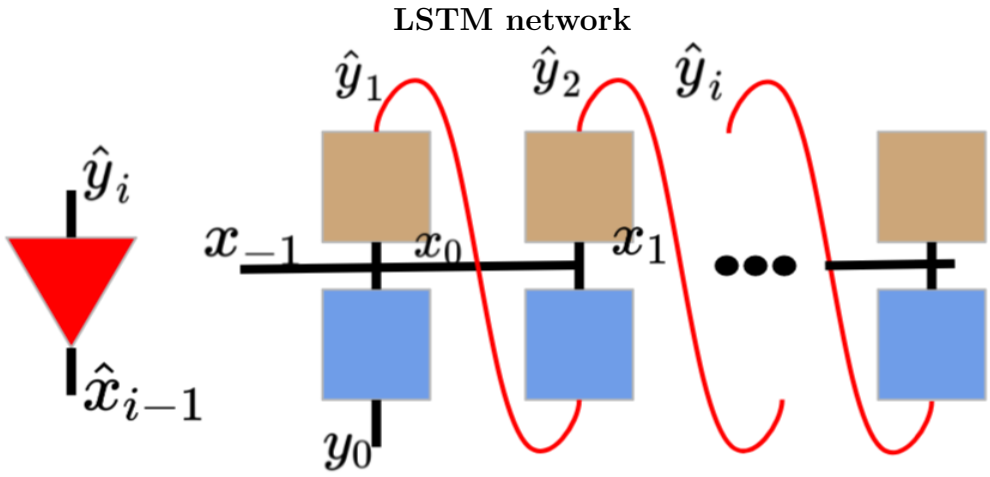

Figure 3.4: **Dynamics Representation using LSTM network (Square Blocks). Here y_i represent the measurement at instant i and x_i(hidden variables) are the states of the dynamics. Initialization of a states (x_{-1})is done by another neural network (in red triangle) which is learnt during the learning process of LSTM. Error is minimized between predicted $\hat{y}_i$ and the actual measurements y_i while learning**

tween kinetic energy and potential energy for a mechanical system. This neural network model takes in angle and angular rate information to predict the Lagrangian of the dynamical system while minimising the loss between the angular acceleration computed from the Lagrangian and the actual observer angular acceleration. In the paper, [8] the author describes the method only to learn the actual non-torquing dynamics only. But in the current method, we extended this to talking dynamics of the satellite. The final equation of the motion with external torque is given by eq[3.24]

$$\ddot{\theta} = (\frac{\partial^2 L}{\partial \dot{\theta}^2})^{-1}(\frac{\partial L}{\partial \theta} - \dot{\theta}\frac{\partial^2 L}{\partial \theta \partial \dot{\theta}} + \tau) \tag{3.24}$$

where
theta is the euler angle vector to represent a satellite.
L is the learnt Lagrangian
τ is the external torque

The depiction of the network used in LNN is shown in Fig 3.5

While running diagnosis for fault detection, LNN would predict a different value for La-

LNN network

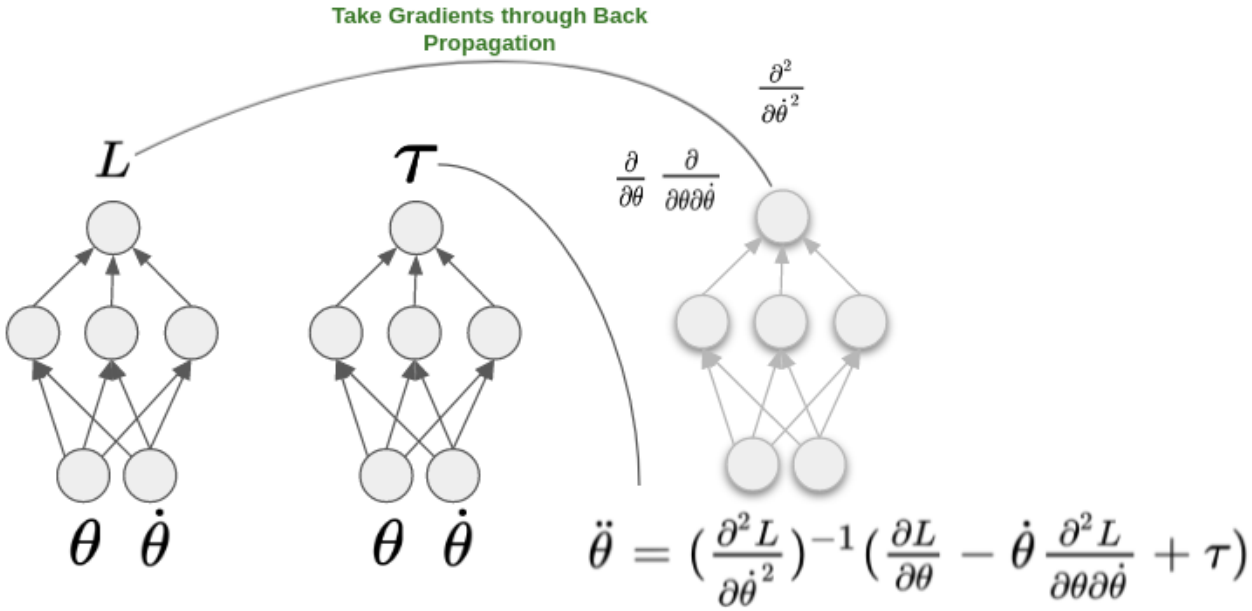

Figure 3.5: **Dynamics Representation using LNN network. Back propagation is used to compute the values of $\frac{\partial^2 L}{\partial \dot{\theta}^2}, \frac{\partial L}{\partial \theta}, \frac{\partial^2 L}{\partial \theta \partial \dot{\theta}}$. Error is minimised between predicted $\ddot{\theta}_{LNN}$ and actual $\ddot{\theta}$ computed using difference scheme.**

grangian and torque when compared against nominal performance data. This difference in time series between LNN and actual measurement can be used to identify faults in the system. Note that in the current system we do not require any torque measurement as it is learnt online. Results of LNN are discussed in the next chapter.

3.4 FDI using time series features

This is the most common method used in identifying faults for any kind of data sets. The idea behind this method is to extract representative features that are much smaller in dimension when compared to the original data and use features for FDI. The basic flow diagram of these methods first includes computing the representative features of the data and then applying various clustering algorithms/anomaly detection algorithms to identify faults. In this section, we first discuss the feature generation techniques followed by various clustering-based fault detection algorithms.

3.4.1 Feature Selection

This section discusses about various feature selection algorithms for a time series.

3.4.1.1 Manual Feature Extractions

As mentioned in [38] various features can be extracted from the data by computing the list of statistics mentioned in Table 3.1. In our time series data, there are 19 columns of data (4 quaternions, 3 body angular rates, 4 reaction wheel speeds, and 8 thrusters). All the feature statistics are evaluated for each column in the time sequence and then concatenated to form a complete feature vector. These feature vectors will be further used to identify faults.

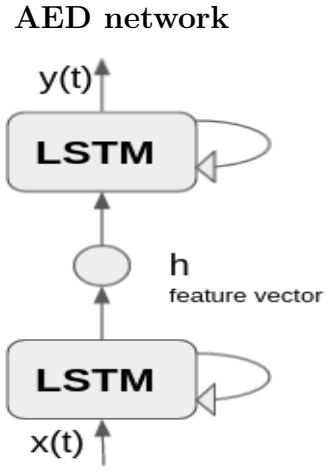

Figure 3.6: **Feature Selection (h) from AED network**

3.4.1.2 Auto Encoder Decoder (AED) Feature Extractions

Autoencoder Decoder network can also be used to generate relevant features for anomaly detection. In the current setup, an LSTM based encoder Decoder network is trained and the output of the encoder is taken as a feature vector for a given time series. The depiction of the LSTM encoder-decoder network is

Feature List		
Sl.No	Feature name	Formula
1.	Mean	M = $\frac{1}{n}\Sigma_t x(t)$
2.	Absolute Mean	AM = $\frac{1}{n}\Sigma_t \lvert x(t)\rvert$
3.	Root Mean Square	RMS = $\sqrt{\frac{1}{n}\Sigma_t x^2(t)}$
4.	Average Power	AVP = $\frac{1}{n}\Sigma_t x^2(t)$
5.	Root Amplitude	ROA = $(\frac{1}{n}\Sigma_t \sqrt{\lvert x(t)\rvert})^2$
6.	Peak	PE = $max(\lvert x(t)\rvert)$
7.	Peak-to-Peak	PEP = $max(x(t)) - min(x(y))$
8.	Variance	VA = $\frac{1}{n-1}\Sigma_t (x(t) - M)^2$
9.	Standard Deviation	STD = $\sqrt{VA}$
10.	Skewness	SK = $\frac{1}{n-1}\Sigma_t (x(t) - M)^3$
11.	Kurtosis	KU = $\frac{1}{n-1}\Sigma_t (x(t) - M)^4$
12.	Waveform Index	WAI = RMS/AM
13.	Crest Index	CRI = PE/RMS
14.	Impulse Index	IMI = PE/AM
15.	Margin Index	MAI = PE/ROA
16.	Skewness Index	SKI= SK/STD^3
17.	Kutosis Index	KUI = KU/STD^4
18.	Frequency Center	FC = $\frac{\Sigma_t x(t)\dot{x}t}{2\pi\Sigma_t x^2(t)}$
19.	Mean Squared Frequency	MSF = $\frac{\Sigma_t \dot{x}^2 t}{4\pi^2\Sigma_t x^2(t)}$
20.	Root Mean Square Freq	RMSF = $\sqrt{MSF}$
21.	Variance Freq	VF = $MSF = FC^2$
22.	Root Variance Freq	RVF = $\sqrt{VF}$

Table 3.1: Manual Features for each data column. Here $x(t)$ is a time sequence

given in Fig 3.6. Several clustering-based algorithms are applied to these features to identify faults as described in the next subsection. Features selected by this method may be entangled for various modes of operation. K-Means loss is applied on the feature vector to disentangle the hidden vectors belonging to different classes. Disentanglement is useful to identify clusters of operation and thus we can use the IGMM algorithm to learn these clusters.

Once IGMM learns all the possible clusters any new sample which is a fault would belong to an existing cluster and outliers belong to a new class that is not present in the existing cluster list. Several deep embedding techniques are available but in the interest of time, only K-Means based clustering is implemented.

Kmeans loss on clusters

To train AED we have reconstruction loss given by eq[3.25]

$$l_{recon} = \Sigma_i (x(i) - y(i))^2 \tag{3.25}$$

where

l_{rcon} is the reconstruction loss

$x(i)$ is input data, $y(i)$ is the output data

As mentioned in [21] K-Means loss can be written as eq[3.26]

$$l_{K-Means} = Trace(H^T H) - Trace(F^T H^T H F), s.t F^T F = I \tag{3.26}$$

Where

H is the data matrix consisting of collection of hidden vectors

F is the cluster selection unitary matrix

If we apply K-Means loss on the hidden data to be minimized along with reconstruction loss, features formed will be well clustered after training. Training procedure alternating the update of network weights and F matrix as mentioned in [21]. Hence total loss function for minimization is given by

$$loss = l_{recon} + l_{K-Means} \tag{3.27}$$

The author also suggests the use of classification loss on fake samples but observation found out this has little effect on learnt features. The above class of algorithms where features/hidden vector are used in minimizing clustering loss are part of a larger class of algorithms called Deep Embedding Clustering (DEC) algorithms. In the interest of time, other DEC algorithms are not studied for this project but these algorithms are very good methods to get proper time-series features.

3.4.2 Supervised Algorithms

Given time series data, it is divided into batches of 50 seconds and the batches are marked pass or fail based on kind of simulation i.e. nominal or failure mode respectively. Features are extracted using manual features from these time windows for all 19 measurements and are concatenated to form a complete feature vector. Extracted features along with the pass/fail markings, forms the data for the supervised learning algorithms.

Several Algorithms are studied especially to establish the bench-mark accuracy for unsupervised learning. Algorithms studied include k-nearest neighbours, Decision trees, Random Forest, Neural Network, AdaBoost, Quadratic discriminant Analysis. The accuracy and ROC curve of these algorithms are presented in the results section.

