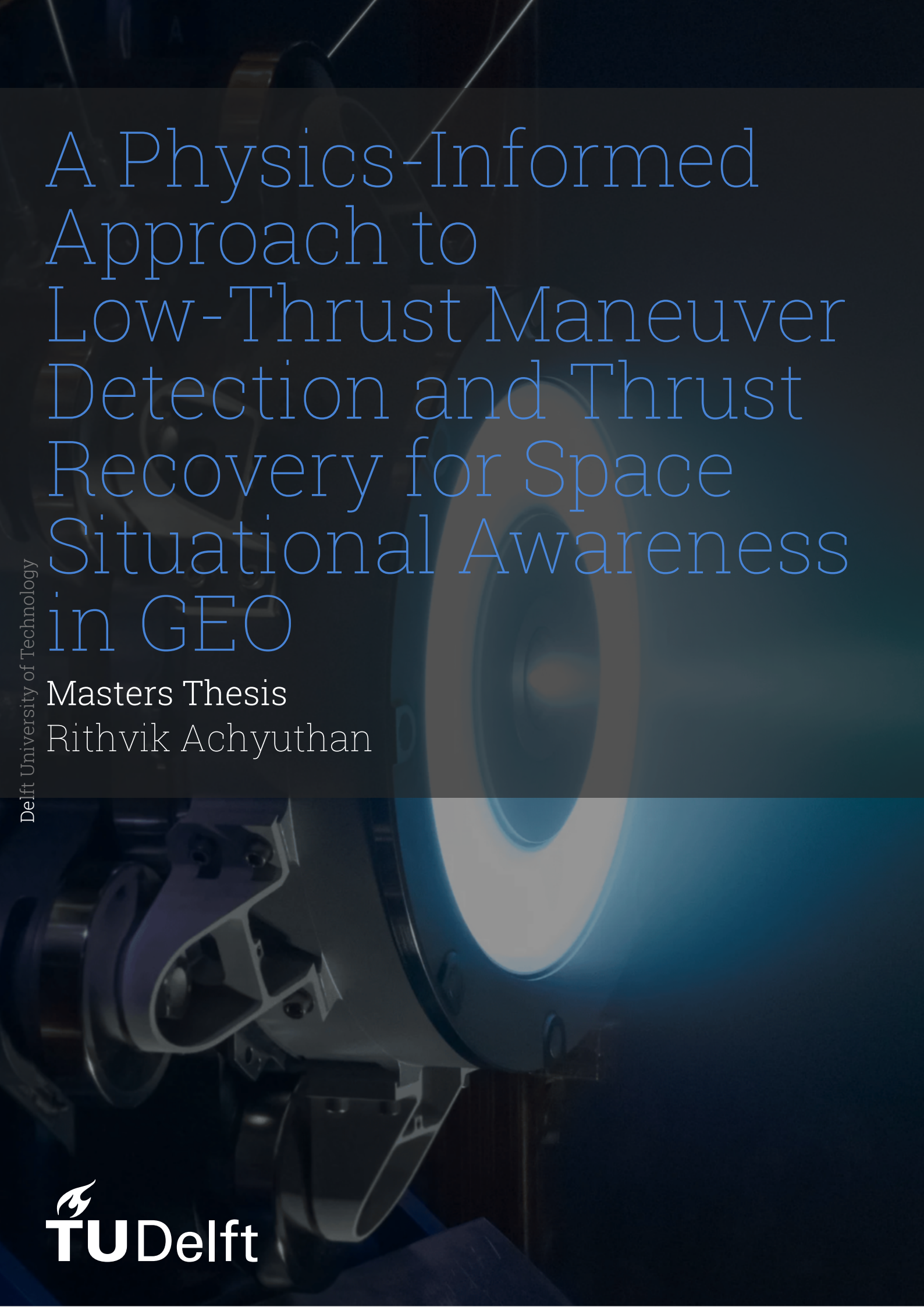


A Physics-Informed Approach to Low-Thrust Maneuver Detection and Thrust Recovery for Space Situational Awareness in GEO

Masters Thesis

Rithvik Achyuthan

Delft University of Technology

TU Delft

A Physics-Informed Approach to Low-Thrust Maneuver Detection and Thrust Recovery for Space Situational Awareness in GEO

by

Rithvik Achyuthan

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday April 24, 2026 at 13:00.

Student number:	5991587	
Project duration:	June 11, 2025 – April 24, 2026	
Thesis committee:	Prof. Dr. S. Gehly,	TU Delft, Supervisor
	Dr. ir. E.G.O. Schrama,	TU Delft, Chair
	Dr. J.G. de Teixeira da Encarnacao,	TU Delft, Examiner

Cover: True Blue: High-Power Propulsion for Gateway by NASA, public domain (Modified)

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

This thesis marks the end of a journey that began with two separate interests, machine learning and space situational awareness and the slow realization that the most interesting problems tend to live at the boundary between fields rather than comfortably inside one. Physics-informed neural networks, applied to the detection of low-thrust maneuvers in GEO, turned out to be exactly that kind of boundary problem: enough orbital mechanics to keep me humble, enough machine learning to keep me curious, and enough unknowns to keep me working late more often than I'd planned.

If I could give one piece of advice to the version of myself who started this master's thesis, it would be this: dream big in scope, but narrow it down early. It is tempting to chase every interesting thread, and I chased quite a few before learning that a thesis is defined as much by what is left out as by what is included. The work that remains is, I hope, sharper for it.

I am grateful to my supervisor, Steve Gehly, for the guidance, the patience, and the willingness to let me wander far enough to learn something before gently steering me back. I would also like to thank the Faculty of Aerospace Engineering at Delft University of Technology for providing the academic home in which this work was carried out, and the DelftBlue support team, whose help in getting me set up on the supercomputer made the computational side of this work possible. To my parents, whose support has been unwavering across continents and time zones, and to my friends in Delft and beyond, who kept me grounded when the work threatened to become all-consuming — thank you. You made the difficult parts bearable and the good parts better.

Rithvik Achyuthan
Delft, April 2026

Abstract

The detection of non-compliant low-thrust maneuvers in Geosynchronous Earth Orbit (GEO) is hindered by the similarity between thrust accelerations and uncertainties in solar radiation pressure (SRP) modelling. In this thesis, a hybrid architecture is proposed that combines a Long Short-Term Memory (LSTM) classifier with a Physics-Informed Neural Network (PINN) formulated as an inverse-thrust recovery solver. A synthetic dataset of 8400 GEO trajectories across three classes, nominal, low-thrust, and mismodeled SRP, is used for training and evaluation. The LSTM classifier achieved over 88% accuracy on a noisy test set, and thrust vectors were recovered by the PINN with a median magnitude error of 0.9% and directional error of 0.6°. Superior performance over both a TLE sliding-window method and an Unscented Kalman Filter is demonstrated.

Nomenclature

Abbreviations

Abbreviation	Definition
AD	Automatic Differentiation
Adam	Adaptive Moment Estimation
AdamW	Adam with decoupled weight decay regularization
CNN	Convolutional Neural Network
DOP853	Dormand-Prince eighth-order Runge-Kutta integrator
ECI	Earth-Centred Inertial
EKF	Extended Kalman Filter
ESA	European Space Agency
GEO	Geosynchronous Earth Orbit
GEODSS	Ground-Based Electro-Optical Deep Space Surveillance
GTO	Geosynchronous Transfer Orbit
IADC	Inter-Agency Debris Coordination Committee
ITU	International Telecommunication Union
L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno
LEO	Low Earth Orbit
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NN	Neural Network
OCBE	Optimal Control Based Estimator
OGS	Optical Ground Station
PINN	Physics-Informed Neural Network
PoL	Pattern of Life
RAAN	Right Ascension of the Ascending Node
ReLU	Rectified Linear Unit
RK4	Fourth-order Runge-Kutta integrator
RNN	Recurrent Neural Network
RSO	Resident Space Object
RTN	Radial-Tangential-Normal
SGP4	Simplified General Perturbations 4
SNR	Signal-to-Noise Ratio
SOD	Statistical Orbit Determination
SPLID	Satellite Pattern-of-Life Identification Dataset
SRP	Solar Radiation Pressure

Abbreviation	Definition
SSA	Space Situational Awareness
SSN	United States Space Surveillance Network
TFC	Thrust Fourier Coefficients
TLE	Two-Line Element
UKF	Unscented Kalman Filter

Symbols

Symbol	Definition
a	Semi-major axis
$\mathbf{a}^{(l)}$	Activation vector of layer l
$\mathbf{a}_{3b}$	Acceleration due to third-body gravitational effects
$\mathbf{a}_{J_2}$	Acceleration due to J2 perturbation
$\mathbf{a}_{\text{natural}}$	Total natural (non-thrust) perturbing acceleration
$\mathbf{a}_{\text{pert}}$	Total perturbing acceleration
$\mathbf{a}_{\text{SRP}}$	Acceleration due to solar radiation pressure
$\mathbf{a}_T$	Thrust acceleration vector
a_{scale}	Acceleration normalisation factor
A/m	Area-to-mass ratio
b	Bias scalar (single neuron)
$\mathbf{b}^{(l)}$	Bias vector of layer l
c	Speed of light
$\mathbf{c}$	Attention context vector
C_R	Dimensionless coefficient of reflectivity
d_h	LSTM hidden state dimension
e	Orbital eccentricity
E	Total mechanical energy
$\mathcal{E}$	Specific orbital energy
f_{pred}	Physics residual
$\mathbf{f}_k$	Feature vector at timestep k
H	Hamiltonian function
h	Specific angular momentum magnitude
$\mathbf{h}$	Specific angular momentum vector
$\mathbf{h}_k$	LSTM hidden state at timestep k
h_p	Altitude above Earth's surface
i	Orbital inclination
J	Cost function (OCBE optimal control)
J_2	Second zonal harmonic coefficient
k	Spring constant (mass-spring system)
L_{data}	Data loss
L_{IC}	Initial condition loss
L_{ODE}	ODE residual loss
L_{physics}	Physics loss

Symbol	Definition
L_{total}	Total loss function
m	Mass of spacecraft
M	Mass of central body (Earth)
$\mathcal{N}$	Neural network function
p	Canonical momentum variable
q	Canonical position variable
$\mathbf{r}$	Position vector
r	Magnitude of position vector
R_0	Reference distance (1 AU)
R_E	Mean equatorial radius of Earth
r_{GEO}	Geosynchronous orbit radius
$r_\odot$	Distance from Sun to spacecraft
$\dot{r}$	Radial velocity
$\mathbf{r}^{\text{ref}}$	Reference trajectory position vector
$\hat{\mathbf{r}}_S$	Unit vector from Sun to spacecraft
s_k	Attention score at timestep k
t	Time
t_0	Initial time
t_c	Collocation time points
T	Thrust magnitude
$\hat{\mathbf{u}}$	Unit vector of thrust direction
$\mathbf{u}(t)$	Control acceleration vector
v_0	Initial velocity (mass-spring system)
$\mathbf{v}$	Velocity vector
v_{GEO}	Circular orbit speed at GEO
W	Neural network weight matrix
$\mathbf{W}^{(l)}$	Weight matrix of layer l
$W_\odot$	Average solar radiation flux at 1 AU
$\mathbf{x}$	State vector
x_0	Initial displacement (mass-spring system)
z	Pre-activation (weighted sum before activation function)
α_k	Temporal attention weight at timestep k
$\boldsymbol{\delta}$	Position deviation from reference trajectory
δa	Semi-major axis deviation from GEO reference
$\delta\mathcal{E}$	Specific orbital energy deviation from GEO reference
δg	Gravity differential
δh	Specific angular momentum deviation from GEO reference
δr	Radial position deviation from GEO reference
δv	Speed deviation from GEO reference
Δv	Change in velocity
Δ_{scale}	PINN output scaling factor
η	Learning rate

Symbol	Definition
λ_{physics}	Weighting hyperparameter for physics loss
λ_T	Thrust scaling factor
μ	Standard gravitational parameter
μ_j	Gravitational parameter of third body
ν	Cylindrical shadow function
σ	Activation function
τ	Normalised time
ω	Argument of perigee / natural frequency
ω_{ref}	Reference angular frequency (Fourier PINN)
$\omega_{\odot}$	Angular velocity of the Sun
ω_{moon}	Angular velocity of the Moon
Ω	Right Ascension of the Ascending Node

1

Introduction

The Geosynchronous Earth Orbit (GEO) regime represents one of the most strategically valuable regions of near-Earth space. At an altitude of approximately 35,786 km, satellites in this regime orbit with a period matching Earth's sidereal day, enabling persistent coverage for telecommunications, meteorology, and national security applications [44]. However, this unique orbital characteristic also makes GEO an increasingly congested and contested environment. As the number of resident space objects continues to grow, maintaining accurate catalogues and ensuring the safety of operational assets has become a pressing challenge in Space Situational Awareness (SSA).

A critical component of SSA is the ability to detect and characterize satellite maneuvers. When a satellite performs an orbital maneuver, its trajectory deviates from the predicted path, potentially leading to catalogue inaccuracies, missed conjunctions, or undetected proximity operations. Traditional maneuver detection techniques, such as geometric analysis of Two-Line Element (TLE) histories and statistical orbit determination using Kalman filters, have proven effective for detecting large, impulsive burns. However, these methods face fundamental limitations when confronted with continuous low-thrust maneuvers. Electric propulsion systems, which are increasingly adopted for stationkeeping and orbit-raising in GEO, produce accelerations on the order of 10^{-7} km/s^2, magnitudes comparable to the uncertainties inherent in modelling perturbations such as solar radiation pressure (SRP) [17][20]. This overlap between the maneuver signal and the perturbation noise floor renders conventional threshold-based detection methods largely ineffective.

Recent advances in machine learning have introduced data-driven approaches to this problem, with techniques ranging from supervised classifiers to recurrent neural networks demonstrating promise in identifying anomalous orbital behaviour [7][16]. Yet, purely data-driven models suffer from well-documented limitations: they are prone to overfitting, lack physical interpretability, and struggle to generalize from synthetic training environments to real-world observations characterized by higher variability and noise [38]. An emerging class of models, known as Physics-Informed Neural Networks (PINNs), offers a potential solution by embedding the governing differential equations of orbital motion directly into the learning process, thereby constraining the

network to produce physically plausible solutions [29].

This thesis investigates whether integrating orbital dynamics into a data-driven framework can overcome the limitations of standard maneuver detection techniques for identifying low-thrust maneuvers in GEO. The proposed approach employs a hybrid architecture comprising two complementary components: a Long Short-Term Memory (LSTM) network with temporal attention for classifying orbital anomalies, and a PINN formulated as an inverse-problem solver for reconstructing the unknown thrust vector. The LSTM leverages engineered features derived from orbital mechanics to distinguish between nominal trajectories, low-thrust maneuvers, and mismodeled SRP signatures. For trajectories flagged as anomalous, the PINN recovers the constant-thrust acceleration by learning the small positional deviation relative to a numerically integrated reference trajectory, with the equations of motion enforced through a physics-based loss function.

The contributions of this work are threefold. First, a synthetic dataset generation framework is developed to produce labelled GEO trajectories across three distinct classes using realistic perturbation models. Second, the LSTM classifier is benchmarked against both a traditional TLE sliding-window method and an Unscented Kalman Filter (UKF), demonstrating superior detection capability for subtle anomalies. Third, the PINN-based inverse solver is shown to recover thrust vectors with median magnitude errors below 1% and median directional errors of approximately 0.6°, providing actionable intelligence that classical filters cannot deliver.

The main body of this thesis is structured as follows. Chapter 2 presents a literature review on the GEO regime, current maneuver-detection practices, low-thrust propulsion, pattern-of-life analysis, and physics-informed neural networks. Chapter 3 establishes the theoretical framework for the astrodynamics and neural network foundations upon which the methodology is built. Chapter 4 details the methodology, including dataset generation, the LSTM classifier architecture, and the PINN inverse-problem formulation. Chapter 5 presents the verification and validation of the individual components. Chapter 6 provides the analysis and comparative results, and Chapter 7 concludes with a discussion of the findings, limitations, and recommendations for future work.

2

Literature Review

2.1. Introduction

In the Geosynchronous Earth Orbit (GEO), the number of Resident Space Objects (RSOs) has been increasing, as in other orbital regimes. While the overall number of RSOs in GEO continues to grow, efforts towards space debris mitigation, particularly improved compliance with end-of-life disposal guidelines, are working to slow the rate of new debris accumulation in this concentrated orbital regime [8]. The particularities of the orbital regime are advantageous for many operators. Relative to a position on Earth, there is next to no movement of the satellite, thus ensuring that a data link between the ground station and the satellite is maintained at all times [4]. This comes with its own set of drawbacks, as it is a highly contested area of interest for telecommunication satellites, surveillance, and military applications. Examples include the documented Luch Olymp satellite, a Russian military satellite that performed rendezvous and proximity operations at distances well within the average separation of 200 kilometres [33]. Therefore, in Space Situational Awareness, it is critical to maintain the safety of this orbital regime to improve collision risk assessments, and thus, there is a catalogue of RSOs that describe their trajectories. What leads to complications in this catalogue is the tracking and data-association problem. When a satellite maneuvers, its post-maneuver state may no longer correlate with its pre-maneuver catalogue entry, potentially leading to misidentification as a new object or a loss of custody [27]. The latter is crucial for cataloguing activities, as it could lead to a loss of accuracy in the catalogue. Moreover, there has recently been an increase in the use of continuous low-thrust propulsion, as it is much more efficient and more flexible than a high-magnitude burn. The disadvantage is that these are much harder to detect over a short period than a high-magnitude burn, and consequently, if these maneuvers are not communicated properly, they could pose a threat to all other objects in the nearby space environment. This literature review will examine past and current methods for maneuver detection, as well as other aspects of astrodynamics in the GEO regime, and will then move towards the research question.

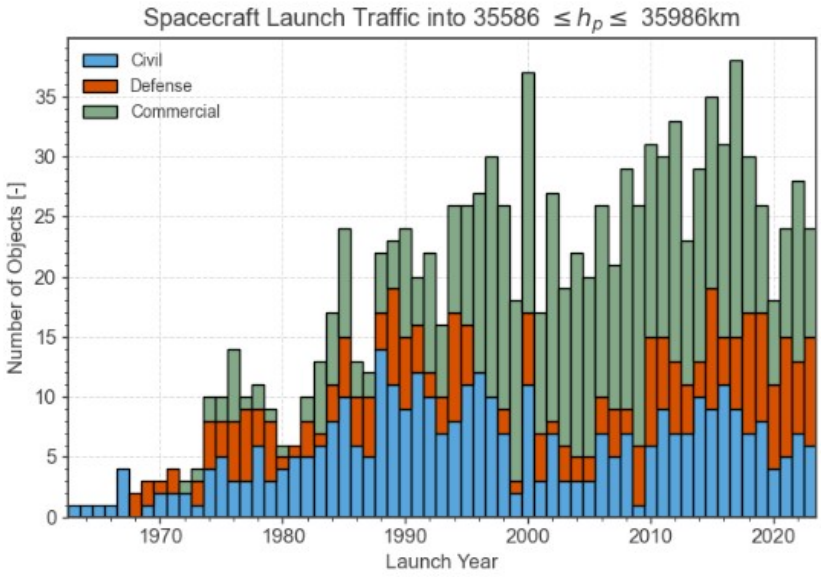


Figure 2.1: Evolution of Space Launches into GEO [13]

2.2. Geostationary Orbit and SSA

The GEO regime is unique among orbital regimes in that, at its characteristic altitude, $h_p = [35786]$ km, the orbital period matches the sidereal day on Earth. This regime is often extended to $h_p \in [35586, 35986]$ km to include GEO graveyard orbits, which are not in synchronous orbit [8]. Thus, as an observer on Earth, satellites in this regime remain at the same point in the sky at all times. This has many advantages, especially in the fields of communications, weather forecasting and national security [4]. Furthermore, due to the distance away from Earth, the orbital dynamics in this regime are quite different. At GEO altitudes, the relative importance of perturbations differs significantly from that in low Earth orbit (LEO). Perturbations that are dominant in LEO, such as atmospheric drag, effects of Earth's magnetic field, and lower-order gravitational harmonics, are greatly diminished. While solar radiation pressure has a similar magnitude across both regimes, its relative effect increases in GEO due to the reduction of other perturbing forces. Additionally, solar and lunar gravitational effects play a more significant role at GEO altitudes [37]. Furthermore, the effect of the non-spherical Earth is different in GEO than in other lower orbits, but it remains noticeable for the satellites. To counteract the effects of these perturbations, satellites in this regime periodically perform station-keeping maneuvers. For example, to counter the effect of the non-spherical Earth, satellites typically perform an East-West station-keeping burn bi-monthly, and a North-South burn annually [5]. Similar burns are performed at regular intervals to counteract the effects of other perturbations on the satellites in this regime. Therefore, maneuvers are performed periodically for each satellite, and their effects must be accounted for.

This is especially true given the orbit's congestion. As highlighted earlier, the unique properties of the orbit affect many operators, and thus, proper space traffic management is required to maintain safety in the orbit. To track satellites within the regime, various systems are operated by different organizations. The United States Space

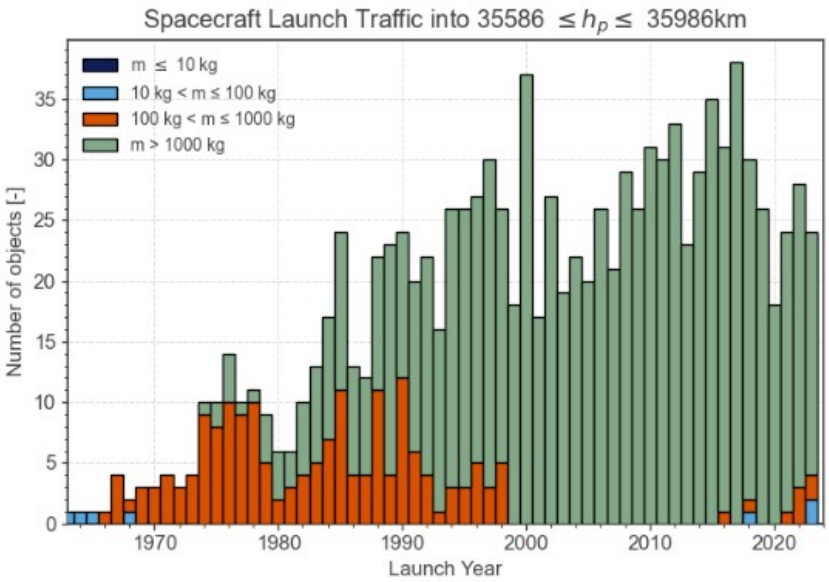


Figure 2.2: Evolution of Space Launches by Mass into GEO [13]

Surveillance Network (SSN), under the United States Space Force, tracks satellites and other RSOs in space [15][42]. This is achieved using a combination of optical and radar sensors worldwide. Part of this system, the Ground-Based Electro-Optical Deep Space Surveillance (GEODSS), catalogues accurate data on many man-made objects in orbit, usually in the format of Two-Line Elements (TLEs). This provides information on the object's Keplerian elements, along with other useful data for orbit propagation using the Simplified General Perturbations Models (SGP) [15]. Systems such as this can detect and catalogue objects in GEO as small as a basketball [42]. The European Space Agency (ESA) also operates its own Optical Ground Station (OGS), which can detect objects as small as 10 cm in diameter in GEO. Thus, current networks are more than capable of tracking satellites in this regime, especially when the vast majority of satellites launched into GEO have a mass of over 1000 kg [13].

Satellite operators frequently co-locate multiple satellites within the same slot in the orbit, which leads to the formation of satellite clusters [40]. These can be cooperative clusters, in which satellites share an orbital slot through coordinated operations, or non-cooperative clusters, which may arise from uncontrolled drift following a loss of satellite control, or, in rare cases, from deliberate, unauthorized proximity operations. In either case, they pose a significant security and operational risk [40]. Given the tight spacing between the satellites in a cluster, RSOs are more difficult to track. Thus, the need for a framework for the orbital regime. To address this, there are broad mandates from the International Telecommunication Union (ITU), which allocates orbital slots and manages the radio-frequency spectrum used for satellite communications to mitigate interference among neighbouring satellites within the regime. While most satellites typically operate at a distance of around 200 kilometres, some have gotten significantly closer, on the order of 10 kilometres, to neighbours [4]. Furthermore, in the GEO regime, it is impractical to bring a satellite to a lower orbit at the end of its life to burn up in the Earth's atmosphere. For one, this requires a significant amount

of fuel, which most operators are unwilling to carry on board, and, due to the weight of the satellites normally sent to GEO, it is not certain that they would fully break up and burn in the atmosphere. Thus, satellites in the regime are recommended by the Interagency Debris Coordination Committee to perform a maneuver at the end of life that raises the perigee to a minimum altitude of 235 km above GEO with a maximum eccentricity of 0.003 [13].

The GEO regime poses significant challenges due to its unique nature. Due to their distance from Earth, debris and inactive satellites in this regime have extremely long orbital lifetimes, on the order of millions of years [8]. Furthermore, as with all orbital regimes, collisions in GEO occur at high relative velocities, and given the average mass of the satellites in the regime, a total breakup would create a significant debris field. Additionally, given the high density of GEO with satellites, a large breakup event could trigger a chain reaction within the orbital band, underscoring the importance of safety in this regime.

2.3. Past and Current Practices in Maneuver Detection

When assessing collision risk or cataloguing RSOs, satellite maneuvers play a significant role [27]. A large number of new detections in the RSO catalogue come from maneuvers, which can lead to a general loss of catalogue accuracy [27]. Thus, detecting whether a maneuver has occurred and characterizing the maneuver's surrounding conditions are crucial for maintaining safety in space. Observations made from post-maneuver states that are correlated with any pre-maneuver state could fail, as they do not correlate to any existing objects. Thus, if this leads to a new object being entered into the catalogue, this would result in a duplication [27]. Therefore, accurate maneuver detection techniques are required to maintain catalogue accuracy.

There are different approaches to tackle this problem. Initially, research focused on detecting anomalies in TLEs, as these were publicly available [18]. Using historical TLE data, which contain orbital information at a particular epoch, non-perturbation-based maneuvers can be detected by observing the time history of TLEs. Given the right conditions, these methods can both detect maneuvers and determine the delta-velocity magnitudes with centimetre-per-second-level precision [18]. However, some of the more common techniques for maneuver detection use maneuver detection filters [43]. These employ orbit determination to detect whether a maneuver has been performed. At its core, this uses classical filters, such as the Extended Kalman Filter (EKF) or the Unscented Kalman Filter (UKF). However, when a target performs an unknown maneuver between the measurement windows, assuming that this maneuver is large enough, the propagators that the estimators are based on become less accurate and become overconfident due to the covariance becoming small [43]. While some methods can improve the robustness of these algorithms, such as covariance inflation and fading memory, in recent years, Multiple Model Filters have emerged as a promising filtering technique for tracking problems [27][43]. This technique is based on a family of basic filters that can be designed to model different aspects of the system's behaviour, together with a probabilistic mixing logic that combines the outputs of the basic filters.

More recently, the problem of maneuver detection has been formulated as part of an Optimal Control Problem, assuming that the satellite will typically perform maneuvers with the lowest propellant consumption [23]. This leads to the Optimal Control Based Estimator (OCBE), which uses an estimator similar to a Kalman Filter to achieve accurate tracking and maneuver reconstruction by first generating optimal control policies. Then, a maneuver-detection algorithm that relies on measurement residuals is used, and finally, a maneuver-reconstruction algorithm that estimates the mismodeled parameters for atmospheric drag and solar radiation pressure [23]. Other models and methods exist as well, based on advances in machine learning (ML), which use neural networks (NNs) as part of the maneuver detection algorithm but, there are severe limitations when it comes to data availability, as a large number of maneuvers are required to properly train the neural network [6][7][16][26]. Most of the research on the subject has focused on LEO, as more data are available, but it has been shown that the same principles can be applied to the GEO regime as well [31]. The difference being that station-keeping maneuvers were included as a priori information, and thus any maneuvers matching the expected timing or magnitude of routine station-keeping were not considered unknown maneuvers.

2.4. Low Thrust Maneuvers

Low thrust propulsion systems are becoming more widely adopted by satellite operators for a variety of applications, from station-keeping to geosynchronous transfers and interplanetary trajectories [1]. In contrast to more traditional chemical propulsion systems, these systems typically employ electric propulsion. This is because chemical systems have a limit on the energy available through the decomposition or combustion of molecular compounds. Electric propulsion systems, on the other hand, make use of energy from solar power or other more abundant sources to accelerate propellant to higher energies [1]. Continuous low-thrust propulsion provides a small, continuous thrust over a long period to change a spacecraft's trajectory. Compared to impulse-based maneuvers, these methods are significantly more fuel-efficient due to their high specific impulse and, as such, are ideal for missions with longer range and lifetime. Furthermore, due to the continuous nature of the propulsion, it can be fine-tuned and controlled continuously, and, finally, it is significantly more cost-effective than impulse-based propulsion, owing to its efficiency and lower required mass[10]. As mentioned earlier, it has been used extensively for station-keeping in GEO and for other orbital maneuvers, such as transferring from a Geosynchronous Transfer Orbit (GTO) to GEO.

As its adoption increases, it becomes necessary to detect whether a satellite has performed a maneuver using low-thrust propulsion. Unlike impulsive thrust maneuvers, a satellite maneuvering with low thrust can be tracked throughout the event [17]; however, it is difficult to distinguish it from natural perturbations, especially with traditional maneuver-detection methods that rely on residuals. Thus, some methods adapt maneuver detection algorithms for high thrust maneuvers that use Batch Least Squares, or Kalman Filters, to determine maneuver characteristics [17], and more recently, use surveillance radar observations over a long period and use filters that rely on Thrust Fourier Coefficients (TFCs), which represent the unknown thrust vector as a truncated

Fourier series to parameterize the continuous maneuver profile over the observation arc [19].

2.5. Pattern of Life Analysis

Satellite Patterns of Life (PoL) are the descriptions of on-orbit behaviour made up of sequences of both natural and non-natural behavioural modes. This includes routine station-keeping, on-orbit maneuvers, and uncontrolled motion in GEO [37]. While these overall behaviours of a satellite can be determined by inspecting the longitudinal position history, the specific patterns associated with these behavioural modes are much more difficult to identify and characterize [32]. Learning these patterns greatly helps determine the “nominal” orbit of GEO satellites, thereby establishing a baseline for attributing certain satellite movements.

To learn these patterns, researchers have turned to ML techniques. Early methods characterized the problem as an unsupervised learning problem, meaning that the training set has no labels and the algorithm must discover the underlying pattern with no prior knowledge. The method used to solve this problem is k-means clustering on time-series longitudinal positions derived from TLEs [32]. The results demonstrate an effective method for characterizing behavioural patterns, with clusters that describe station-keeping and satellite drift. However, within these clusters, it is difficult to infer the magnitude, direction, and, especially, the propulsion type of the satellites. Similarly, for satellite drift, insights on drift rates or the use of electric propulsion are not discernible [32].

To this end, the Satellite Pattern-of-Life Identification Dataset (SPLID) supports future studies on the PoLs, enabling comparisons of different AI approaches to the problem. SPLID consists of synthetic space object data and true space object data generated from Vector Covariance Messages and high-accuracy ephemerides [37]. This dataset, which provides multivariate time-series data including longitude and drift rate, was used in a competition, from which additional ML methods to characterize the PoL of GEO satellites emerged. The results of this competition highlighted that different models had “varying degrees of success” because they were better suited for different tasks. For example, gradient-boosted trees proved highly effective at the classification task by learning thresholds from physics-derived features. Conversely, deep learning models such as the Convolutional Neural Network (CNN), which can exploit distinct patterns across maneuver types, were more adept at pattern recognition [38]. Despite the overall performance, generalizing to real-world data, including TLE-derived ephemerides, was challenging due to the greater variability and complexity. These could be mitigated to some extent by employing additional data augmentation techniques to better balance the dataset [38]. Thus, ongoing research in the field aims to better understand and characterize the PoL of GEO satellites to improve predictability and tracking.

2.6. Physics Informed Neural Networks

Physics-informed neural networks (PINNs) were developed for solving forward and inverse problems governed by parametric differential equations [35]. This is then reg-

ulated by the physics of the problem. In other words, PINNs leverage the known physical equations to better inform the learning process. This has recently been shown to be useful for orbit determination using known differential equations, such as the circular restricted three-body problem, achieving accuracies comparable to classical least-squares methods [36]. However, it has the advantage that it does not require a precise initial guess and removes integration as a source of numerical error. Further work is underway to regularize the training process so that it can be used to predict cislunar orbit determination or within the GEO belt [3].

Due to the unique nature of PINNs, they can account for unmodelled dynamics by enforcing the known governing equations as constraints. Any discrepancy between the network's output and the enforced physics is then attributable to unmodelled forces, such as continuous thrust, allowing the network to isolate and reconstruct these unknown accelerations. Furthermore, PINNs have been applied to construct trajectories that satisfy optimization problems with varying constraints, particularly in scenarios characterized by higher degrees of non-linearity [10].

PINNs, as a whole, are relatively new in ML and especially in astrodynamics. As such, PINNs still face limitations that hinder their viability in many applications. PINNs, like other machine learning methods, suffer from overfitting dynamics during training and require specialized methods to mitigate it [3]. Furthermore, hyperparameter tuning is complex, and training times are typically long. The specific strategies employed to address these challenges in this thesis are discussed in Chapter 4.

2.7. Limitation in Current Practices

Thus, to summarize the limitations across the fields studied in this review, there is a need for higher-fidelity tracking and awareness within the GEO belt due to its high concentration of satellites. Situations such as the inspector satellite scenario highlight the need for proper management of space traffic, as spy or uncooperative satellites can get very close to other satellites within the belt. Maneuver detection is needed to predict whether these situations arise, but with the advent of low-thrust maneuvers, it is difficult to determine whether a maneuver is due to station-keeping or to a non-periodic maneuver. This distinction is critical because routine station-keeping can be accounted for in catalogue predictions, whereas non-periodic maneuvers may indicate unauthorized proximity operations, collision-avoidance maneuvers, or orbital transfers that pose a direct threat to neighbouring assets.

ML techniques are useful for solving such problems, and for the GEO belt, ML models can predict the PoLs of satellites. However, these models exhibit low transferability from synthetic to real-world datasets due to the greater variability in the data sources. A solution recently introduced to the field of SSA uses PINNs to solve nonlinear differential equations, such as those arising in orbit determination [36]. However, these approaches were 'physics-informed' primarily through feature engineering, using features derived from physics, such as drift rate and its derivatives. They did not incorporate the governing differential equations of orbital motion directly into the model's

loss function or architecture. This distinction is critical. Despite their performance on synthetic data, these models struggled to generalize to real-world data [38]. This suggests that these pattern-matching-based models (CNNs, LSTMs, and gradient-boosted trees) are not robust to the greater variability and complexity of real-world observations. While these issues could be alleviated to some extent by employing additional data augmentation techniques, this points to a fundamental gap [38]. Thus, ongoing research in the field aims to better understand and characterize the PoL of GEO satellites for improved predictability and tracking, particularly by developing models that generalize better than the current state-of-the-art.

2.8. Research Question

From the limitations identified in this review, the following primary research question is formulated:

Main Research Question

What are the limitations of standard maneuver detection techniques for detecting low-thrust maneuvers, and how can these limitations be overcome by integrating orbital dynamics into a data-driven model?

To systematically address this question, the following sub-questions are defined:

Sub-question 1

What accuracy does a machine learning model need to achieve to be adequate for maneuver detection in GEO using low-thrust orbit control?

Sub-question 2

What are the key limitations of a baseline physics-agnostic model in identifying low-thrust maneuvers from synthetic data?

Sub-question 3

How can a PINN be formulated to specifically model continuous low-thrust trajectories and the associated orbital perturbations in GEO?

Sub-question 4

How does the performance of the PINN-based PoL model compare to the baseline model in terms of detection accuracy, robustness to data noise, and transferability from synthetic to real-world data?

3

Theoretical Framework

The goal of this chapter is to provide a strong foundation upon which the thesis will be built. It will focus on two main sections. The astrodynamics of GEO, low-thrust maneuvers, and NNs, with a more detailed examination of PINNs.

3.1. Astrodynamics of GEO and Low-Thrust Maneuvers

This section details the astrodynamics and physics of orbits in GEO and low-thrust maneuvers. The astrodynamical framework will be presented first, followed by the equations governing low-thrust maneuvers. It will be seen that, for a PINN, understanding the physical equations to be optimized is critical to the problem.

3.1.1. Reference frames

Unless otherwise specified, all equations are presented in the Earth-Centred Inertial (ECI) reference frame. This reference frame is a right-handed system with Earth's centre of mass at its origin. The $\hat{\mathbf{x}}_\mathbf{I}$ axis is pointed towards the mean equinox at J2000 along the equatorial plane. The $\hat{\boldsymbol{z}}_\mathbf{I}$ axis is defined as the direction of Earth's mean rotational pole at J2000, and finally, $\hat{\mathbf{y}}_\mathbf{I}$ is orthogonal to $\hat{\mathbf{x}}_\mathbf{I}$ and $\hat{\boldsymbol{z}}_\mathbf{I}$ to complete the reference system.

3.1.2. Two-Body Problem

At its simplest, the motion of a spacecraft about a body is modelled by the two-body problem. A spacecraft with mass m rotates around a body with mass M under the gravitational influence of the central body with a gravitational parameter μ. The assumption is that in this scenario, $M \gg m$, thus the mass of the spacecraft can be considered negligible. In the most basic form of the two-body problem, it is also assumed that the central body is a point mass and that the only force acting on the spacecraft is the gravitational attraction of M on m. This relationship is modelled by equation 3.1, which gives the resulting acceleration experienced by the spacecraft due to the two-body problem.

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} \tag{3.1}$$

In this equation, $\mathbf{r}$ is the position vector of the spacecraft, and thus $\ddot{\mathbf{r}}$ is the resulting acceleration vector of the spacecraft, while r is the magnitude of the position vector.

The solution to this equation yields a closed-form conic section, but for high-precision orbit modelling, perturbations from various sources must also be accounted for. For a spacecraft in GEO, these perturbations are dominated by the non-spherical Earth, solar radiation pressure and third-body effects from the Sun and Moon. Equation 3.2 details the difference with an extra acceleration term $\mathbf{a}_{\text{pert}}$ added to signify the accelerations caused by the external factors.

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_{\text{pert}} \tag{3.2}$$

This term, $\mathbf{a}_{\text{pert}}$ is the sum of the accelerations caused by the perturbations, and the following sections will contextualize them.

3.1.3. Perturbations Related to the Non-Spherical Earth

In the previous section, the central body was assumed to be a point mass. This simplification is insufficient for high-precision orbital dynamical analysis, as the Earth is more accurately an oblate spheroid with a non-uniform mass distribution. This complex gravitational field can be modelled as a spherical harmonic expansion. The terms of this expansion are grouped by their geometric properties. The zonal harmonics (where $m = 0$) describe variations with latitude and are denoted by J_n coefficients. The most dominant of all perturbation terms is the second zonal harmonic, J_2, which accounts for Earth's equatorial bulge. The perturbing acceleration from the J_2 term is given by equation 3.3.

$$\mathbf{a}_{J_2} = -\frac{3\mu J_2 R_E^2}{2r^5}\left[\left(1 - 5\frac{r_z^2}{r^2}\right) r_x \hat{\mathbf{x}}_{\mathbf{I}} + \left(1 - 5\frac{r_z^2}{r^2}\right) r_y \hat{\mathbf{y}}_{\mathbf{I}} + \left(3 - 5\frac{r_z^2}{r^2}\right) r_z \hat{\mathbf{z}}_{\mathbf{I}}\right] \tag{3.3}$$

In this equation, J_2 is the constant coefficient for the second zonal harmonic, R_E is the mean equatorial radius of Earth, $[r_x, r_y, r_z]$ are the components of the spacecraft's position vector $\mathbf{r}$, and r is its magnitude. The primary effect of the J_2 perturbation is a long-term secular precession of the orbit, primarily affecting the Right Ascension of the Ascending Node (RAAN) and the Argument of Perigee.

While J_2 is the largest term, the "East-West" drift relevant to GEO satellites is caused by the sectoral harmonic $J_{2,2}$ (where $m = n$) which models the ellipticity of Earth's equator. This creates the so-called GEO gravity wells, two stable and two unstable equilibrium longitude points. The tesseral harmonics (where $m \neq 0$ and $n \neq 0$) also contribute to longitudinal variations, though to a lesser extent. Together, these terms create a non-uniform gravitational field that drives longitudinal drift. To counteract this, GEO satellites must perform regular East-West station-keeping maneuvers. These maneuvers, which are a key part of their PoL, are typically performed every 2-3 months with a small Δv of around 0.27 m/s [5].

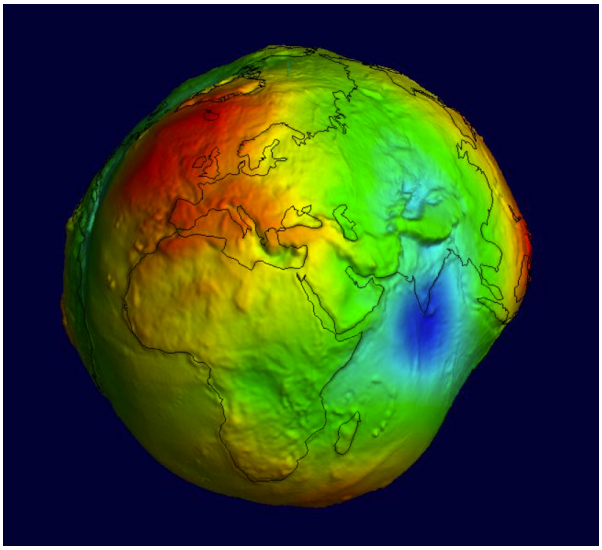


Figure 3.1: The Earth's geoid, representing an equipotential surface of its gravity field, significantly deviates from a perfect sphere (exaggerated for visualization) [24].

3.1.4. Perturbations Related to Third Bodies

Spacecraft in GEO also experience gravitational effects from so-called "third bodies". The largest contributors to this are the Sun and Moon, but other planets, such as Venus, Mars, and Jupiter, also have an effect, albeit much smaller. Because the Sun and Moon are in orbits inclined relative to the Earth's equatorial plane, their gravitational pull induces significant out-of-plane acceleration, thereby causing the satellite's orbital inclination to drift. The relative acceleration due to these effects is given in equation 3.4.

$$\mathbf{a}_{3b} = \mu_j \left(\frac{\mathbf{r}_j - \mathbf{r}_i}{\|\mathbf{r}_j - \mathbf{r}_i\|^3} - \frac{\mathbf{r}_j}{r_j^3} \right) \tag{3.4}$$

In this equation, all vectors are in the ECI frame: the subscript i denotes the spacecraft and j denotes the third body. Thus, μ_j is the gravitational parameter of the third body, $\mathbf{r}_i$ is the position vector of the spacecraft, and $\mathbf{r}_j$ is the position vector of the third body in the ECI frame. This equation gives the relative acceleration by subtracting the acceleration of the Earth (caused by the third body) from the direct pull of the third body on the spacecraft. Some calculated values for the magnitude of these accelerations are shown in Table 3.1. Based on these results, the magnitude of acceleration from the Sun and Moon is at least five orders of magnitude greater than that of the other planets. Therefore, only these two sources will be considered as significant third-body perturbations.

Celestial Body	Magnitude of Acceleration (m/s^2)
Sun	3.34×10^{-6}
Moon	3.62×10^{-6}
Venus	1.08×10^{-11}
Jupiter	1.13×10^{-11}
Mars	3.04×10^{-13}

Table 3.1: Maximum third-body perturbations on a GEO satellite from different celestial bodies. Values are calculated for a simplified scenario in which the Sun is in quadrature, and all other bodies are in conjunction.

3.1.5. Perturbations Related to Solar Radiation Pressure

Another dominant non-gravitational perturbation affecting spacecraft in GEO is Solar Radiation Pressure (SRP). This is a force exerted on the spacecraft as photons from the Sun are reflected or absorbed. This continuous “push” away from the Sun causes secular and long-periodic variations in the satellite’s orbital elements, most notably its eccentricity, which must be corrected with station-keeping maneuvers. The acceleration from SRP is commonly modelled using the ”cannonball model” shown in Equation 3.5 [9]:

$$\mathbf{a}_{\text{SRP}} = \frac{W_{\odot} C_R}{c} \frac{A}{m} \left(\frac{R_0}{r_{\text{sun}}} \right)^2 \hat{\mathbf{r}}_S \tag{3.5}$$

Here, $W_{\odot}$ is the average solar radiation flux at 1 AU (approx. $1361 W/m^2$), C_R is the spacecraft’s dimensionless coefficient of reflectivity, A/m is the area-to-mass ratio and r_{sun} is the instantaneous distance from the Sun to the spacecraft.The $\left(\frac{R_0}{r_{\text{sun}}}\right)^2$ term is a critical scaling factor. Due to the eccentricity of Earth’s annual orbit around the Sun, this term accounts for an approximate $\pm 3.4\%$ variation in the SRP force throughout the year. The vector $\hat{\mathbf{r}}_S$ is the unit vector from the Sun to the spacecraft, expressed in the ECI reference frame. While a spacecraft in GEO experiences sunlight for most of the year, it enters Earth’s shadow (eclipse) at the equinoxes, causing this force to drop to zero. A high-fidelity model would include a shadow function; however, in this framework, only the sunlit case will be considered. The main source of error from the SRP model comes from the reflectivity and the area-to-mass ratio terms, as there are not simple constants in practice. A spacecraft’s cross-sectional area and its reflective properties change over time depending on its orientation. Thus, more accurate models exist, such as the N-plate model or high-fidelity approaches, but they are not considered because they stray from the research questions [9].

3.1.6. The Full Dynamical Model

The above sections present the perturbing accelerations affecting a spacecraft in GEO. There are other sources; however, these are all significantly smaller than the ones presented and thus will be considered negligible for low-thrust maneuver detection. The perturbing accelerations provided are the most useful as they dictate the PoL of a GEO

spacecraft, namely the reasons for station-keeping. The perturbing effects caused by the non-spherical Earth require East/West station-keeping maneuvers of around 0.27 m/s every 2-3 months, while the third body effects require North/South station-keeping maneuvers of around 45-50 m/s every year [5]. The third major perturbing acceleration, SRP, introduces “noise” into the dynamical model due to uncertainty in the cross-sectional area-to-mass ratio and the coefficient of reflection. As explained in section 3.1.5, these factors change depending on the orientation of a satellite and are a cause of high uncertainty in real-world data. Thus, the full form of equation 3.2 is shown in equation 3.6:

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_{J2} + \sum \mathbf{a}_{3b} + \mathbf{a}_{\text{SRP}} \tag{3.6}$$

3.1.7. Low Thrust Maneuvers

In addition, to detect and quantify low-thrust acceleration, an additional acceleration term must be added. Synthetic data will have a_{SRP} as a deterministic signal, while the uncertainty in real-world data could have this acceleration as “noise” that must be distinguishable from the acceleration coming from low-thrust maneuvers. This changes the equation 3.6 to the following:

$$\ddot{\mathbf{r}} = \underbrace{-\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_{J2} + \sum \mathbf{a}_{3b} + \mathbf{a}_{\text{SRP}}}_{\mathbf{a}_{\text{natural}}} + \mathbf{a}_T(t) \tag{3.7}$$

This shows that the a_{natural} represents all perturbing forces, while $\mathbf{a}_T(t)$ represents the unknown thrusting maneuver that is to be detected.

Low-thrust maneuvers involve a spacecraft using a propulsion system, typically electric propulsion, which applies a sustained, small thrust over a long duration. The difference between a low-thrust spiral and an impulsive Hohmann transfer is shown in Figure 3.2, showing the gradual increase in the orbital semi-major axis with the low-thrust spiral comapred to the instantaneous increase resulting from an impulsive burn. This results in a very small instantaneous acceleration, on the order of the uncertainty in the perturbations caused by SRP. This makes it difficult to detect using traditional filter-based methods that rely on detecting large, sudden residuals [17].

Mathematically, this unknown acceleration vector is modelled in equation 3.8 [44]:

$$\mathbf{a}_{\text{T}}(t) = \frac{T(t)}{m(t)}\hat{\mathbf{u}}(t) \tag{3.8}$$

where $T(t)$ is the thrust magnitude, $m(t)$ is the instantaneous mass of the spacecraft, and $\hat{\mathbf{u}}(t)$ is the unit vector of the thrust direction. Unlike the large, impulsive Δv maneuvers used for nominal station-keeping, this continuous force results in a slow, gradual spiral trajectory.

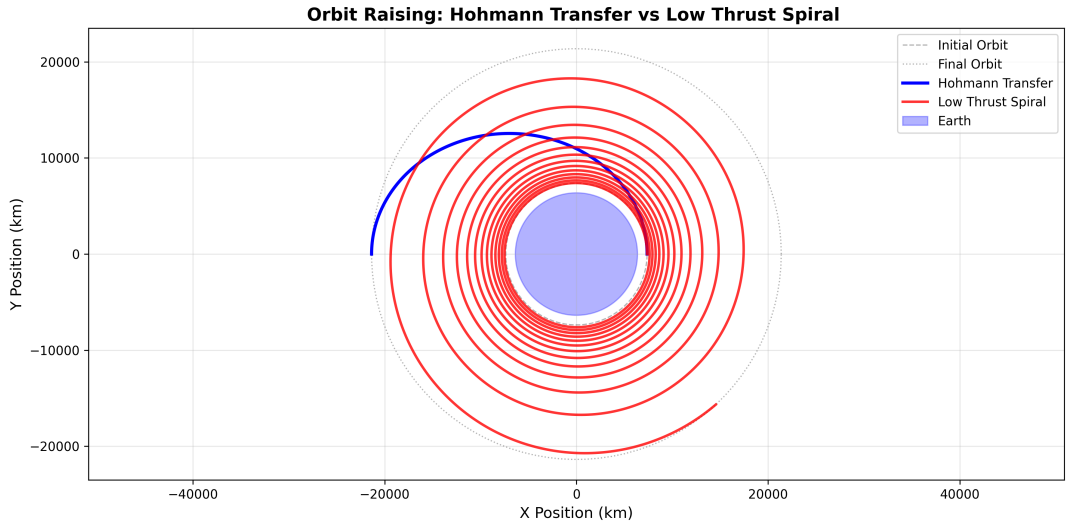


Figure 3.2: Comparison of an impulsive Hohmann transfer (blue) and a continuous low-thrust spiral (red) for raising a spacecraft's orbit. While an impulsive maneuver achieves orbit change through discrete, high-thrust burns, low-thrust propulsion systems produce continuous, small acceleration, resulting in a gradual, spiralling trajectory.

When observed with sparse, noisy data, this spiral is incredibly difficult to distinguish from natural orbital drift or mismodeled "noise" in $\mathbf{a}_{\text{natural}}$. Figure 3.3 shows how a mismodeled SRP A/m ratio affects the magnitude of the perturbed position at GEO. Thus, to accurately detect a low-thrust maneuver, noisy, sparse spacecraft observations must be processed by a model to distinguish between what is required and what is noise.

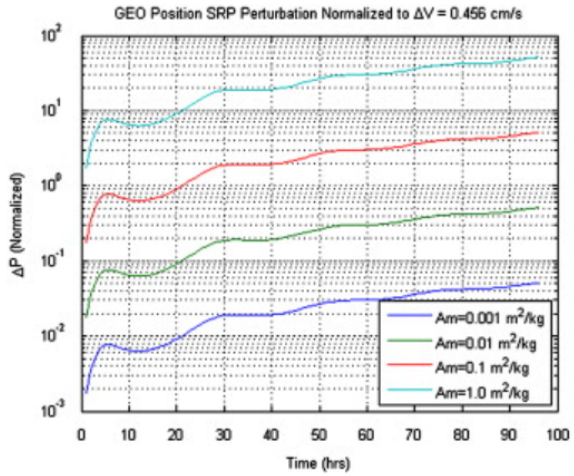


Figure 3.3: Maneuver normalized solar radiation pressure perturbations on a GEO object vs. time [17].

3.2. Neural Networks

Within engineering, it is a common task to find a function that describes a set of observed data points. This is often accomplished with methods such as least-squares

regression to fit a polynomial $y = f(x)$ to the data. A NN is a biologically inspired model that can be understood in a similar way: it is a highly flexible, powerful computational tool for function approximation. NNs are part of the field of ML and are exceptionally well-suited to learning the "rules" of highly complex, non-linear systems from data, even when the underlying function is unknown.

3.2.1. Architecture

Neural networks are often visualized as a whole; however, they are composed of individual neurons or nodes that serve as computational units performing mathematical operations. This unit is a form of linear regression followed by a non-linear "clipping" function. A single neuron takes n inputs ($\mathbf{x} = [x_1, ..., x_n]$). It performs two steps:

1. **Step 1: Weighted Sum**. It computes a weighted sum of its inputs, just like in linear regression. Each input x_i has a corresponding weight w_i. The neuron adds a bias b. This bias is a "tunable" intercept that shifts the function.

$$z = (w_1x_1 + w_2x_2 + ... + w_nx_n) + b \quad \text{or, in vector form:} \quad z = \mathbf{W} \cdot \mathbf{x} + b \tag{3.9}$$

2. **Step 2: Activation Function.** The result z is then passed through a non-linear activation function $\sigma(z)$. This is the most critical part. Without this nonlinearity, a 100-layer deep network would mathematically collapse into a single linear function. Common activation functions include:

   - **ReLU (Rectified Linear Unit):** $\sigma(z) = \max(0, z)$. This is the modern default. It is computationally simple and helps solve key training problems.
   - **Sigmoid (Logistic)**: $\sigma(z) = 1/(1 + e^{-z})$. This constrains any input value to the range [0,1].
   - **Tanh (hyperbolic Tangent)**: $\sigma(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ Like the sigmoid function, it constrains the input to the range [-1,1]. This zero-centred property often helps the network's optimization converge more efficiently.

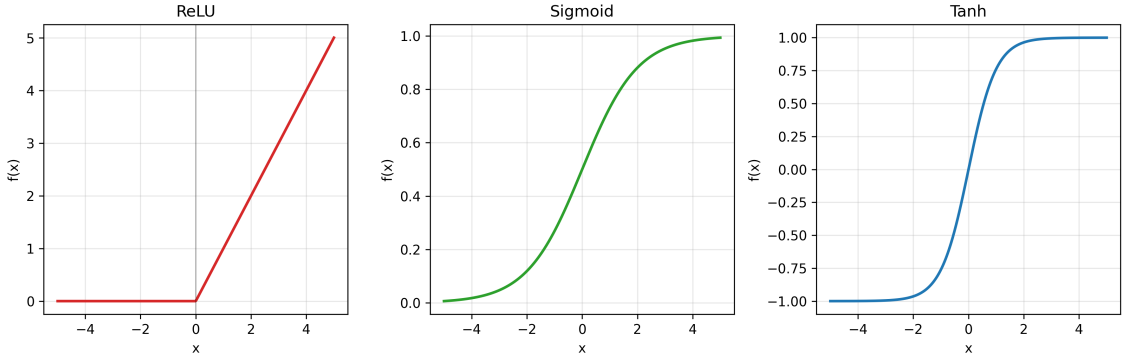


Figure 3.4: Visualization of the Different Activation Functions

Thus, the complete mathematical operation for a single neuron, mapping its input vector $\mathbf{x}$ to its scalar output a, is given by:

$$a = \sigma(z) = \sigma(\mathbf{W} \cdot \mathbf{x} + b) \tag{3.10}$$

At its simplest form, a feedforward neural network consists of many of these neurons, spread in different layers:

- **Input Layer:** Receives the raw input vector, for example, time t
- **Hidden Layers:** One or more layers of neurons, where the network's high-level feature learning and non-linear understanding of the function occurs
- **Output Layer:** The final layer that produces the prediction, for example, the three components of a velocity vector $\dot{\mathbf{r}}(t)$.

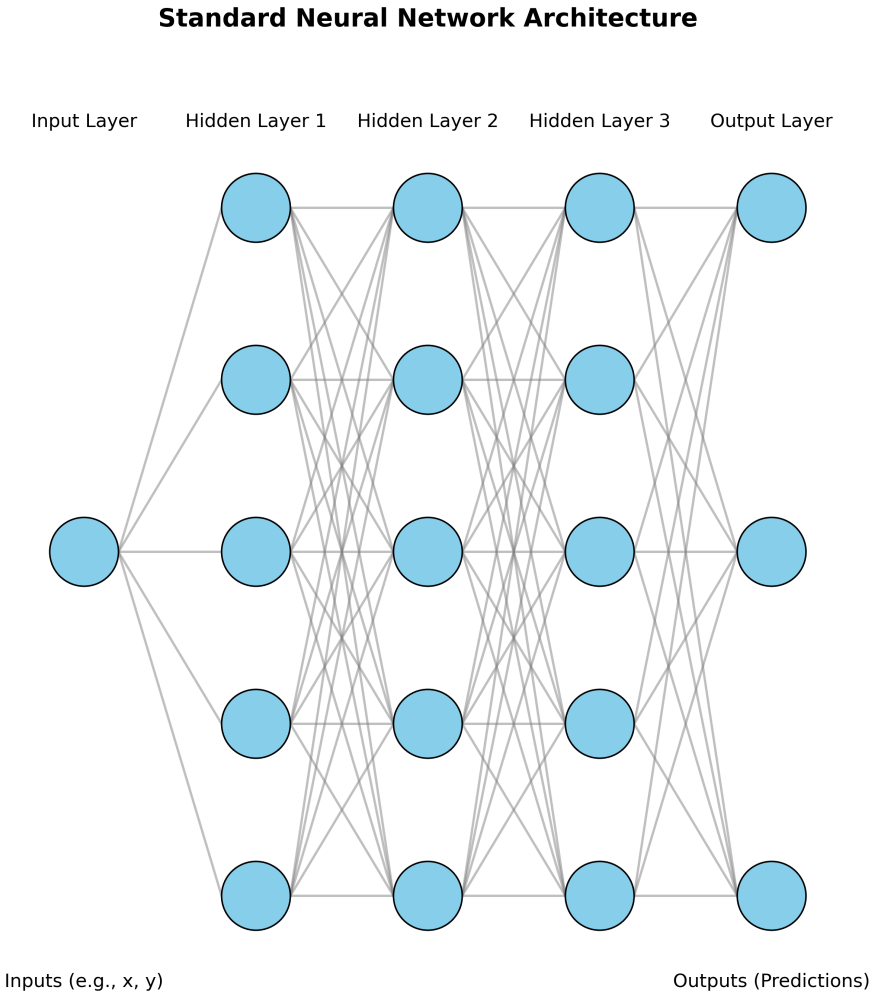


Figure 3.5: Simple Feedforward Neural Network Architecture with 3 Hidden Layers

The weights W and biases b of all neurons constitute the complete set of tunable elements, or coefficients, that the network learns. A full network is a composite function in which the output of one layer serves as the input to the next. The output of a layer l can be defined as $\mathbf{a}^{(l)}$. This is a vector whose elements are the outputs of the neurons in that layer. The input $\mathbf{x}$ is the activation of layer 0, or $\mathbf{a}^{(0)}$. For a simple network, with a hidden layer l and an output layer $l+1$ the full equation mapping the input $\mathbf{x}$ to the

final prediction $\hat{\mathbf{y}}$ is given by the following set of equations:

$$\mathbf{a}^{(l)} = \sigma^{(l)}(\mathbf{W}^{(l)}\mathbf{x} + \mathbf{b}^{(l)}) \tag{3.11}$$

Equation 3.11 shows the calculation of the hidden layer l, where the input $\mathbf{x}$ is given to the hidden layer. Each neuron in the layer computes its activation, yielding the activation vector $\mathbf{a}^{(l)}$. For the next layer, $l+1$, the output $\mathbf{a}^{(l)}$, becomes the input ($\mathbf{x}^{(l)} = \mathbf{a}^{(l)}$. Thus, the full equation can be written as:

$$\hat{\mathbf{y}} = \sigma^{l+1}((\mathbf{W}^{(l+1)}\mathbf{x}^{(l)} + \mathbf{b}^{(l+1)}) \tag{3.12}$$

This entire process is known as forward propagation and is repeated through all layers within the neural network.

3.2.2. The Learning Process

"Learning" is the process of finding the optimal values for all the weights $\mathbf{W}^{(l)}$ and biases $\mathbf{b}^{(l)}$ to best map inputs to outputs. This is framed as a mathematical optimization problem. It consists of a loss function and an optimization procedure. The loss function (L) is defined to measure the accuracy of the network's predictions. In regression problems, the Mean Squared Error (MSE) is a common loss function.

$$L_{\text{data}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_{\text{true}}^{(i)} - \hat{\mathbf{y}}_{\text{pred}}^{(i)})^2 \tag{3.13}$$

This is the optimal loss function under the assumption that the observational errors are normally distributed, a common assumption in many physical systems. The optimization problem that follows aims to find the set of all $\mathbf{W}$ and $\mathbf{b}$ that minimize L_{data}. This is achieved using an optimizer, commonly gradient descent. The algorithm is quite complex, but the core is the update rule:

$$w_{\text{new}} = w_{\text{old}} - \eta \cdot \frac{\partial L}{\partial w} \tag{3.14}$$

The network calculates the derivative of the loss with respect to a weight. This vector is multiplied by a small step-size η called the learning rate. Thus, conceptually, a derivative is a vector that points in the direction of the greatest error. At each step, defined by the learning rate, the network moves the weight away from this direction, slightly reducing the error. This process is repeated thousands or millions of times. The algorithm used to compute these millions of derivatives is called backpropagation, which applies the chain rule of calculus backward through the network. Learning can be performed in various ways. Batch Gradient Descent calculates the gradient for the entire dataset. This is accurate but incurs a heavy computational cost and is infeasible for large datasets. Stochastic Gradient Descent computes the gradient for a single point at a time. This is quite fast, but training can be unstable due to "noisy" updates. The standard approach in modern learning is Mini-Batch Gradient Descent, which computes the gradient from a small, random "batch" of data, balancing stability and computational efficiency.

In practice, standard gradient descent is often inefficient. Modern approaches use adaptive optimizers such as Adam (Adaptive Moment Estimation), which compute an individual learning rate for each weight in the network. This is done with two key concepts:

1. **Momentum**: It keeps an exponentially decaying "running average" of past gradients (the first moment). This is like a ball rolling downhill—it builds up momentum, allowing it to "roll through" small local minima and accelerate faster in the correct, consistent direction.
2. **Adaptive Learning Rates**: It also keeps an exponentially decaying "running average" of the squares of past gradients (the second moment). This term helps to scale the learning rate for each weight. Weights with large, consistent gradients will have their learning rate effectively reduced, whereas weights with sparse or small gradients will receive larger updates.

By combining these two concepts, the Adam optimizer converges much faster than standard gradient descent due to momentum, while also being more stable and robust due to the adaptive learning rates.

3.2.3. Overfitting and Generalization

This standard data-driven approach is powerful, but it is fundamentally a "black box." It has no a priori knowledge of the system's underlying physics. It simply finds the best mathematical fit to the data provided. This leads to overfitting, which occurs when the network becomes too flexible. It begins to "memorize" training data, including random noise, rather than learning the true underlying signal. An overfitted model will perform well on the data it was trained on, but poorly on new, unseen data. This is the generalization gap. This limitation is precisely why more sophisticated models are required, leading to a physics-informed approach.

3.3. Physics Informed Neural Networks

Physics-informed neural networks (PINNs) are an advanced architecture that mitigates overfitting. A PINN is a "grey-box" model that bridges the gap between data-driven machine learning and classical physics-based modelling. The idea is not only to train the network to fit the data, but also to enforce the laws of physics simultaneously. This is done by keeping the same feedforward network architecture, but instead changing the loss function fundamentally.

3.3.1. Hybrid Loss Function

A PINN's learning is driven by a hybrid loss function that is a weighted sum of two distinct components:

$$L_{\text{total}} = L_{\text{data}} + \lambda_{\text{physics}} L_{\text{physics}} \tag{3.15}$$

Here, λ_{physics} is a weighting hyperparameter that balances the importance of the two terms. Data loss (L_{data}) is the exact same loss function as in a standard network (3.2.2). It is the Mean Squared Error (MSE) between the network's predictions $\hat{\mathbf{r}}(t_i)$ and the known, sparse, and potentially noisy observation data $\mathbf{r}_{\text{obs}}$.

$$L_{\text{data}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{r}_{\text{obs}}(t_i) - \hat{\mathbf{r}}(t_i))^2 \tag{3.16}$$

The objective of this term is to ensure that the network solution is grounded in reality. The second part of this loss, (L_{physics}) or the physics loss, is the core innovation of the PINN. This term measures how well the network's output obeys the governing differential equations. Within the context of astrodynamics, the full spacecraft equation of motion, equation 3.6, is known as the physics residual as seen in equation 3.17. The network predictions will be physically valid only if they make the residual equal to 0. Automatic differentiation (AD) is then applied to this neural network because the composite function in section 3.2.1 is infinitely differentiable. Thus, using automatic differentiation, the exact, analytical derivatives of the network's output $\hat{\mathbf{r}}(t)$ with respect to its input t can be calculated, implying this is not a numerical approximation. AD applies the chain rule exactly through every layer of the network to find the true derivatives $\dot{\hat{\mathbf{r}}}(t)$ and $\ddot{\hat{\mathbf{r}}}(t)$. Thus, this is advantageous as it removes numerical integration as a source of error.

$$f_{\text{pred}} = \ddot{\hat{\mathbf{r}}}(t) - \mathbf{a}_{\text{natural}}(t, \hat{\mathbf{r}}(t), \dot{\hat{\mathbf{r}}}(t)) \tag{3.17}$$

Finally, the physics loss is computed everywhere, including regions with no data. Random points are created in time, t_c, within the training window. This is known as collocation points. The physics loss is thus the MSE of the residual f_{pred}, calculated over all these collocation points.

$$L_{\text{physics}} = \frac{1}{M}\sum_{i=1}^{M} \| f_{\text{pred}}(t_c^{(i)}) \|^2 \tag{3.18}$$

As with data loss, the physics loss guides the network's solution toward a physically plausible trajectory across the vast empty spaces between real data points. It is important to note that while AD eliminates numerical integration as a source of error, the physics loss in equation 3.18 functions as a soft constraint. The residual is only evaluated at a finite set of collocation points, and there is no mechanism that forces it to be exactly zero everywhere. The network is thus driven toward physically plausible solutions in the vicinity of these points, but strict satisfaction of the governing equations across the entire domain is not guaranteed.

3.3.2. Learning Process

While Adam is effective for initial training and handles noisy gradients well, PINNs commonly employ the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm for convergence. L-BFGS is a quasi-Newton method that approximates the inverse Hessian matrix using only the most recent gradient evaluations, rather than storing the full matrix as in standard BFGS. This makes it memory-efficient for problems with many parameters while still exploiting curvature information to take more informed steps towards the minimum. Unlike first-order methods such as Adam, which use fixed adaptive learning rates, L-BFGS performs a line search at each step

to determine the optimal step size, enabling superlinear convergence near a minimum [21]. For PINNs, this typically means L-BFGS can extract several additional orders of magnitude of precision from a solution that Adam has already brought close to the minimum, making it the preferred final-stage optimizer. When the optimizer minimizes the L_{total}, it is forced to find a set of network weights and biases that satisfy both conditions simultaneously. This creates a "push and pull" on the network's solution:

- The "Pull" of Data: L_{data} "pulls" the network's prediction toward the known, sparse observation points. It acts as an "anchor" to reality.
- The "Push" of Physics: L_{physics} "pushes" the network's solution away from any physically implausible trajectory. It serves as a powerful physics-based regularizer that enforces the governing laws throughout the vast empty spaces between data points.

This "grey-box" approach, which leverages known physical laws to guide the learning process, fundamentally solves the "black-box" limitations. Specifically, it allows the network to generalize from sparse data by guiding the physics at all collocation points and interpolating in a physically sound manner. It is also able to handle noisy data as the physics loss term will "push" back and penalize any solution that tries to fit that noise, making the network inherently more robust.

4

Methodology

4.1. Current State of Practice in Maneuver Detection

The detection of maneuvers is fundamentally an anomaly detection problem. Historically, the tracking of RSOs was a cataloguing process: a deterministic process of maintaining state vectors for objects that were largely assumed to be passive or operating on predictable schedules. Maneuver detection is one of the methods in the orbital domain that provides the essential intelligence required to maintain custody of active assets, ensure the safety of flight for neighbouring spacecraft, and identify potentially hostile changes. Thus, the objective is to identify deviations from a predicted nominal trajectory that cannot be explained by natural perturbations. Currently, this is approached through three primary methodologies: geometric analysis of public catalogues, statistical orbit determination and data-driven machine learning techniques.