3.4.3 Unsupervised Algorithms: Outlier/Novelty Detection Algorithms

In this section Unsupervised Anomaly detection algorithms are applied to the features generated by the time series. Anomaly detection algorithms for fault identification include One-Class Support Vector Machine (OC-SVM), Isolation Forest (IF), Clustering Algorithms like K-Means, IGMM were adjusted to fit in anomaly detection algorithms, Self Organising Maps (SOM). The subsequent section describes the methodology of usage of these algorithms. Density-based algorithms like DB Scan, Local Outlier Factor(LOF) are avoided in this project as hyperparameter tuning has become very hard with these algorithms.

3.4.3.1 One Class Support Vector Machine (OC-SVM)

OC-SVM generally identifies anomalies by first training on nominal data and learns a non-linear boundary around the nominal data. Any data point outside the boundary is identified as an anomaly. The formulation of OC-SVM is as shown in [32] and [34]. Features generated from the methods mentioned in section 3.4.1 are used in OC-SVM to identify anomalies.

3.4.3.2 Isolation Forest (IF)

As mentioned in [19] IF identifies anomalies in data by counting the number of steps it takes to isolate a data point using a randomly picked features column. The idea here is any anomalous point takes fewer steps to isolate itself while an inlier would take a lot more steps to isolate itself from the given data set. Similar to OC-SVM, IF first learn on nominal data set then use this knowledge to identify the outliers. The data set used for IF is the same data set used in OC-SVM. The illustration of IF is shown in Fig 3.7. In the Fig 3.7 x_i is an inlier and x_o is an outlier. It can be seen that the number of steps taken to isolate x_o is much lesser compared to steps taken to isolate x_i.

3.4.3.3 K-Means(KM)

K-Means Algorithms is a clustering algorithm used to identify cluster centres given the predefined number of cluster in the data. Clustering algorithms can be used in anomaly detection as mentioned in [25] and [5]. To identify anomalies using KM we first train the K-Means algorithm

Isolation Forest [19]

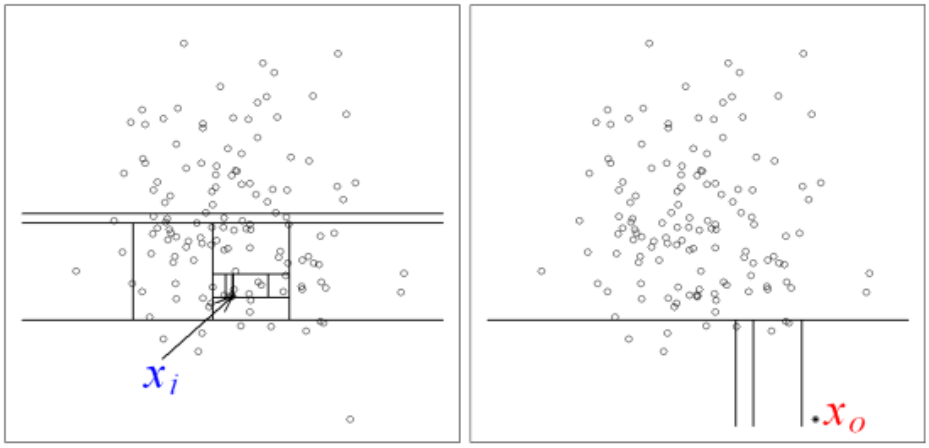

Figure 3.7: **Illustration of Isolation Forest Anomaly detection. x_i takes more steps to isolate itself as it is internal point and x_o takes less number of steps to isolate itself as it is an outlier**

on nominal data points (extracted features) to identify all possible clusters. Now based on the distance of a new point from the cluster centres we can determine the anomaly.

Implementation of KM Outlier detection is done using the distance measure. We know that a data point belongs to a particular class if the distance to that class is minimum compared to all other classes. To compute an outlier we compute the minimum distance among all the distances that data point makes to all the clusters. The minimum distance is greater than a thresh hold (average minimum distance) anomaly is identified. Results of KM anomaly detection is shown in Fig 3.8. We can see 3 different clusters in Fig 3.8. All the anomaly points are detected as black stars.

3.4.3.4 Infinite Gaussian Mixture Model (IGMM)

IGMM is a natural extension to the KMeans anomaly detection. KM algorithm inherently required the number of clusters required to identify the clusters and anomaly score is computed based on the distance from the cluster. These properties can be naturally designed using IGMM algorithm as mentioned in [29],[30]. IGMM is implemented by parameterizing the means and covariance of GMM with a Dirichlet distribution over the existing clusters. The best way to understand IGMM is through the Chinese Restaurant Process (CRP). An illustration of CRP is given in Fig 3.9.

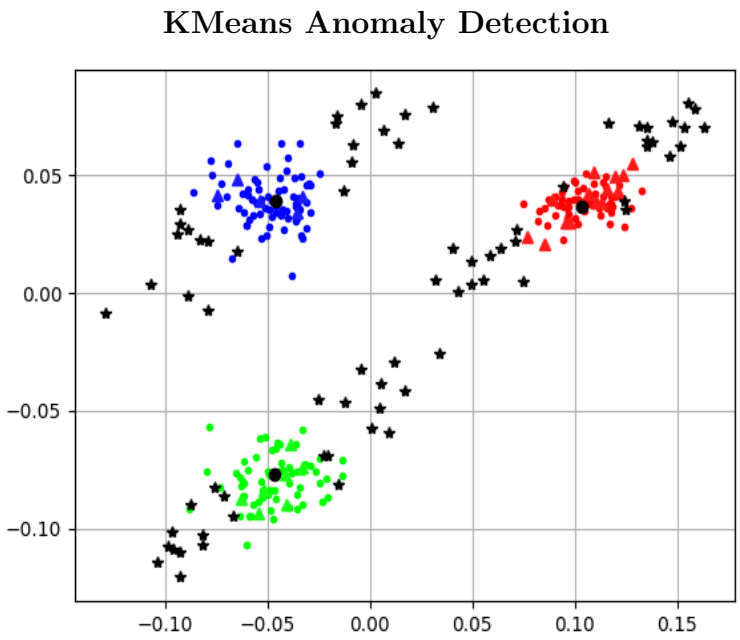

Figure 3.8: **Illustration of K-Means Anomaly detection**

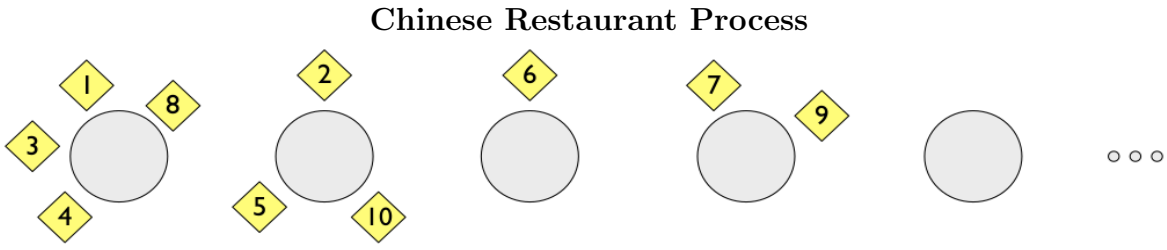

Figure 3.9: **Illustration of CRP**

In a CRP, a restaurant has n number of tables. Each table has people with similar taste over the kind of food on the table. When a new person enters the restaurant he has to decide to join the existing table based on the similarity of the taste or to join a new table of his own. If the person started his table, effectively he joined his own cluster. This idea can be extended to identifying new clusters in the same as the new person identifies a new cluster or belonging to an existing cluster through similarity measure. At each iteration of IGMM i^{th} data point can identify itself to a cluster based on sampling from probability distribution in eq[3.28]. If the data point has a high probability for a new cluster then it's identified as an anomaly else otherwise. Fig 3.10 gives an result of identifying clusters using IGMM. All the sky blue points in the figure are identified as anomalies.

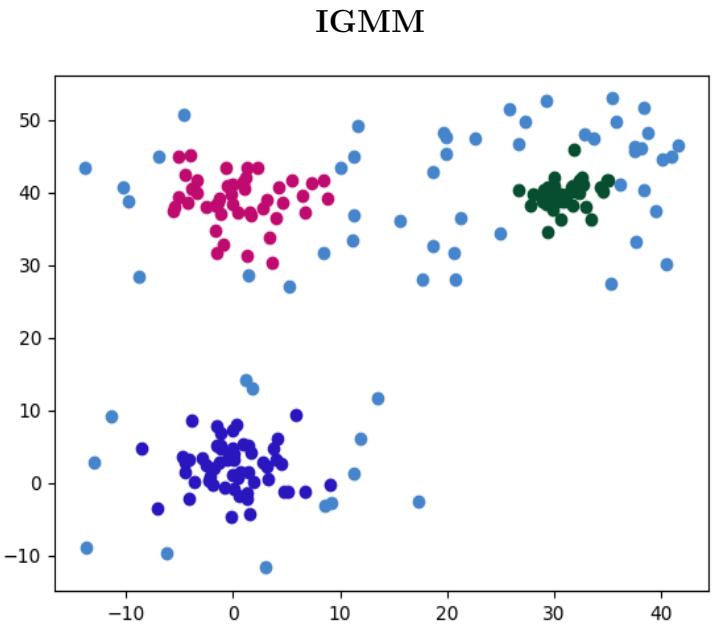

Figure 3.10: **Anomaly detection using IGMM. Red, Blue and Green points are learnt marked with their clusters (learnt using IGMM). Sky blue points are identified as possible points belonging to a new cluster anf thus identified as anomalies**

$$P(c_i = k|C_{-i}, \alpha) = \begin{cases} \frac{n_k}{n-1+\alpha} N(x_i, \mu_k, \Sigma_k) & k \in K \\ \frac{\alpha}{n-1+\alpha} & k \notin K \end{cases} \tag{3.28}$$

Where

x_i is the data point

$N()$ is the normal density function

k is cluster name
K all possible clusters at that point of time
$\alpha = 1/K$ is the Dirichlet process parameter
μ_k is the mean of cluster k
Σ_k is the co-variance matrix of cluster k

Thus using IGMM we can identify **new clusters**. The application of IGMM in FDI is quite straightforward. Any outlier will not belong to any of the existing clusters and hence the probability of forming a new cluster will be high. This probability measure can be used to identify faulty data points. Initial IGMM is trained for all the nominal data points so that all the clusters are learnt. Once all the clusters are learnt any new point will be evaluated for its cluster using eq[3.28]

3.4.3.5 Self Organizing Maps (SOMs)

SOMs map multidimensional data onto lower-dimensional sub spaces where geometric relationships between points indicate their similarity. The reduction in dimensionality that SOMs provide allows us to visualize and interpret data. SOMs generate subspaces with an unsupervised learning neural network trained with a competitive learning algorithm. Neuron weights are adjusted based on their proximity to "winning" neurons (i.e. neurons that most closely resemble a sample input). Training over several iterations of input data sets results in similar neurons grouping together and vice versa.

Self Organizing Map

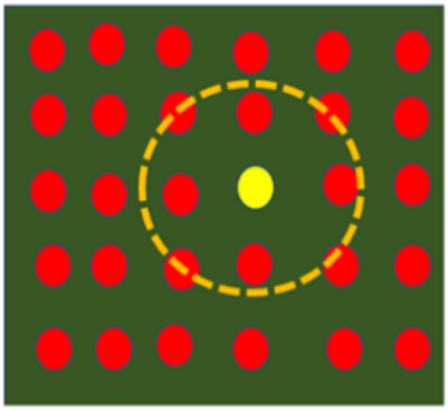

Figure 3.11: **SOM Grid structure**

In a SOM a 2D grid of size n by m is created and at each grid point, a weight vector of the same dimension of the input vector is initialized randomly. The best matching unit (BMU) is computed for a data point and the weights around the BMU are also updated while learning as shown in Fig 3.11. The learning algorithm looks like in eq[3.29] to eq[3.33]

$$sim = ||x(t) - w_{BMU}(t)|| \tag{3.29}$$

$$W(t+1) = W(t) + \Theta(t)L(t)(x(t) - W(t)), \forall W \in neighbourhood(W) \tag{3.30}$$

$$L(t) = L_0 e^{-\frac{t}{\lambda}} \tag{3.31}$$

$$\Theta(t) = e^{-\frac{sim^2}{2\sigma^2(t)}} \tag{3.32}$$

$$\sigma(t) = \sigma_0 e^{-\frac{t}{\lambda}} \tag{3.33}$$

Here $\Theta(t)$ is a neighbourhood weight, $L(t)$ is the learning rate decay. λ, L_0, σ_0 are some hyper-parameters. $W(t)$ is the weight values at time t at each node and note that the weight update decreases as the distance from the node increases with the help of influence parameter $\Theta(t)$.