4.1.1. Geometric Analysis and TLE Processing

The most fundamental form of maneuver detection relies on the geometric analysis of orbital element time-histories, typically derived from TLE catalogues. This method is the operator baseline, historically used to detect large impulsive burns. This is an indispensable tool for long-term trend analysis. The TLE data format provides a continuous record of mean Keplerian elements. While TLEs have lower accuracy than higher-fidelity state representations, the geometric signal-to-noise ratio (SNR) — that is, the ratio of the orbital element change caused by a maneuver to the inherent noise in the TLE fitting process — is sufficient to identify significant orbital changes when processed through algorithmic filters [18]. The guiding principle of geometric maneuver detection is rooted in the principle of ballistic continuity; an object in orbit, absent of active thrust, follows a trajectory dictated strictly by the natural forces, described in section 3.1. Propagators such as Simplified General Perturbations-4 (SGP4) are designed to model these natural ballistic forces analytically. Thus, maneuver detection can be framed as a continuous hypothesis test, in which the null hypothesis assumes that the satellite is in free flight. When the residual difference between a propagated state and an observed state exceeds a calibrated threshold, the null hypothesis is rejected and a maneuver is flagged.

The term “operator baseline” refers to the standard, deterministic methods used by surveillance entities to track objects for which maneuver plans are not shared, such as uncooperative or foreign satellites. It is characterized by a few visual factors that operators used before machine learning classifiers: determinism, interpretability and accessibility. Determinism refers to fixed logical gates, i.e. “if $\Delta a > X$, then Maneuver”. Interpretability refers to the metrics used to map directly to physical controls, for example, the semi-major axes and inclination being mapped directly to thrusters for station-keeping maneuvers, and finally, accessibility refers to only needing the publicly available TLE data and standard propagators. Thus, with this information, large impulsive burns can be detected, making it the standard for evaluating any model.

To further the accuracy of TLE-based maneuver detection, certain methods have been developed. One method is to perform a sliding-window analysis of TLE time histories. This approach recognizes that a single TLE may be erroneous due to the inherent noise in the TLE fitting process, and thus relies on the trend of a sequence of TLEs to distinguish genuine orbital changes from fitting artifacts. Thus, rather than comparing two points in time, the method examines a sequence of TLEs and fits a smooth polynomial or a ballistic trajectory to a window of N TLEs. It predicts the $(N+1)^{th}$ TLE based on the trend of the previous N. If the actual $(N+1)^{th}$ TLE deviates significantly from the trend line, a maneuver is flagged [18]. By fitting a trend to the previous N TLEs, this approach reduces the influence of noise in the historical data used to establish the baseline trajectory. However, a single erroneous TLE at the detection point can still trigger a false alarm, which remains a limitation of the method.

Another geometric approach to the maneuver detection problem, is known as the “Space Event Detection Method” [25]. This method abstracts the problem away from pure geometry and into the domain of energy conservation. Since energy is a constant of motion in ideal Keplerian mechanics, it is considered a robust metric. The algorithm calculates the specific energy for every TLE. It then applies a statistical filter, for example, a 3-sigma threshold relative to the running variance allowing it to be computationally efficient at detecting in-plane maneuvers. It is thus also less dependent on precise propagation from SGP4 because it compares the derived property rather than the original state vector.

While TLE-based methods provide an accessible and computationally efficient baseline for maneuver detection, they are subject to several fundamental limitations. First, the accuracy of TLEs is inherently limited by the SGP4 propagation model, which introduces modelling errors that can mask subtle maneuver signatures. Second, these methods operate on mean Keplerian elements with update intervals on the order of hours to days. Consequently, they are tuned for larger impulsive maneuvers, but are poorly suited for gradual low-thrust maneuvers, whose instantaneous effect on the orbital elements is well below the TLE now floor. Finally, TLE-based approaches are limited to detecting only if a maneuver has occurred. They cannot reconstruct the satellite’s maneuver, including thrust magnitude and direction. These shortcomings motivate the need for higher-fidelity approaches such as statistical orbit determination and prediction, which operate on higher-cadence observations and can provide more detailed maneuver characterizations.

4.1.2. Statistical Orbit Determination

Statistical orbit determination (SOD) is the process of tracking RSOs by reconciling an imperfect physics-based model with noisy sensor data. In the context of maneuver detection, SOD provides a framework for hypothesis testing, where a null hypothesis (H_0) is formulated, often assuming that an object is following its known orbit and allowing for natural perturbations. An alternative hypothesis (H_1) is then formed; often, an unknown thrust force is acting on the object. If an RSO undergoes a maneuver, the resulting deviations in the state vector manifest as an increase in the residuals, i.e., the difference between the predicted position and the incoming sensor measurements. Consequently, the detection of a maneuver is not merely an identification of movement but a statistical assessment of when the divergence from the expected orbital model exceeds defined covariance thresholds, necessitating a reset or an adaptation of the estimation process.

Numerous algorithms have been developed and used to track maneuvering satellites. The most common of these are Kalman filters, but in the presence of unknown maneuvers, their performance suffers [19]. Thus, different variants of Kalman filters are used, such as Extended Kalman Filters (EKF) and Unscented Kalman Filters (UKF), which are nonlinear variants of the basic Kalman filter, and within these, various techniques have been used to account for the presence of an unknown maneuver [45]. These techniques broadly fall into the category of process-noise methods, in which the unknown dynamics are modelled as additional uncertainty in the state prediction step. The process noise covariance matrix quantifies the filter's assumption about the magnitude of unmodelled forces. Large process noise allows the filter to place greater weight on incoming measurements than on its dynamical model, at the cost of noisier state estimates. The first is State Noise Compensation, which adds stochastic noise to the dynamical model to artificially inflate the covariance, thereby keeping the filter responsive to new measurements. This approach increases the filter's adaptability but does not modify the state vector itself, meaning it cannot directly estimate the unknown accelerations [34]. An alternative is Dynamic Model Compensation, which augments the state vector to include acceleration terms. This method models the unknown acceleration as a first-order Gauss-Markov process, introducing both a deterministic component to be estimated and a stochastic component [39]. This is a relatively simple approach, but it faces challenges in determining the equivalent noise level and related noise statistics in real time [19]. Another approach to solving the problem of an unknown maneuver is to use input detection and estimation. The goal is to detect and estimate an unknown input acceleration directly from the available measurement data. Thus, it is assumed that the satellite state is estimated without having a maneuver model. One such example is the use of thrust-Fourier-coefficients (TFC) to explicitly estimate an unknown control acceleration. These are 14 coefficients used to model spacecraft maneuvers by representing the thrust vector using a Fourier series [12]. More sophisticated models leverage optimal control principles to estimate maneuvers without requiring high-order parametrization, such as TFCs. The Optimal Control Based Estimator (OCBE) shifts the problem from a parameter estimation to a functional optimization framework [23]. The OCBE operates on the principle that a satellite operator will generally seek to satisfy objectives while minimizing propellant consumption. Consequently, the unknown maneuver is modelled as an optimal

control policy that minimizes a specific cost function, typically defined as:

$$J = \frac{1}{2}\int_{t_0}^{t_f} ||\mathbf{u}(t)||^2 dt \tag{4.1}$$

In this cost function, $\mathbf{u}(t)$ represents the control acceleration. By applying Pontryagin's Minimum Principle, the OCBE solves the Two-Point Boundary Value Problem to determine the thrust profile that best bridges the gap between the pre-maneuver state and the post-maneuver observations using costate equations to favour measurement estimates over a ballistic propagation [2][23]. Unlike standard sequential filters that may become "overconfident" as covariance shrinks, the OCBE uses an iterative process: it first reconstructs the most likely optimal control policy, then uses this policy to refine the state estimate through a smoother architecture [23].

As mentioned before, the decision to reject H_0 in favour of H_1 is typically governed by a Chi-squared test where the squared Mahalanobis distance of the residuals is compared against a threshold defined by the desired false-alarm rate. This spike is instantaneous for impulsive maneuvers, but for continuous low-thrust maneuvers, the residuals grow gradually and are often difficult to distinguish from unmodeled or mismodelled perturbing forces. Once a maneuver has been statistically confirmed, the SOD process must be adapted to prevent filter divergence. This is typically achieved either by artificially increasing the uncertainty to allow the filter to place greater trust in new measurements than in the previous orbital model, or by a full state reset. By integrating these statistical tests with optimal control-based reconstruction, SSA architectures can maintain a high-fidelity catalogue even in the presence of frequent, low-magnitude orbital transfers.

4.1.3. Data-Driven Machine Learning Techniques

More recently, with advances in machine learning algorithms, data-driven techniques are proliferating into the realm of maneuver detection. While SOD relies on the explicit modelling of natural perturbations to identify anomalies, ML techniques offer a data-driven alternative that focuses on pattern recognition within orbital time-series data. In recent years, the rise of high-cadence sensor networks and the availability of large-scale synthetic datasets, such as SPLID, have enabled the training of complex models capable of classifying maneuver types without a priori knowledge of the satellite's propulsion system. However, a significant challenge arises from the transition from synthetic training environments to the high variability of real-world observations, where sensor noise and sparse data gaps can lead to a high false-alarm rate if the model is not properly regularized by physical constraints [38].

As with any ML-based solution, there are certain drawbacks. First, while a statistical model can explain why a maneuver was flagged, such as using the Mahalanobis distance showing a statistical deviation, most machine learning models are "black boxes" in that they may flag an event without providing a physical reason. Second, as this is a data-driven approach, large amounts of data are required for training, and these data must be properly labelled. TLE data are available for many satellites, but there is no straightforward way to identify which ones exhibit maneuvers. For a truly global

system, thousands of maneuvers are needed for training. In real-world scenarios, this may not be feasible due to data availability; therefore, techniques such as data augmentation or synthetic data generation are used to bridge the gap. Although not perfect, as synthetic data cannot replicate real-world data, it can be used as a first step [38]. Furthermore, the type of data varies across models. Some models use raw TLE or state vectors for their computations, whereas others require feature engineering to convert the raw data into features such as drift rate or orbital energy, particularly for baseline ML models. These are often preferred because they are considered constants of motion in an unperturbed environment, making deviations much easier for these models to detect than raw data.

In ML, various techniques are used for maneuver detection, which can be divided into supervised classification and unsupervised anomaly detection. Supervised learning involves training models on labelled datasets where the ground truth of the maneuver is known. It has been demonstrated that, by transforming raw TLE data into physics-based features, such as mean longitude, drift rate, and inclination, machine learning classifiers can achieve high accuracy in distinguishing between different behavioural modes. Roberts and Linares [31] demonstrated that using algorithms such as Random Forests and Gradient Boosted Trees, models can effectively learn the decision boundaries that define nominal stationkeeping maneuvers, and thus allow for the detection of abnormal maneuvers that fall outside of the learned thresholds. However, it was also shown that if a satellite performs a maneuver type not present in the training set, the classifier may fail to categorize it correctly.

Because orbital data is inherently a time series, Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks are frequently employed [16][6]. Unlike static classifiers, LSTMs have ”memory” cells that enable them to process sequences of states, making them adept at recognizing gradual deviations frequently observed in maneuvers with low-thrust propulsion. These architectures are often used in an AutoEncoder configuration: the model learns to reconstruct nominal trajectories with high precision, and any observation that results in a high reconstruction error is flagged as a potential maneuver. This approach mitigates the need for labelled maneuver data, focusing instead on defining the statistical bounds of normal behaviour. This is the essence of Unsupervised learning, specifically Self-Supervised Learning, as the label is the input data itself [16].

4.1.4. Synthesis and Limitations

The current landscape of maneuver detection is defined by a fundamental trade-off between the interpretability of physics-based models and the flexibility of data-driven architectures. Geometric and statistical models, such as TLE processing and SOD, provide a transparent, physical and mathematically rigorous framework grounded in the principle of ballistic continuity. However, these methods remain highly sensitive to model mismatch, particularly when subtle signatures of low-thrust maneuvers can be easily obscured by uncertainties arising from mismodeled natural perturbations.

Conversely, data-driven architectures, specifically recurrent architectures like LSTMs, excel at identifying these gradual deviations by learning complex temporal dependencies without an explicit dynamical model, making them more adaptable to scenarios

where the dynamics are not fully known or modelled. Yet, their inherent “black box” nature and reliance on high-fidelity labelled datasets limit their reliability and generalizability when transitioned from synthetic training environments to real-world observations. This dichotomy highlights a critical need for a hybrid approach: a model that maintains the flexibility of neural networks while being strictly regularized by the governing differential equations of orbital motion. Such a framework would allow for the detection of maneuvers with the robustness of a physical propagator and the adaptive learning of a data-driven system.

4.2. Dataset Generation

4.2.1. Problem Formulation

The research questions in Section 2.8 formulate the problem as a machine-learning task for low-thrust maneuvers in GEO orbits and distinguish accelerations resulting from these thrusting maneuvers from mismodeled accelerations due to poor SRP estimates. Thus, a dataset is required to train a model on this, and as such, a synthetic dataset needs to be created with GEO trajectories split into three distinct classes:

- **Class 0 - Nominal:** Correct physical model, no maneuvers
- **Class 1 - Low-thrust:** Continuous constant thrust acceleration superimposed on the nominal dynamics
- **Class 2 - Bad SRP:** Incorrect area-to-mass ratio in the solar radiation pressure model superimposed on nominal dynamics

These distinct classes enable a model to train and recognize the underlying patterns within each orbit.

4.2.2. Initial State Generation

Satellites are initialized at the GEO radius $r_{\text{GEO}} = R_{\oplus} + 35{,}786$ km $\approx 42{,}164$ km, with a circular-orbit speed

$$v_{\text{GEO}} = \sqrt{\frac{\mu_{\oplus}}{r_{\text{GEO}}}} \approx 3.075 \text{ km/s}$$

The initial state $\mathbf{x}_0 = [\mathbf{r}_0, \mathbf{v}_0]^\top \in \mathbb{R}^6$ is generated by applying a RAAN rotation $R_z(\Omega)$ followed by an inclination rotation $R_x(i)$ to a reference in-plane state, where $\Omega \sim \mathcal{U}(0, 360)$ and $i \sim \mathcal{U}(0, 5)$, both in degrees.

4.2.3. Equations of Motion

The satellite state $\mathbf{x} = [\mathbf{r}, \mathbf{v}]^\top$ evolves according to equation 3.7, where the thrust acceleration coming from the low-thrust accelerations is zero for classes 0 and 2. This uses two-body gravity, the J2 perturbation, SRP, and third-body effects from the Sun and Moon. The Sun and Moon also need to be propagated, and they can be done in simplified circular orbits in the equatorial planes at their respective angular velocities: $\omega_{\odot} = 2\pi/(365.25 \times 86400)$ rad/s and $\omega_{\leftmoon} = 2\pi/(27.32 \times 86400)$ rad/s.

For the SRP modelling, equation 3.1.5 is used, but is multiplied by a cylindrical shadow function, $\nu \in [0, 1]$ [30]. This simplification neglects time spent in the penumbra, which is negligible for the test cases in a typical geosynchronous orbit. For Class 2, the area-to-mass ratio is varied from its nominal value, thereby introducing the unmodeled SRP bias.

For Class 1, a constant inertial thrust vector $\mathbf{a}_T = a_T \hat{\mathbf{u}}_T$ is added in the ECI frame, with a small magnitude a_T and direction $\hat{\mathbf{u}}_T$ drawn uniformly on the unit sphere. This simplification ensures generality of the training data by avoiding encoding assumptions about specific maneuver strategies, such as along-track or cross-track thrusting.

4.2.4. Numerical Integration

Each trajectory is integrated using the DOP853 scheme (8th order Dormand-Prince Runge-Kutta) with 16-48 hour arcs. This integrator was chosen for several reasons. Every time an integrator evaluates $\dot{\mathbf{x}}$, it invokes the full perturbation model, which is computationally expensive. Lower-order integrators, such as RK4 (Runge-Kutta 4), accumulate local truncation errors every step. For a GEO orbit sampled at 10-minute intervals over 48 hours, these errors compound over 288 steps, and thus the small velocity errors will translate to growing position errors over time because of the feedback between position and gravitational acceleration. RK4 would require a very small step size to keep errors below the noise level of the required measurements. Thus, the DOP853 scheme is preferred because it uses 13 function evaluations per step and achieves a much smaller local error than RK4, enabling the use of larger step sizes for equivalent accuracy [11]. Furthermore, DOP853 includes embedded 5th- and 3rd-order estimates, which enable adaptive step-size control: the integrator automatically tightens the step size when the trajectory's derivative curves are sharp. For example, near an eclipse transition in SRP, to prevent Gibbs-like oscillations that might incorrectly be classified as a maneuver and these very low orders of magnitude, and relax it during smooth stretches. The tolerances are also adjusted to keep the local relative error sufficiently low to ensure that the differences between Class 0 and Class 1 trajectories are due to physics rather than integration artifacts. Thus, while an Rk4 integrator can be used for this purpose and will likely produce acceptable results for propagation arcs of this length, the main advantage of a variable step integrator is that the step-size dependent error does not need to be characterized. This reduces setup effort while ensuring that local truncation error remains below a specified threshold throughout the integration.

4.3. Orbit Anomaly Classifier

4.3.1. Feature Engineering

Each raw state sequence $[\mathbf{r}_k, \mathbf{v}_k]$ is mapped to 13 derived orbital features per timestep that amplify anomaly signatures relative to the GEO reference orbit, shown in table 4.1.

The features are normalized to zero mean and unit variance using statistics computed from the training split. A normalized scalar time $\tau_k = t_k/t_N$ is appended, giving a 14-dimensional input vector at each step.

Index	Feature	Formula
0	Radial deviation	$\delta r = \lvert\mathbf{r}\rvert - r_{\text{GEO}}$
1	Speed deviation	$\delta v = \lvert\mathbf{v}\rvert - v_{\text{GEO}}$
2	Specific energy deviation	$\delta\mathcal{E} = \frac{1}{2}v^2 - \frac{\mu_\oplus}{r} - \mathcal{E}_{\text{GEO}}$
3	Angular momentum deviation	$\delta h = \lvert\mathbf{r} \times \mathbf{v}\rvert - h_{\text{GEO}}$
4	Radial velocity	$\dot{r} = (\mathbf{r} \cdot \mathbf{v})/r$
5	Osculating eccentricity	$e = \lvert\mathbf{v} \times \mathbf{h}/\mu_\oplus - \hat{\mathbf{r}}\rvert$
6	Semi-major axis deviation	$\delta a = -\mu_\oplus/(2\mathcal{E}) - r_{\text{GEO}}$
7–9	RTN position	ECI residual decomposition into $(\hat{\mathbf{r}}, \hat{\mathbf{t}}, \hat{\mathbf{n}})$ frame
10–12	RTN velocity	ECI state vector decomposition into RTN frame

Table 4.1: Feature Mapping Breakdown

The reason for this mapping is that the raw state, $[\mathbf{r}, \mathbf{v}]$, is expressed in ECI coordinates. The absolute position values dominate the signal and contain almost no anomaly information. A nominal orbit and an anomalous orbit have nearly identical absolute positions. The features are engineered to isolate the orbital-mechanics signal that distinguishes the three classes, removing the dominant GEO reference and expressing everything as deviations or invariants.

Thus, Index 0 and 1 refer to the deviations in the GEO reference. A nominal GEO satellite drifts very slowly from its reference radius and speed due to perturbations. A low-thrust satellite will drift more rapidly, and in a direction correlated with the thrust. A mismodeled SRP acceleration will also exhibit drift in the satellite position, but the pattern is driven by solar geometry rather than by a constant inertial force. Furthermore, in an unperturbed two-body orbit, the orbital energy is conserved, and it only changes very slowly in a nominally perturbed orbit. When thrust is present, the rate of energy change is expressed as $\dot{\mathcal{E}} = \mathbf{a}_T \cdot \mathbf{v}$, a term that grows linearly with time. This makes $\delta\mathcal{E}$ a sensitive and monotonically growing signature of low thrust. Mismodeled SRP does not directly inject energy in the same way, and thus it helps distinguish between Class 1 and 2. Index 3 is used because the angular momentum vector h determines the shape of the orbit. Since these synthetically generated orbits are all circular at GEO, this vector should remain constant for nominal trajectories. Thrust in the tangential direction changes h rapidly; thrust in the radial direction changes it much less. Mismodeled SRP, which acts radially away from the Sun, will produce a different δh signature than a constant inertial thrust. Additionally, this effect is captured in indices 4 and 5, where any change in radial velocity indicates that the orbit is becoming elliptical rather than remaining circular, thereby affecting the eccentricity over time. Low thrust signatures would show as a time-varying non-zero $\dot{r}$, while mismodeled SRP will produce an oscillating $\dot{r}$. Index 6, the semi-major axis, is tightly coupled with the energy. While this does not provide an independent view, since the two are

related nonlinearly ($a \propto 1/\mathcal{E}$), retaining both can aid learning by providing the optimizer with multiple gradient paths to the same underlying signal. The last six indices, 7-12, decompose the position and velocity into the Radial-Transverse-Normal (RTN) frame of reference. This rotates with the satellite, where $\hat{\mathbf{R}}$ points away from Earth, $\hat{\mathbf{T}}$ points along the velocity, and $\hat{\mathbf{N}}$ is the orbit normal. These components represent the deviation from the nominal GEO reference orbit expressed in this local frame. Decomposing into RTN separates the state into three physically distinct directions, each with different sensitivity to the anomaly classes. For example, SRP acts primarily along the Sun-satellite direction, which projects onto all three RTN components depending on the satellite's orbital position, producing a characteristic oscillatory pattern as the satellite orbits. Low-thrust maneuvers, depending on their inertial direction, can produce persistent secular growth in one or more RTN components. By providing all three components to the network, the classifier can learn to distinguish among these temporal patterns without making assumptions about the thrust or perturbation direction.

4.3.2. LSTM with Temporal Attention

The primary classifier is an LSTM with an additive attention mechanism. As mentioned earlier, time-series problems such as this are better solved using an RNN or an LSTM. A single snapshot of the orbit gives almost no information, since all three classes can look identical in a given moment. The distinguishing features arise when orbital elements evolve over time. A simple feedforward network treats each timestep independently. It cannot detect that the orbital energy, for example, has been growing linearly over an orbit, a key signature of low thrust. The LSTM, however, is designed precisely to process sequences and learn what patterns over time are predictive of a class. Thus, given a sequence of feature vectors $\mathbf{f}_{1:N} \in \mathbb{R}^{14}$, the forward pass proceeds as:

1. **LSTM encoding:** $\{\mathbf{h}_k\}_{k=1}^{N} = \text{LSTM}(\mathbf{f}_{1:N})$, with $L = 3$ layers and hidden dimension $d_h = 256$ and N is the number of timesteps in the observation arc. Here, $\mathbf{h}_k$ is the hidden state of the LSTM, a compressed memory of everything seen up to time k. At each timestep k, the LSTM takes the current input, the 14-dimensional feature vector, $\mathbf{f}_k$ and the previous hidden state $\mathbf{h}_{k-1}$ and produces an updated hidden state $\mathbf{h}_k$. Internally, it uses three gates, all learned from data:

    - **Forget Gate:** This decides what fraction of the previous memory to erase. If the orbit suddenly changes behaviour, such as thrust onset partway through, the gate can effectively reset the memory.
    - **Input Gate:** This decides how much of the new observation to write into memory. Early timesteps, where the trajectory is ambiguous, contribute less, whereas later timesteps with clear anomaly signatures contribute more.
    - **Output Gate:** This controls what part of the memory is passed forward as the hidden state $\mathbf{h}_k$ to be seen by downstream layers.

    With $L = 3$ stacked LSTM layers, each layer learns increasingly abstract temporal representations. The first layer may capture local dynamics, such as $\dot{r}$ oscillations; the second might integrate these into energy-drift trends; and the third may relate those trends to global class signatures.

2. **Attention scoring:** Simply averaging all the hidden states wastes information, even with 3 LSTM layers. Especially for longer arcs, the anomaly signature may be most pronounced towards the end, when thrust-induced deviation has had time to accumulate. A simple mean would dilute the signal with the early, ambiguous portion of the trajectory. Thus, the attention mechanism learns a scalar weight α_k for each timestep, using the scoring variables formulated in equation 4.2. These weights are computed by a small two-layer network applied to each $\mathbf{h}_k$ for each time step, then normalized via softmax, which rescales a vector so that its elements lie between 0 and 1, and sum to 1 as seen in equation 4.3.

$$s_k = \mathbf{v}_a^\top \tanh(\mathbf{W}_a \mathbf{h}_k + \mathbf{b}_a) \tag{4.2}$$

$\mathbf{W}_a \in \mathbb{R}^{d_a \times d_h}$ is the attention weight matrix, projecting the hidden state into an attention latent space of dimension d_a. This allows the model to learn which combinations of features in the hidden state are important. $\mathbf{b}_a \in \mathbb{R}^{d_a}$ is the bias vector, providing a learnable offset. $\mathbf{v}_a^\top \in \mathbb{R}^{1 \times d_a}$ is the weight vector collapsing the d_a-dimensional latent representation down to a single scalar score s_k

$$\alpha_k = \frac{\exp(s_k)}{\sum_{j=1}^{N} \exp(s_j)} \tag{4.3}$$

3. **Weighted context vector:** The final context vector, $\mathbf{c} = \sum_k \alpha_k \mathbf{h}_k$, is a weighted average that emphasizes whichever timesteps are most informative for the classification decision. These weights are learned end-to-end from data: the model discovers which parts of the orbit are discriminative without any domain knowledge being hardcoded.
4. **Classification head:** The context vector, $\mathbf{c}$, is a 256-dimensional summary of the entire trajectory, as per the number of hidden dimensions, d_h. It is passed through two fully connected layers with ReLU activations and dropout, then a final linear layer to produce 3 logit values. These logits are converted to class probabilities via softmax: $\hat{\mathbf{y}} = \text{Softmax}(\mathbf{W}_3 \cdot \text{ReLU}(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1 \mathbf{c} + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3)$.

4.3.3. Training

The model is trained with cross-entropy loss using AdamW, a variant of the Adam optimizer that decouples weight decay regularization from the adaptive gradient update, providing more effective regularization against overfitting [22]. Early stopping based on validation loss is typical for classifiers.

The full classifier pipeline is shown in Figure 4.1. The model comprises four functional blocks. First, the input block accepts a variable-length sequence of 14-dimensional feature vectors, with sequence length N ranging from 96 to 288 depending on the arc duration. Second, the encoder consists of three stacked LSTM layers of 256 hidden units each, in which the output sequence of layer is fed as input to the layer + 1; this this produces a sequence of hidden states $h_1, ..., h_N$ that compresses the observed dynamics up to each timestep. Third, a temporal attention mechanism is applied to the hidden state sequence: each h_k is projected through a learned linear layer into a

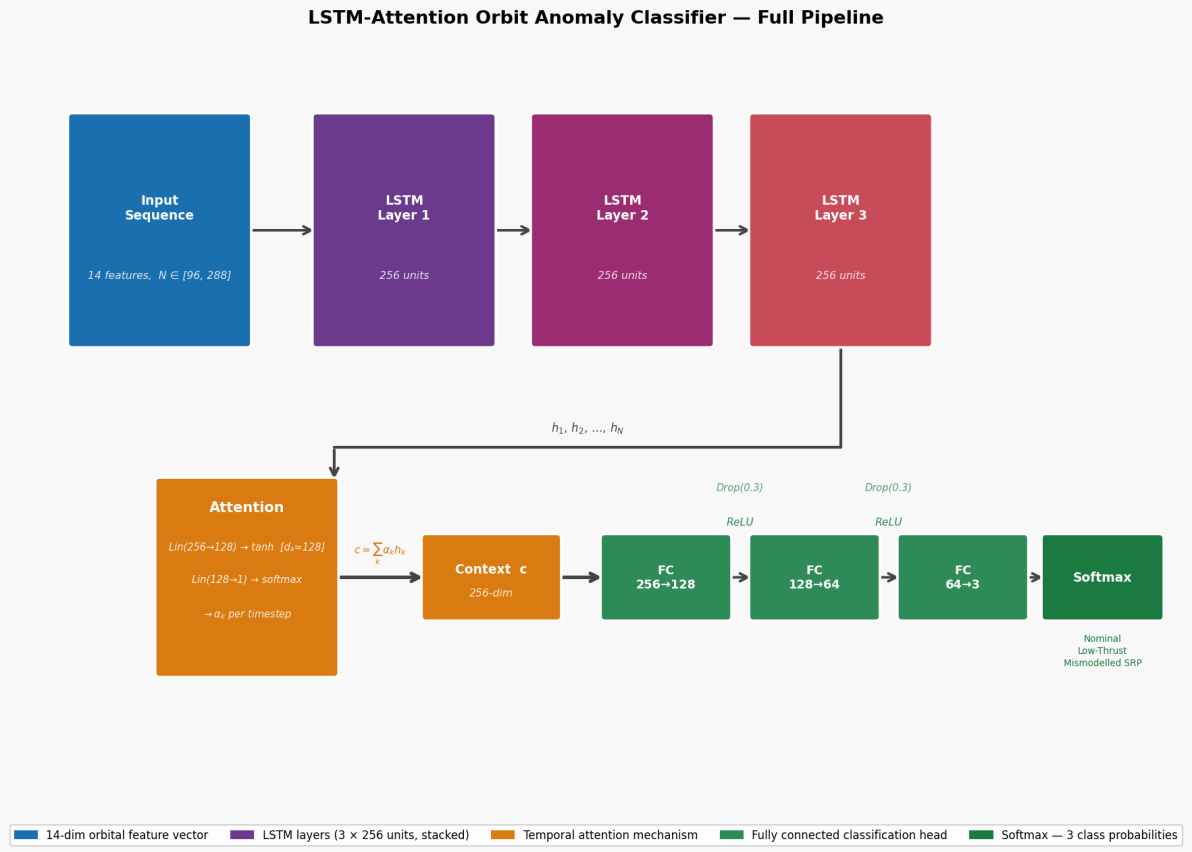


Figure 4.1: Architecture of the LSTM-Attention Orbit Anomaly Classifier

128-dimensional attention space, passed through a tanh nonlinearity, and scored by a second linear layer whose outputs are softmax-normalized to yield attention weights $\alpha_k$that sum to unity. A single context vector **c** is then formed as the weighted sum of the hidden states, producing a fixed 256-dimensional summary of the trajectory regardless of its original length. Fourth, the context vector is passed through a classification head comprising three fully connected layers with ReLU activations and dropout with a rate of 0.3 applied between layers; a final softmax converts the logits into a probability distribution over the three classes: Nominal, Low-Thrust, and Mismodelled SRP. The network is trained end-to-end using a cross-entropy loss with the AdamW optimizer, and early stopping is triggered on validation loss.

4.4. PINN - Inverse Thrust Recovery

4.4.1. Problem Statement

The objective of the PINN in this context is to solve the following problem: Given a noisy observed trajectory $[\hat{\mathbf{r}}_k, \hat{\mathbf{v}}k]_{k=1}^{N}$ of a suspected maneuvering satellite and its known initial conditions $\mathbf{x}_0$, recover the unknown constant thrust vector $\mathbf{a}_T \in \mathbb{R}^3$. Thus, instead of using a PINN to predict an orbit forward in time, this is an inverse problem to recover an unknown constant thrust vector $\mathbf{a}_T$ that caused it. The naive approach is to fit a neural network directly to the observed positions, then compute its second derivative and subtract known accelerations to isolate thrust. The problem is that an

orbit at GEO has positions of $\sim 42{,}000$ km, meaning a neural network has to learn a function with output values five orders of magnitude larger than the thrust signal, usually ($\sim 10^{-8}$ km/s^2 times T^2). Small relative errors in the neural network output lead to large errors in the estimated thrust. A PINN solves this by encoding physics into the loss, but a well-conditioned representation of the orbit is still required.

It is important to note that, unlike the LSTM classifier, the inverse PINN is not a traditional supervised learning model. It is tained per orbit arc, where each arc constitutes a unique inverse problem with its own observations, reference trajectory, and unknown thrust to recover. The network weights encode the solution to that specific problem, analogous to classic orbit determination. Consequently, there is no dataset split in the traditional sense; the solution is validated by the magnitude of the data and physics residuals, as well as by the physical plausibility of the recovered thrust.

4.4.2. Reference Trajectory Decomposition

A reference trajectory $\mathbf{r}^{\text{ref}}(t), \mathbf{v}^{\text{ref}}(t)$ is first generated by numerically integrating the perturbed two-body equations from the same $\mathbf{x}_0$ with $\mathbf{a}_T = 0$. The total position is then decomposed as:

$$\mathbf{r}(t) = \mathbf{r}^{\text{ref}}(t) + \boldsymbol{\delta}(t) \tag{4.4}$$

where $\boldsymbol{\delta}(t)$ is a small correction (~ 1–40 km) due entirely to the unknown thrust. The PINN is trained to predict only $\boldsymbol{\delta}(t)$, which is orders of magnitude better conditioned than fitting the full orbit. The reason for this is that, after one full GEO orbit, of thrust at 10^{-8} km/s^2, the accumulated position deviation is at most ~ 30 km, against a background orbit of $\sim 42{,}000$ km. By subtracting the reference, the PINN's task shifts from fitting a very large carrier to fitting a small deviation. Thus, working with differences relative to a known solution rather than the full solution. One consequence of this is that the PINN output is later initialized near zero, meaning it starts with a nearly correct answer, and the optimizer then makes small refinements. Without the reference trajectory, the optimizer would have to build the entire orbit from scratch.

4.4.3. Neural Network Architecture

A PINN can be parameterized such that the network output is structured so that the initial conditions are satisfied by construction instead of being learned through the loss function as seen in equation 4.5.

$$\boldsymbol{\delta}(\tau) = \tau^2 \cdot \mathcal{N}(\tau; \theta) \cdot \Delta_{\text{scale}} \tag{4.5}$$

Here, $\tau = (t - t_0)/T \in [0, 1]$ is the normalized time and $\mathcal{N}$ is a fully-connected network (4 hidden layers, 64 neurons each, Tanh activations, output $\in \mathbb{R}^3$). The τ^2 factor exactly enforces the initial conditions $\boldsymbol{\delta}(0) = \mathbf{0}$ and $\dot{\boldsymbol{\delta}}(0) = \mathbf{0}$ without any penalty term and is thus a hard constraint. Consequently, if the assumed initial state deviates from the true state, this introduces a systemic bias at $\tau = 0$ that the network cannot correct for. This hard constraint is preferable to the alternative, which requires careful tuning of the initial conditions, as it can be enforced analytically. The consequence is that

the optimizer can focus entirely on fitting the physics and matching the data, without spending iterations to maintain the initial conditions.