While applying SOMs to FDI algorithm we first train the SOM with nominal data. When anomalous data is tested for BMU we record the distance of the BMU for the anomalous data point. The distance measure of anomalous point will be quite high compared to no anomalous data and thus can be used in fault detection. In the current project 95 % confidence interval over trained/nominal data is used as a threshold. Fig 3.12 gives the training and anomaly detection results for the SOM algorithm.

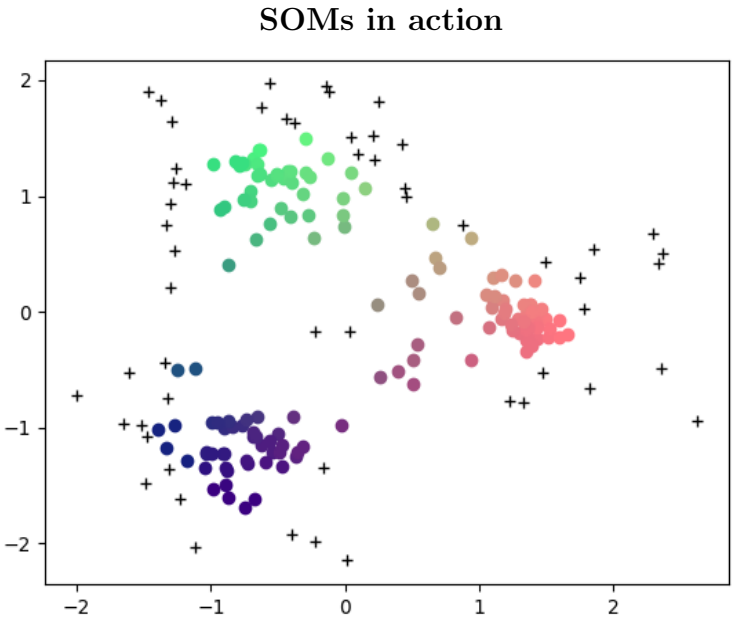

Figure 3.12: **Anomaly detection using SOMS where similar cluster points are identified with similar color (One can see a color gradient depicting the measure of similarity). Out liers are identified based on similarity measure (Euclidean Distance in this case)with the existing nodes in the SOM grid.**

Chapter 4

Methodology: Navigation Solution

Extended Kalman Filter (EKF) algorithm can be used for the estimation of states for nonlinear forward model. A complete description of the EKF algorithm is described in [16] and [31].A practical approach to implement the Kalman filter for satellite navigation is described [23]. Here in the current chapter we try to apply the EKF on satellite state propagation (Navigation) with our own modifications on the EKF algorithm to improve the navigation accuracy.

Given our dynamics model, the formulation of EKF requires the computation of partial derivatives of forward propagation model (dynamics model) and measurement model with respect to state vector. As mentioned earlier in chapter 2 IGRF2015 geomagnetic field measurement model and Sun Ephemeris model are considered to be black-box functions and hence computation of partial derivatives with respect to state-vector is implemented as a central difference scheme on dynamics/measurement model. Euler integration is assumed for forward propagation. Dynamics model under Euler integration assumption is given by

4.1 Dynamics Model

$$\vec{r}_{t+1} = \vec{r}_t + \vec{v}_t \delta t \tag{4.1}$$

$$\vec{v}_{t+1} = \vec{v}_t - \frac{\mu}{||\vec{r}_t||_2^2} \hat{r}_t \delta t \tag{4.2}$$

$$sa_{t+1} = sa_t + \omega_{er} \delta t \tag{4.3}$$

$$q_{t+1} = q_t \tag{4.4}$$

where

$\vec{r}_t$ is the position vector at time t

$\vec{v}_t$ is the velocity vector at time t

sa_t is the Sidereal Angle at time t
q_t is the quaternion at time t
ω_{er} is the earths angular rate
$deltat$ is the step size

Note the Dynamics Model does not change whether the satellite is in eclipse/non-eclipse zone of the orbit. Using eq[4.1] to eq[4.4] the final equations of the EKF are given by

4.2 Prediction Step

$$\vec{x}_{t+1}^{-} = f(\vec{x}_t) \tag{4.5}$$
$$P_{t+1}^{-} = Q_t + F_x(\vec{x}_t) P_t F_x(\vec{x}_t)' \tag{4.6}$$

where
$\vec{x} = \begin{bmatrix} \vec{r}_t, & \vec{v}_t, & sa_t, & q_t \end{bmatrix}'$
f is the dynamics model represented in eq[4.1] to eq[4.4]
F_x is the partial derivative of the dynamics model f w.r.t state vector evaluated at $\vec{x}_t$
P_t is the estimated state co-variance matrix at time t
Q_t is the Process co-variance matrix estimated at time t

4.3 Estimation of process co-variance matrix

In the prediction step, only part that is not known to us the process covariance matrix Q. In this section we describe various methods to learn Q matrix online.

Selection of process co-variance matrix Q is done using an iterative method based on [4]. Innovation Based Adaptive estimation used to estimate the Q matrix. In this method innovation during the estimation process is used to update the process co-variance matrix. Process co-variance matrix is estimated as co-variance of the difference ${x_t}^+$ and ${x_{t-1}}^+$ where ${x_t}^+$ is the estimate at time t. For some forward model ϕ and state vector x_t such that $x_{t+1} = \phi(x_t) + w_t, E(w_t w_t^T) = Q$, estimated process co-variance matrix(Q) can be written as

$$\hat{w}_{t-1} = {x_t}^+ - \phi({x_{t-1}}^+) \tag{4.7}$$
$$= {x_t}^+ - {\hat{x}_t}^- \tag{4.8}$$

$$= K_t(z_t - h(\hat{x_t}^-)) \tag{4.9}$$
$$= K_t\vec{V_t} \tag{4.10}$$
$$\hat{Q}_{t-1} = E(\hat{w}_{t-1}\hat{w}'_{t-1}) \tag{4.11}$$
$$= K_t E(\vec{V_t}\vec{V_t}')K'_t \approx K_t\vec{V_t}\vec{V_t}'K'_t \tag{4.12}$$

where
$\hat{x_t}^-$ is the prediction at time t from estimate at time t-1
z_t is the measurement st time t
h is the measurement model
$\vec{V_t}$ is the innovation/measurement residual st time t
$E(.)$ is the expectation operation

4.3.1 Q-learning

In [4] author describes the method to update both measurement covariance matrix (R) and process covariance matrix (Q). We only update the Q matrix from eq[4.12] assuming the R matrix is accurately computed while ground calibration. Instead of updating Q from eq(4.12) instantly we update the Q matrix using a forgetting factor α to average out Q over a period of time. The iterative equation to update Q matrix is given by

$$Q_{t+1} = \alpha Q_t + (1-\alpha)(K_{t+1}\vec{V}_{t+1}\vec{V}'_{t+1}K'_{t+1}) \tag{4.13}$$

where
Q Process covariance matrix
α factor for design, $0 < \alpha < 1$
K is Kalman gain
d is innovation i.e. (actual measurement - predicted measurement)

4.3.2 Moving Average

Q matrix can also be updated using the moving average method as shown in eq[4.14]

$$Q_{t+1} = \frac{1}{n}\Sigma_{i=n-t}^{t+1} K_i\vec{V}_i\vec{V}_i'K'_i \tag{4.14}$$

Note that in the above method there will be a $t/2$ seconds delay in estimating Q matrix as it takes in only the data from past data time steps. To avoid this we can use some time of estimation to have a regression fit between estimated states and the estimated Q matrix to

minimize the effect of the delay.

4.3.3 Gaussian Process Regression (GPR)

GPR is a regression method that used a Gaussian kernel to fit data. We used GPR to find a relation between the estimated state and the computed process co-variance matrix. To estimate the covariance matrix at a time step t we computed the covariance of data over window (length W) between $(t - \frac{W}{2}, t + \frac{W}{2})$. By doing so we can resolve the delay caused by the Q estimation process in eq[4.14]. But for this method to work, we need future values of Q which is not possible and hence we collect data for GPR for some burn-in time T to use it later to estimate the Q matrix. Thus using this method we can write
$Q(s) = GPR(X(1:t), X(s), Qest(1:t)) \forall (t < T) \& (s > T)$
where
$Qest(t) = co - variance(X(t - \frac{W}{2}, t + \frac{W}{2})) \forall t < T$
X is the state vector
Note here that Qest is 11 $\times$ 11 matrix resulting in 66 GPR computations for each element of Q (since Q is symmetric).

4.3.4 Flow diagram for Q computation

Note Q matrix can only be updated after the computation of innovation and Kalman gain at the time (t+1) i.e immediately after update equations (eq[4.15]-eq[4.19]). A data flow diagram is given in Fig. 4.1 explaining the EKF estimation process.

4.4 Update

$$\vec{V}_{t+1} = y_{t+1} - h(\vec{x}^{-}_{t+1}) \tag{4.15}$$
$$S_{t+1} = R + H_x(\vec{x}^{-}_{t+1}) P^{-}_{t+1} H_x(\vec{x}^{-}_{t+1})' \tag{4.16}$$
$$K_{t+1} = P^{-}_{t+1} H_x(\vec{x}^{-}_{t+1})' \vec{S}^{-1}_{t+1} \tag{4.17}$$
$$\vec{x}_{t+1} = \vec{x}^{-}_{t+1} + K_{t+1} \vec{V}_{t+1} \tag{4.18}$$
$$P_{t+1} = P^{-}_{t+1} - K_{t+1} S_{t+1} K'_{t+1} \tag{4.19}$$

where
$\vec{V}_{t+1}$ is the innovation used in eq[4.7]
h is the measurement model described in eq(2.11) and eq(2.12)
y_t is the measurement at time t+1
R is measurement co-variance matrix

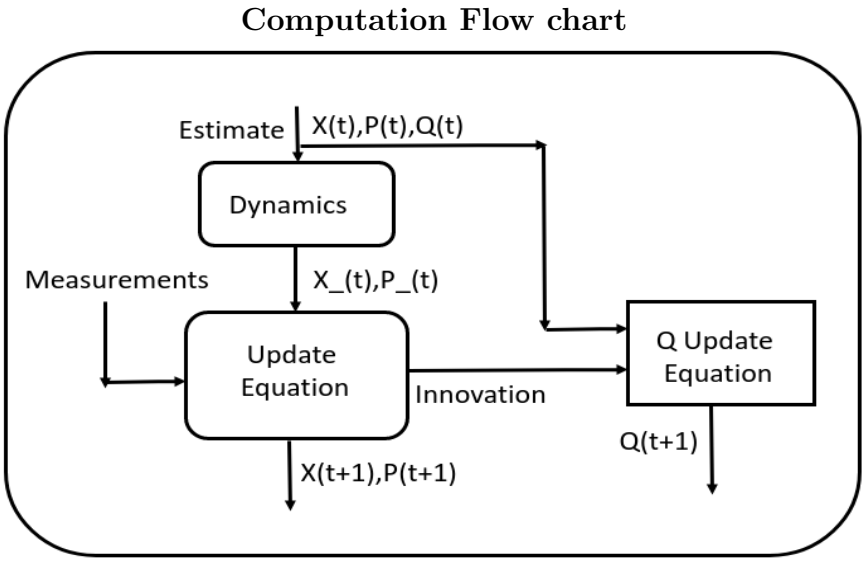

Figure 4.1: **Flow chart describing the flow of data in EKF algorithm with Q update. Estimates from the previous iteration used along with the current estimate to update the Q matrix which will be used in the subsequent cycle**

H_x is the partial derivative of the measurement model h w.r.t state vector evaluated at $\vec{x}_{t+1}^{-}$
K_{t+1} is the computed kalman gain

Note while computing H_x we have to compute quantities like $\frac{\partial(IGRF2015(\vec{r},\vec{v},sa))}{\partial\vec{r}}$, $\frac{\partial(IGRF2015(\vec{r},\vec{v},sa))}{\partial\vec{v}}$, $\frac{\partial(IGRF$
These partial derivatives are approximated using central difference scheme as define below

$$\frac{\partial(IGRF2015(\vec{r},\vec{v},sa))}{\partial\vec{r}} \approx \frac{IGRF2015(\vec{r}+\vec{\Delta r},\vec{v},sa)-IGRF(\vec{r}-\vec{\Delta r},\vec{v},sa)}{2\vec{\Delta r}}$$

$$\frac{\partial(IGRF2015(\vec{r},\vec{v},sa))}{\partial\vec{v}} \approx \frac{IGRF2015(\vec{r},\vec{v}+\vec{\Delta v},sa)-IGRF(\vec{r},\vec{v}+\vec{\Delta v},sa)}{2\vec{\Delta v}}$$

$$\frac{\partial(IGRF2015(\vec{r},\vec{v},sa))}{\partial\vec{v}} \approx \frac{IGRF2015(\vec{r},\vec{v}+,sa+\delta sa)-IGRF(\vec{r},\vec{v},sa-\delta sa)}{2\delta sa}$$

With these above formulations EKF is implemented and the result are discussed in the subsequent section.