Δ_{scale} is set to about 1.5 times the maximum observed position deviation between the thrusted orbit and the reference. Neural networks with a Tanh activation function are calibrated to produce outputs of order 1. This is chosen to give the network some headroom, as it does not need to hit its maximum output to represent the largest correction. If the PINN outputs $\mathcal{N}(\tau) \sim \mathcal{O}(1)$ and the physical correction is ~ 20 km, then without Δ_{scale} the NN would need to output 20. This is bad for a Tanh-based network as this would require large pre-activation values, pushing the activations into the saturated tails where gradients vanish and training stalls. By setting Δ_{scale} to 20 and ensuring that the PINN outputs 1, all internal activation-function weights lie within a natural range, making the optimization landscape smoother. This is similar to feature normalization in supervised learning. In practice, Δ_{scale} can always be computed before training begins, since it only requires the maximum observed position deviation between the observed trajectory and a reference propagation with no thrust. This value serves as a reasonable upper bound on the magnitude of the correction, regardless of whether thrust is present, since if no thrust occurred, the deviation would be small, and the PINN output would remain near zero.

The thrust is a learnable parameter $\mathbf{a}_T = \tilde{\mathbf{a}}_T \cdot \lambda_T$ where $\lambda_T = 10^{-8}$ km/s^2 keeps the raw parameter $\tilde{\mathbf{a}}_T$ order-unity for numerical stability. This is done for the same reason as Δ_{scale} If the optimizer directly updates a parameter with the small thrust magnitudes, gradient steps of $\sim 10^{-4}$ would move the thrust by $10^{-4}/10^{-8} = 10^4$ times the physical value, significantly higher than what is required. Therefore, by using a raw parameter of order 1 and applying the scale factor at the end, the optimizer's default learning rates yield sensible physical step sizes. The scaling also keeps the thrust penalty term $|\tilde{\mathbf{a}}_T|^2$ comparable in magnitude to the other loss terms.

4.4.4. Loss Function

The total loss is represented as:

$$\mathcal{L} = \lambda_d \mathcal{L}_{\text{data}} + \lambda_p \mathcal{L}_{\text{phys}} + \lambda_r |\tilde{\mathbf{a}}_T|^2 \tag{4.6}$$

Data loss penalizes the mismatch between the predicted correction $\delta(\tau_k)$ and the observed deviation from the reference trajectory $\Delta\hat{\mathbf{r}}_k = \hat{\mathbf{r}}_k - \mathbf{r}^{\text{ref}}(t_k)$, and similarly for the predicted velocity deviation $\dot{\boldsymbol{\delta}}$, computed via AD and observed velocity deviation. The input position and velocity vectors are in ECI with some Gaussian noise. Position observations alone are ambiguous, as many different velocity profiles can produce the same position at each snapshot. Including velocity constraints, the rate of change of the correction directly constrains the thrust estimate. This helps resolve the state-observability problem more quickly, especially in short-arc scenarios where the curvature of $\delta(t)$ may not be sufficiently distinct from that of a slightly different initial velocity. After 24 hours sampled at 10-minute intervals, there are around 145 positions and 145 velocity observations. This yields 870 scalar constraints on three thrust-unknown parameters under the assumption of a constant thrust profile, making the system highly over-determined, which is essential for robust recovery.

Phase	Optimizer	Iterations	Thrust	Purpose
1	AdamW (lr $= 10^{-3}$)	2000	Frozen	Fit $\boldsymbol{\delta}$ to trajectory deviation.
Warm-start	—	—	—	Direct algebraic estimate from $\langle \ddot{\boldsymbol{\delta}}/T^2 - \delta g \rangle$ to initialize $\mathbf{a}_T$ without gradient descent.
2	AdamW (lr $= 10^{-4}$)	3000	Unfrozen	Joint refinement of $\boldsymbol{\delta}$ and $\mathbf{a}_T$.
3	L-BFGS	Until Convergence	Unfrozen	High-precision convergence for final residual minimization.

Table 4.2: Three-phase Optimization Procedure for PINN Reconstruction

Position deviations are in km, and the velocity deviations are in km/s; thus, different physical units with different magnitudes. Without normalization, whichever term has larger raw values numerically dominates the gradient. The adaptive scales, $\sigma_r = \max(|\Delta\hat{\mathbf{r}}|)$ and $\sigma_v = \max(|\Delta\hat{\mathbf{v}}|)$ ensure both terms contribute comparably.

Physics loss enforces Newton's second law for the correction:

$$\mathcal{L}_{\text{phys}} = \frac{1}{\lambda_T^2} \left\langle \left| \frac{\ddot{\boldsymbol{\delta}}}{T^2} - \underbrace{\left[\mathbf{a}(\mathbf{r}^{\text{ref}} + \boldsymbol{\delta}) - \mathbf{a}(\mathbf{r}^{\text{ref}})\right]}_{\text{gravity differential}} - \mathbf{a}_T \right|^2 \right\rangle$$

This residual is derived from the equation of motion for the correction:

$$\ddot{\boldsymbol{\delta}} = \mathbf{a}_{\text{pert}}(\mathbf{r}^{\text{ref}} + \boldsymbol{\delta}, t) - \mathbf{a}_{\text{pert}}(\mathbf{r}^{\text{ref}}, t) + \mathbf{a}_T$$

where the estimated thrust, a_T is a learnable parameter, $\mathbf{a}_{\text{pert}}$ is the full perturbed acceleration from the same model used to generate the reference trajectory, computed via two successive ADs of the PINN with respect to τ and λ_T^2 is a heuristic scaling factor based on the expected thrust magnitude. The gravity differential $[\mathbf{a}(\mathbf{r}^{\text{ref}} + \boldsymbol{\delta}) - \mathbf{a}(\mathbf{r}^{\text{ref}})]$ captures the nonlinear change in gravitational acceleration caused by the correction, similar to a tidal force term. This term will hereby be referred to as δg. Because $\boldsymbol{\delta} \ll |\mathbf{r}^{\text{ref}}|$, this differential is small, and the PINN only has to produce accelerations of order $\sim 10^{-8}$ km/s^2 rather than the full $\sim 10^{-3}$ km/s^2 gravitational acceleration. The residual that must be zero is precisely thrust, and thus, the physics loss directly informs the neural network that any unexplained acceleration is thrust.

4.4.5. Optimization

The optimization follows a three-phase procedure, shown in table 4.2

The thrust warm-start in Phase 1 → Phase 2 directly solves $\mathbf{a}_T \approx \langle \ddot{\boldsymbol{\delta}}/T^2 - [\mathbf{a}(\mathbf{r}^{\text{ref}} + \boldsymbol{\delta}) - \mathbf{a}(\mathbf{r}^{\text{ref}})] \rangle$ over interior trajectory points. These are the endpoints at which the τ^2 constraint forces δ, yielding a close initial estimate and dramatically accelerating Phase 2 convergence.

The L2 regularization term $\lambda_r |\tilde{\mathbf{a}}_T|^2$ penalizes large thrust magnitude, where $\lambda_r = 0.001$ in Phase 2 and 0.00001 in Phase 3. Without regularization, the optimizer can explain any trajectory noise by inflating the thrust estimate. The regularization embeds the prior belief that thrust is small. Weaker regularization in Phase 3 allows L-BFGS to converge tightly to the true thrust without being held back, once the PINN correction is already well-initialized from the previous two phases.

The reason for the three-phase optimization is that the joint problem, fitting the correction and estimating the thrust simultaneously from a random initialization, is extremely ill-conditioned. The thrust enters the loss only through the physics residual, but the residual itself is meaningless until the neural network correction is already close to the data. A randomly initialized neural network has $\boldsymbol{\delta} \approx 0$ everywhere, so $\ddot{\boldsymbol{\delta}} \approx 0$, so the physics loss gradient pushes thrust toward zero regardless of the true value. Thus, the problem must be broken into stages.

The first phase is a data-only Adam optimization. The PINN learns to fit the observed trajectory deviation $\Delta\hat{\mathbf{r}}$ as accurately as possible. The thrust is not updated in this phase; therefore, it remains frozen at zero. This phase is intended to be fast and well-posed because it merely approximates the function. After Phase 1, $\boldsymbol{\delta}(\tau)$ closely matches the true position and velocity deviations, indicating the neural network has captured all the information about how the orbit actually moved.

The Warm-start phase is an algebraic estimate of the thrust. Before activating the physics loss, an initial thrust estimate is extracted directly from the fitted correction. Since $\boldsymbol{\delta}$ now closely tracks the true deviation, the physics equation $\ddot{\boldsymbol{\delta}}/T^2 \approx \delta g + \hat{\mathbf{a}}_T$ can be rearranged: $\hat{\mathbf{a}}_T \approx \langle \ddot{\boldsymbol{\delta}}/T^2 - \delta g \rangle$. This yields a good initial thrust estimate without performing gradient descent on the physics loss, avoiding a cold start in which the physics gradient is uninformative.

Phase 2 is a joint Adam optimization with both the neural network and the thrust active, with the physics loss introduced alongside the data loss. The neural network and thrust are refined together. The learning rate is reduced since the neural network is already close and large steps would break the carefully established fit. This phase primarily refines the thrust direction and magnitude while maintaining data fit.

Phase 3 switches the optimizer to L-BFGS, the preferred optimizer for PINNs. This is a quasi-Newton method that uses a low-rank approximation to the inverse Hessian to take curved steps toward the minimum. It usually converges superlinearly near a minimum, meaning errors shrink faster than any fixed-rate schedule. Compared with an Adam optimizer, such as that used in the previous steps, a first-order method with fixed adaptive learning rates can stall near the minimum. The L-BFGS method is used to extract the last digits of precision from a well-initialized solution, using a strong Wolfe line search to ensure sufficient decrease at each step. The thrust regularization is also significantly reduced here, so that the converger can settle at the physically

correct value rather than being biased towards zero.

Finally, AD is used throughout the computation to compute the velocity and acceleration of the correction. The finite differences are never used, and instead, the derivatives are calculated analytically.[28] Thus, $\dot{\boldsymbol{\delta}} = \partial\boldsymbol{\delta}/\partial\tau$ and $\ddot{\boldsymbol{\delta}} = \partial^2\boldsymbol{\delta}/\partial\tau^2$ are exact to machine precision. Finite differences are approximations of derivatives and thus introduce truncation errors proportional to the step size. In the physics loss, because the second derivatives computed are extremely small, truncation errors cannot be avoided and are comparable to the thrust signal. AD has zero truncation error and is typically favoured for machine learning tasks, at the cost of only a small constant multiple of the forward pass. This function is one of the core reasons PINNs are implemented in differentiable frameworks rather than as wrappers around a black-box solver.

Several limitations of this approach should be noted. The per-arc training cost is higher than recursive estimators such as the UKF, though each arc is independently solvable and therefore trivially parallelizable. Validation is necessarily residual-based, a circularity shared with classical orbit determination; independent confirmation requires withheld observations or operator maneuver logs. Finally, the approach inherits the reference trajectory's dynamical model, meaning that unmodelled perturbations are absorbed into the thrust estimate.

5

Verification and validation

To arrive at the methodology presented in the previous section, some parts of the neural network are tested in advance to better understand the capabilities of a PINN. These will be presented in the following sections, followed by the verification and validation of the test run.

5.1. Mass Spring System

The first verification test uses a mass-spring system. The system is governed by a simple differential equation, making it well-suited for testing the PINN's capabilities. The system assumes a perfect, undamped spring, and thus, the analytic solution gives an undamped sinusoidal wave. The system in this verfication test is governed by the equation:

$$m\ddot{x} + kx = 0$$

This has an analytic solution of:

$$x(t) = x_0 \cos(\omega t) + \frac{v_0}{\omega} \sin(\omega t), \quad \omega = \sqrt{k/m} = 2$$

which simplifies to $x(t) = \cos(2t)$. The PINN uses 3 hidden layers of 128 neurons each, using Tanh activations between them. The layers map the single input, time, to the 128 neurons in each of the three layers, and then produce the position as an output. The loss function used for this is

$$L = L_{ODE} + 10L_{IC}$$

where L_{ODE} is the mean squared error (MSE) of the residual, $m\ddot{x} + kx$, evaluated with automatic differentiation. L_{IC} is the MSE on the predicted initial conditions. Hyperparameter tuning sets the IC weight to 10 to force the network to prioritize initial conditions, which would otherwise be overshadowed by the much larger ODE loss. The optimizer used here is purely L-BFGS, as it is the standard choice for PINNs. This initial test serves to determine how the PINN learns a simple differential equation and to identify its limitations [29]. Figure 5.1 shows the results in the training domain, and the training parameters are shown in Table 5.1.

Training Parameter	Type	Value
m	Constant	1 kg
k	Constant	4 kg/s^2
x_0	Initial Condition	1 m
v_0	Initial Condition	0 m/s
Epochs	Hyperparameter	5000
Initial Learning Rate	Hyperparameter	0.1

Table 5.1: Training Parameters for Basic Undamped Mass-Spring PINN

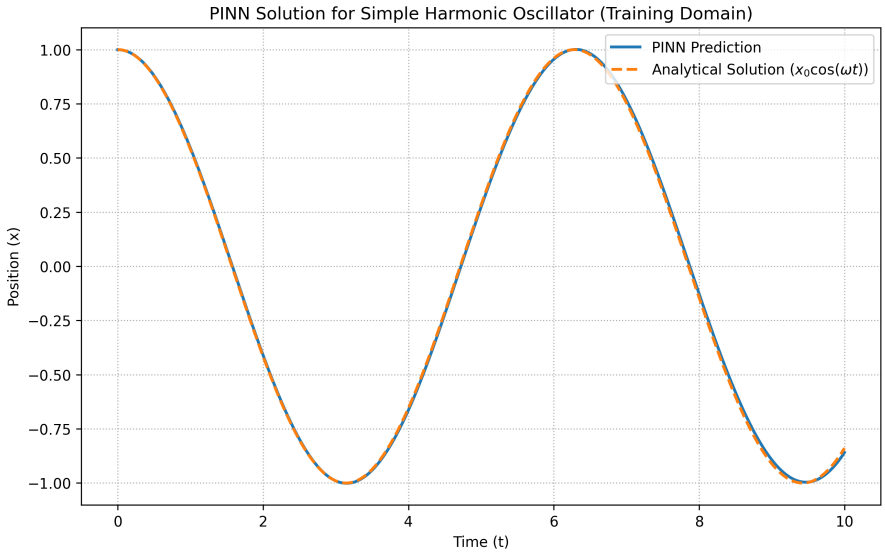


Figure 5.1: Result of the PINN on Training Domain

The result is that the PINN closely follows the expected trajectory with minimal error. The trajectory is sinusoidal as expected. However, this test also reveals a limitation. Figure 5.2 shows what happens on the test set when the model is asked to extrapolate. The error increases rapidly when the model is asked to predict trajectories beyond the training time horizon, as shown in the residual-over-time plot.

Standard MLPs exhibit a spectral bias, meaning they tend to learn low-frequency, smooth functions and struggle with oscillations, particularly at higher frequencies. For this test, $\omega = 2$, which is small enough to be manageable, but extrapolation beyond the training window degrades quickly as the network reverts to a flat or slowly varying output.

5.2. Extrapolation

To address the core issue of extrapolation, three different variants of the PINN are selected and tested. While a mass-spring system is a linear system, the periodic effects inform how a PINN could perform when used in a non-linear system for orbital

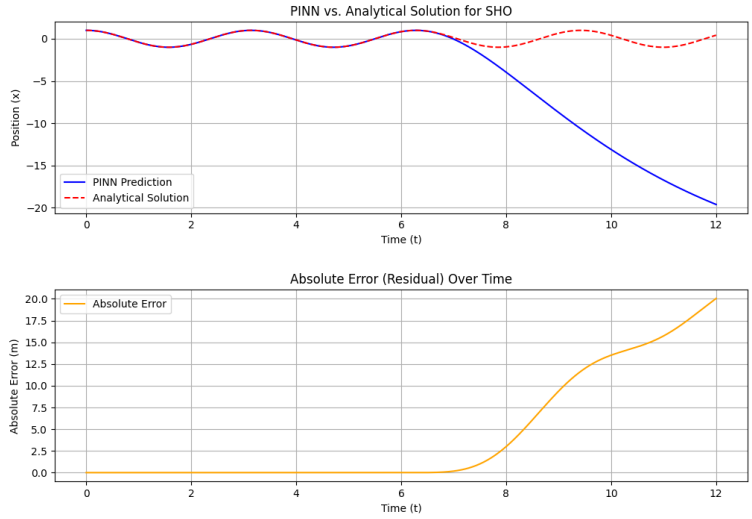


Figure 5.2: Test Case of PINN with Extrapolation (Top) and Residuals Over Time (Bottom)

mechanics. Thus, with the simpler system, testing is performed using a Fourier-based PINN, a Hamiltonian PINN, and an LSTM-PINN, and the results are shown in Figure 5.3.

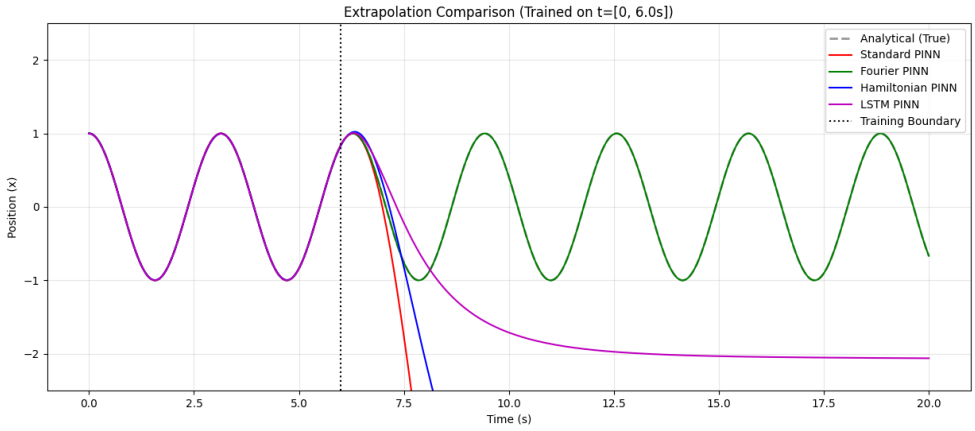


Figure 5.3: Comparison in Extrapolation Between PINN Variants

5.2.1. Fourier PINN

The Fourier PINN directly addresses spectral bias. The architecture comprises 2 hidden layers of 128 neurons, each with a Tanh activation function. The core difference is that time is transformed:

$$t \mapsto \mathbf{z}(t) = [\sin(\omega_{\text{ref}} \cdot t),\ \cos(\omega_{\text{ref}} \cdot t)]$$

resulting in two inputs, and one output. $\omega_{ref} = 2$, which is exactly the true natural frequency of the system. The trigonometric functions evaluated at the correct frequencies already form the foundation of the analytical solution. The network now only

needs to learn the appropriate linear combination of these features, rather than discovering oscillations from scratch [41]. Thus, if ω_{ref} is set correctly, then it is able to extrapolate beyond the training window as seen in Figure 5.3. In reality, ω_{ref} will not be known and would have to be added as a learnable parameter, but for this case, ω_{ref} is set as a fixed value for simplicity. The loss function used is the same as in the original PINN, with the only difference arising from the input encoding.

5.2.2. Hamiltonian PINN

The Hamiltonian PINN is structurally the most different approach from the original. It maps 1 input, time, to a single 128-neuron hidden layer with a Tanh activation function, yielding an output 2D state vector (q, p). Instead of directly fitting $x(t)$, it learns the phase-space trajectory and enforces Hamilton's equations as the physics constraint. The Hamiltonian for a spring-mass system is the total mechanical energy:

$$H(q,p) = \underbrace{\frac{p^2}{2m}}_{\text{kinetic}} + \underbrace{\frac{1}{2}kq^2}_{\text{potential}}$$

Hamilton's equations of motion follow from this:

$$\dot{q} = \frac{\partial H}{\partial p} = \frac{p}{m}, \qquad \dot{p} = -\frac{\partial H}{\partial q} = -kq$$

The loss function enforces both equations simultaneously:

$$L = \underbrace{\text{MSE}\left(\dot{q} - \frac{p}{m}\right)}_{L_q} + \underbrace{\text{MSE}\left(\dot{p} + kq\right)}_{L_p} + 10 \cdot L_{IC}$$

The Hamiltonian formulation is symplectic, meaning it preserves the geometric structure of conservative systems [14]. Energy is implicitly conserved as a consequence of the structure, rather than enforced as a separate penalty. While a standard PINN is able to satisfy $m\ddot{x} + kx = 0$ while still having slowly drifting energy, the Hamiltonian PINN cannot, because q and p are coupled through physical equations. Thus, in theory, it makes it more robust for long-time integration and extrapolation. However, as seen in Figure 5.3, this is not entirely the case, as the Hamiltonian PINN starts to deviate quickly from the analytic trajectory soon after extending beyond the training window.

5.2.3. LSTM-PINN

As explained before, an LSTM processes each time point as a single-step sequence. Thus, the architecture is a 2-layer LSTM with 64 hidden units, followed by a linear readout that maps to 1 output dimension. The input shape is $[\text{batch}, 1, 1]$, a batch of scalars, sequence length 1 and feature size 1. The motivation for using an LSTM is its internal gated memory cells, which can, in principle, maintain state across a sequence, making it a natural candidate for time-series and dynamical systems. The loss system is the same as the standard PINN. In practice, this variant also fails to

extrapolate, as shown in Figure 5.3, and it does not extrapolate correctly either. One reason is that the sequence length was kept at 1, thereby preventing the model from leveraging temporal dependencies.

These tests demonstrate that the Fourier PINN holds the most promise for extrapolating periodic systems. The unperturbed two-body problem is itself a harmonic oscillator with a well-defined sidereal period, making it a natural candidate for Fourier-based positional encoding. However, when perturbations such as J2, third-body effects, and SRP are included, the motion is no longer strictly periodic. These forces introduce secular drifts and long-period oscillations that cannot be captured by a single reference frequency. Thus, while the Fourier PINN's success on the spring-mass system suggests that harmonic embedding could be beneficial for encoding the dominant orbital frequency, it would not fully resolve the extrapolation problem for perturbed GEO orbits. Furthermore, the Hamiltonian PINN is unsuited for this application because SRP is a non-conservative force, meaning the total energy is not conserved and thus cannot serve as a valid Hamiltonian constraint.

5.3. Two-Body Verification

Translating the PINN to the perturbed orbital two-body problem introduces significant challenges compared to the spring-mass system. The input dimensionality increases to 7, with the six state-vector components $\mathbf{x} = [\mathbf{r}, \mathbf{v}]^\top$ and one from time, t and the output is a three-dimensional position vector. The highly nonlinear nature of orbital dynamics also requires a deeper, more expressive network architecture.

The inputs are first z-score normalized using precomputed statistics from the dataset. This is necessary because the raw state values span a wide range of magnitudes. Positions are on the order of 10^4 km, while velocities are on the order of 10^0 km/s. Without normalization, the larger values would dominate the gradient updates during training. The normalized inputs are then projected from 7 dimensions to 512 dimensions through a linear layer. This projection maps the low-dimensional input into a higher-dimensional latent space where the network has more capacity to represent the complex, non-linear relationships in orbital dynamics.

The core of the architecture consists of 8 residual blocks. Each block comprises two linear layers with layer normalization (LayerNorm) and tanh activations, connected by a skip connection from the block input to its output. The skip connection means each block only needs to learn a small correction Δx to its input rather than a full transformation. This is critical for deep networks because, without skip connections, gradients can vanish as they propagate backward through many layers, preventing the earlier layers from learning effectively. With 8 stacked blocks, the network can represent increasingly complex features of orbital dynamics while maintaining stable training. Finally, the 512-dimensional output is projected back to 3 dimensions and denormalized to recover physical position values.

The physics loss function is computed component-wise using AD to get velocity from position and acceleration from velocity. The residual is, $L_{\text{phys}} = \text{MSE}\left(\frac{\hat{\mathbf{a}}}{a_{\text{scale}}}, \frac{\mathbf{a}_{\text{true}}}{a_{\text{scale}}}\right)$ where $a_{\text{scale}} = \mu / r_{\text{GEO}}^2$ normalizes the acceleration to a dimensionless quantity. This

scaling is essential because without it, the raw acceleration values would cause numerical precision issues and an imbalance in the loss term.

Thus, after training and fine-tuning the model, the PINN can replicate a GEO orbit, including orbital perturbations, as shown in Figure 5.4, which shows the component-wise residuals between the predicted orbit and the true orbit.

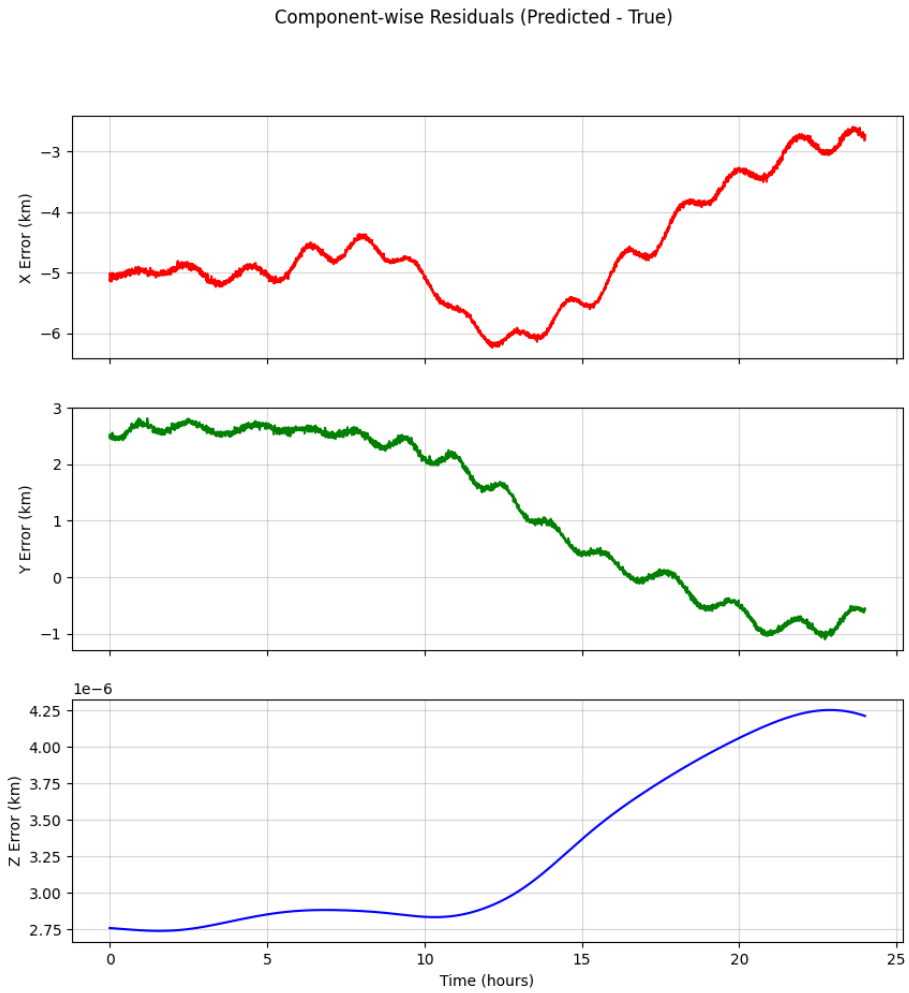


Figure 5.4: Residuals Between Predicted and Truth For GEO Reconstruction

After extensive fine-tuning, the positional residuals are on the order of a few kilometres in GEO distance. This demonstrates that the PINN can be used for orbit reconstruction, and that the errors are sufficiently small to enable basic maneuver-detection tests. Low-thrust maneuvers themselves may not be detected, but station-keeping maneuvers are modelled as impulsive burns to test whether the PINN could detect a sufficient deviation from its baseline to classify it as a maneuver.

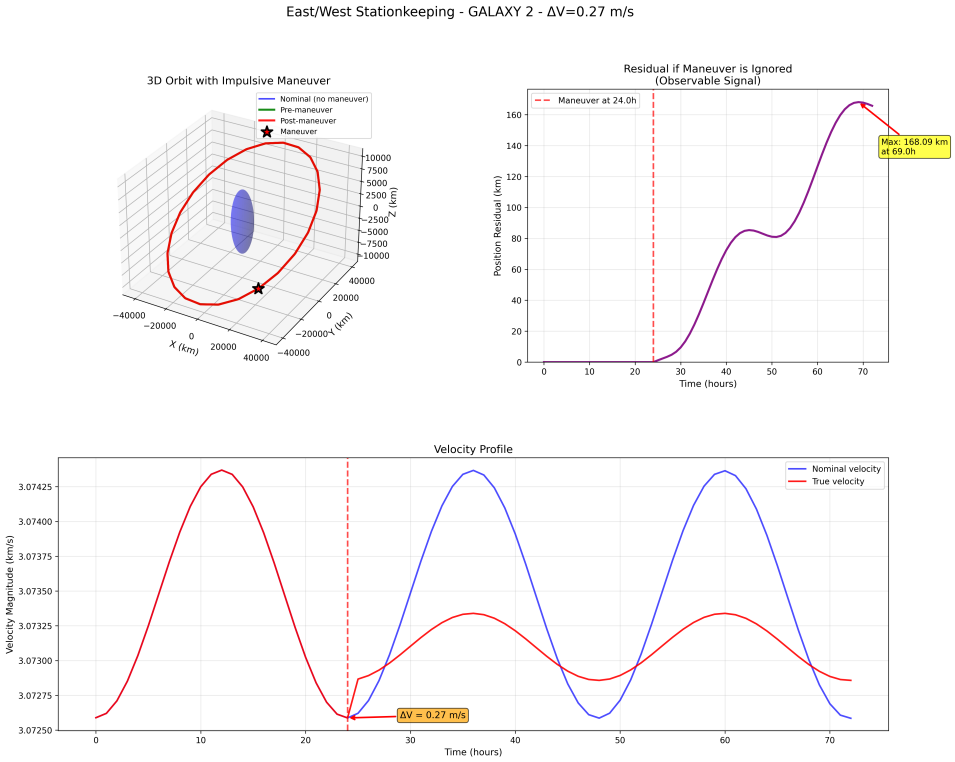


Figure 5.5: Maneuver Analysis of Worst Case East-West Stationkeeping Maneuver

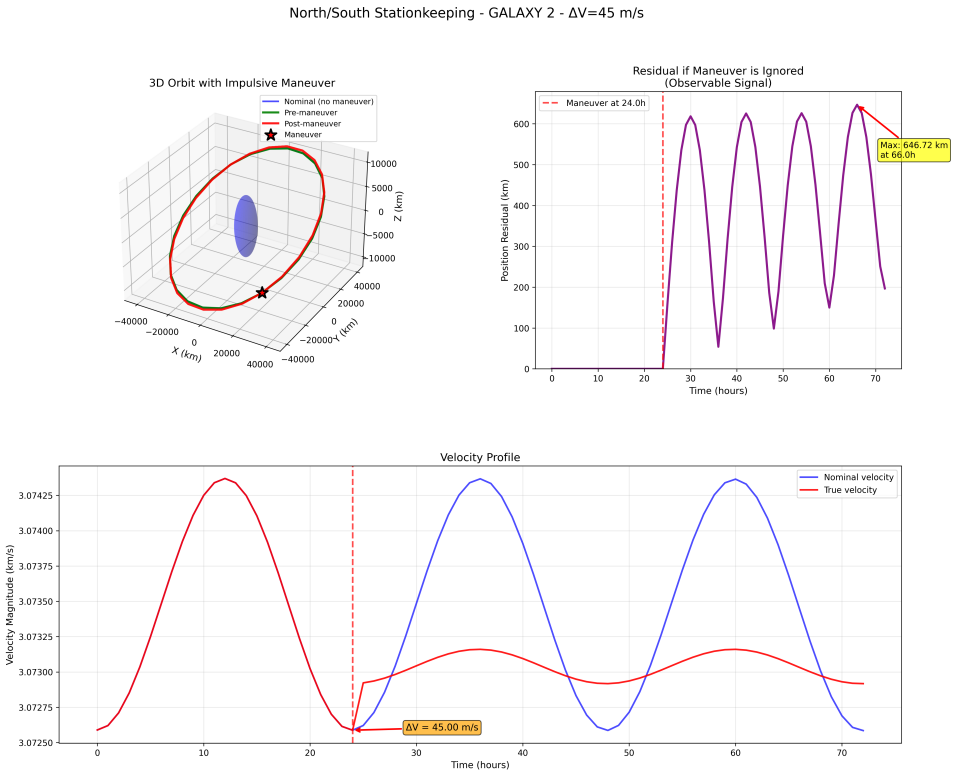


Figure 5.6: Maneuver Analysis of Worst Case North-South Stationkeeping Maneuver

Figures 5.5 and 5.6 show what would happen to the position residuals if a maneuver

occurred at the quoted stationkeeping burns [5]. The orbits are propagated with an SGP4 propagator, and after one full orbit, the stationkeeping impulsive burns are applied. The post-maneuver state is then propagated using SGP4, and the residuals are compared with the pre-maneuver state. The top right plot of both Figures 5.5 and 5.6 show the position residuals, while the bottom plot shows the velocity profile. These values are all much higher than the residual error in the PINN reconstruction, indicating that maneuver detection is indeed possible, if the burn were impulsive. However, the PINN used to show the residuals in Figure 5.4 is remodelled for the two-body problem from the standard PINN used for the spring-mass system.

Consequently, a Fourier, Hamiltonian, LSTM, and the standard PINN are compared to assess their responses to orbital dynamics. A standard PINN with Tanh activation learns a smooth mapping from time to position, but Tanh saturates asymptotically to ± 1 for large inputs. When $t > 24$ h and the network is taken out of its training distribution, the activations push into saturation, and the output collapses toward a constant as seen in Figure 5.8.

The Fourier PINN is the only one that struggles to determine the initial conditions; as such, it can identify the periodic nature but cannot recover the correct shape of the orbit. The top-right graph shows that the position error over time is significant due to this issue, whereas the bottom-left graph shows that, in the x-direction, the periodic nature is preserved. This arises from a frequency mismatch due to a slight offset between the learned frequencies and the true orbital frequency, leading the predicted orbit to drift in phase.

The Hamiltonian PINN suffers from Hamiltonian leakage. The PINN enforces energy conservation as a constraint, but it learns an approximation to the true Hamiltonian. If the learned energy is slightly overestimated by even a small fraction, integrating the equations of motion derived from this slightly wrong Hamiltonian produces an orbit with slightly more energy than reality. This is an issue with Hamiltonians as they conserve a Hamiltonian exactly, but not necessarily the correct one. Since a PINN is not a symplectic integrator, due to it being a function approximator that minimizes a loss function, there is no structural guarantee that its learned solution preserves phase-space over long periods. In practice, this means that even if the PINN's learned Hamiltonian is very close to the true one, small violations of the symplectic structure accumulate over time, causing the predicted trajectory to slowly drift off the true energy manifold. This is why the Hamiltonian PINN performs well within the training window but diverges during extrapolation.

For the LSTM-PINN, the hidden state is updated by "seeing" each timestep in sequence. Within training, the hidden state is continuously grounded by the physics loss and data at each step. At the training boundary, the hidden state contains a compressed representation of 24 hours of orbital dynamics. As the network continues to roll out beyond that boundary, the hidden state receives no corrective signal from new training data and begins to drift toward whatever the recurrent dynamics settle on, typically a fixed point rather than a sustained oscillation. The output plateauing to roughly the mean position is the LSTM's attractor behaviour dominating over the orbital dynamics. The LSTM-PINN is uniquely susceptible to attractor behaviour during

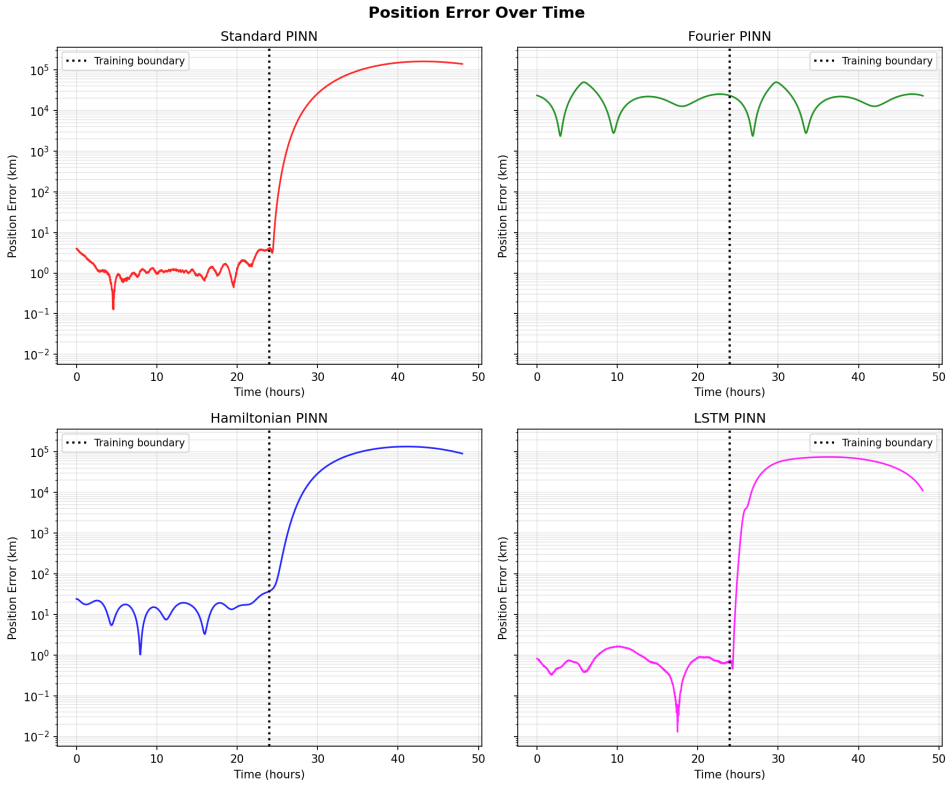


Figure 5.7: Absolute Position Error Over Time for Different PINN Models with a) Standard PINN (Top Left), b) Fourier PINN (Top Right), c) Hamiltonian PINN (Bottom Left), and d) LSTM-PINN (Bottom Right)

extrapolation because the LSTM is itself a recurrent dynamical system, whose gated update equations are inherently dissipative. As time progresses beyond the training window, the forget gate progressively removes energy from the hidden state, drawing it toward a fixed-point attractor of the LSTM's transition function, and causing the predicted output to collapse to a constant rather than continue oscillating. The three other architectures are immune to this; the Fourier PINN maps all time inputs onto the unit circle via sine and cosine functions, ensuring no out-of-distribution inputs regardless of how large t grows; the Standard and Hamiltonian PINNs are feedforward networks with no internal state and therefore no recurrent attractor dynamics, with the Hamiltonian formulation additionally embedding energy conservation directly into the training objective.

A side-by-side comparison of the methods mentioned above is shown in Figures 5.7 and 5.8. Figure 5.7 shows the position error over time for each model. The Fourier PINN is the only one that retains the system's periodic dynamics beyond the training window; however, it also fails to properly capture the system's dynamics and thus exhibits high error during the training window. Furthermore, the error on the Hamiltonian PINN increases smoothly and monotonically after the bounds of the training window, showing how small violations of symplectic structure accumulate over time. The orbit is slowly leaving the correct energy manifold and not catastrophically failing, as the Standard PINN does. Figure 5.8 shows the x-position over time for each model, highlighting some of the issues discussed in the models. Namely, with the standard

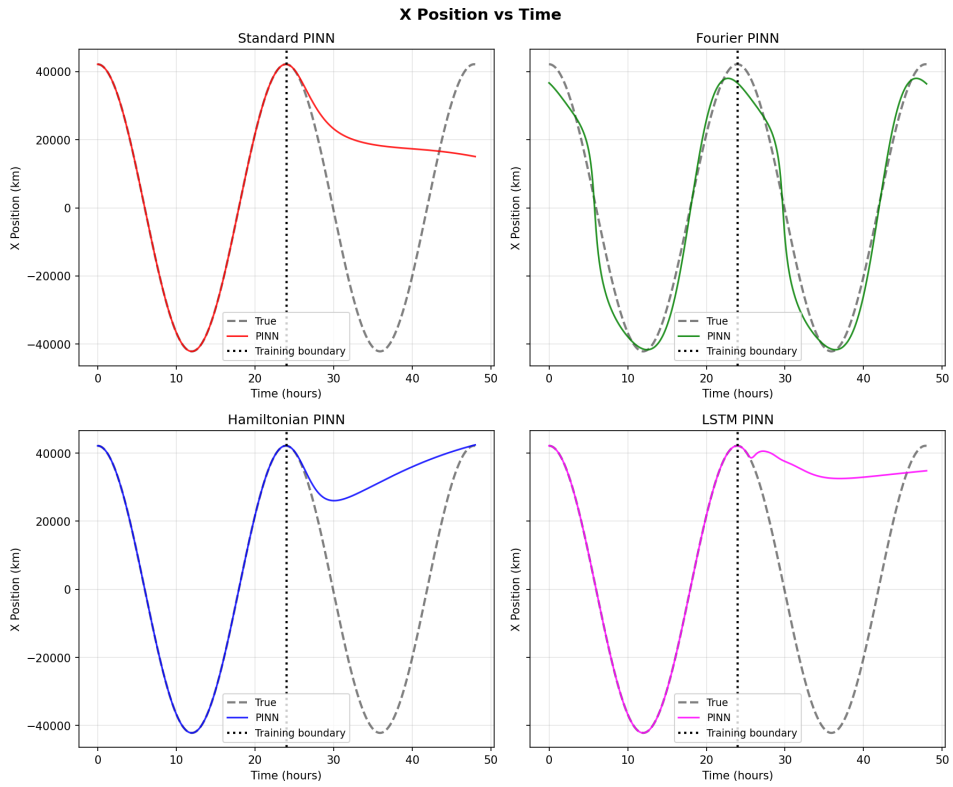


Figure 5.8: Position in X Over Time for Different PINN Models with a) Standard PINN (Top Left), b) Fourier PINN (Top Right), c) Hamiltonian PINN (Bottom Left), and d) LSTM-PINN (Bottom Right)

PINN, the output collapses to a constant value because the activations approach saturation. The LSTM-PINN tracks the true trajectory very well until the training boundary, at which point it immediately ceases oscillation, exhibiting the fixed-point attractor.

Thus, none of these models is usable for extrapolation, thus changing the overall problem from an extrapolation problem to an interpolation problem, leading to the inverse problem described in the Chapter 4. Furthermore, the training for each of these models is on a single orbit, with a specific initial condition. When the initial condition is varied, the loss plateaus at local minima, resulting in very high errors. Figure 5.9 demonstrates this. This is the same plot as 5.4, but with the initial condition set to a different point on the GEO orbit as opposed to the trained point, which is at $\mathbf{r} = [42000, 0, 0]^\top$ km and using the trained weights of the standard PINN to propagate the test orbit.

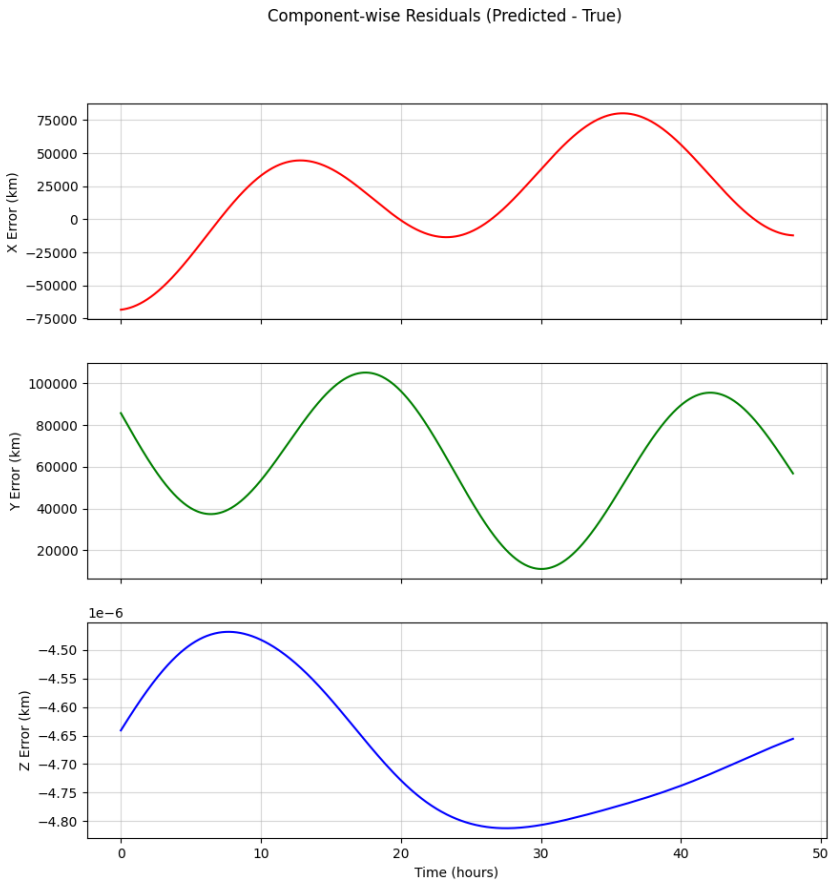


Figure 5.9: Residuals When the PINN is Used to Reconstruct a General GEO Orbit

The failure of the generalized PINN to reconstruct orbits across varying initial conditions highlights the 'failure to generalize' problem prevalent in deep learning for high-precision astrodynamics. While a network can be overfitted to a single orbit with residuals of a few kilometres, it cannot yet replace numerical integrators for high-fidelity propagation across the GEO belt. Consequently, this study pivots from using PINNs as propagators to using them as residual estimators. By employing a DOP853 integrator to establish a high-fidelity reference trajectory and utilizing the PINN solely to solve the inverse problem of thrust recovery, the architecture leverages the strengths of both frameworks: the deterministic precision of classical mechanics and the flexible optimization of neural networks.

6

Results & Analysis

6.1. Dataset Creation

According to the methodology presented in Chapter 4, the first step was to create the dataset. One synthetic dataset was created using the DOP853 scheme, with random initial conditions along the GEO regime. All are circular orbits with identical altitude and velocity. The RAAN is drawn from a uniform distribution between 0 and 360 degrees, while the orbital inclination is drawn from a uniform distribution between 0 and 5 degrees. The RAAN and inclination are randomized independently of each other and of the class label, ensuring that all three classes share the same distribution of initial orbital geometries. This provides diversity in the Sun-relative geometry across the dataset, preventing the classifier from associating a particular solar illumination condition with a specific class. The remaining characteristics for the dataset generation are presented in the following tables. Table 6.1 presents the integration parameters and initial conditions common to all three classes. The arc length is drawn from a uniform distribution between 16 and 48 hours, and the state vector is sampled every 600 seconds (10 minutes). The resulting trajectories are noise-free; a noisy variant of the dataset is created separately by adding zero-mean Gaussian noise independently to each component of the position and velocity vectors. Position noise, with a standard deviation of 50 m, and velocity noise, with a standard deviation of 5 mm/s, are adopted as representative of high-cadence radar tracking at GEO distances [**tombasco_orbit_nodate**].

Parameter	Value	Note
Integration Method	DOP853	Chosen integrator for propagation
rtol	1×10^{-10}	Relative tolerance
atol	1×10^{-10}	Absolute tolerance
Minimum Arc Length	16 h	Lower Bounds of the Observation Arc
Maximum Arc Length	48 h	Upper Bounds of the Observation Arc
Sampling Interval	600 s	Time Between Observations
RAAN	$\mathcal{U}[0, 360]$	RAAN Initialization
Inclination	$\mathcal{U}[0, 5]$	Inclination Initialization

Table 6.1: Initial Conditions and Parameters for Dataset Generation

For each class, the specific parameters are presented in table 6.2. The thrust is applied at the beginning of the arc and is of constant magnitude and direction for the duration of the arc. It is important to note that the Class 1 thrust profiles do not model realistic stationkeeping maneuvers, which are typically executed in specific directions relative to the orbit (e.g., along-track for East-West corrections, cross-track for North-South corrections) at predictable intervals. Instead, a constant thrust of lower magnitude than that typically used for stationkeeping is applied in a random inertial direction for the duration of the arc for generality [20]. This deliberate simplication serves two purposes. First, it tests the classifier and PINN under a worst-case scenario in which no assumptions about the thrust direction can be exploited, whereas a realistic stationkeeping maneuver with a known direction and magnitude would be easier to detect. Second, it ensures that the model generalizes to unknown maneuvers, such as orbit transfers or proximity operations, in which the thrust direction is unknown and not aligned with any predictable station-keeping strategy.

Parameter	Value
Nominal A/m	0.02 m^2/kg
Thrust	None

(a) Class 0 - Nominal Parameters

Parameter	Value
Nominal A/m	0.02 m^2/kg
Thrust Magnitude	$\mathcal{U}[10^{-10}, 10^{-8}]$ km/s^2
Thrust Direction	Random Direction, Isotropic in ECI

(b) Class 1 - Low Thrust

Parameter	Value
A/m	$\mathcal{U}[0.005, 0.08]$ m^2/kg
Thrust	None

(c) Class 2 - Mismodeled SRP

Table 6.2: Summary of Parameters for Each Class

In total, 8400 samples are produced, split into 6000 for training, 1200 for validation, and 1200 for testing, with each split containing equal numbers of scenarios for each class. Thus, the 6000 training samples are 2000 each for the three classes, and the same holds for validation and testing. Figure 6.1 shows the position and velocity magnitudes over time for a representative sample of each class. The nominal orbit (Class

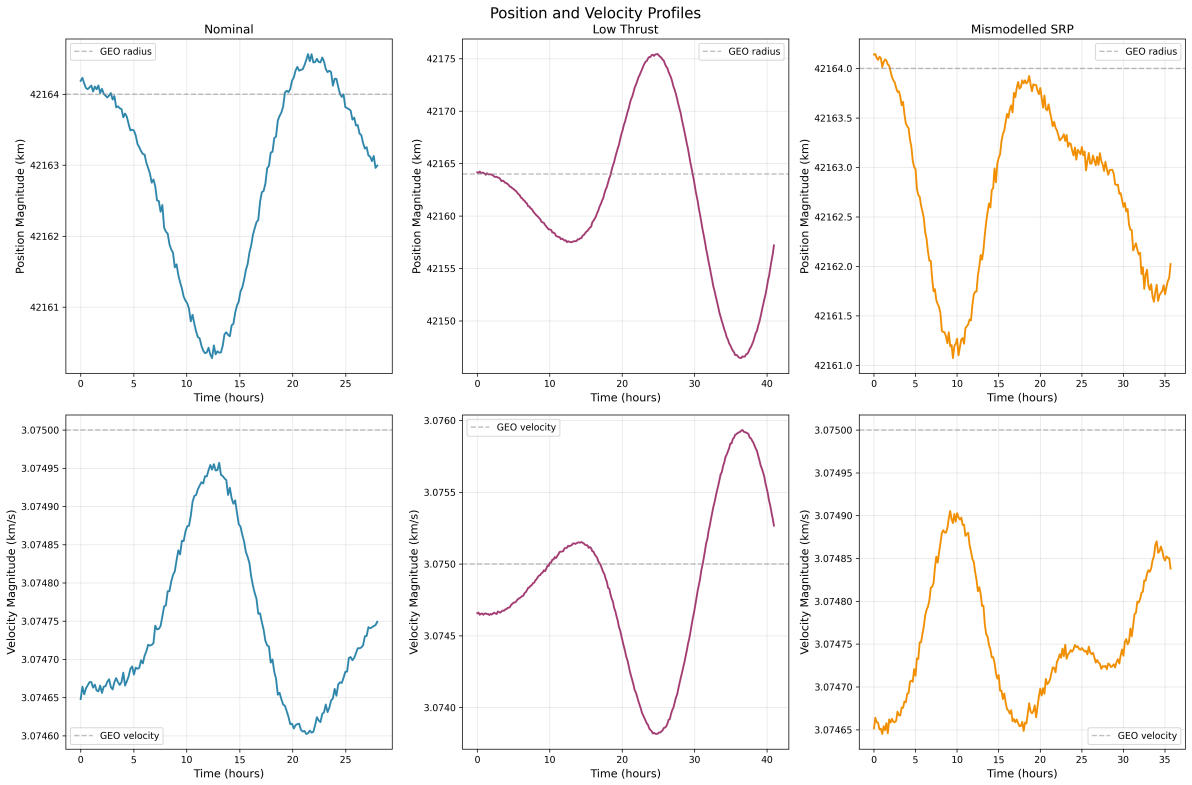


Figure 6.1: Position and Velocity Profiles for a Sample Orbit of Each Class

0) exhibits small, smooth oscillations around the GEO reference values, driven by the natural perturbations. The low-thrust orbit (Class 1) shows a gradual drift in both position and velocity away from the GEO reference, consistent with the continuous application of a constant thrust. The mismodeled SRP orbit (Class 2) displays oscillatory behaviour similar to the nominal case but with a different amplitude, reflecting the altered A/m ratio affecting the SRP force. These profiles illustrate the subtle nature of the differences between the three classes; at the scale of absolute position and velocity, the trajectories are nearly indistinguishable, which is precisely why the feature engineering described in Section 4.3.1 is necessary to amplify the discriminative signals.

Figure 6.2 presents three-dimensional views of sample trajectories for each class. All three classes trace nearly identical paths at GEO altitude, further emphasizing that the raw state vectors provide minimal discriminative information. The differences between classes manifest as deviations on the order of kilometres over the arc duration, which are imperceptible at the scale of the full orbit.

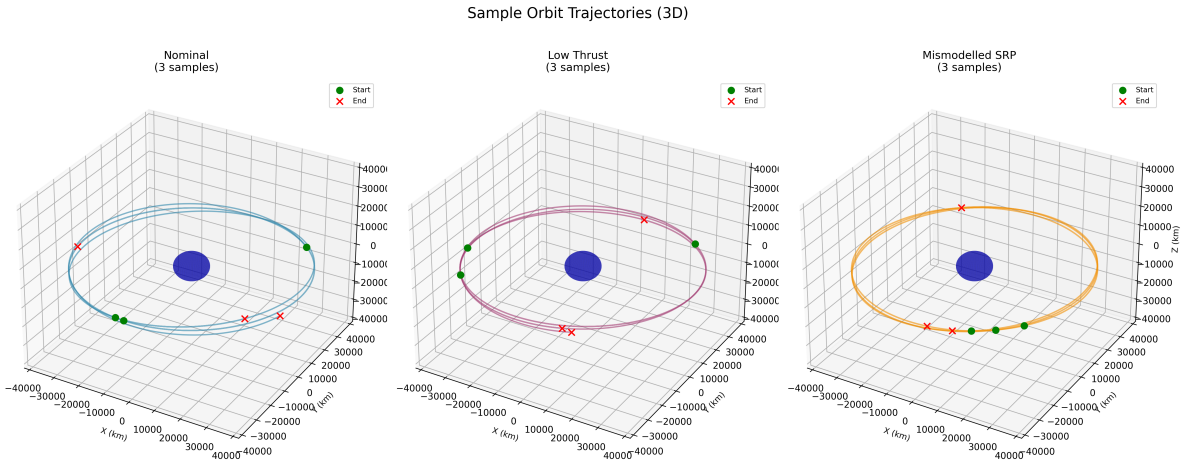


Figure 6.2: 3D Sample Orbits

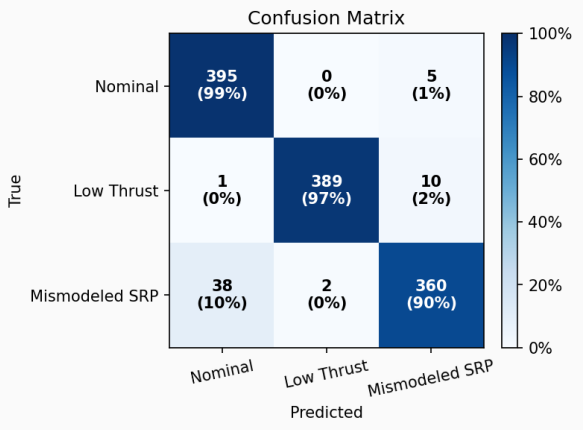
6.2. Classifier

Using the methodology that was discussed previously in Chapter 4, the LSTM classifier is trained on both the noise-free and noisy datasets presented in Section 6.1. Each trajectory's raw state vectors $\mathbf{x} = [r, v]^\top$ are mapped to the 14-dimensional feature vector defined in Table 4.1, normalized using statistics computed from the training split. The classifier receives the full feature time-series for each arc and outputs a probability distribution over the three classes. The model is trained on 6000 samples (2000 per class) using cross-entropy loss with AdamW, and validated on 1200 samples. The best model based on the validation loss is selected and it is then evaluated on a held-out test set of 1200 samples.

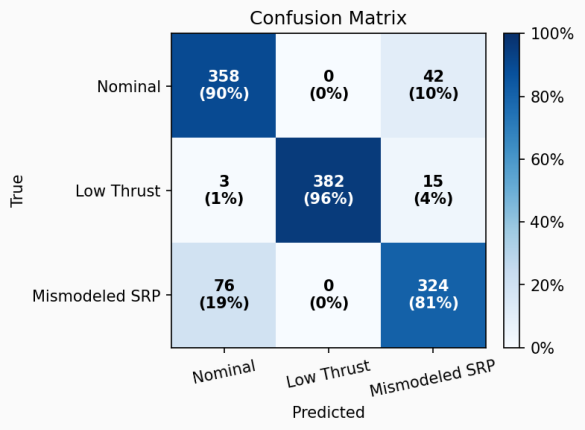
Figure 6.3 shows the training progress on the noise-free dataset. Both the training and validation losses decrease over the first 150 epochs, then plateau, with the best validation accuracy of 95.3% reached at epoch 150, referring to the number of correct predictions over the total number of samples. The gap between the training and validation losses after epoch 150 indicates mild overfitting, which is mitigated by early stopping, which selects the model with the highest validation performance.

Figure 6.3: Training and Validation Loss (left) and Accuracy (right) Curves for the LSTM Classifier

6.2.1. Test Set Results



(a) Confusion Matrix of LSTM Trained on Noise-Free Data



(b) Confusion Matrix of LSTM Trained on Noisy Data

Figure 6.4: Confusion Matrices of Both LSTMs

The classification results on the test set are provided in the confusion matrix in Figure 6.4. This matrix is used to evaluate the performance of a classification model by comparing predicted values against the ground truth, summarizing correct and incorrect predictions. An indicator of a well-trained network is that most of the information lies on the main diagonal, indicating that the truth and predictions agree, or that most predictions are true positives. Information in the off-diagonal entries in that a class's column is an indicator of a false positive, meaning that the classifier predicted a sample as a certain class, when it is not that class. For example, a sample that is not a low-thrust maneuver is predicted as a low-thrust maneuver. This matters because it would trigger an unnecessary investigation into a satellite that is not actually maneuvering. Similarly, the information in the off-diagonal entries of a class's row corresponds to false negatives, meaning that the sample is a certain class but the classifier predicted it as something else. For example, a low thrust maneuver sample is incorrectly predicted as a mismodeled SRP. This is important because the maneuver remains undetected. Figure 6.4a shows the confusion matrix for the noise-free dataset, in-

dicating that most of the classifications lie on the diagonal. This shows that the true positive rate is very high, providing a good baseline for comparing the confusion matrix of the noisy dataset, shown in Figure 6.4b. The number of false positives increases significantly here; however, all cases involve the mismodeled SRP being classified as nominal. Additionally, the number of false negatives has increased because slightly more low-thrust maneuvers are misidentified. Therefore, the classifier demonstrates strong overall performance on both datasets, with the noise-free accuracy of 95.3% degrading to 88.7% under realistic measurement noise. Critically, the model's ability to detect low-thrust maneuvers remains robust, with most of the performance degradation isolated to the SRP-nominal boundary, a regime where the physical signals are fundamentally ambiguous.

The quality of a classifier can also be assessed using precision, recall, and F1 Score. Precision is a measure of how correct the predictions are. The formula is given by:

$$\frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

where TP is true positive and FP is false positive. Recall is the proportion of positive outcomes that are predicted. It is given by:

$$\frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

where FN is false negative. Finally, the F1-score is the harmonic mean of precision and recall. This is given by:

$$\frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

This is standard for measuring the validity of a model, and Figures 6.5 and 6.6 show the values for the trained classifiers. Both figures show that the classifier achieves very high scores across all three classes, with an average F1 score of 94.5% for the classifier trained on the noise-free dataset and 88.8% for the classifier trained on the noisy dataset. These high F1 scores demonstrate the robustness of the classifier, especially with the highest precision, recall and F1 scores shown for the low-thrust maneuver.

6.2.2. Ablation Study

To quantify the contribution of each feature to the classifier's performance, a single-feature ablation study is conducted. In this study, each feature is removed from the input, the classifier is retrained, and the resulting accuracy is compared with the baseline of 88.7% obtained with all 14 features on the noisy dataset. A larger drop in accuracy indicates a more important feature. Figure 6.7 presents the results ranked by accuracy drop. Eccentricity is the single most important feature, with its removal resulting in a 42.7 percentage-point drop in accuracy. This is consistent with the physical analysis, as eccentricity captures whether the orbit is becoming elliptical, which is a direct indicator of both thrust-induced drift and SRP-driven oscillations. Radial velocity and RTN radial velocity rank second and third, each causing approximately

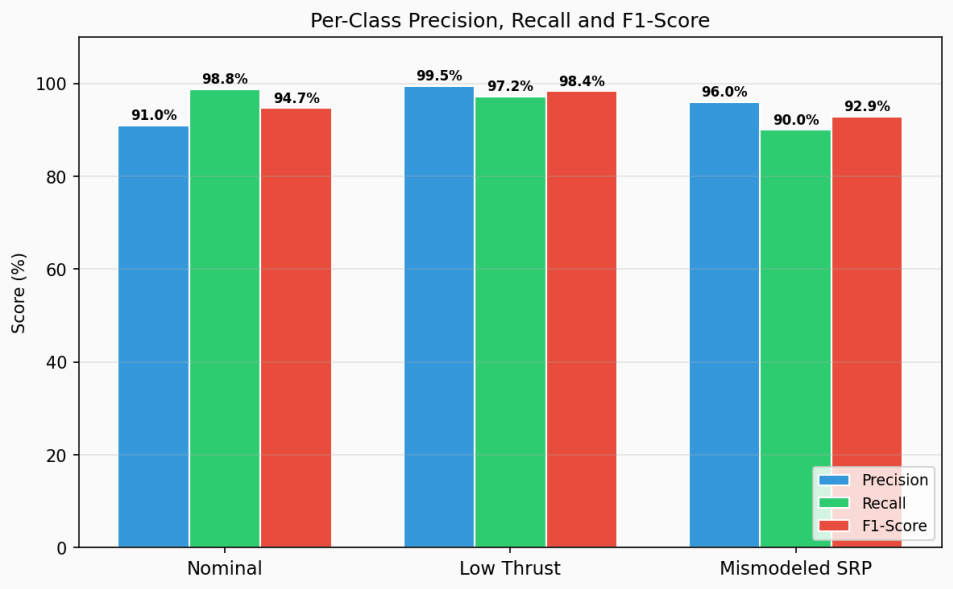


Figure 6.5: Precision, Recall and F1-Score for Each Class for LSTM Trained on Noise-Free Data

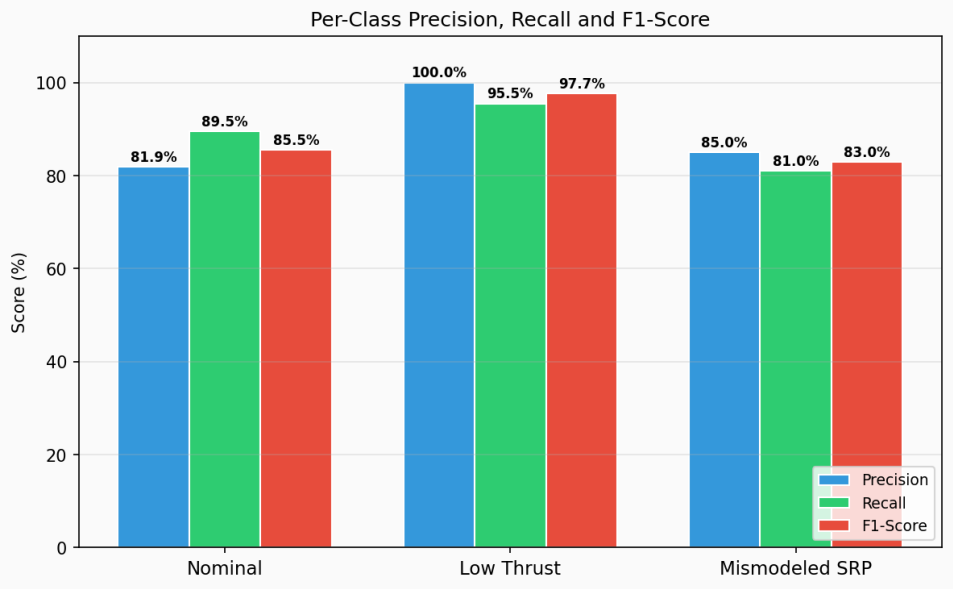


Figure 6.6: Precision, Recall and F1-Score for Each Class for LSTM Trained on Noisy Data

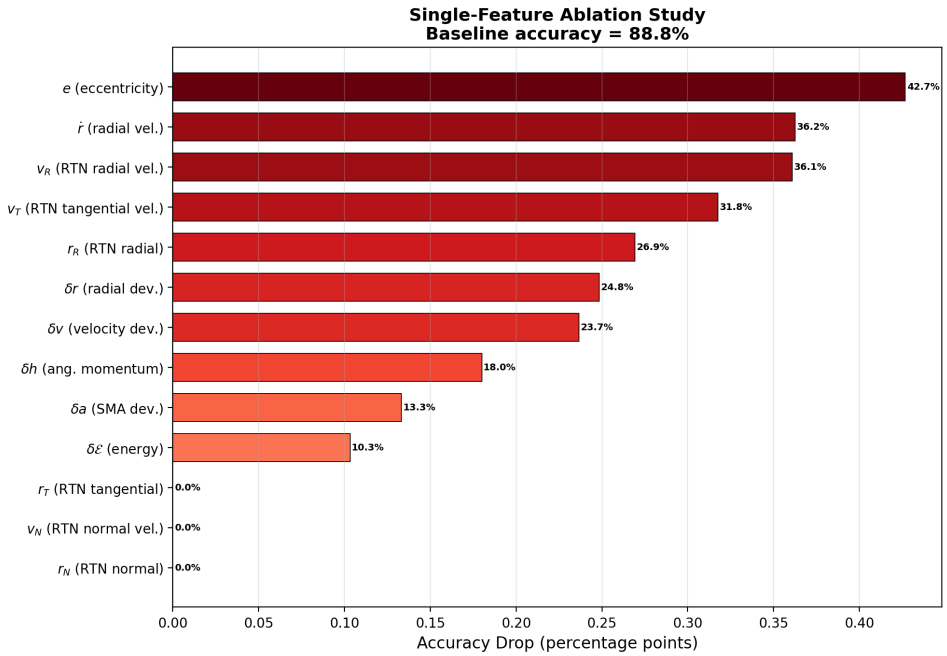


Figure 6.7: Single-Feature Ablation Study

a 36 percentage-point drop. These features capture the rate of change of the orbit shape, providing complementary temporal information to the eccentricity. There is a redundancy here, as these two are mathematically equivalent for circular orbits; hence, the very similar accuracy drops in the ablation study. This redundancy was an oversight in the feature design, meaning the classifier effectively has 13 independent features rather than 14. The RTN radial components and radial deviation also contribute significantly, with decreases of approximately 27% to 24%, respectively. Notably, the three lowest-ranked features, RTN tangential position, RTN normal position, and RTN normal velocity, contribute no measurable improvement, suggesting they could be removed without affecting performance.