Chapter 5

Experiment & Results

5.1 Sensor,Actuator FDI

5.1.1 Experiment Setup

For sensor/actuator fault attitude dynamics are considered as per the eq[2.1] & eq[2.2]. Inertia matrix is taken from an already flown satellite (Hysis). The values of the inertia matrix are given by

$$Inertia = \begin{bmatrix} 100, 0, 0 \\ 0, 80, 0 \\ 0, 0, 120 \end{bmatrix}$$

Three-axis Gyros were considered for rate measurement along with the 3 mutually orthogonal principal directions. Similarly, 3 reaction wheels actuators are assumed to be mounted along the orthogonal direction along the principal axis of the satellite. Reaction wheels considered here are 10 Nms wheels at 3600 RPM. A Proportional Derivative (PD) controller is used to control the satellite attitude. With this parameter in loop, we could identify both the sensor and actuator faults as shown below for a regulator problem.

5.1.2 Results (Model Based)

In these simulations, Gyro 1 sensor fault was simulated between 100 to 250 seconds and Reaction Wheel 1 Actuator fault was simulated between 300 to 350 seconds. At 120 seconds Gyro sensor fault was identified and the output of the first gyro observer was taken in the loop for proper control. These faults are identified with green lines for gyro sensor fault and red lines for actuator fault. The black line is the time from which gyro 1 observer output is taken in the

loop.

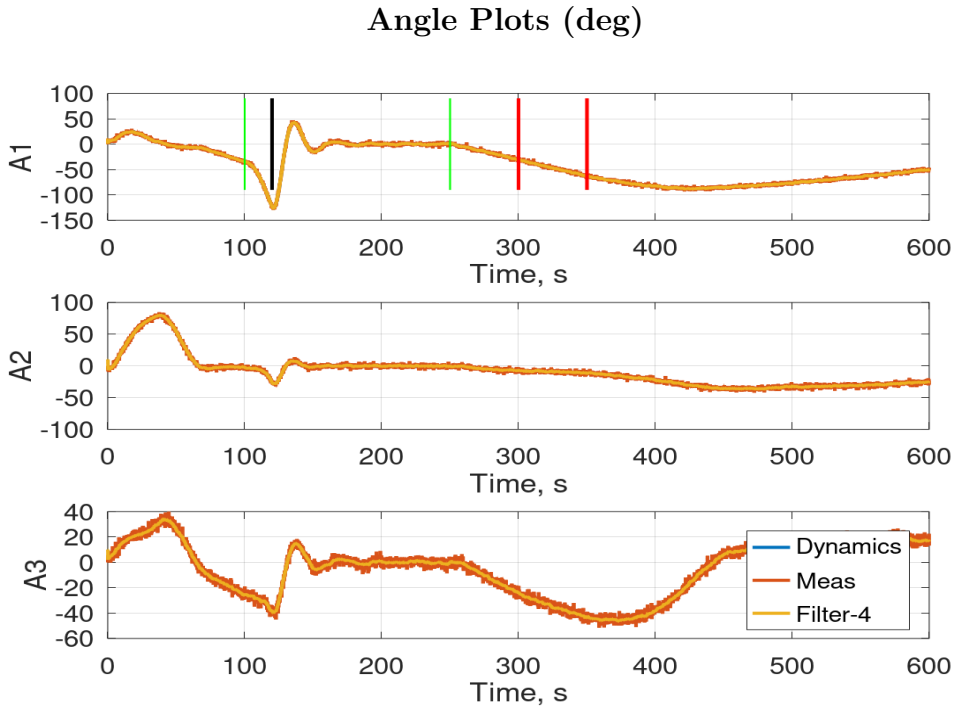

Figure 5.1: **Figure showing the variation of angles during failures. Gyro 1 failure happened between the green lines .Wheel1 actuator failure happened between the red lines.**

When there was a sensor fault at 100 seconds we can see that there is an angle change. But while there was a wheel failure there was none as the system already reached its study state as shown in Fig. 5.1.

Fig. 5.2 shows a gyro stuck fault from 100 to 250 seconds but recovery can seen from 120 seconds onwards.

Fig. 5.3 shows no residue in Angle 1 and Rate 1 channel during gyro 1 failure as these channels are not driven by gyro 1 output. Similarly, residue in wheel 1 channel is not observed during the wheel failure while it can be seen in all other channels. This information can be used for identifying the sensor and actuator failure.

5.1.3 Results (Model Learning)

For model learning algorithms like LSTM and LNN, a total of 10000 time-series samples are generated from validated simulation software. These time-series samples are broken down into a 50-second window sample for training purpose. During training LSTM/LNN network is trained with nominal time series and are then tested with a mix of faulty and nominal time series. Data

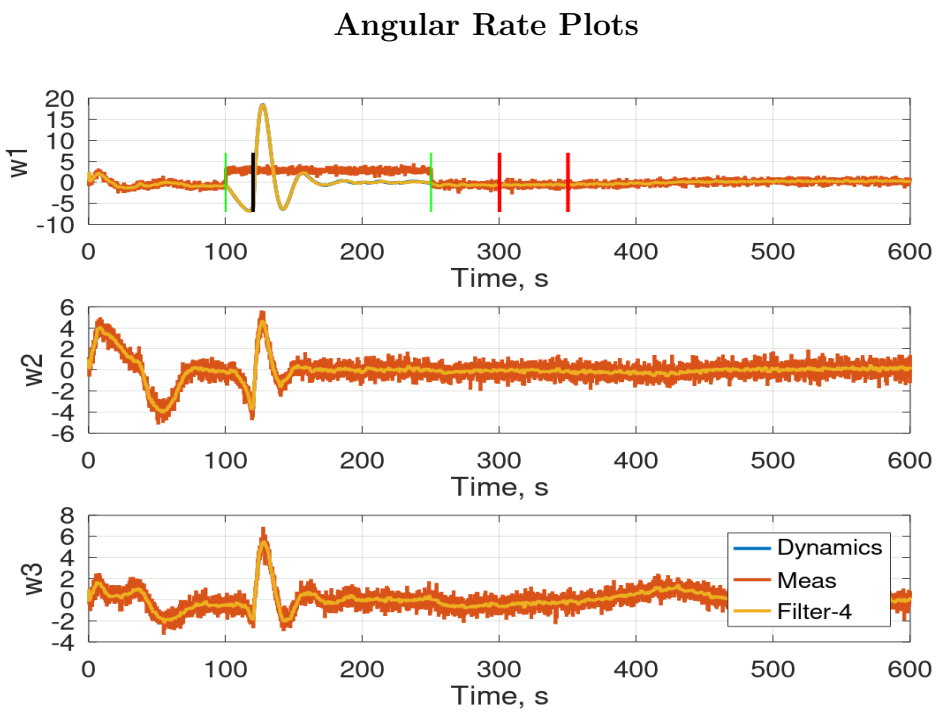

Figure 5.2: **Figure showing the gyro fail identification at 120 seconds (black line) and a corrective action brought back the angle errors to zeros subsequently. We can see measurement is faulty in axis 1 while there convergence of attitude using filter data.**

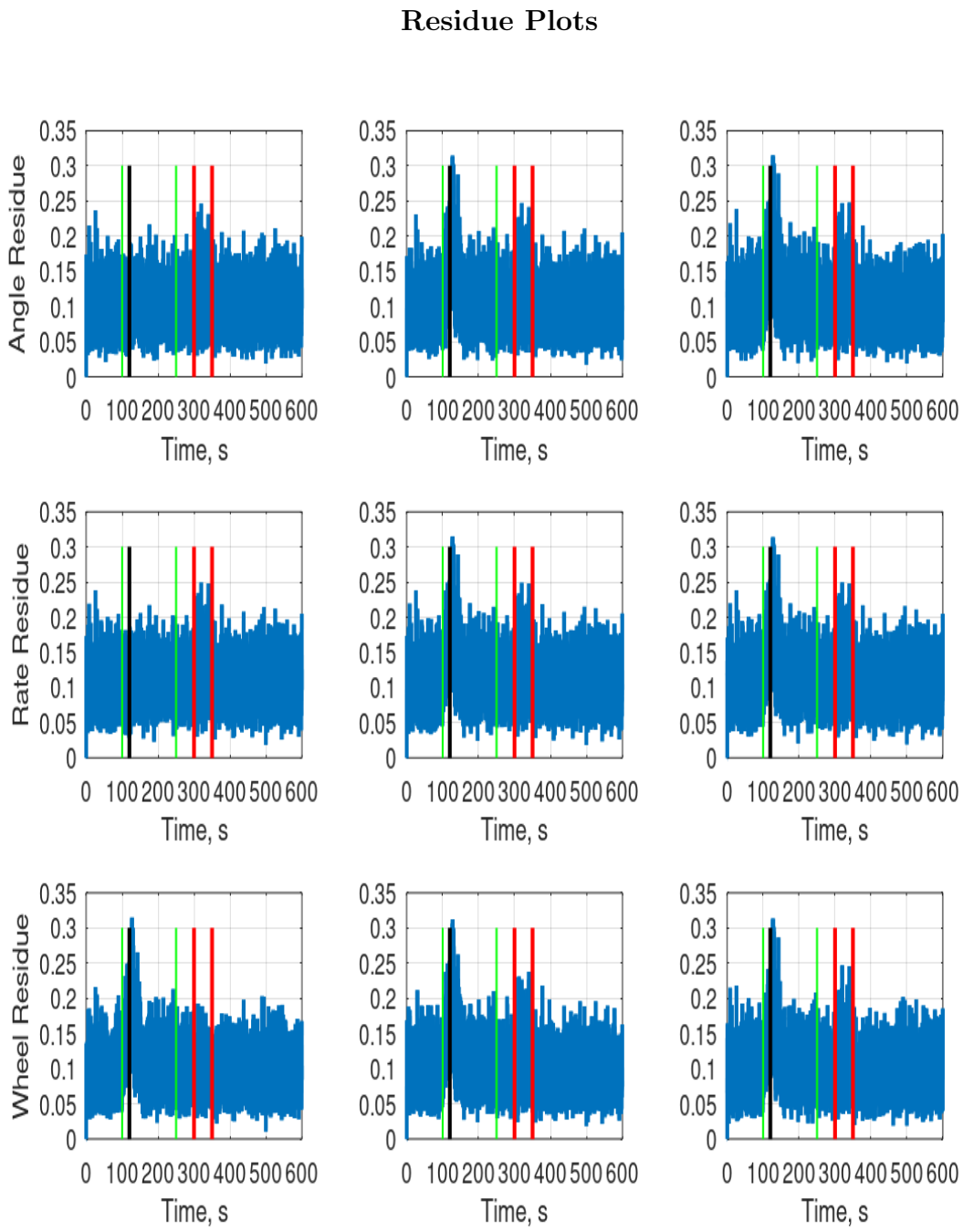

Figure 5.3: **Figure showing the variation of Residue during failures. During sensor failure (between green lines) there is no spike in Angle residue 1 and rate residue 1 while there is a residue increase in all other places. During the actuator failure (between the red lines) same phenomenon can be observed with wheel 1 residue.**

available during the actual telemetry (TM) will be of a similar format where most of the data available will be nominal with no failure data. Any failure has to be identified as new novelty detection.

5.1.3.1 LSTM Network

In LSTM Network we compute the difference between predicted output and actual observation from the data. The magnitude of these differences are stored for nominal cases and the distribution of these differences is used to identify faults by selecting samples outside the 95 % of the learnt distribution. The accuracy obtained by this method is **75 %**. Receiver Operating Characteristics of the learnt model is shown in Fig 5.4

LSTM Network - ROC

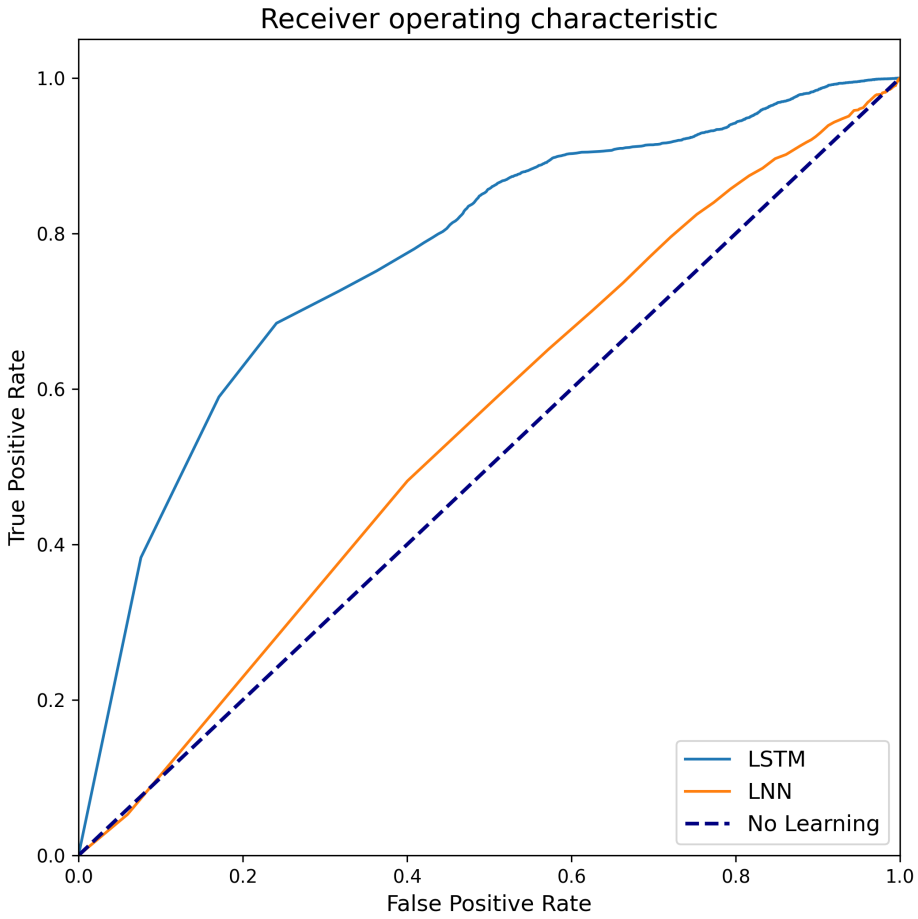

Figure 5.4: **ROC curve of learnt LSTM Network**

Accuracy				
Sl.No	Algorithm	Manual	AED	AED-KMeans
1.	OV-SVM	0.66	0.59	0.52
2.	Iso Forest	0.73	0.72	0.73
4.	K-means	0.80	0.73	0.67
5.	SOM	0.76	0.73	0.67
6.	IGMM	0.72	0.53	0.57

Table 5.1: Accuracy with different Anomaly detection algorithms

5.1.3.2 LNN Network

Similar to the LSTM network LNN network also learns the model from the nominal data set. It is important to note the LNN works with models which satisfy Newtonian dynamics and hence it's a good fit for satellite attitude control system but may not be a good fit for all kinds of dynamics. The accuracy obtained by this method is **54 %**. Receiver Operating Characteristics of the learnt model is shown in the Fig 5.4

5.1.4 Results (Time Series Features)

In this results section, we discuss the performance of various anomaly detection algorithms on extract features. Features are selected as mentioned in section 3.4.1. We first present the supervised learning results followed by unsupervised anomaly detection algorithms.

5.1.4.1 Supervised Learning Algorithms

Fig 5.5 gives the roc curves along with their accuracies of various supervised algorithms.

One can see that the best accuracy obtained is **91.4 %** for KNN but other classifiers are not too behind.

5.1.4.2 Anomaly/Novelty Detection Algorithms

In this subsection, we compared the performance of different anomaly detection algorithms. We compare the performance of various algorithms with different feature data sets. Fig 5.6 gives the ROC for various anomaly detection algorithms using manual features. Fig 5.7 & Fig 5.8 gives the same with features from AED and AED-Kmeans respectively. We can see that the accuracy achieved by the Manual features is better than other feature extraction methods. Table 5.1 gives the accuracies of anomaly detection algorithms applied on satellite data.

Supervised - Accuracy,ROC

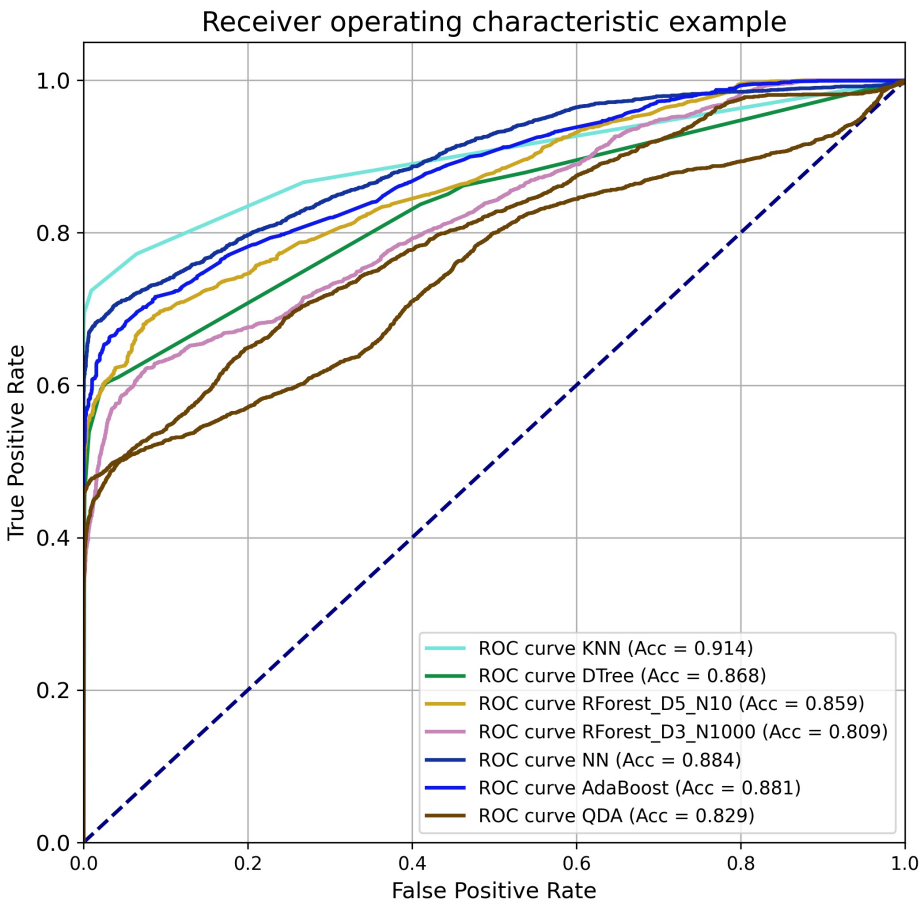

Figure 5.5: **Accuracy & ROC using supervised learning algorithms**

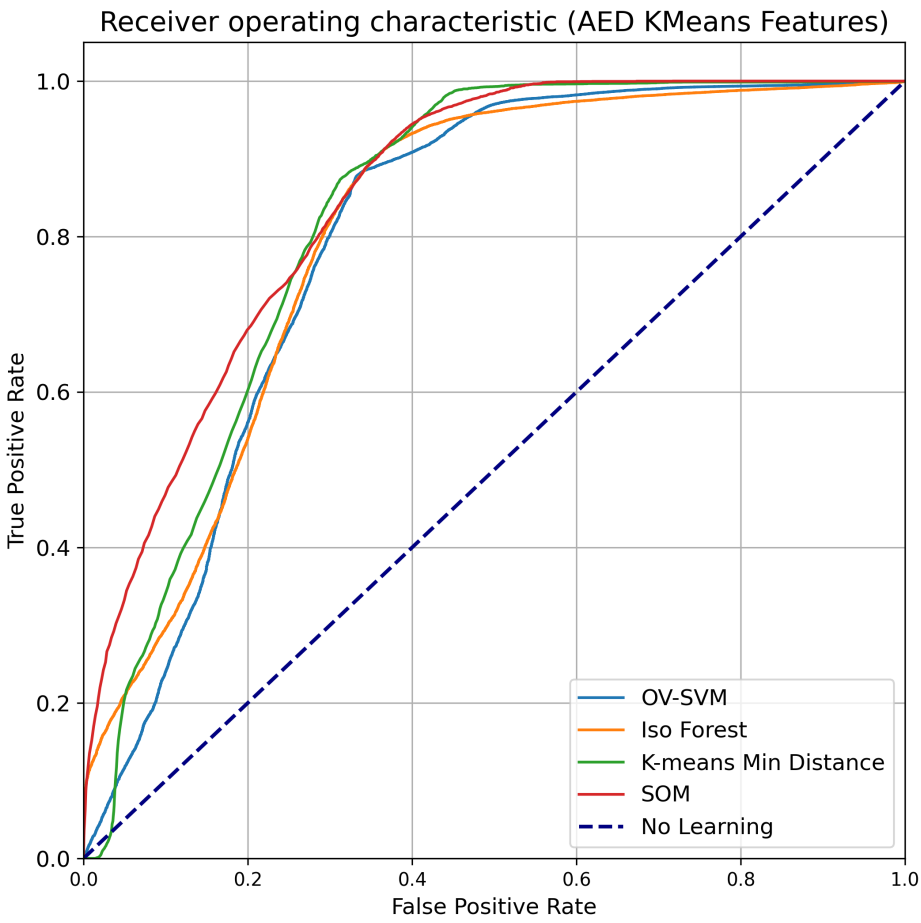

Figure 5.6: **ROC using Anomaly Detection learning algorithms with Manual Features**