Figure 6.8 breaks down the feature importance per class, revealing that the features serve different roles for different classification tasks. The nominal class is highly sensitive to nearly all features, with eccentricity removal alone causing an 88% accuracy drop, indicating that the classifier relies heavily on eccentricity to establish what a nominal orbit looks like. In contrast, the low-thrust class shows minimal sensitivity to any single feature, with all individual drops below 5%. This indicates that the classifier identifies low-thrust maneuvers by combining temporal patterns across multiple features rather than relying on any single discriminative signal. The mismodeled SRP class falls between these two extremes, with eccentricity (40.8%), radial velocity (48.2%), and RTN radial velocity (46.2%) being the primary discriminators. This aligns with the physical expectation that mismodeled SRP manifests primarily through altered radial forcing, which directly affects the eccentricity and radial velocity profiles. The robustness of the low-thrust detection to single-feature removal is a noteworthy finding. It suggests that the LSTM's temporal attention mechanism is learning to integrate subtle, correlated signals across multiple features.

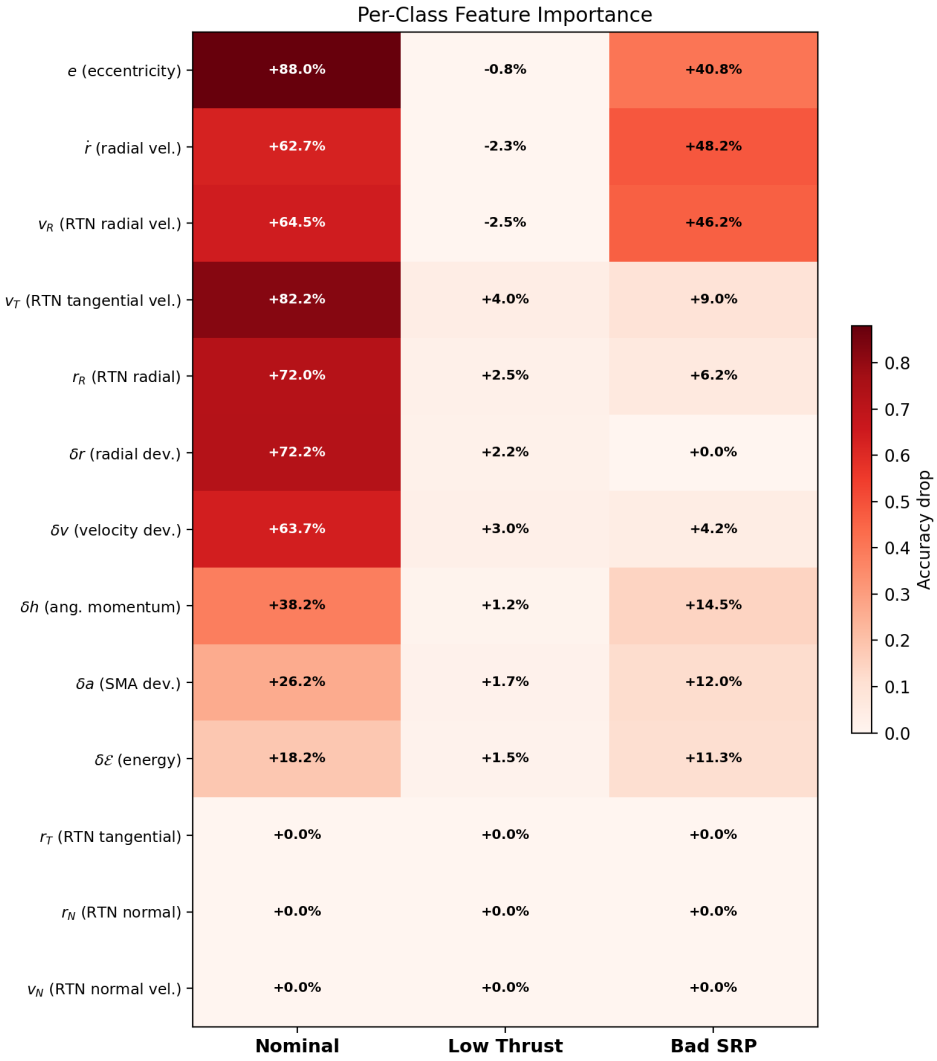
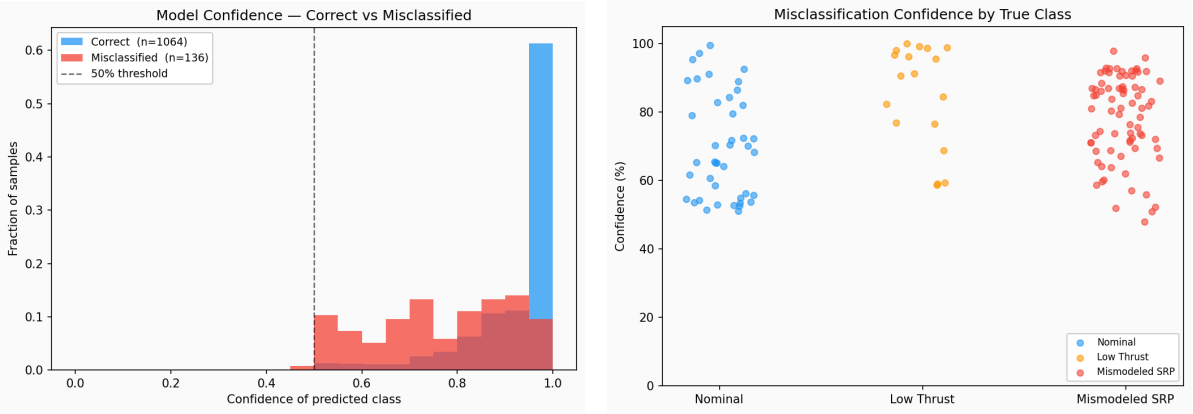


Figure 6.8: Ablation Study on Each Class



(a) Model Confidence Normalized

(b) Misclassification Confidence per Class

Figure 6.9: Overview of Model Confidence on the LSTM Trained on Noisy Data

6.2.3. Misclassification Investigation

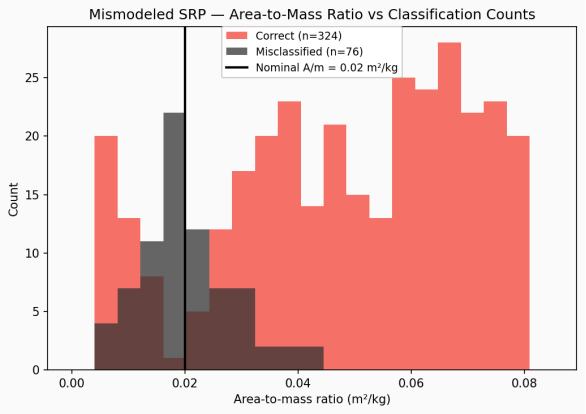
As shown in the confusion matrices in Figure 6.4, most misclassifications are from a mismodeled SRP orbit being misclassified as a nominal orbit or vice versa. Investigating the misclassifications further, Figure 6.9 gives an overview of the classifier performance. The model's confidence is defined as the maximum softmax probability over the three output classes, i.e., the probability assigned to the predicted class. The histogram in Figure 6.9a shows the distribution of the model's confidence in its predicted class, stratified by whether the sample is correctly classified or misclassified, with both being superimposed on one another. The y-axis gives the fraction of samples in each confidence bin, so the bars for each group sum to 1. Correctly classified samples are concentrated near confidence = 1, indicating the model is highly certain when it is right. Misclassified samples are spread more broadly, but many still fall within high-confidence regions, indicating that the model is often wrong yet remains confident. Looking further, by splitting the misclassified confidences by class in Figure 6.9b, it is clear that the mismodeled SRP case is the one in which the model was most confident but wrong, as indicated by the high concentration of data points towards the top of the plot for this class. This indicates a fundamental ambiguity in the feature space, rather than a model deficiency.

Issues arise when the A/m ratio of the mismodeled SRP deviates by only 1-15% from the nominal ratio, suggesting a noise floor. The model is very confident that the orbit is nominal because it exhibits very similar characteristics, but because the A/m ratio is perturbed, this assumption is incorrect. This is shown in the metadata analysis for the noisy dataset in Figure 6.10a. The figure shows the density distribution of the A/m ratio for correctly classified and misclassified Class 2 samples. There is a large spike in the number of misclassified, Class 2 samples near the nominal A/m, with the number of misclassified A/m progressively decreasing as the perturbed A/m moves farther from the nominal. While this supports the noise floor, this is not necessarily an issue with the model. This is a dataset ambiguity, since the accumulation of errors from the mismodeled SRP results only in a deviation from the nominal trajectory of a few centimetres at most, which is well below the detectable threshold, and thus these errors are almost impossible to overcome. This is shown in Figure 6.11, where

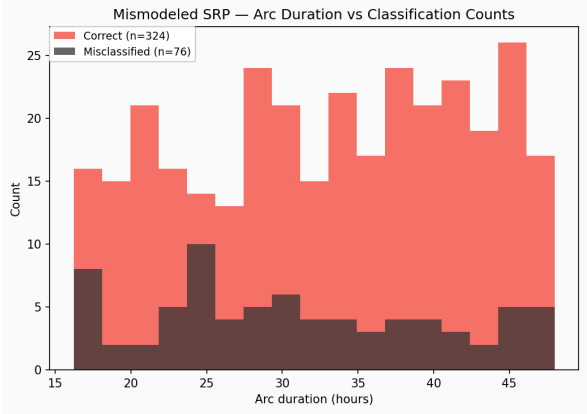
the position deviations from the mismodeled SRP cases and the background nominal cases are very similar; thus, the misclassified lines genuinely appear to be nominal to the classifier. Thus, these orbits feature oscillation amplitudes comparable to naturally occurring orbital perturbations, providing no discriminative signal above the noise floor. Another possible reason for the misclassification of the SRP is that the arc length is too short, so errors do not have enough time to accumulate and produce a noticeable difference. As such, the model becomes uncertain and picks the wrong class, as is also seen in the metadata analysis in Figure 6.10b.

There are also cases in which a low-thrust maneuver is classified as a mismodeled SRP. The reason for this is that the thrust is directed close to the Sun's direction. Furthermore, because the acceleration from the thrust is very close in magnitude to that from SRP, the model becomes confused and assumes it is a mismodeled SRP, thereby creating a geometric degeneracy. Thus, the model knows something is wrong, but it cannot distinguish which signal is causing it because both produce the same kind of slowly varying, Sun-geometry-correlated perturbation.

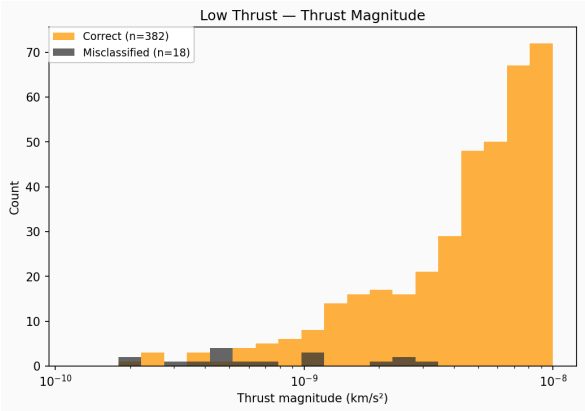
There are also a few cases where a low-thrust maneuver is misclassified. Figure 6.10c shows that the distribution of misclassified low-thrust maneuvers is highest when the magnitude is the least. The figure shows the counts of misclassified and correctly classified low-thrust maneuvers, similar to those in the previous figure. However, as seen in the confusion matrices in Figure 6.4, the total number of misclassified low-thrust maneuvers is small, which explains the low overall counts. Despite this, the majority of misclassified maneuvers occur at lower thrust magnitudes, indicating why they are misclassified.



(a) Distribution of Area-to-Mass Ratio for Successes and Failures of the Mismodeled SRP Class by Counts



(b) Distribution of Arc Lengths for Successes and Failures of the Mismodeled SRP Class by Counts



(c) Distribution of Acceleration Magnitude for Successes and Failures of the Low Thrust Maneuver Class by Counts

Figure 6.10: Metadata Analysis

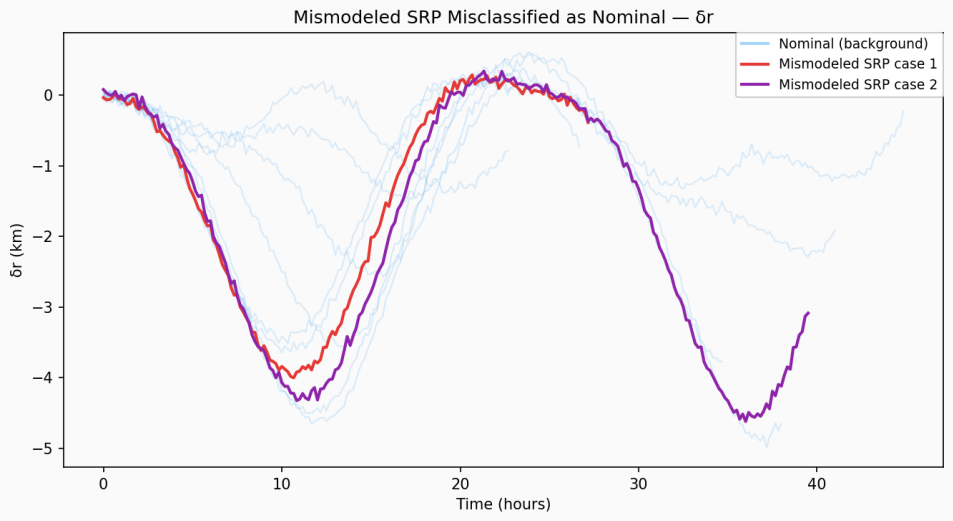


Figure 6.11: Feature Trajectories of Position Deviation when Mismodeled SRP is Incorrectly Classified as Nominal

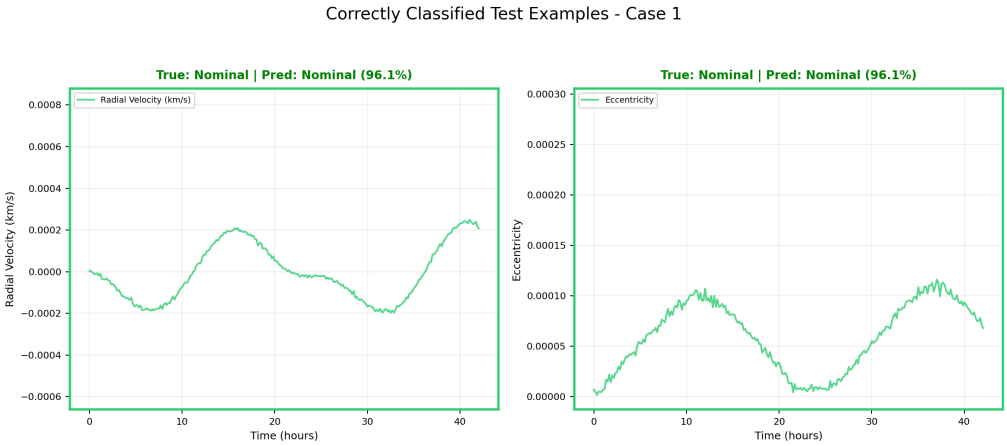
The metadata analysis reveals that misclassifications are primarily concentrated in

scenarios featuring low-magnitude thrust or minimal A/m deviations. This identifies a detectability threshold at which the anomalous signal is submerged beneath the Gaussian noise floor of the observations. Based on the analysis of Figure 6.10a, this floor is at around $\pm 15\%$ of the nominal A/m ratio as a conservative estimate. For maneuvers, this floor is at 6×10^{-9} km/s^2.

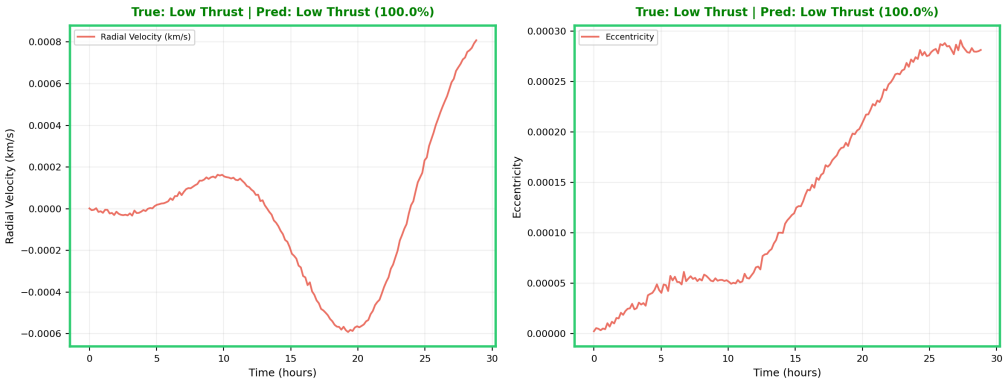
6.2.4. LSTM Decision Example

To provide physical intuition for the classifier's decision, Figures 6.12 and 6.13 present representative examples of correct classifications and misclassifications, showing the two most discriminating features identified in the ablation study: eccentricity and radial velocity. Figure 6.12 shows three correctly classified noise-free examples. The nominal orbit (Figure 6.12a) exhibits smooth, bounded oscillations in both features, with eccentricity remaining on the order of 10^{-5} and radial velocity on the order of 10^{-4} km/s. The classifier correctly identifies this, though with relatively low confidence (57.8%), suggesting that the sample lies near a decision boundary. The low-thrust orbit (Figure 6.12b) shows a qualitatively different signature. The eccentricity increases secularly to approximately 3.5×10^{-4}, an order of magnitude larger than in the nominal case, while the radial velocity exhibits an increasing amplitude, consistent with an increasingly elliptical orbit. The classifier identifies this with 100% confidence, as the secular growth provides an unambiguous discriminative signal. The mismodeled SRP case (Figure 6.12c) displays oscillatory behaviour in both features, similar in shape to the nominal case, but with a notably different amplitude envelope. The eccentricity oscillates with a growing peak amplitude across successive orbits, and the radial velocity shows a corresponding increase in oscillation magnitude. This amplitude modulation, driven by the altered A/m changing the SRP force, provides the discriminative signal that the classifier identifies with 99.5% accuracy.

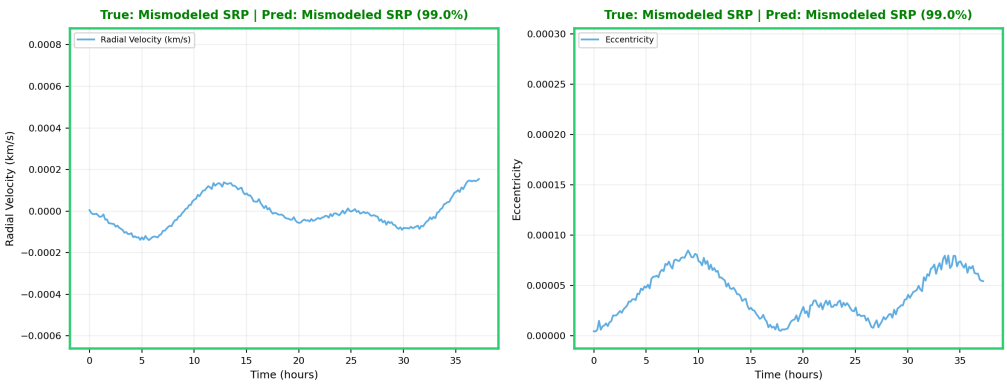
Figure 6.13 shows two misclassified noise-free examples that illustrate the failure modes identified in the confusion matrix. In Figure 6.13a, a low-thrust maneuver is classified as mismodeled SRP with 98.7% confidence. Comparing the eccentricity profile to the mismodeled SRP case, in Figure 6.12c reveals why: the thrust direction produces an oscillatory eccentricity pattern rather than the secular growth seen in Figure 6.12b, mimicking the periodic signature that SRP would produce. The classifier detects an anomaly but attributes it to the wrong source. In Figure 6.13b, a mismodeled SRP orbit is misclassified as nominal with 75.2% confidence. Both the eccentricity and radial velocity profiles are nearly indistinguishable from the nominal case in Figure 6.12a, confirming the A/m deviation is too small to produce a detectable signal above the perturbation background. This corresponds to the detectability noise floor identified in the metadata analysis.



(a) Correctly Classified Nominal Class

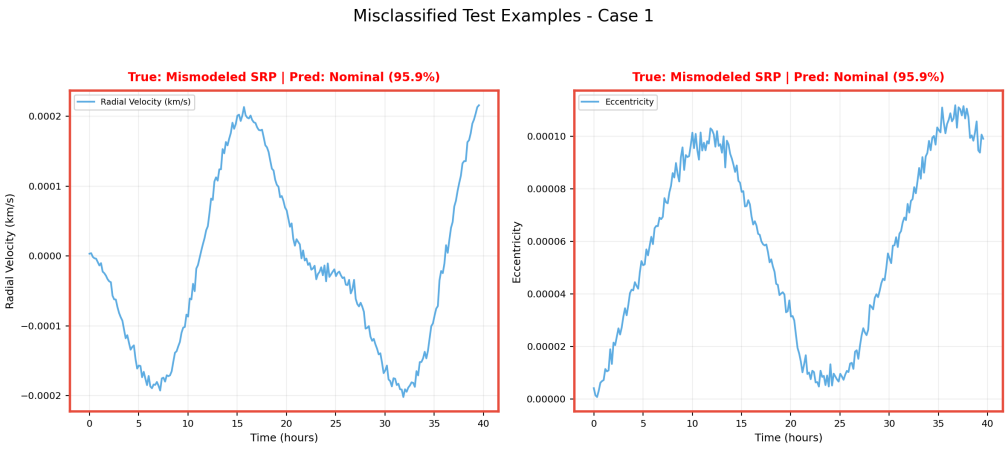


(b) Correctly Classified Low Thrust Maneuver Class

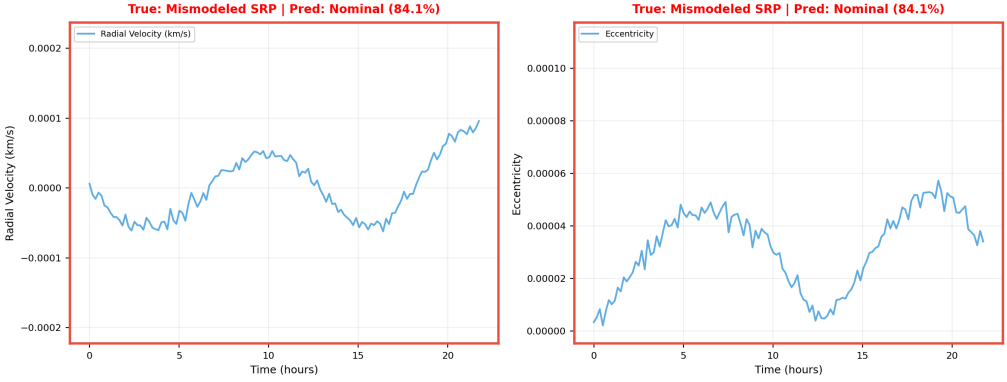


(c) Correctly Classified Mismodeled SRP Class

Figure 6.12: Correctly Classified Test Examples



(a) Misclassified Low Thrust Maneuver Class



(b) Misclassified Mismodeled SRP Class

Figure 6.13: Misclassified Test Examples

6.2.5. Early Detection

An operationally relevant metric for any maneuver detection system is the speed with which it can identify an anomaly, not just whether it can detect it given a full observation arc, but also the minimum observation time required for a confident detection. To assess this, the LSTM classifier trained on the noisy dataset is evaluated on progressively longer sub-arcs of each test trajectory. Starting from the initial epoch, the observation window is incrementally extended by ten minutes, and the classifier is queried at each step. A detection is registered when the classifier first assigns the correct class with a confidence exceeding 70% based on the results shown in Figure 6.9a.

Figure 6.14 shows the cumulative detection rate as a function of elapsed observation time for all three classes. Low-thrust maneuvers are detected earliest, with 50% of cases identified within 8.4 hours and 90% within 16.5 hours of observation. This is consistent with the secular nature of the thrust signal, as the energy drift and eccentricity growth accumulate monotonically, providing an increasingly strong discriminative signal over time. Nominal orbits require longer observations to be classified with confidence, achieving 90% detection at approximately 25 hours, as the classifier needs

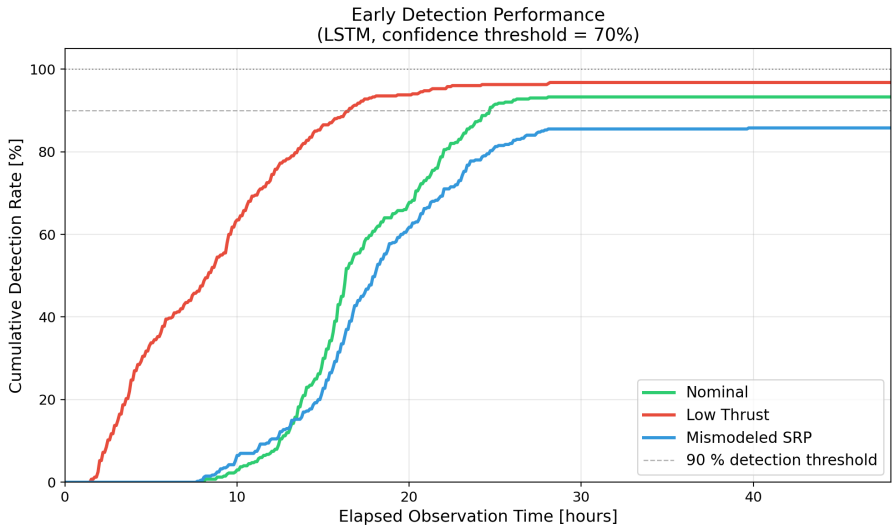


Figure 6.14: Cumulative Detection Rate Curves for Each Class with 70% Confidence Threshold

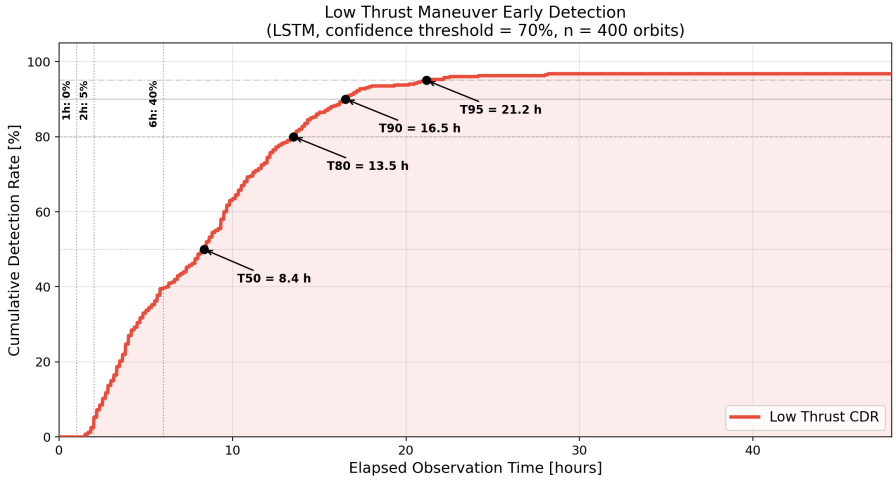


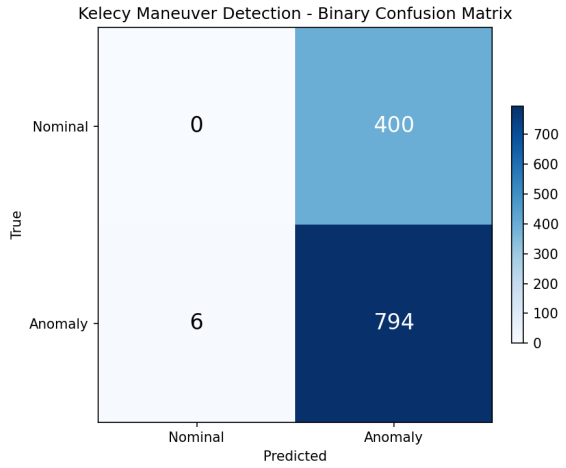
Figure 6.15: Cumulative Detection Rate Curve for Maneuver Detection with Different Confidence Thresholds

sufficient arc length to rule out anomalous trends. The mismodeled SRP class is the slowest to detect, and plateaus below 90%, consistent with the detectability limitations identified in the previous section, i.e. many of these cases are never confidently classified because their signal remains below the noise floor regardless of arc length Figure 6.15 provides a detailed view of the low-thrust detection timeline. After 2 hours of observation, only 5% of low-thrust maneuvers are detected, as the accumulated deviation is still too small to distinguish from natural perturbations. By 6 hours, 40% are detected, and the median detection time (T50) is 8.4 hours or around one third of the GEO orbital period. The 80th- and 90th-percentile detection times are 13.5 and 16.5 hours, respectively, with 95% of cases detected within 21.2 hours. The curve plateaus at approximately 96%, indicating that the remaining 4% of low-thrust cases have thrust magnitudes too small to produce a detectable signal even over a full 48-hour arc. These results provide a practical guideline for operational deployment: a surveillance system using this classifier could expect to flag the majority of low-thrust

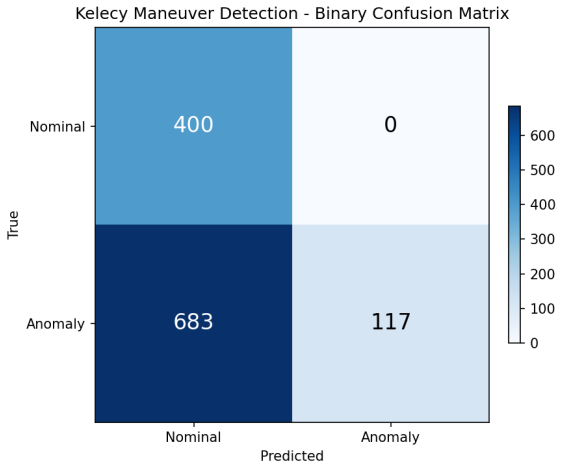
maneuvers within half a GEO orbital period.

6.2.6. Comparison With Geometric Analysis

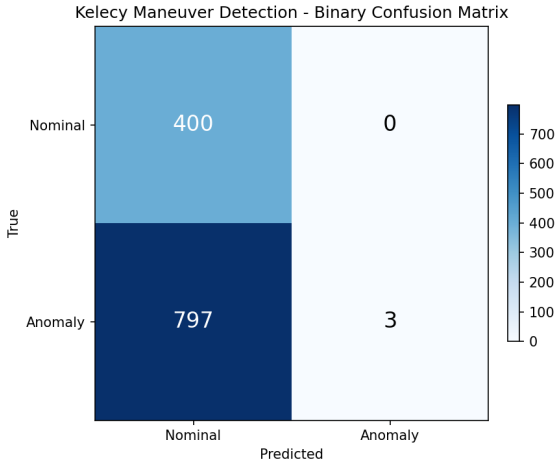
As established in Chapter 4, the standard operator baseline for maneuver detection involves the geometric analysis of TLE histories. Kelecy et al. show that a sliding window analysis of TLEs can provide insight into the timing and magnitude of impulsive [18]. Thus, to validate the classifier results, the sliding-window model was compared with the LSTM classifier. The test set remained unchanged, and the results from the sliding-window model are shown below. The sliding window model yields only a binary output, as it cannot determine the anomaly type; it only indicates whether one is present. At each timestep, a polynomial is independently fit to the orbital energy values in the window immediately before and the window immediately after that point. Both polynomials are extrapolated to the midpoint between the two windows, and the difference is recorded. In a nominal orbit, the two fits should agree closely at that midpoint, so the differences stay near zero. A maneuver abruptly changes the orbital energy, causing the leading-window fit to extrapolate to a systemically different value than the trailing-window fit, producing a sustained deviation in the differences, seen in Figure 6.17. The robust mean and standard deviation of that orbit's entire difference series are computed iteratively by discarding points more than three standard deviations from the current mean, repeating this three times. The mean and standard deviation are used to place the threshold bands. Figure 6.16 shows the confusion matrix with this anomaly detector for different decision boundaries. This was tested on the same noisy dataset as 6.4b. Figure 6.16c shows that with the 3-σ bounds, almost all samples were considered to be nominal. This shows that the miss rate is very high. Furthermore, if the bounds are lowered to 1.5-σ as in Figure 6.16a, the opposite occurs: almost all samples are now considered anomalies. A 2-σ bound yields better results as seen in Figure 6.16b, with more anomalies detected correctly than the others; however, the majority are still misclassified. The confusion matrices use the test set from the noisy dataset, and since the sliding window can only produce a binary output, the resulting confusion matrix is smaller than that in Figure 6.23.



(a) Confusion Matrix of the Kelecy Maneuver Detection Model Using the Noisy Dataset with 1.5-σ Decision Bounds



(b) Confusion Matrix of the Kelecy Maneuver Detection Model Using the Noisy Dataset with 2-σ Decision Bounds



(c) Confusion Matrix of the Kelecy Maneuver Detection Model Using the Noisy Dataset with 3-σ Decision Bounds

Figure 6.16: Overview of Confusion Matrices of the Kelecy Maneuver Detection Model With Different Decision Bounds

Figure 6.17 shows that the discontinuity in the predicted orbital energy between two adjacent polynomial fits rarely, or never, exceeds the 3-σ detection threshold. Since the deviation must exceed the detection threshold to be considered an anomaly, the detection threshold bounds are too high to capture the subtle differences between the mismodeled SRP and the nominal trajectories. In other words, this indicates that the SNR is so low that the anomalous signal is confined to the model's noise floor. When lowering the bounds to 1.5-σ, the opposite happens, as it was shown in Figure 6.16. Now, almost all the samples are being flagged, including nominal samples. Finally, when using a 2-σ bounds detection threshold, some more anomalies are flagged based on the orbital energy deviations, but a lot are still missed, mostly samples representing mismodeled SRP. Since the magnitudes of errors for these signals are so low, the applied bounds are too high to detect any anomalies unless the arc length is sufficiently long for a low-thrust maneuver to deviate enough from the nominal orbit to cross the thresholds. Alternatively, if the bounds are too tight, it starts misclassifying nomi-

nal orbital motion as an anomaly. This is demonstrated in Figure 6.18, which shows the peak sigma values across all the classes for a 2-σ decision boundary, where the detected anomalies are mostly low-thrust maneuvers. The peak sigma value reported for an orbit is the largest absolute deviation from that robust mean, expressed as a multiple of that robust standard deviation, analogous to the maximum Mahalanobis distance, but the “covariance” here is just a univariate robust variance. For the other classes, especially the mismodeled SRP, the 3-σ bounds filter out all deviations, and, likely, since the errors are periodic, the model will find them difficult to detect, while the 1.5-σ bounds do the opposite and trigger a false alarm on all the nominal orbits. With 2-σ bounds, a noise floor for the low-thrust maneuvers, or a noise ceiling for the nominal orbits is shown here. This implies that tighter decision bounds risk increasing the false-alarm rate, whereas a looser threshold flags fewer maneuvers. When examining Figure 6.18 for the 2-σ bounds, an additional limitation becomes apparent, in that the distribution of the peak sigma values for both nominal samples and mismodeled SRP samples is very similar. This analysis demonstrates the fundamental limitation of univariate threshold-based methods: no single sigma threshold can simultaneously maintain a low false-alarm rate for nominal orbits while detecting the subtle, persistent signatures of low-thrust maneuvers and mismodeled SRP.