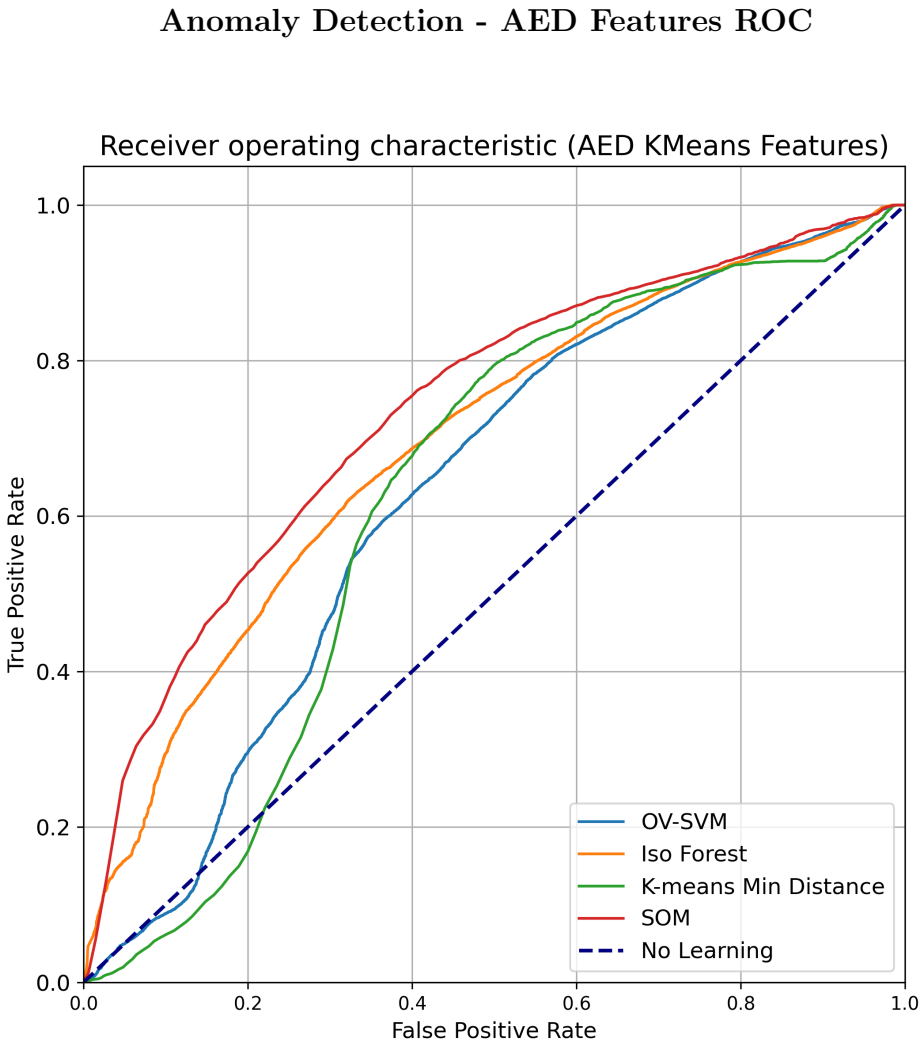

Figure 5.7: **ROC using Anomaly Detection learning algorithms with auto encoder decoder network features**

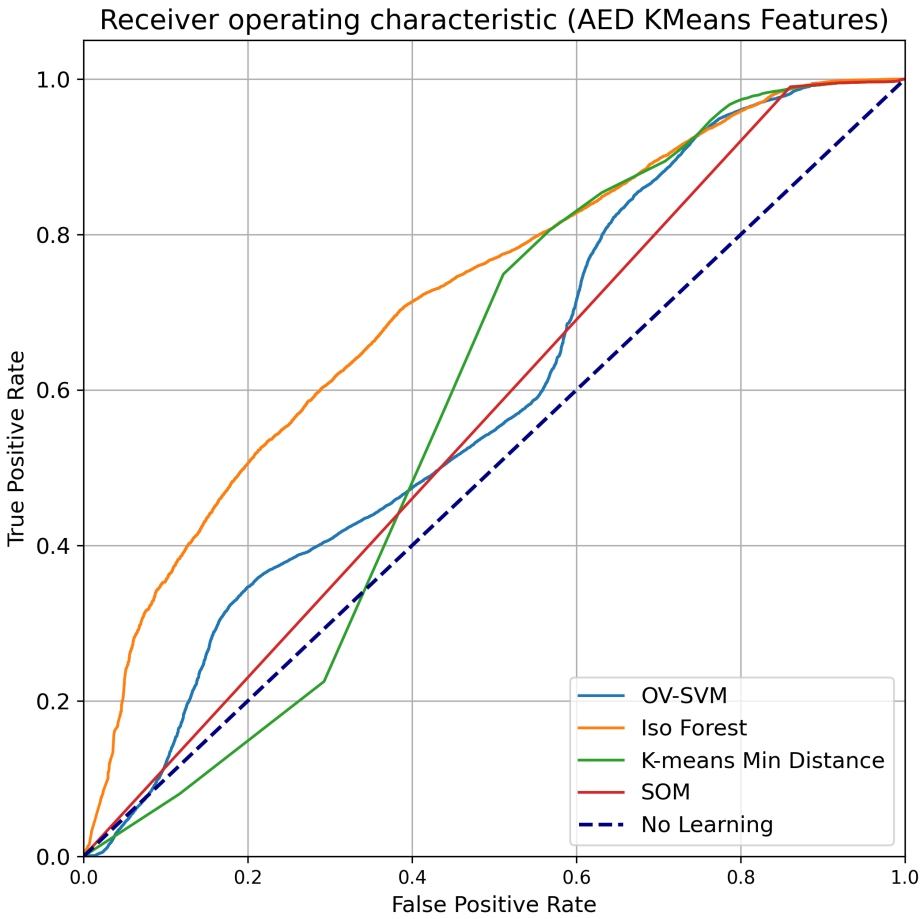

Figure 5.8: **ROC using Anomaly Detection learning algorithms with AED + KMeans features**

5.2 Navigation Solution using Magnetometer and Sun Sensor

5.2.1 Experiment Setup

Simulations are done with a polar orbit which has a 97^o inclination orbit to maintain orbital regression. In the polar orbits, the variation of magnetic field direction is relatively high (angle between two consecutive measured vectors) near the poles when compared against the field near the equator, which is helpful during the estimation. Satellite is equipped with a 3-axis magnetometer along with the Yaw, Roll and Pitch axis along with 4 modules of 4-pi sun-sensors. Each module of 4-pi sun-sensor is mounted with 3 sensors mounted in three orthogonal directions. Each 4-pi sun-sensor modules is mounted such that there will full coverage of 4-pi steradians around the satellite irrespective of the Sun's position to satellite.

Measurement data from sensors are assumed to be available for simulations in real-time at 8 sec sampling time. An initial positions error of $60\tilde{K}m$ and velocity error of $30\tilde{m}/s$ is assumed in each axis along an initial error of 1 deg in attitude. Typically these initial estimates in practice will be significantly less compared to what is being used here. Different αs is assumed in the current simulations and the best $\alpha = 0.8$ is selected based on these simulations. The entire simulation duration is about 20000 seconds ($3Orbits$). Simulation results show the convergence of position, velocity and quaternions with eclipse. The grey areas in the plots represent the eclipse duration.

5.2.2 Results

Fig. 5.9 shows the position RMSE values to time with different Q update schemes. It's very clear from the plots that Q update with $\alpha = 0.8$ and moving average scheme are more accurate than other schemes. AvgQ scheme takes the Q matrix vale to be the average of all the Q matrices at each time instant during $\alpha = 0.8$ experiment. Q computed using GPR method also showed good convergence properties especially during the initial phases. With this experiment, we could able to achieve accuracy close to 10 km (average RMSE).

Fig. 5.10 gives the Velocity RMSE with respect to time with different experimental set up. An average velocity RMSE of 30 m/s is observed consistently in all plots.

Fig. 5.11 gives the angle errors in all three axes during the estimation process while Fig. 5.12 gives error in sidereal angle.Except for $\alpha = 0.5$, sidereal angle estimation is accurate in all other experiments.

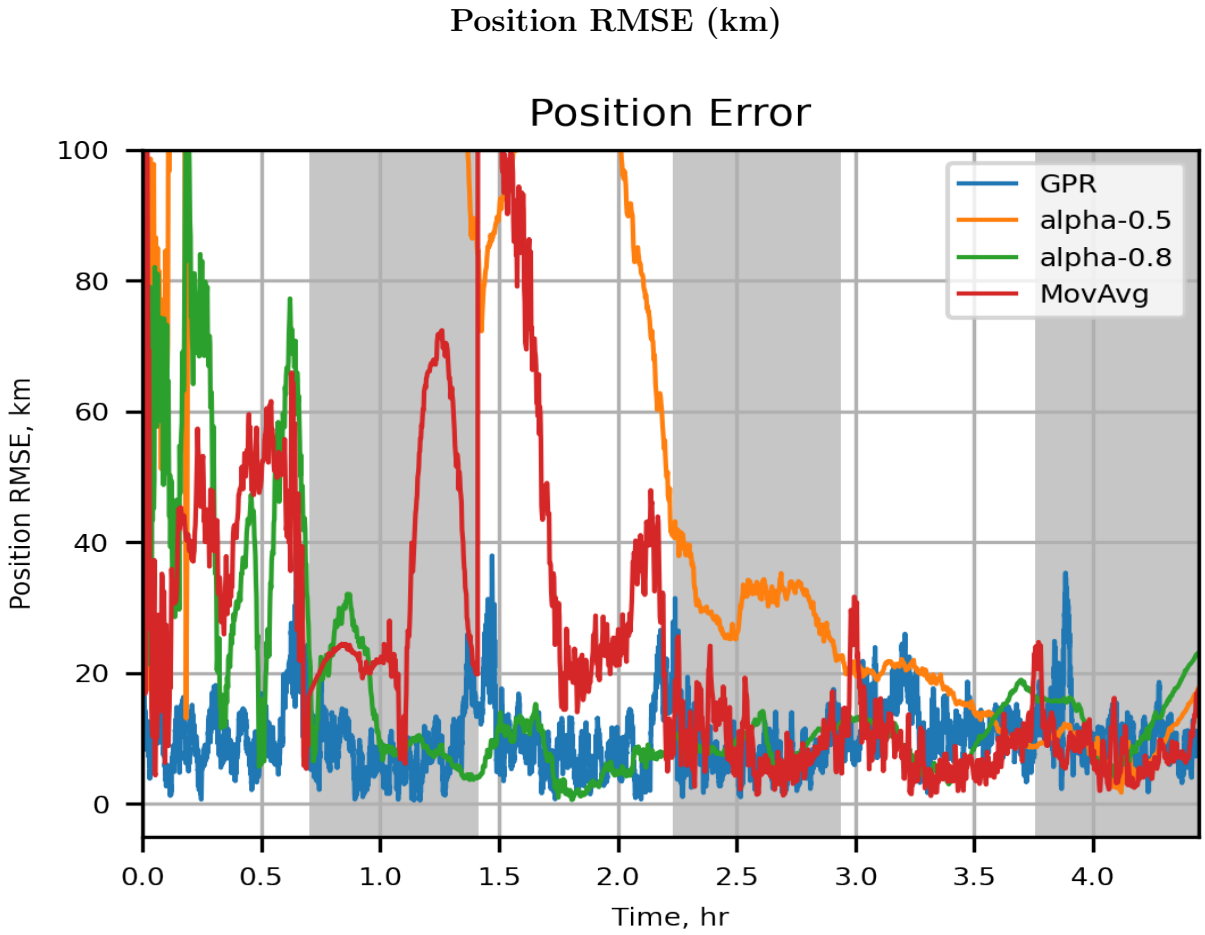

Figure 5.9: **Figure showing RMSE of position error with different Q update schemes**

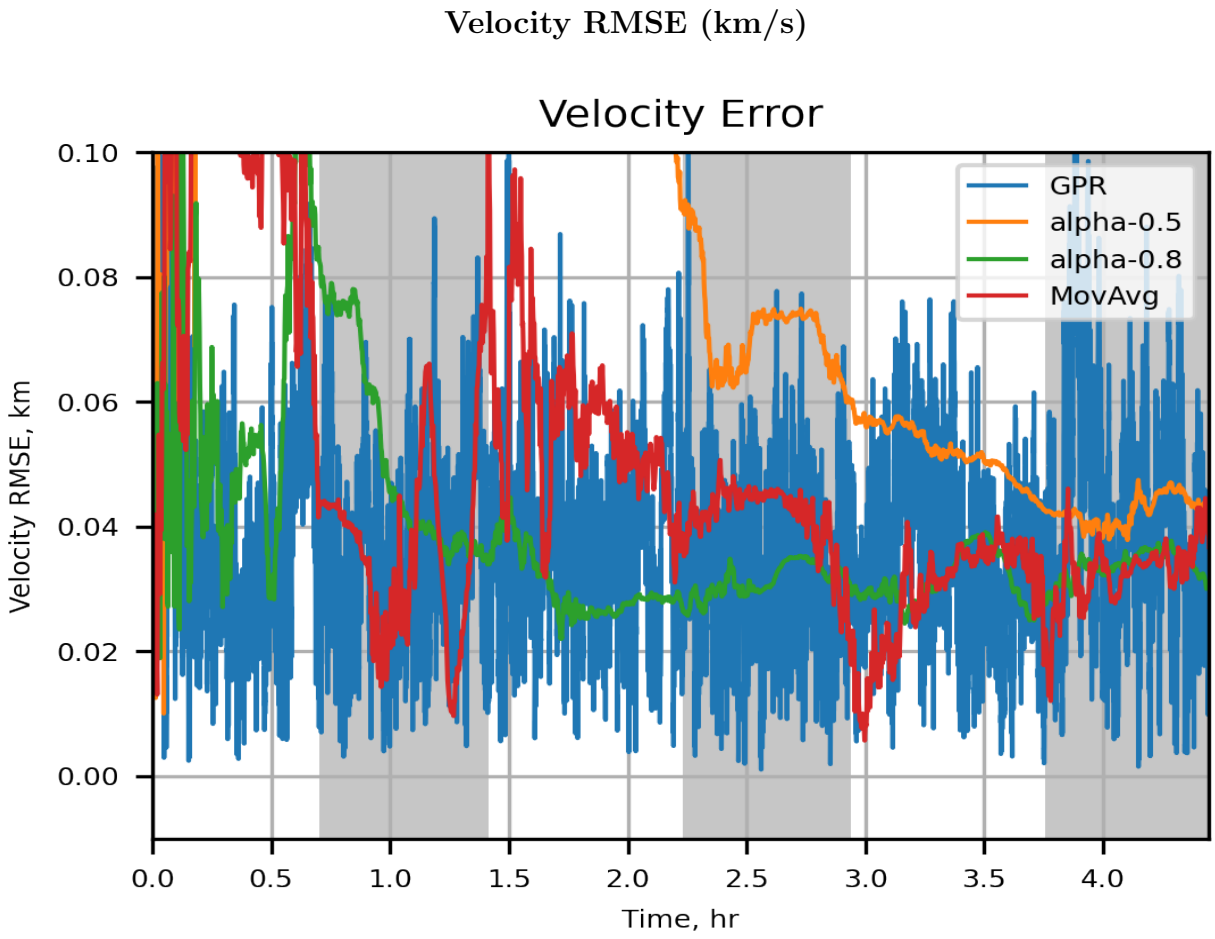

Figure 5.10: **Figure showing RMSE of Velocity error with different Q update schemes**

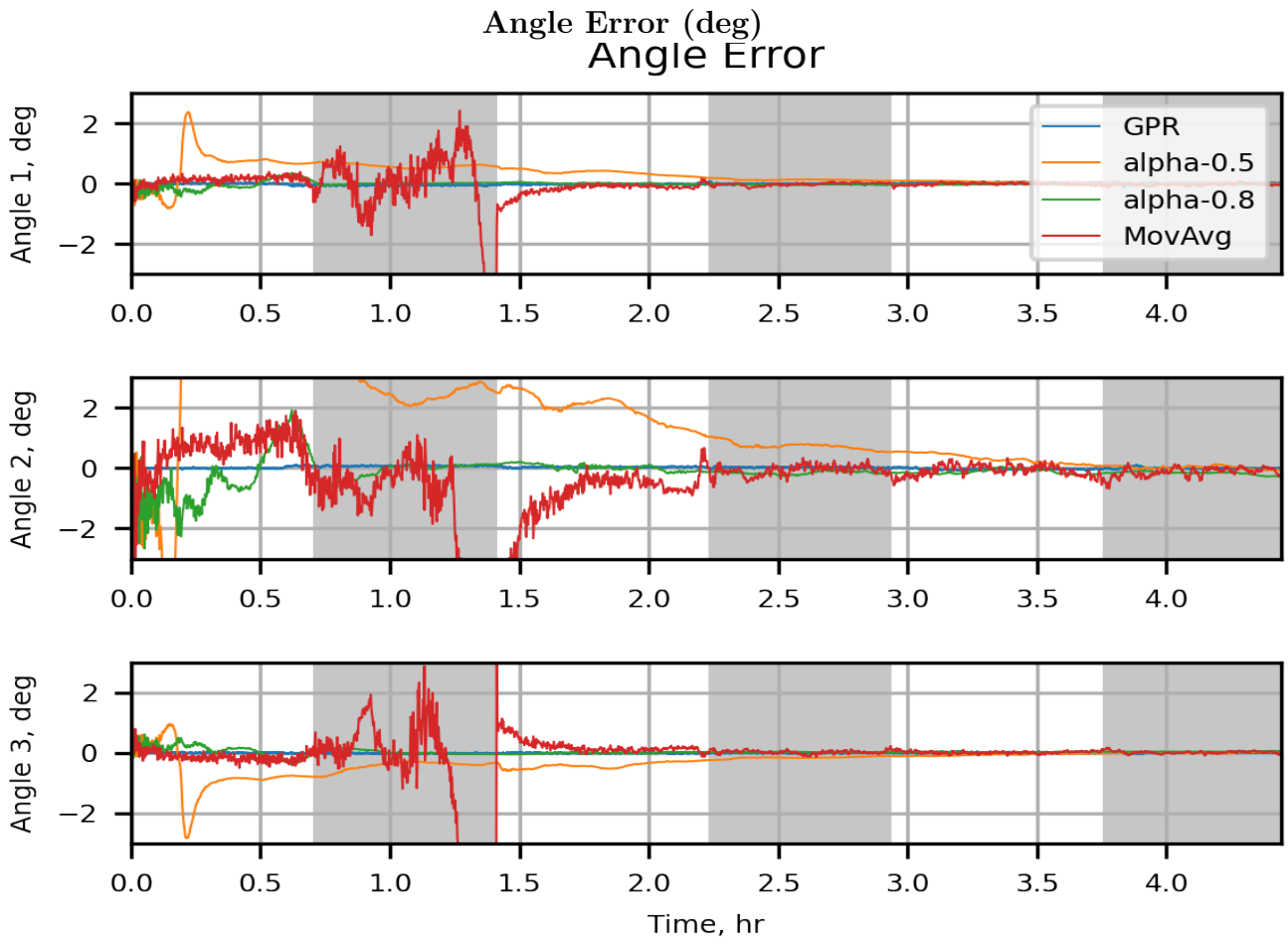

Figure 5.11: **Figure showing Angle Errors in all three axis with different Q update schemes**

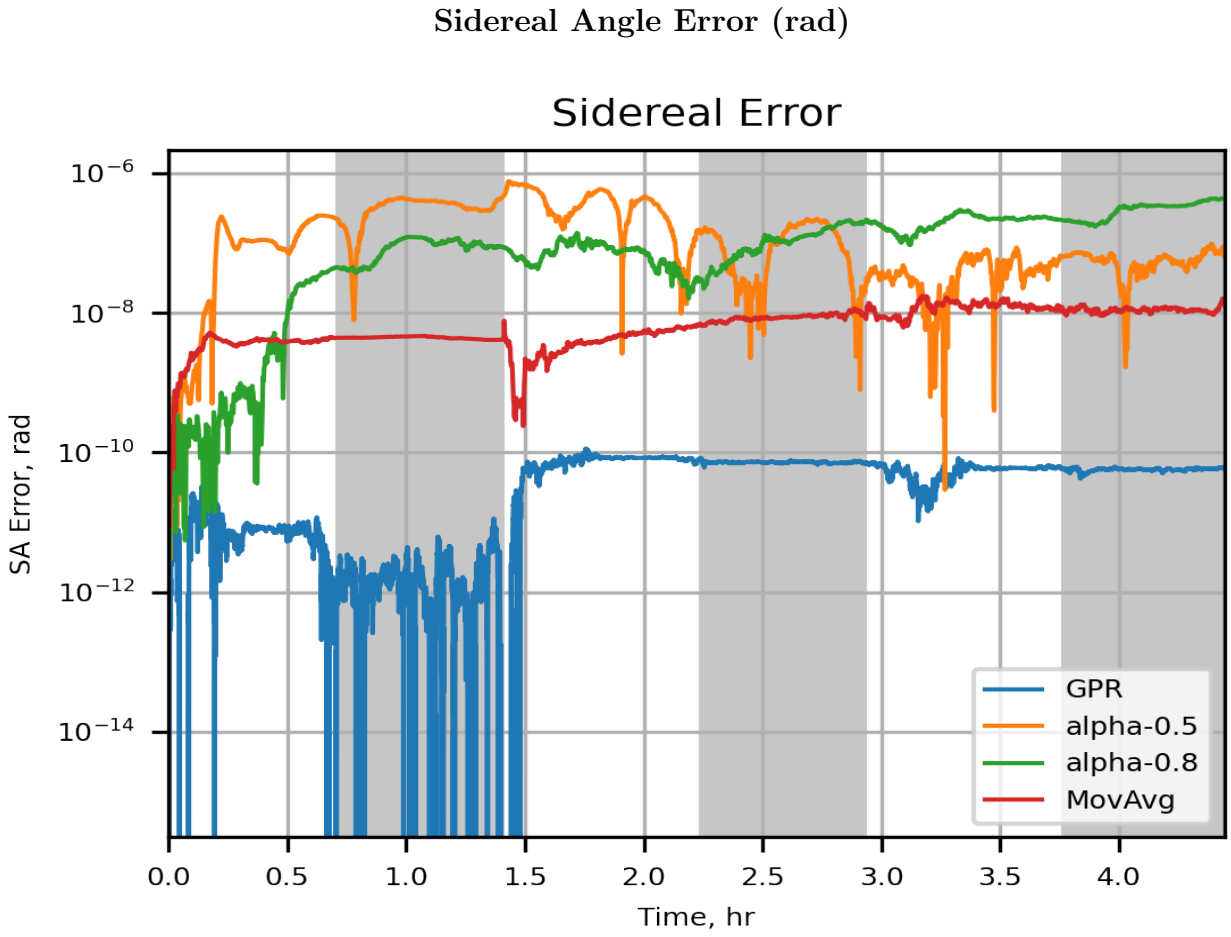

Figure 5.12: **Figure showing Sidereal Angle Errors with different Q update schemes**

Chapter 6

Conclusion

6.1 Fault Detection & Identification

1. Improved the existing model-based GOS algorithm to identify actuator faults along with sensor faults.
2. Successfully realised model learning based dynamics identification using LSTM & LNN network and used this model for FDI
3. Realised several Anomaly detection algorithms for FDI in satellite system
4. From the various experiments its been found out that manual features served better for FDI
5. Some failure profiles matched natural dynamics of satellite system resulting in misclassification
6. Training with longer sequences of time series data has improved the accuracy of model learning/anomaly detection algorithms

6.2 Navigation Solution

1. We could able to achieve a mean RMSE of 15 Km in position and 30 m/s in velocity which is better than results presented in [36] but not as good as results presented in [12] and [15]
2. A maximum attitude estimation error of 0.2^o is observed in each axis which is either comparable/better than all the results present in the references

3. Implementing Q (process covariance matrix) update equation has greatly improved variance in estimation.

4. Estimation took nearly 6000 seconds to converge to a constant error band

5. Results obtained during this estimation process especially in position, velocity estimation are very sensitive to the sensor noise.

6. Attitude errors are less sensitive to position errors

7. Algorithm is sensitive to the sampling rate at which data is available. Since the estimation of velocity is necessarily a differentiation operation, a lower sampling rate does not necessarily increase the accuracy of estimation.

8. During eclipse estimation error will get affected by the final value of quaternion error at the end of non-eclipse duration.