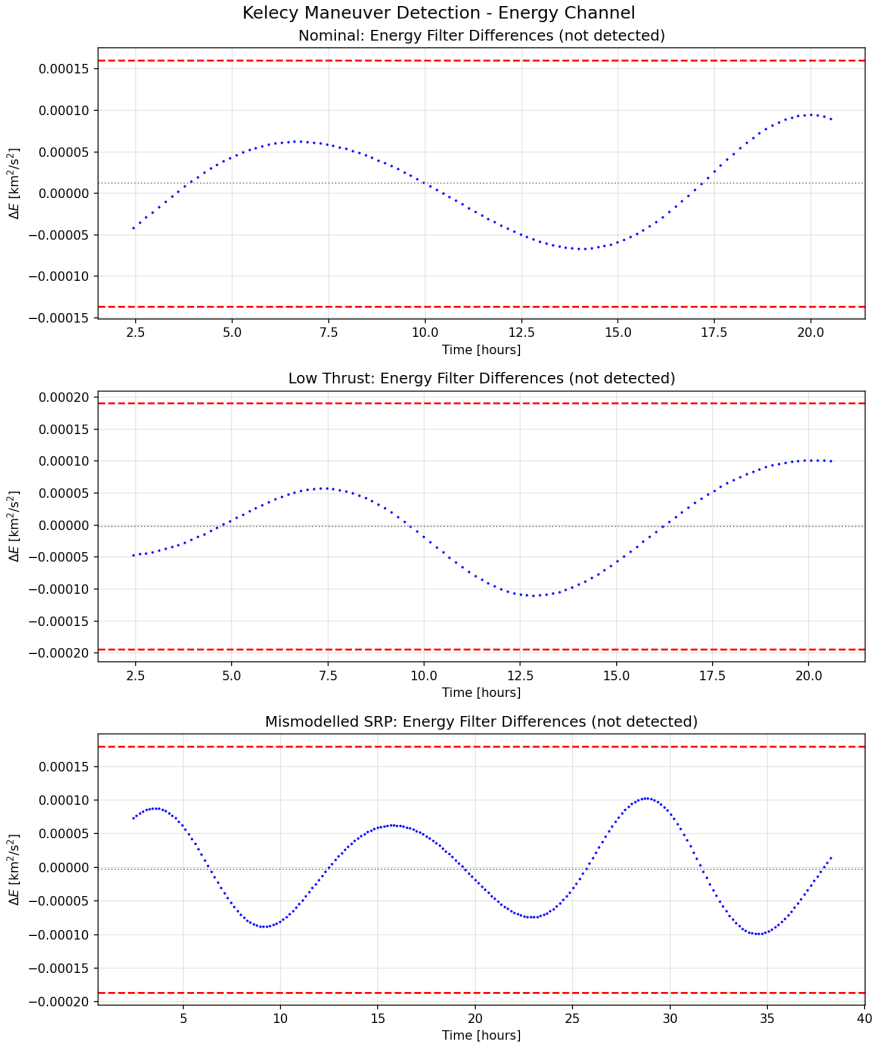


Figure 6.17: Example Orbits for Each Class and How They Were Classified with 3-σ Decision Boundaries

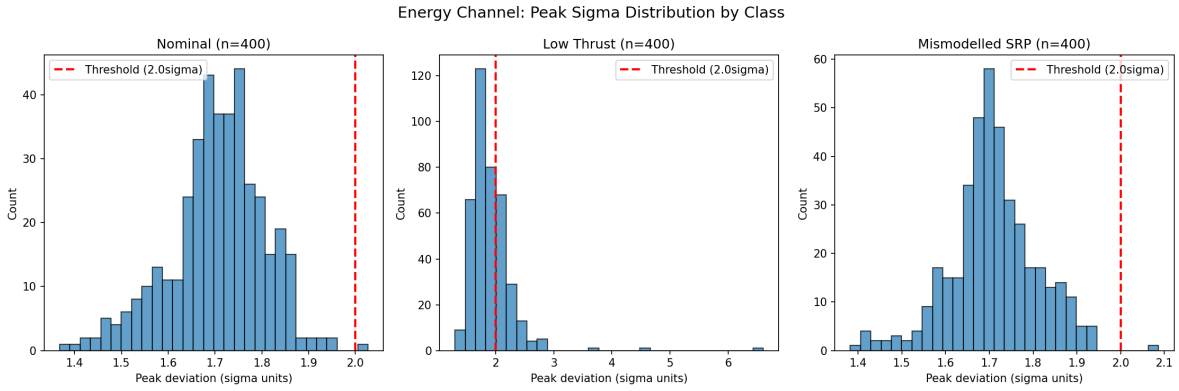


Figure 6.18: Peak Sigma Distribution per Class for 2-σ Decision Boundaries

This comparison underscores the need for a multivariate approach. While the sliding window model relies on univariate thresholds of derived properties like specific energy,

the LSTM classifier's ability to ingest 13 remapped features and learn temporal dependencies through its attention mechanism allows it to distinguish subtle patterns that a simple 3σ check ignores. Consequently, the LSTM provides a significantly more robust solution for the "quiet" maneuvers that increasingly define the modern GEO environment.

6.3. PINN Solver

The second component of the hybrid architecture is the inverse thrust recovery problem. For trajectories classified as low-thrust by the LSTM, the PINN is used to recover the unknown constant thrust vector. As described in Section 4.4, a reference trajectory is generated by propagating the perturbed two-body equations from the same initial conditions as the given trajectory with zero thrust. The PINN then learns the small positional deviation between the reference and the observed trajectory, with the thrust recovered through the physics-informed loss function.

The PINN is evaluated on 400 low-thrust test samples, the same used by the LSTM, from the noisy dataset described in Section 6.1. For each sample, the three-phase optimization is performed independently, and the recovered thrust vector is compared against the ground truth in terms of magnitude and angular errors. Figure 6.19 shows the loss convergence for a representative sample. The three optimization phases are clearly distinguishable. In Phase 1, the data loss decreases rapidly as the PINN learns to fit the observed trajectory deviation with thrust frozen at zero. The transition to Phase 2, where the physics loss is introduced and thrust is unfrozen, produces a visible increase in the total loss as the AdamW optimizer balances the competing objectives. By Phase 3, both loss terms have settled, and the L-BFGS optimizer drives the solution toward its final minimum. The thrust convergence plot confirms that the three components stabilize to values in the correct order of magnitude.

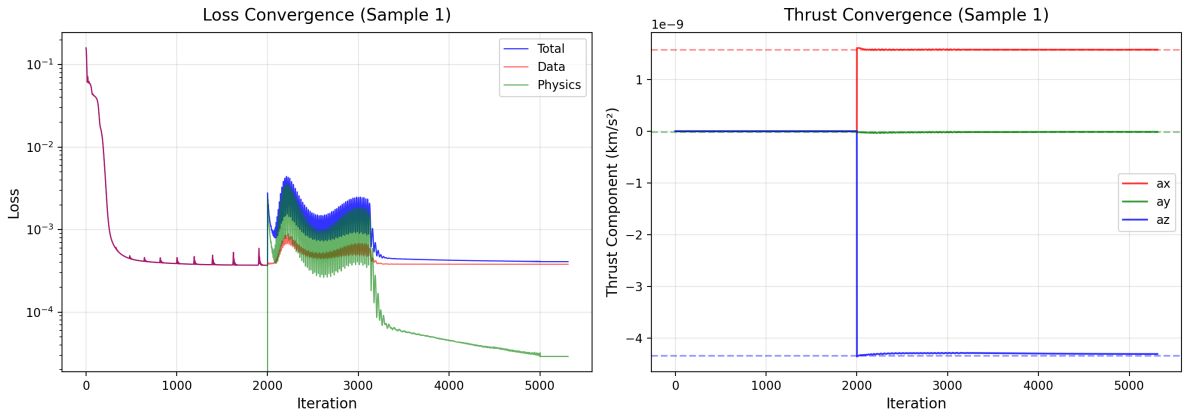


Figure 6.19: Training Curve for PINN

Figure 6.20 compares the recovered thrust magnitude against the ground truth for all 400 low-thrust test samples on a log-log scale. The dashed line indicates perfect recovery. The majority of estimates cluster tightly around this line across two orders of magnitude (10^{-10} to 10^{-8} km/s^2), demonstrating that the PINN reliably recovers thrust

magnitudes even at accelerations comparable to the SRP perturbation noise floor. Some scatter is visible at the lowest thrust magnitudes, where the SNR is smallest, and the deviation from the reference trajectory provides less information to the optimizer.

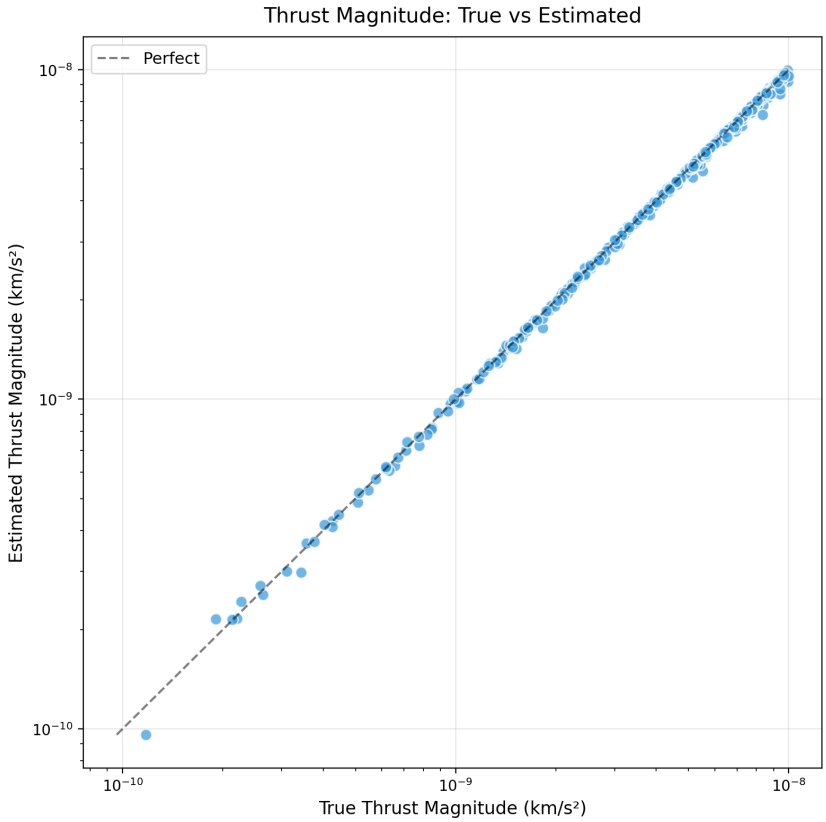


Figure 6.20: Evaluation of Thrust Magnitude

Figure 6.21 decomposes the recovery into the three ECI thrust vector components. All three components show strong correlation with the ground truth, confirming that the PINN recovers not only the thrust magnitude, but also its direction. The X and Z components, which span the equatorial plane, show tight clustering around the parity line across their full range of approximately $\pm 10^{-8}$ km/s^2. The Y-component exhibits slightly more scatter, particularly at intermediate magnitudes. This is consistent with the orbital geometry of GEO. Since orbits are near-equatorial, with inclinations between 0 and 5 degrees, the out-of-plane component produces a smaller positional deviation for a given thrust magnitude than the in-plane components, reducing its observability and making it harder for the optimizer to constrain precisely. The outliers visible in the X-component, where a few points deviate noticeably from the parity line, correspond to cases where the thrust is predominantly in the Y-Z plane, leaving the small X-component susceptible to limited observability; an issue which occurs when one component is an order of magnitude smaller than the others, resulting in a physics loss landscape becoming flat in that direction.

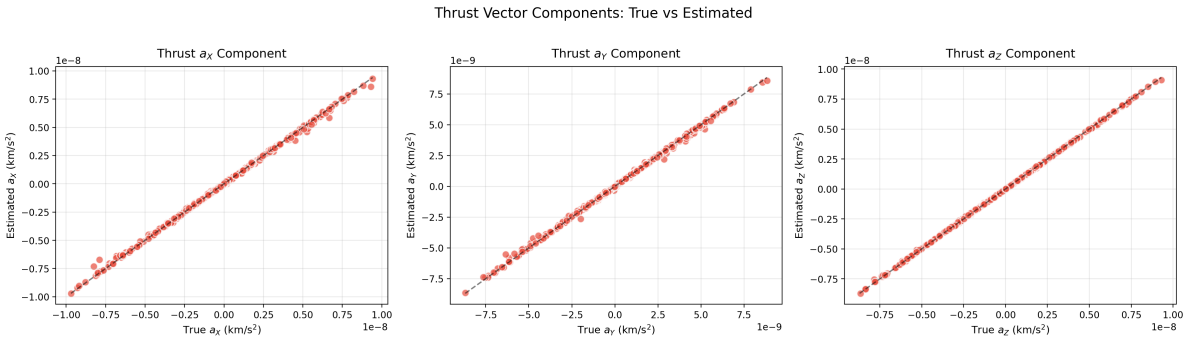


Figure 6.21: Evaluation of Thrust Magnitude by Component

Figure 6.22 shows the distribution of the magnitude and directional errors across all 400 test samples. Both distributions are heavily right-skewed, with most samples concentrated at low error values. The median magnitude error is 0.9%, and the median angular error is 0.6 degrees, indicating that for the typical case, the PINN recovers the thrust vector with high fidelity. Over 70% of samples achieve magnitude errors below 2.5% and angular errors below 1.5 degrees.

However, the long tails of both distributions indicate a population of outliers with significantly degraded performance, with magnitude errors reaching approximately 19% and angular errors up to 13 degrees. To understand the sources of error in the tail of the distribution, the relationship between thrust magnitude, arc duration, physics loss, and recovery accuracy is examined. Two distinct failure modes emerge, and critically, they are anti-correlated. The first failure mode manifests as elevated physics loss. The dominant correlate is the true thrust magnitude (r = 0.634 in log-log space), with the top 10% of physics loss values corresponding to a mean thrust of 8.28×10^{-9} km/s^2, compared to 4.69×10^{-9} km/s^2 for the remainder. Arc duration is a secondary factor (r = 0.454), as longer arcs accumulate more residual evaluation points. This is not a convergence failure, as the thrust recovery for these high-physics-loss samples remains accurate, with a median magnitude error of approximately 2.6%. Rather, the elevated physics loss is a scaling artifact: the residual of the equations of motion is proportional to the thrust magnitude, so larger thrusts produce larger absolute residuals even when the PINN fits the trajectory correctly. The second failure mode manifests as high-magnitude and angular errors despite low physics loss. These cases are concentrated at the lowest thrust magnitudes, near 10^{-10} km/s^2. At this level, the thrust-induced positional deviation is comparable to or smaller than the measurement noise, leaving insufficient signal for the optimizer to constrain the thrust vector. The PINN finds a solution that satisfies the physics, hence the low residual, but converges on an incorrect thrust because the data simply does not contain enough information to distinguish the true thrust from nearby alternatives. Samples in the 10^{-10} to 10^{-9} km/s^2 range exhibit a mean magnitude error of 3.7% and angular error of 3.4 degrees, compared to 1.4% and 0.8 degrees for the 10^{-9} to 10^{-9} km/s^2 range. The anti-correlation between these two failure modes has an important practical implication: the physics loss alone is not a reliable diagnostic of thrust estimation accuracy across different thrust magnitudes. A high physics loss does not indicate poor recovery, and a low physics loss does not guarantee accurate recovery. A more robust quality indicator

would need to account for the expected thrust magnitude, for example, by normalizing the physics residual by the estimated thrust or by incorporating component-wise sensitivity analysis.

In a real-world scenario, where the magnitudes of thrust are much higher, around 10^{-7} km/s^2, the failure modes are actually non-issues. Real thrusts will further increase the physics loss, but because high physics loss does not correlate with poor thrust recovery, normalizing the physics loss by thrust magnitude will eliminate the resulting scaling artifact. Additionally, the second failure mode does not occur in real-world scenarios, as the expected thrust is orders of magnitude higher than that required to produce it. The 10^{-9} km/s^2 bin gives 1.4% magnitude error and 0.8 degree angular error, and in real-world scenarios, the SNR will be considerably better, meaning the optimization landscape is more strongly constrained and the PINN should converge more reliably.

In real-world scenarios, thrust magnitudes for electric propulsion are typically in the 10^{-7}km/s^2 regime, well above the range where the PINN encounters difficulties. The 10^{-9} km/s^2 bin already achieves a 1.4% magnitude error and a 0.8-degree angular error; at operationally realistic thrust levels, the positional deviation from thrust would be orders of magnitude larger, more strongly constraining the optimization landscape. The results presented here, therefore, constitute a conservative lower bound on real-world performance. However, this conclusion assumes the measurement noise remains comparable. The Gaussian noise model used in this study is optimistic relative to real tracking data, which contains correlated errors, systematic biases and sparse observation gaps that could partially erode this SNR advantage.

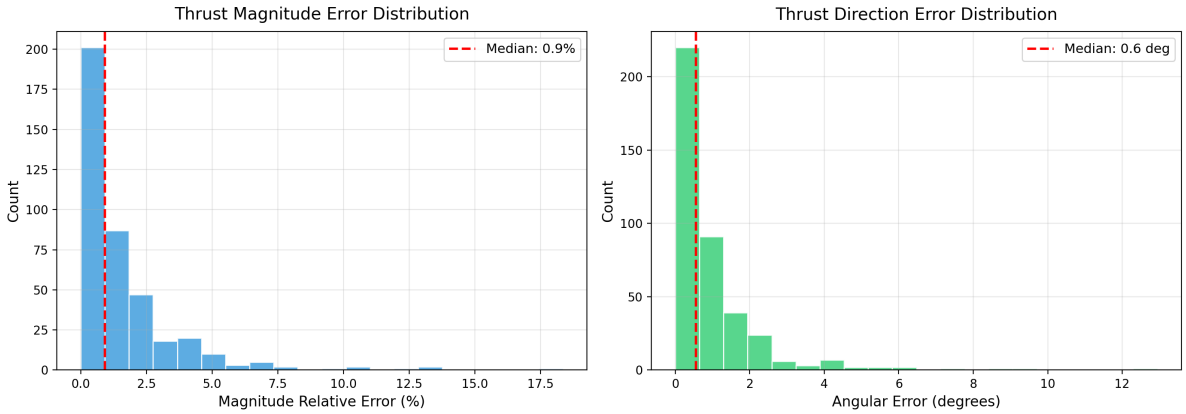


Figure 6.22: Distribution of Magnitude and Angular Error

The final results are better shown in Table 6.3, which highlights two samples that caused more issues than the rest.

Metric	Median	Best	Worst
Magnitude Error	0.915%	0.001%	18.35%
Angular error	$0.560°$	$0.019°$	$12.95°$

Table 6.3: Final Results Across 10 Test Samples of Thrust Reconstruction

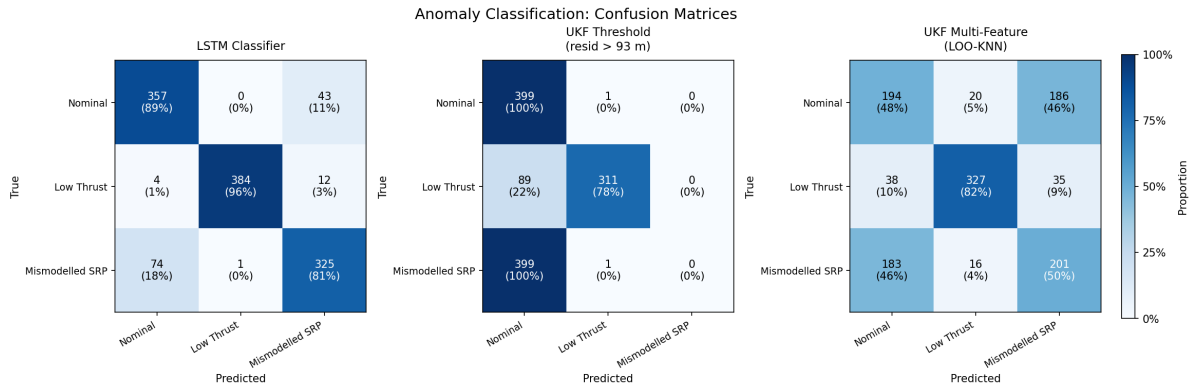


Figure 6.23: Confusion Matrices for LSTM and Both UKF Variants

6.4. Comparison with Statistical Analysis

To provide a high-fidelity benchmark, the proposed architecture was compared against an Unscented Kalman Filter. Unlike the sliding-window TLE approach, the UKF operates on the same high-cadence state vectors used by the LSTM and the PINN. The UKF is specifically designed to handle the non-linearities of orbital mechanics by using deterministic sampling of so-called sigma points to propagate the state mean and covariance. [45]. The UKF, as it stands, has no knowledge of the anomaly, as it simply tracks the orbit as best it can using its nominal dynamics model.

6.4.1. Comparison Between LSTM and UKF

To benchmark the LSTM classifier, its performance is compared with that of the UKF operating as an anomaly detector. Since the UKF is not inherently a classifier, a residual threshold is applied to its output: if the RMS position residual exceeds a calibrated threshold, the orbit is flagged as anomalous. Unlike the LSTM, which provides a three-class output, the UKF can only distinguish between nominal and anomalous behaviour, as it has no mechanism to determine the source of the anomaly. Figure 6.23 compares the confusion matrices of both models on the same test set. The LSTM achieves superior overall accuracy, particularly in distinguishing mismodeled SRP from nominal orbits. The UKF detects most low-thrust maneuvers, as these produce position residuals that grow beyond the threshold, but classifies nearly all mismodeled SRP cases as nominal. the filter passively absorbs the small SRP bias into its state update, preventing the residual from exceeding the detection threshold. The threshold is determined via a brute-force sweep, rather than being manually tuned. The detection threshold is determined by sweeping 200 evenly-spaced candidates between the minimum and maximum residual values across the test set, selecting the threshold that maximizes binary classification accuracy (maneuver vs. non-maneuver). This brute-force approach yields the UKF the optimal threshold for the given dataset, yet it still fails to separate mismodeled SRP from nominal behaviour because both produce residuals of comparable magnitude.

To give the UKF a second, more favourable evaluation, a multi-feature classification approach is also tested. Six UKF output metrics are used as input features: position and velocity RMS errors against the ground truth, position and velocity RMS innova-

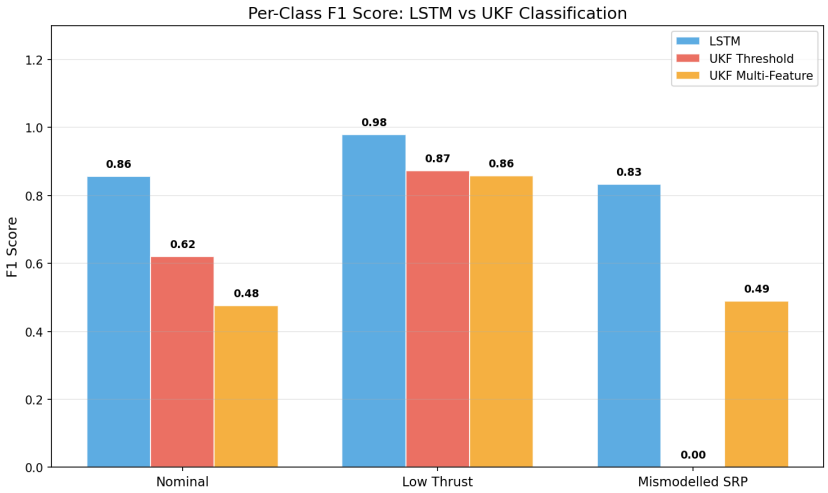


Figure 6.24: Per Class F1 Score Between LSTM and UKF

tions, mean position residual, and peak position residual. These features are classified using a K-nearest neighbours (KNN) algorithm, a non-parametric method that assigns a class to a sample based on the majority vote of its k closest neighbours in the feature space. In this case, $k = 3$ is used with distance weighting, meaning that closer neighbours exert a stronger influence on the classification decision than more distant ones. Due to the small size of the test set (approximately 30 orbits), a conventional train-test split would leave too few samples for a reliable fit. Instead, leave-one-out (LOO) cross-validation is employed: each orbit is held out in turn as the sole test sample, with the classifier trained on the remaining $n1$ samples. This process is repeated n times, so that each orbit receives exactly one prediction, and the features are re-scaled for each fold to prevent information leakage. This approach provides the UKF with the best opportunity for three-class classification by leveraging all available filter diagnostics simultaneously, rather than relying on a single residual threshold. Even with this evaluation, Figure 6.23 shows that while the distribution is better with this new approach, there is still a lot of confusion between the nominal and mismodeled SRP classes when compared to the LSTM.

Furthermore, Figure 6.24 shows the F1-score metric for the dataset from each model. Since this comes from the confusion matrix, it indicates that all three models are adept at identifying a low-thrust maneuver, but the UKF particularly struggles to distinguish the mismodeled SRP from the nominal samples.

Figure 6.25: UKF Position and Velocity Distributions

Figure 6.25 explains this phenomenon. As described in Section 4.1.2, the UKF treats unmodelled accelerations as stochastic process noise rather than as a deterministic signal. Both nominal orbits and mismodeled SRP orbits yield low residuals because the filter uses the nominal A/m model. The small SRP bias is passively absorbed into the state correction at each measurement epoch without the filter recognizing that a systematic modelling error is present. The LSTM, by contrast, learns the temporal patterns that distinguish the three classes from the engineered feature trajectories, enabling it to identify anomalies that remain invisible to the filter's residual statistics. Consequently, only low-thrust orbits produce position residuals large enough to exceed the UKF detection threshold. The multi-feature KNN approach partially alleviates this limitation, correctly identifying approximately 50% of the mismodeled SRP cases. By combining six filter diagnostics rather than relying on a single residual threshold, the KNN can exploit subtle differences in the relationships among the metrics, for example, when the innovation sequence and the position error against truth diverge slightly, indicating a systematic modelling error rather than pure measurement noise. However, the improvement is modest because the underlying features are all derived from the same UKF, which has already absorbed much of the SRP signal into its state estimate. The additional metrics provide some discriminative power but cannot recover information that the filter has already suppressed. The LSTM remains significantly more effective because it operates on physics-derived features computed independently of any filter, preserving the full SRP signature in the input data.

6.4.2. Comparison Between UKF and PINN

When comparing the UKF to the PINN, the task varies significantly. Both are now evaluated in terms of state estimation accuracy. The PINN obtains this from its fitted correction trajectory, whereas the UKF obtains it from its filtered state. The PINN additionally provides the thrust magnitude and direction, which the baseline UKF cannot do, as thrust is not included in its state vector. Figure 6.26 shows the PINN position RMS error, compared to the UKF position RMS error.

When comparing the PINN to the UKF in terms of state estimation accuracy, it is important to note that the two methods solve fundamentally different problems. The UKF is explicitly designed as a state estimator, in that its objective is to minimize the po-

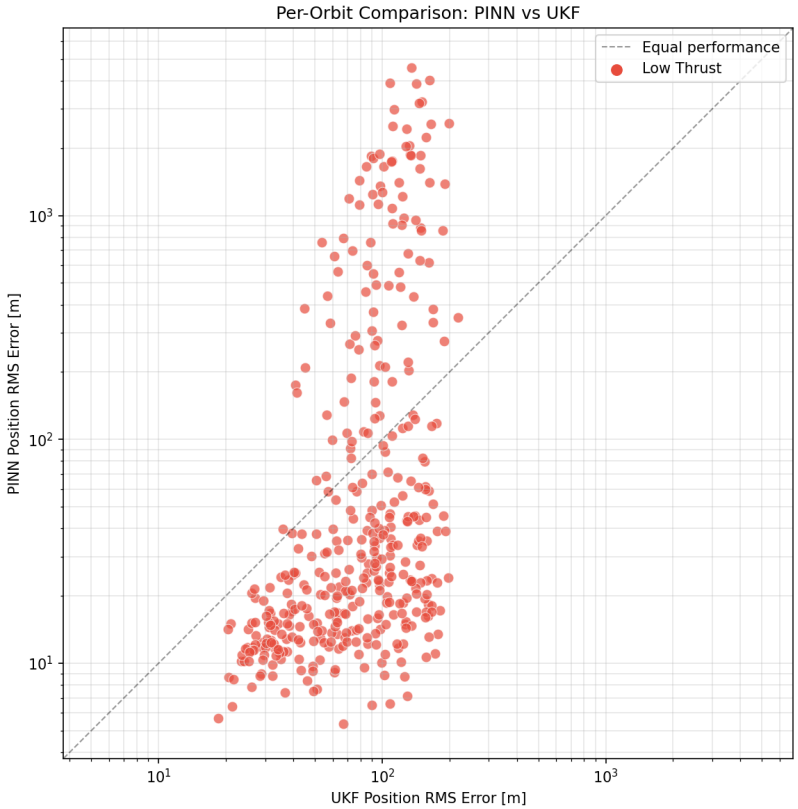


Figure 6.26: Comparison of PINN and UKF Position Residuals for Each Class

sition and velocity error at each epoch. The PINN, by contrast, is trained to recover the unknown thrust vector; the state estimation accuracy is a byproduct of fitting the correction trajectory δt to the observations, not the primary optimization target. Despite this, Figure 6.26 shows that the PINN achieves lower position RMS error than the UKF for the majority of low-thrust cases. The bulk of the samples cluster in the lower-left region of the plot, well below the equal-performance line, indicating that the PINN's global batch optimization over the full observation arc yields a smoother, more accurate trajectory reconstruction than the UKF's sequential filtering. This is because the PINN has access to the entire arc simultaneously and enforces dynamical consistency through the physics loss, whereas the UKF processes measurements causally and can only correct its estimate as new data arrives, introducing a recursive estimation lag when tracking a continuously maneuvering object.

However, a population of outliers is visible above the parity line, where the PINN's position error significantly exceeds that of the UKF, despite the UKF remaining within the 50–200 m range. The PINN outliers are attributable to ballistic-error accumulation due to imperfect thrust recovery. Because the PINN fits a single continuous trajectory with a constant thrust vector, any error in the recovered thrust is effectively double-integrated over the arc duration, producing position errors that scale as $\frac{1}{2}a_T \cdot t^2$. The two compounding factors are thrust magnitude and arc duration. Larger thrusts amplify the absolute error in the recovered acceleration, and longer arcs give this error more time to accumulate. For example, a 2.9% magnitude error on a 47-hour arc

with a thrust magnitude of $9.9 \times 10^{-9}\ km/s^2$ produces a predicted position error of approximately 4.1 km, consistent with the observed error of 4.6 km.

The UKF is not susceptible to this effect because it updates the state at each measurement epoch rather than integrating a trajectory from a fixed thrust estimate. Its position error remains bounded by the observation noise regardless of arc length or thrust magnitude. This reflects a fundamental difference between the two approaches: the PINN recovers the thrust vector by fitting a physically consistent trajectory, while the UKF tracks the evolving state without estimating the cause of the motion. The PINN's position outliers are therefore not a convergence failure but an inherent trade-off of the trajectory-fitting approach since the same mechanism that enables thrust recovery also causes small thrust errors to compound over long arcs.

To provide a more direct comparison with the PINN's thrust-recovery capability, the baseline UKF is augmented to estimate the thrust vector in addition to the orbital state. The baseline UKF operates on a 6-dimensional state vector $\mathbf{x} = [r, v]^\top$, treating the reflectivity coefficient C_R as a fixed parameter and absorbing unmodelled accelerations through process noise. The augmented UKF extends the state vector to 10 dimensions [r, v, C_R, a_T] to co-estimate both the reflectivity coefficient and the three-component thrust acceleration vector. Including C_R as an estimated parameter is necessary because, in real-world scenarios, uncertainty in the spacecraft's reflectivity directly affects the SRP acceleration. If C_R is held fixed at an incorrect value, the resulting SRP modelling error could be absorbed into the thrust estimate, biasing the recovery. By co-estimating C_R, the filter can separate SRP uncertainty from the thrust signal, providing a fairer comparison with the PINN which operates with a known SRP model.

The augmented state introduces several changes to the filter architecture. The UKF handles non-linearity by propagating a set of deterministically chosen sample points, known as sigma points, through the equations of motion. These points are selected to capture the mean and covariance of the state distribution, and after propagation, the transformed mean and covariance are reconstructed from the propagated points. For the 6-dimensional baseline, this requires 13 sigma points; the 10-dimensional augmented state requires 21. The process noise model is extended to include three independent channels: state-noise compensation for position and velocity, a random-walk process for C_R, and a random-walk process for the thrust components. Both C_R and the thrust are modelled as constant in the equations of motion, with the random walk process noise as the sole mechanism allowing them to evolve over time. Crucially, the propagator evaluates SRP using the current estimate of C_R for each sigma point rather than a fixed constant, allowing the filter to adapt its SRP model as observations accumulate. Since only position and velocity are directly observed, C_R and thrust are not independently observable, and thus, they must be inferred entirely from how the trajectory deviates from the expected dynamics. This makes the augmented UKF a challenging but fair benchmark: it attempts the same inverse problem as the PINN, recovering an unknown thrust from trajectory observations, but within the sequential Kalman filter framework rather than a global batch optimization.