9. GPR Q update has greatly improved the accuracy in navigation solution when compared to Q-Learning & Moving Average Scheme

Bibliography

[1] Pieter Abbeel, Adam Coates, Michael Montemerlo, Andrew Y. Ng, and Sebastian Thrun. Discriminative Training of Kalman Filters. In *Robotics: Science and Systems I*. Robotics: Science and Systems Foundation, June 2005. ISBN 9780262701143. doi: 10.15607/RSS.2005.I.038. URL http://www.roboticsproceedings.org/rss01/p38.pdf. 4

[2] Leonardo Aguayo and Guilherme A. Barreto. Novelty Detection in Time Series Using Self-Organizing Neural Networks: A Comprehensive Evaluation. *Neural Processing Letters*, August 2017. ISSN 1370-4621, 1573-773X. doi: 10.1007/s11063-017-9679-2. URL http://link.springer.com/10.1007/s11063-017-9679-2. 4

[3] H.-S. Ahn and S.-H. Lee. Gyroless Attitude Estimation of Sun-Pointing Satellites Using Magnetometers. *IEEE Geoscience and Remote Sensing Letters*, 2(1):8–12, January 2005. ISSN 1545-598X. doi: 10.1109/LGRS.2004.840608. URL http://ieeexplore.ieee.org/document/1381338/. 6

[4] Shahrokh Akhlaghi, Ning Zhou, and Zhenyu Huang. Adaptive adjustment of noise covariance in Kalman filter for dynamic state estimation. In *2017 IEEE Power & Energy Society General Meeting*, pages 1–5, Chicago, IL, July 2017. IEEE. ISBN 9781538622124. doi: 10.1109/PESGM.2017.8273755. URL http://ieeexplore.ieee.org/document/8273755/. 4, 35, 36

[5] Omar Alghushairy, Raed Alsini, Terence Soule, and Xiaogang Ma. A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams. *Big Data and Cognitive Computing*, 5(1):1, December 2020. ISSN 2504-2289. doi: 10.3390/bdcc5010001. URL https://www.mdpi.com/2504-2289/5/1/1. 28

[6] David J. Atkinson, Mark L. James, and R. G. Martin. SHARP: automated monitoring of spacecraft health and status. pages 859–869, Orlando, FL, January 1990. doi:

10.1117/12.21136. URL http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=943280. 4

[7] Anastasios Bellas, Charles Bouveyron, Marie Cottrell, and Jérôme Lacaille. Anomaly detection based on confidence intervals using som with an application to health monitoring. *Advances in Intelligent Systems and Computing*, 295, 07 2014. doi: 10.1007/978-3-319-07695-9_14. 4

[8] Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks, 2020. 4, 22, 23

[9] Sylvain Fuertes, Gilles Picart, Jean-Yves Tourneret, Lotfi Chaari, André Ferrari, and Cédric Richard. Improving Spacecraft Health Monitoring with Automatic Anomaly Detection Techniques. In *SpaceOps 2016 Conference*, Daejeon, Korea, May 2016. American Institute of Aeronautics and Astronautics. ISBN 9781624104268. doi: 10.2514/6.2016-2430. URL https://arc.aiaa.org/doi/10.2514/6.2016-2430. 5

[10] Zhiwei Gao, Carlo Cecati, and Steven Ding. A Survey of Fault Diagnosis and Fault-Tolerant Techniques Part II: Fault Diagnosis with Knowledge-Based and Hybrid/Active Approaches. *IEEE Transactions on Industrial Electronics*, pages 1–1, 2015. ISSN 0278-0046, 1557-9948. doi: 10.1109/TIE.2015.2419013. URL http://ieeexplore.ieee.org/document/7076586/. 3, 15

[11] Zhiwei Gao, Carlo Cecati, and Steven X. Ding. A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches. *IEEE Transactions on Industrial Electronics*, 62(6):3757–3767, June 2015. ISSN 0278-0046, 1557-9948. doi: 10.1109/TIE.2015.2417501. URL http://ieeexplore.ieee.org/document/7069265/. 3, 15

[12] Ke Han, Hao Wang, Binjie Tu, and Zhonghe Jin. Pico-satellite Autonomous Navigation with Magnetometer and Sun Sensor Data. *Chinese Journal of Aeronautics*, 24(1):46–54, February 2011. ISSN 10009361. doi: 10.1016/S1000-9361(11)60006-X. URL https://linkinghub.elsevier.com/retrieve/pii/S100093611160006X. 6, 54

[13] Wei Huang and Xiaoxin Su. Design of a Fault Detection and Isolation System for Intelligent Vehicle Navigation System. *International Journal of Navigation and Observation*, 2015: 1–19, January 2015. ISSN 1687-5990, 1687-6008. doi: 10.1155/2015/279086. URL https://www.hindawi.com/journals/ijno/2015/279086/. 4, 15, 16

[14] Rolf Isermann. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer, Berlin ; New York, 2006. ISBN 9783540241126. OCLC: ocm61703226. 3, 15

[15] Y. Kim and G. Vukovich. Satellite orbit and attitude estimation using three-axis magnetometer. *International Journal of Space Science and Engineering*, 2(3):276, 2014. ISSN 2048-8459, 2048-8467. doi: 10.1504/IJSPACESE.2014.064202. URL http://www.inderscience.com/link.php?id=64202. 6, 54

[16] J. M Lewis, Sudarshan Kumar Dhall, S Lakshmivarahan, and Cambridge University Press. *Dynamic data assimilation: a least squares approach*. Cambridge University Press, Cambridge, U.K., 2009. ISBN 9780511526480. URL https://doi.org/10.1017/CBO9780511526480. OCLC: 852208330. 34

[17] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200209, April 2021. ISSN 1364-503X, 1471-2962. doi: 10.1098/rsta.2020.0209. URL https://royalsocietypublishing.org/doi/10.1098/rsta.2020.0209. 5

[18] Steve Lindsay and Diane M. Woodbridge. Spacecraft State-of-health (SOH) Analysis via Data Mining. In *SpaceOps 2014 Conference*, Pasadena, CA, May 2014. American Institute of Aeronautics and Astronautics. ISBN 9781624102219. doi: 10.2514/6.2014-1733. URL https://arc.aiaa.org/doi/10.2514/6.2014-1733. 5

[19] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, 6(1):1–39, March 2012. ISSN 1556-4681, 1556-472X. doi: 10.1145/2133360.2133363. URL https://dl.acm.org/doi/10.1145/2133360.2133363. 4, 28, 29

[20] J. Ma and S. Perkins. Time-series novelty detection using one-class support vector machines. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 3, pages 1741–1745, Portland, OR, USA, 2003. IEEE. ISBN 9780780378988. doi: 10.1109/IJCNN.2003.1223670. URL http://ieeexplore.ieee.org/document/1223670/. 4

[21] Qianli Ma, Jiawei Zheng, Sen Li, and Garrison Cottrell. Learning representations for time series clustering. 12 2019. 5, 27

[22] Dubravko Miljković. Fault detection methods: A literature survey. May 2011. 1, 3, 15

[23] Howard Musoff and Paul Zarchan. *Fundamentals of Kalman Filtering: A Practical Approach, Third Edition*. American Institute of Aeronautics and Astronautics, Reston, VA, 2009. ISBN 9781600867200. URL https://doi.org/10.2514/4.867200. OCLC: 884705084. 34

[24] Ahmad M. Mustafa, Gbadebo Ayoade, Khaled Al-Naami, Latifur Khan, Kevin W. Hamlen, Bhavani Thuraisingham, and Frederico Araujo. Unsupervised deep embedding for novel class detection over data stream. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1830–1839, Boston, MA, December 2017. IEEE. ISBN 9781538627150. doi: 10.1109/BigData.2017.8258127. URL http://ieeexplore.ieee.org/document/8258127/. 5

[25] Zahra Nazari, Seong-Mi Yu, Dongshik Kang, and Yousuke Kawachi. Comparative Study of Outlier Detection Algorithms for Machine Learning. In *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies - ICDLT '18*, pages 47–51, Chongqing, China, 2018. ACM Press. ISBN 9781450364737. doi: 10.1145/3234804.3234817. URL http://dl.acm.org/citation.cfm?doid=3234804.3234817. 4, 28

[26] Shaobo Ni and Cui Zhang. Attitude Determination of Nano Satellite Based on Gyroscope, Sun Sensor and Magnetometer. *Procedia Engineering*, 15:959–963, 2011. ISSN 18777058. doi: 10.1016/j.proeng.2011.08.177. URL https://linkinghub.elsevier.com/retrieve/pii/S187770581101678X. 6

[27] Daehyung Park, Yuuna Hoshi, and Charles C. Kemp. A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, July 2018. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2018.2801475. URL http://ieeexplore.ieee.org/document/8279425/. 5

[28] R. J. Patton and J. Chen. Review of parity space approaches to fault diagnosis for aerospace systems. *Journal of Guidance, Control, and Dynamics*, 17(2):278–285, March 1994. ISSN 0731-5090, 1533-3884. doi: 10.2514/3.21194. URL https://arc.aiaa.org/doi/10.2514/3.21194. 4

[29] Carl Rasmussen. The infinite gaussian mixture model. volume 12, pages 554–560, 04 2000. 4, 29

[30] Sylvia. Richardson and Peter J. Green. On Bayesian Analysis of Mixtures with an Unknown Number of Components (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(4):731–792, 1997. ISSN 13697412. doi: 10.1111/1467-9868.00095. URL http://doi.wiley.com/10.1111/1467-9868.00095. 4, 29

[31] Simo Sarakka. *Bayesian filtering and smoothing*. Cambridge University Press, Cambridge, United Kingdom, 2013. ISBN 9781139344203 9781107619289 9781107030657. OCLC: 884914132. 34

[32] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. volume 12, pages 582–588, 01 1999. 28

[33] Marcel J. Sidi. *Spacecraft dynamics and control: a practical engineering approach*. Number 7 in Cambridge aerospace series. Cambridge Univ. Press, Cambridge, 1997. ISBN 9780521787802 9780521550727. OCLC: 34691034. 8

[34] David M.J. Tax and Robert P.W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, Jan 2004. ISSN 1573-0565. doi: 10.1023/B:MACH.0000008084.60811.49. URL https://doi.org/10.1023/B:MACH.0000008084.60811.49. 28

[35] Stephan Theil, Pontus Appel, and Alexander Schleicher. Low Cost, Good Accuracy - Attitude Determination Using Magnetometer and Simple Sun Sensor. 6

[36] Julie Thienel, Rick Harman, Itzhack Bar-Itzhack, and Mike Lambertson. Results of the Magnetometer Navigation (MAGNAV) Inflight Experiment. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Providence, Rhode Island, August 2004. American Institute of Aeronautics and Astronautics. ISBN 9781624100758. doi: 10.2514/6.2004-4749. URL http://arc.aiaa.org/doi/10.2514/6.2004-4749. 6, 12, 54

[37] Massimo Tipaldi and Bernhard Bruenjes. Spacecraft health monitoring and management systems. In *2014 IEEE Metrology for Aerospace (MetroAeroSpace)*, pages 68–72, Benevento, Italy, May 2014. IEEE. ISBN 9781479920693. doi: 10.1109/MetroAeroSpace.2014.6865896. URL http://ieeexplore.ieee.org/document/6865896/. 5

[38] Cunsong Wang, Ningyun Lu, Yuehua Cheng, and Bin Jiang. A telemetry data based diagnostic health monitoring strategy for in-orbit spacecrafts with component degradation. *Advances in Mechanical Engineering*, 11(4):168781401983959, April 2019. ISSN 1687-8140, 1687-8140. doi: 10.1177/1687814019839599. URL http://journals.sagepub.com/doi/10.1177/1687814019839599. 5, 25

[39] Montri Wiboonrat. Developing Diagnostics and Prognostics of Data Center Systems Implementing with Condition-Based Maintenance. In *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, pages 4901–4906, Washington, DC, October 2018. IEEE. ISBN 9781509066841. doi: 10.1109/IECON.2018.8591203. URL https://ieeexplore.ieee.org/document/8591203/. 5

[40] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V. Chawla. A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1409–1416, July 2019. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v33i01.33011409. URL https://aaai.org/ojs/index.php/AAAI/article/view/3942. 4