A validation scenario is constructed to compare the augmented UKF against the PINN

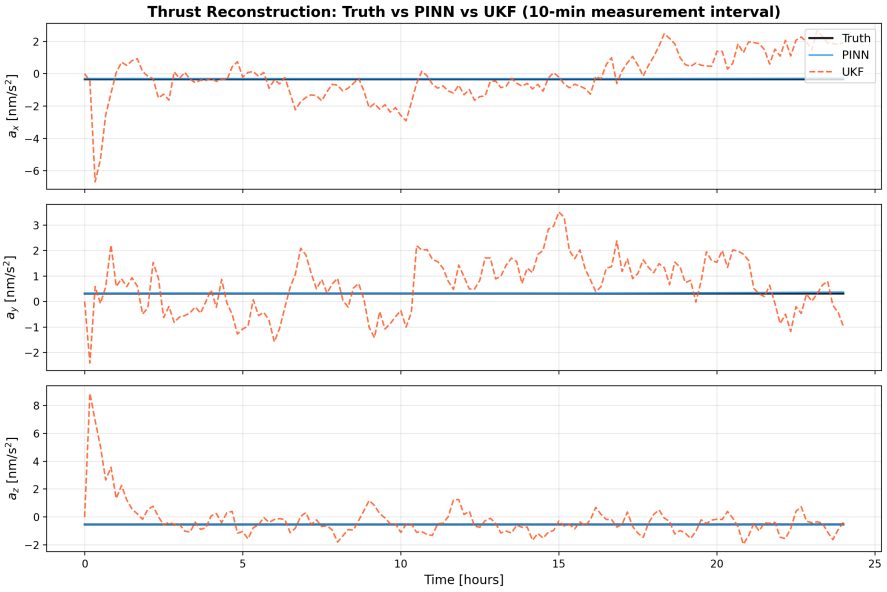


Figure 6.27: Thrust Reconstruction Timeseries Comparing the PINN and the Augmented UKF With 10-Minute Sampling

under progressively more challenging conditions. The scenario uses a dataset similar to that described in Section 6.1, with the only difference being that, now, C_R is drawn randomly from $\mathcal{U}[1.1, 1.5]$, while A/m ratio remains fixed. Both methods co-estimate C_R and thrust simultaneously, though only the thrust recovery results are presented, as this is the primary objective of the comparison. The scenario uses a constant thrust vector and tests measurement frequencies of 10, 50, and 120 minutes, providing a direct comparison under conditions closest to the original PINN evaluation while also evaluating how both methods degrade when observations become sparse.

Figures 6.27, 6.28 and 6.29 shows representative thrust reconstruction time-histories for the 10-minute, 50-minute, and 120-minute measurement intervals, respectively. In each plot, the black line represents the true constant thrust, the blue line shows the PINN estimate, and the dashed orange line shows the augmented UKF estimate.

At the 10-minute interval, the PINN recovers all three thrust components with high accuracy, lying nearly on top of the truth throughout the arc. The UKF estimate, by contrast, exhibits significant oscillation around the true values, particularly in the first few hours as the filter converges from its zero-initialized thrust state. Even after this transient, the UKF continues to fluctuate because measurement noise propagates through the sequential state updates. This illustrates the fundamental difference between the two approaches: the PINN produces a single, globally optimized constant vector, while the UKF outputs a noisy time series that must be post-processed to extract a mean thrust estimate.

At the 50-minute interval, the PINN maintains accurate recovery across all components, with only minor deviations from the truth. The UKF oscillations become more pronounced with fewer measurements to constrain the estimate, and the transient convergence period extends further into the arc. At the 120-minute interval, the PINN begins to show visible drift, particularly in the X-component, where the estimate diverges

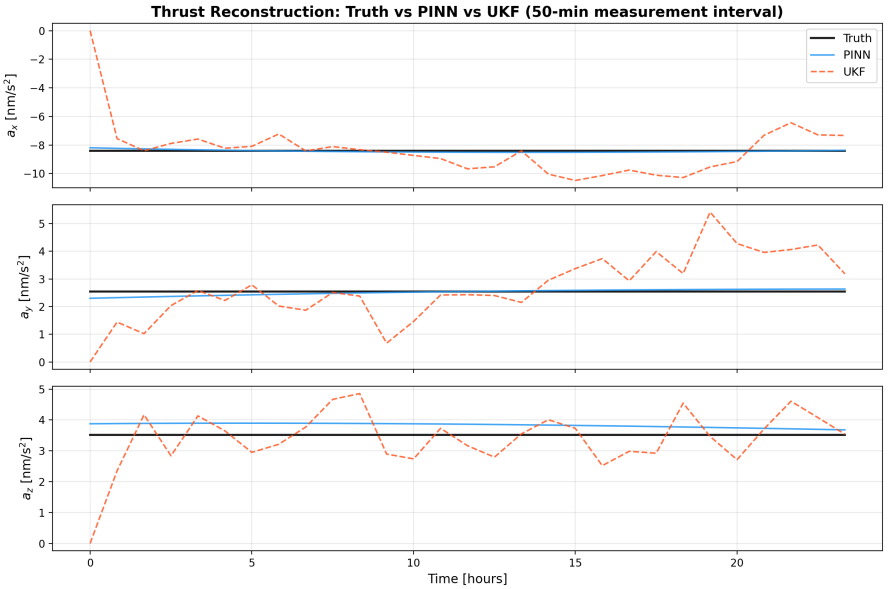


Figure 6.28: Thrust Reconstruction Timeseries Comparing the PINN and the Augmented UKF With 50-Minute Sampling

over the second half of the arc. This is consistent with the reduced observability from sparse data, as fewer measurements provide less constraint on the correction trajectory, allowing the physics loss to admit a wider range of plausible solutions. The UKF shows comparable degradation, with larger oscillations and less stable convergence.

Figure 6.30 summarizes the comparison across all test samples using the mean absolute error (MAE) of the thrust vector, defined as:

$$\text{MAE} = \frac{1}{3N} \sum_{i=1}^{N} \sum_{j \in \{x,y,z\}} |\hat{a}_j(t_i) - a_j(t_i)|$$

where N is the number of timesteps and the sum is taken over all three ECI components at each epoch. For the PINN, which recovers a single constant vector, the estimated thrust is the same at every timestep, so the MAE reduces to the mean component-wise absolute error of that constant estimate. For the UKF, which produces a time-varying thrust estimate, the MAE captures both the bias and the variance of the sequential estimate over the arc. At the 10-minute interval, the PINN achieves a median MAE approximately one order of magnitude lower than the UKF. At 50 minutes, the PINN retains a clear advantage, though the gap narrows. At 120 minutes, the two methods converge to comparable accuracy, with the PINN's median accuracy slightly exceeding that of the UKF. Notably, the PINN's error bars increase substantially as the measurement frequency decreases, reflecting its sensitivity to observation density. As a batch method, its accuracy depends directly on the number of data points used to constrain the fit. The UKF's error, by contrast, remains relatively stable across all three intervals, as its sequential architecture is inherently designed to operate with sparse, irregularly sampled measurements. This crossover point suggests that the PINN is the preferred method when high-cadence observations are available, while the UKF provides more robust performance under data-sparse conditions.

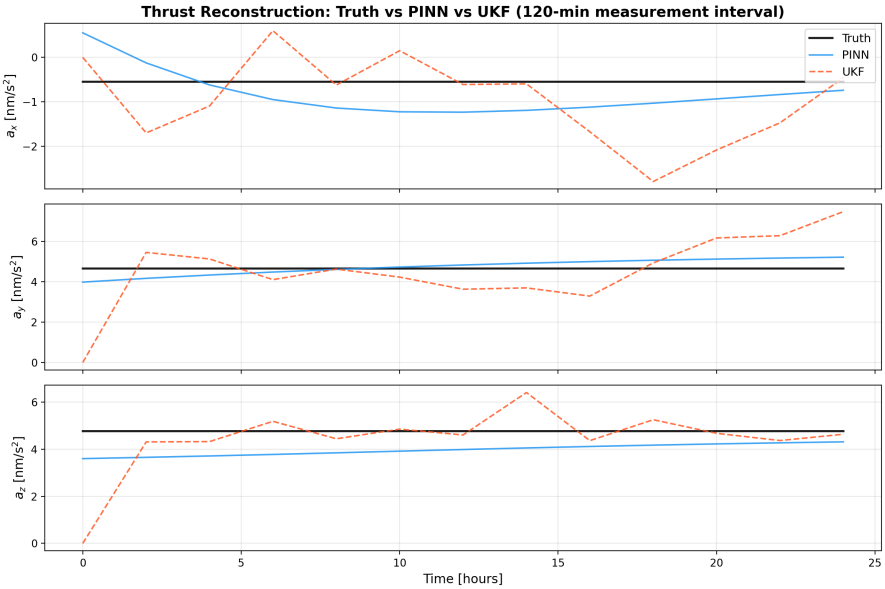


Figure 6.29: Thrust Reconstruction Timeseries Comparing the PINN and the Augmented UKF With 120-Minute Sampling

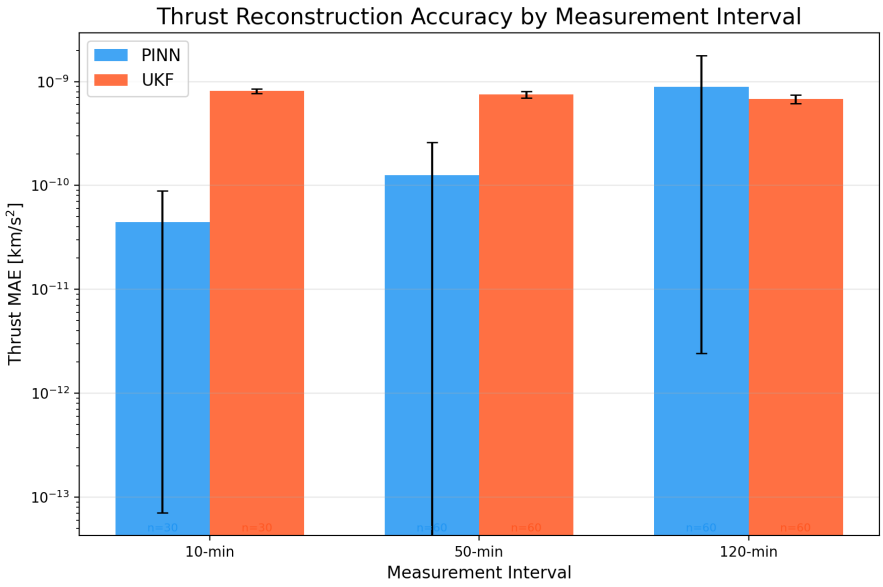


Figure 6.30: Thrust MAE Comparison Between the PINN and Augmented UKF for Different Sampling Intervals

6.5. Synthesis of Comparative Results

The comparative analysis across the four distinct methodologies reveals a clear hierarchical progression in maneuver-detection capability. The operator baseline, represented by the TLE sliding-window method, proves fundamentally inadequate for the modern GEO environment. Its reliance on univariate 3-σ thresholds based on mean Keplerian elements imposes a prohibitively high detectability threshold. Lowering the threshold shows a clear noise floor, at which point, the signatures from the low-thrust maneuvers are no longer distinguishable from nominal or mismodeled SRP trajectories.

The transition to high-fidelity statistical orbit determination via the Unscented Kalman Filter (UKF) improves tracking precision but highlights a critical observability gap. The sequential, recursive nature of the UKF treats unmodeled accelerations as stochastic process noise. By passively absorbing these deviations into the state update at each epoch, the UKF masks the unique temporal signatures of mismodeled SRP and low-magnitude thrust. This results in a high false-negative rate for mismodeled SRP events and introduces a significant recursive lag in low-thrust detection, as the filter chases the maneuvering satellite rather than characterizing the force driving the motion.

The LSTM classifier represents a significant step forward, achieving over 88% overall accuracy by exploiting multivariate temporal patterns that neither threshold-based nor filter-based methods can capture. Its primary limitation, the confusion between mismodeled SRP and nominal orbits at small A/m deviations, reflects a fundamental observability limit rather than a model deficiency.

For thrust reconstruction, the augmented UKF and PINN offer complementary strengths. At high measurement cadences (10-minute intervals), the PINN achieves an MAE approximately an order of magnitude lower than the augmented UKF, owing to its global batch optimization and physics-constrained loss function. However, as observations become sparse (120-minute intervals), the PINN's accuracy degrades significantly while the UKF remains relatively stable, reflecting its architectural robustness to irregular and low-cadence data. This crossover suggests that the optimal choice between the two methods depends on the available observation infrastructure.

Collectively, these results demonstrate that integrating orbital dynamics into the learning process, through physics-derived features in the LSTM and physics-constrained optimization in the PINN, provides capabilities that classical methods cannot match under favourable observation conditions, while highlighting the continued relevance of sequential filtering approaches when data availability is limited.

7

Conclusion

This research investigated the intersection of deep learning and orbital mechanics to address the increasingly critical problem of low-thrust maneuver detection in GEO. The following sections address the research questions posed at the beginning of the study.

7.1. Sub Questions

Sub-question 1

How accurate does a machine learning model need to be to follow real trajectories to be adequate for maneuver detection in GEO?

The adequacy of a machine learning model for GEO maneuver detection is defined by its ability to resolve signals below the sensor noise floor. The study utilized a measurement noise of 50 m in position and 5 mm/s in velocity. For a model to be adequate, its internal reconstruction error (the residuals on nominal orbits) must be significantly lower than the cumulative deviation caused by a maneuver. Analysis showed that a standard PINN used as a propagator resulted in residuals of several kilometres, which is insufficient for detecting low-thrust maneuvers of magnitude 10^{-8} km/s^2 over short arcs. However, by adopting a reference-trajectory decomposition, the model shifted to an interpolation task, in which residuals were reduced to sub-meter levels. This indicates that a model's reconstruction residuals on nominal orbits must be significantly smaller than the positional deviation caused by the maneuver over the observation arc, otherwise the maneuver signal is indistinguishable from the model's own error.

Sub-question 2

What are the key limitations of a baseline physics-agnostic model in identifying low-thrust maneuvers from synthetic data?

Baseline physics-agnostic models, such as standard LSTMs or MLPs, suffer from three primary limitations: Spectral Bias, Lack of Extrapolation, and Black-Box Interpretability. Preliminary verification and validation on mass-spring and two-body

systems demonstrated that physics-agnostic networks naturally learn low-frequency, smooth functions and struggle to maintain orbital phase over time. Without the governing ODEs, these models exhibit energy leakage, where the predicted satellite gradually drifts off its physical manifold. Furthermore, they provide no ”reason” for a classification; they cannot distinguish between a maneuver and an SRP modelling error if the data distributions overlap, as they lack the causal link provided by Newton's second law.

Sub-question 3

How can a PINN be formulated to specifically model continuous low-thrust trajectories and the associated orbital perturbations in GEO?

To overcome the gradient pathology inherent for NNs in the GEO regime, where the absolute magnitudes of the position vector dwarf the subtle signals of low-thrust maneuvers, the PINN was formulated as an Inverse Residual Estimator. This implementation relied on several key architectural innovations to ensure numerical stability and convergence. First, the problem was restructured through Reference Trajectory Decomposition, which models the position deviation $\boldsymbol{\delta}(t)$ rather than the full state $\mathbf{r}(t)$. By focusing solely on the correction term, the numerical conditioning of the network's output is improved by approximately five orders of magnitude. Second, the model employs Hard-Constraint Parameterization via a τ^2 scaling factor. This analytically ensures that the initial conditions are satisfied at $t = t_0$, thereby eliminating the need for manually tuned initial condition penalty weights in the loss function. Finally, a Three-Phase Optimization strategy was utilized, incorporating an algebraic warm-start to initialize the unknown thrust vector. This allows the subsequent joint refinement stages with Adam and L-BFGS to converge from a physically plausible starting point. Together, these elements enable the PINN to treat thrust as a learnable parameter, strictly bounded by a high-fidelity perturbation model that incorporates J_2 effects, solar radiation pressure, and third-body lunar and solar gravity.

Sub-question 4

How does the performance of the PINN-based model compare to the baseline models?

The hybrid LSTM/PINN architecture demonstrated a marked improvement in performance compared to both the TLE-based sliding-window and the UKF baseline. In terms of detection accuracy, the LSTM-based classifier achieved an overall success rate of over 94%. This represents a significant capability leap, as the model successfully navigated the SRP-thrust geometric degeneracies that typically cause TLE models to fail and UKFs to passively ”absorb” anomalous accelerations via stochastic process noise. Regarding robustness to data noise, the PINN's global optimization approach over the entire observation arc provided a distinct advantage over the UKF's sequential update logic. While the UKF remains sensitive to filter lag and covariance collapse during unmodeled events, the PINN effectively smoothed the Gaussian noise, maintaining a median thrust-magnitude recovery error below 2%. Furthermore, the model exhibits superior transferability compared to classical filters. Whereas a UKF

requires precise, scenario-specific tuning of the process noise covariance matrix to remain stable, the PINN's inherent reliance on the underlying differential equations makes it naturally more adaptable to real-world scenarios where satellite-specific parameters, such as the area-to-mass ratio or exact propulsion characteristics, may not be known a priori.

7.2. Primary Research Question

Main Research Question

What are the limitations of standard maneuver detection techniques for detecting low-thrust maneuvers, and how can these limitations be overcome by integrating orbital dynamics into a data-driven model?

Standard techniques are limited by a fundamental "Observability-Interpretability" trade-off. TLE-based methods are interpretable but lack the sensitivity to detect low-thrust signals. Sequential filters, such as the UKF, are highly sensitive but cannot distinguish among different types of unmodelled accelerations, often masking mismodelled accelerations as stochastic noise.

These limitations are overcome by integrating orbital dynamics through a hybrid, physics-informed framework. The data-driven component (LSTM) provides the pattern recognition needed to flag anomalies in high-dimensional feature spaces, while the physics-informed component (PINN) provides the dynamical regularization required to reconstruct the maneuver. By embedding the differential equations of motion directly into the loss function, the model is forced to find a solution that is not only statistically consistent with the sensor data but also physically consistent with the laws of astrodynamics. This integration effectively "unmasks" the maneuver from background noise, providing a robust and explainable tool for maintaining asset custody in GEO.

7.3. Limitations of the Proposed Framework

One of the most significant theoretical hurdles identified during the evaluation phase is the presence of geometric degeneracy between solar radiation pressure and active thrust. When a constant low-thrust vector aligns closely with the Sun-Earth line, the resulting orbital perturbation mirrors the periodic signature of a mismodeled area-to-mass coefficient. In these configurations, the LSTM classifier faces a fundamental observability limit: the two physically distinct forces produce nearly identical trajectory deviations over the short observation arcs used in this study. Consequently, the model may occasionally struggle to distinguish between these two and converge to the wrong answer, identifying that an anomaly is present but failing to correctly characterize its source due to the lack of a distinct discriminative signal.

Beyond these geometric constraints, the framework's sensitivity is strictly bounded by a detectability noise floor dictated by the precision of the input sensor data. For accelerations with magnitudes reaching toward 10^{-10} km/s^2 or area-to-mass ratio errors of less than 5%, the anomalous signal strength falls below the standard deviation of the Gaussian noise added to the positions and velocities. This creates a physical "gray

zone" in which low-thrust behaviour remains effectively submerged within the 50 m-range error typical of geosynchronous surveillance. This limitation underscores that the effectiveness of physics-informed models is ultimately tethered to the quality of the tracking network, as even rigorous dynamical regularization cannot recover a signal with insufficient signal-to-noise ratio.

The PINN-based inverse reconstruction exhibited two distinct failure modes that are, notably, anti-correlated. At high thrust magnitudes and long arc durations, the physics loss is elevated; not due to poor convergence, but because the residual of the equations of motion scales with the thrust magnitude. In these cases, the thrust recovery remains accurate, and the elevated physics loss is a scaling artifact rather than a diagnostic of failure. Conversely, at the lowest thrust magnitudes near 10^{-10} km/s^2, the physics loss converges to a low value but the recovered thrust can be significantly inaccurate. The thrust-induced positional deviation is comparable to the measurement noise, leaving insufficient information to constrain the thrust vector, and the optimizer settles on a physically plausible but incorrect solution. This anti-correlation means that the physics loss alone is not a reliable diagnostic of thrust-estimation accuracy across different thrust regimes, i.e., a high physics loss does not indicate poor recovery, and a low physics loss does not guarantee accurate recovery. Additionally, when a single thrust component is an order of magnitude smaller than the others, the physics loss landscape becomes flat along the minor component, yielding a near-degenerate inverse problem in which multiple thrust directions satisfy the constraint to a similar degree.

The PINN also assumes a constant thrust vector over the entire observation arc. Real low-thrust maneuvers may vary in magnitude and direction over time, particularly during orbit-raising or complex proximity operations. In such cases, the recovered constant vector represents a time-averaged approximation whose physical meaning degrades as the thrust profile becomes more dynamic. Extending the formulation to recover time-varying profiles would require a fundamentally different thrust parameterization, which is left for future work.

The comparison with the augmented UKF revealed that the PINN's accuracy advantage is contingent on high-cadence observations. At 10-minute measurement intervals, the PINN achieves approximately an order of magnitude lower thrust MAE than the augmented UKF. However, at 120-minute intervals, the PINN's accuracy degrades to a level comparable to the UKF, as the reduced number of data points provides insufficient constraint on the correction trajectory. The UKF, by contrast, maintains relatively stable performance across all measurement cadences due to its sequential architecture, which is inherently designed to operate with sparse data. This crossover suggests that the framework is best suited for surveillance architectures with observation cadences of approximately 50 minutes or less.

The entire framework is validated exclusively on synthetic data with independent Gaussian measurement noise. Real-world tracking data contains correlated errors, systematic sensor biases, and irregular observation gaps that are not captured by this noise model. While the Gaussian noise levels used in this study (50 m position, 5 mm/s velocity) are representative of high-quality radar tracking at GEO distances, the

absence of correlated and systematic error sources means the demonstrated performance should be interpreted as an upper bound. The extent to which these factors erode the PINN's accuracy advantage over classical methods remains an open question requiring validation on real-world observations.

Finally, the study confirmed that current PINN architectures are structurally better suited for interpolation and residual estimation than for standalone orbital propagation. Preliminary tests demonstrated that, although a network can accurately fit a specific trajectory, it fails to generalize across varying initial conditions or to extrapolate beyond the training window without significant phase drift. This behaviour necessitated the continued reliance on high-fidelity numerical integrators, such as the DOP853 scheme, to provide a deterministic baseline for the reference trajectory. By acknowledging that the PINN cannot yet match the long-term precision of classical mechanics for general motion, the framework was intentionally designed as a hybrid approach that leverages the deterministic precision of classical solvers alongside the flexible optimization capabilities of adaptive neural network estimators.

7.4. Recommendations for Future Work

The findings of this research provide a foundation for several promising avenues of future study. A primary recommendation is to extend the PINN formulation to recover time-varying thrust profiles. The current constant-thrust assumption limits its applicability to scenarios in which the thrust magnitude and direction remain fixed over the observation arc. Parameterizing the thrust as a learnable function of time, for example, using a truncated Fourier series or a secondary neural network, would enable recovery of realistic maneuver profiles including orbit-raising spirals and phased stationkeeping burns.

The per-arc training requirement of the PINN, while justified by its flexibility, remains a computational limitation. Transfer learning from previous arcs could partially mitigate this cost by using converged network weights as an initialization for subsequent arcs with similar orbital geometries, accelerating convergence when successive thrust profiles are similar. However, this does not constitute true generalization as a fully amortized model that infers the thrust vector in a single forward pass, without per-arc optimization, remains an open research direction. Additionally, the PINN's sensitivity to the reference trajectory's dynamical model can be partially mitigated by co-estimating uncertain parameters, such as C_R, alongside the thrust, as demonstrated by the augmented UKF in this study. Incorporating similar parameter estimation into the PINN loss function is a natural extension, though model error cannot be fully eliminated without an exact force model.

Furthermore, the transition from high-fidelity synthetic datasets to real-world orbital observations represents a critical next step for validating the framework's robustness. While synthetic data provided a controlled environment for testing, real-world data from public catalogues or commercial space surveillance networks introduces complexities such as irregular sensor outages, varying lighting conditions for optical sensors, and unmodeled satellite-specific perturbations. Future research should focus on domain adaptation and transfer learning, in which the model is pre-trained on large-

scale synthetic datasets and then fine-tuned on a smaller corpus of confirmed historical maneuvers. This would ensure that the PINN's physics loss can handle the "noise-heavy" reality of actual orbital maintenance without succumbing to the overfitting issues common in purely data-driven models.

The crossover in measurement frequency identified in the augmented UKF comparison suggests a hybrid operational architecture: the PINN could be deployed when high-cadence observations are available, with the augmented UKF serving as a fallback during data-sparse intervals. Developing an automated selector that routes arcs to the appropriate method based on observation density and expected thrust regime would maximize the strengths of both approaches.

Finally, a direct performance comparison between the PINN-based reconstruction and the OCBE would provide a definitive benchmark for the architecture's efficiency. While this study compared the PINN to the UKF, the OCBE represents the current state-of-the-art in functional optimization for maneuver reconstruction. Investigating how the PINN's parameterized approach, in which thrust is recovered as a learned constant, compares with the OCBE's use of Pontryagin's Minimum Principle would clarify the trade-offs between computational speed and reconstruction fidelity. Specifically, determining whether the PINN can achieve OCBE-like accuracy with fewer iterations or less expert-level tuning of cost functions would further justify the integration of differentiable physics into modern space situational awareness pipelines.

References

[1] Thawar Arif. *Aerospace Technologies Advancements*. en. BoD – Books on Demand, Jan. 2010. ISBN: 978-953-7619-96-1.

[2] Vadim Azhmyakov. “Chapter 9 - Conclusion and Perspectives”. In: *A Relaxation-Based Approach to Optimal Control of Hybrid and Switched Systems*. Ed. by Vadim Azhmyakov. Butterworth-Heinemann, 2019, pp. 385–393. ISBN: 978-0-12-814788-7. DOI: 10.1016/B978-0-12-814788-7.00015-1. URL: https://www.sciencedirect.com/science/article/pii/B9780128147887000151.

[3] G Badura et al. “Regularizing training of physics informed neural networks (pinns) for cislunar orbit determination via transfer learning”. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS), Maui Economic Development Board*. 2024.

[4] Ed Betar. “Space Traffic Management is Critical for National Security”. en. In: *Contemporary Issues in Air and Space Power* 3.1 (May 2025). Publisher: Air and Space Power Centre, bp46274180. DOI: 10.58930/bp46274180. URL: https://ciasp.scholasticahq.com/article/137701-space-traffic-management-is-critical-for-national-security.

[5] Donald Chu et al. “GOES-R STATIONKEEPING AND MOMENTUM MANAGEMENT”. In: *29th Annual AAS Guidance and Control Conference*. 2006.

[6] R Cipollone, E Raviola, and P Di Lizia. “AN UNSUPERVISED LEARNING-BASED MANOEUVRE DETECTION METHOD FOR RESIDENT SPACE OBJECT PATTERN OF LIFE CHARACTERISATION”. en. In: *9th European Conference on Space Debris*. Apr. 2025.

[7] Phil DiBona et al. “Machine learning for RSO maneuver classification and orbital pattern prediction”. en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (2019).

[8] ESA Space Debris Office. *ESA’S ANNUAL SPACE ENVIRONMENT REPORT*. LOG GEN-DB-LOG-00288-OPS-SD. ESA, Mar. 2025. URL: https://www.sdo.esoc.esa.int/environment_report/Space_Environment_Report_latest.pdf.

[9] Ariadna Farres, Cassandra Webster, and David Folta. “High fidelity modeling of SRP and its effect on the relative motion of Starshade and WFIRST”. In: *2018 Space Flight Mechanics Meeting*. 2018, p. 2227.

[10] Thomas Goldman and Kevin Cowan. “AN UNSUPERVISED PHYSICS-INFORMED NEURAL NETWORK FOR FINDING OPTIMAL LOW-THRUST TRANSFER TRAJECTORIES WITH THE DIRECT METHOD”. en. MA thesis. Delft University of Technology, Aug. 2024.

[11] Ernst Hairer, Syvert Norsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Vol. 8. Jan. 1993. ISBN: 978-3-540-56670-0. DOI: 10.1007/978-3-540-78862-1.

[12] Jennifer S. Hudson and Daniel J. Scheeres. "Orbital Targeting Using Reduced Eccentric Anomaly Low-Thrust Coefficients". In: *Journal of Guidance, Control, and Dynamics* 34.3 (2011), pp. 820–831. ISSN: 0731-5090. DOI: 10.2514/1.51336. URL: https://doi.org/10.2514/1.51336.

[13] INTER-AGENCY SPACE DEBRIS COORDINATION COMMITTEE. *IADC Space Debris Mitigation Guidelines*. English. Tech. rep. IADC-02-01. IADC, Sept. 2007. URL: https://www.unoosa.org/documents/pdf/spacelaw/sd/IADC-2002-01-IADC-Space_Debris-Guidelines-Revision1.pdf.

[14] Pengzhan Jin et al. "SympNets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems". In: *Neural Networks* 132 (2020), pp. 166–179.

[15] Nicholas L. Johnson. "U.S. space surveillance". en-US. In: *Advances in Space Research* 13.8 (Aug. 1993), pp. 5–20. ISSN: 0273-1177. DOI: 10.1016/0273-1177(93)90563-Q. URL: https://www.sciencedirect.com/science/article/pii/027311779390563Q.

[16] Ryo Kato et al. "Validity Evaluation of Anomaly Detection Using LSTM AutoEncoder for Maneuver Detection". en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (Sept. 2023).

[17] Tom Kelecy and Moriba Jah. "Detection and orbit determination of a satellite executing low thrust maneuvers". en-US. In: *Acta Astronautica* 66.5-6 (Mar. 2010), pp. 798–809. ISSN: 0094-5765. DOI: 10.1016/j.actaastro.2009.08.029. URL: https://www.sciencedirect.com/science/article/pii/S0094576509004287.

[18] Tom Kelecy et al. "Satellite Maneuver Detection Using Two-line Element (TLE) Data". en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (Jan. 2007).

[19] Hyun Chul Ko and Daniel J. Scheeres. "Maneuver Detection with Event Representation Using Thrust Fourier Coefficients". In: *Journal of Guidance, Control, and Dynamics* 39.5 (Feb. 2016), pp. 1080–1091. ISSN: 0731-5090. DOI: 10.2514/1.G001463. URL: https://doi.org/10.2514/1.G001463.

[20] Lincheng LI et al. "Design of low-thrust control in the geostationary region for station keeping". en. In: (2019). DOI: 10.13009/EUCASS2019-723. URL: https://www.eucass.eu/doi/EUCASS2019-0723.pdf.

[21] Dong C. Liu and Jorge Nocedal. "On the limited memory BFGS method for large scale optimization". en. In: *Mathematical Programming* 45.1 (Aug. 1989), pp. 503–528. ISSN: 1436-4646. DOI: 10.1007/BF01589116. URL: https://doi.org/10.1007/BF01589116.

[22] Ilya Loshchilov and Frank Hutter. "Fixing Weight Decay Regularization in Adam". In: *CoRR* abs/1711.05101 (2017). arXiv: 1711.05101. URL: http://arxiv.org/abs/1711.05101.

[23] Daniel P Lubey and Daniel J Scheeres. “An Optimal Control Based Estimator for Maneuver and Natural Dynamics Reconstruction”. en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (Jan. 2015).

[24] Felix Öhlinger et al. *The satellite-only gravity field model GOCO2025s*. 2025-07-07. DOI: 10.3217/f48p8-8h651. URL: https://repository.tugraz.at/records/f48p8-8h651.

[25] Russell Patera. “Space Event Detection Method”. In: *Journal of Spacecraft and Rockets - J SPACECRAFT ROCKET* 45 (May 2008), pp. 554–559. DOI: 10.2514/1.30348.

[26] Nicholas Perovich, Zachary Folcik, and Rafael Jaimes. “Applications of Artificial Intelligence Methods for Satellite Maneuver Detection and Maneuver Time Estimation”. en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (Sept. 2022).

[27] Lorenzo Porcelli et al. “Satellite maneuver detection and estimation with radar survey observations”. en-US. In: *Acta Astronautica* 201 (Dec. 2022), pp. 274–287. ISSN: 0094-5765. DOI: 10.1016/j.actaastro.2022.08.021. URL: https://www.sciencedirect.com/science/article/pii/S0094576522004258.

[28] PyTorch. *The Fundamentals of Autograd — PyTorch Tutorials 2.10.0+cu128 documentation*. Nov. 2024. URL: https://docs.pytorch.org/tutorials/beginner/introyt/autogradyt_tutorial.html.

[29] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (Feb. 2019), pp. 686–707. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2018.10.045. URL: https://www.sciencedirect.com/science/article/pii/S0021999118307125.

[30] L Rickman and Carlos R Ortiz Longo. *Method for the Calculation of Spacecraft Umbra and Penumbra Shadow Terminator Points*. en. Tech. rep. NASA-TM-3547. 1995.

[31] Thomas G Roberts and Richard Linares. “Geosynchronous satellite maneuver classification via supervised machine learning”. en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (June 2021).

[32] Thomas G Roberts, Haley E Solera, and Richard Linares. “Geosynchronous satellite behavior classification via unsupervised machine learning”. In: *9th Space Traffic Management Conference, Austin, TX*. Vol. 3. 2023.

[33] Thomas G. Roberts and Carson Bullock. *A sustainable geostationary space environment requires new norms of behavior*. en-US. Aug. 2020. URL: https://sciencepolicyreview.org/2020/08/a-sustainable-geostationary-space-environment-requires-new-norms-of-behavior/.

[34] Bob Schutz, Byron Tapley, and George H. Born. *Statistical Orbit Determination*. en. Elsevier, June 2004. ISBN: 978-0-08-054173-0.

[35] Andrea Scorsoglio, Luca Ghilardi, and Roberto Furfaro. “A Physic-Informed Neural Network Approach to Orbit Determination”. en. In: *The Journal of the Astronautical Sciences* 70.4 (Aug. 2023), p. 25. ISSN: 2195-0571. DOI: 10.1007/s40295-023-00392-w. URL: https://doi.org/10.1007/s40295-023-00392-w.

[36] Andrea Scorsoglio et al. “Orbit determination pipeline for geostationary objects using physics-informed neural networks”. In: *AIAA SCITECH 2024 Forum*. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan. 2024. DOI: 10.2514/6.2024-1862. URL: https://arc.aiaa.org/doi/10.2514/6.2024-1862.

[37] Peng Mun Siew et al. “AI SSA Challenge Problem: Satellite Pattern-of-Life Characterization Dataset and Benchmark Suite”. en. In: *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (Sept. 2023).

[38] Peng Mun Siew et al. “Satellite Pattern-of-Life Identification Challenge: Competition Design and Results”. In: *25th Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS), Maui, HI*. Sept. 2024.

[39] Robert A. Singer. “Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets”. In: *IEEE Transactions on Aerospace and Electronic Systems* AES-6.4 (July 1970), pp. 473–483. ISSN: 1557-9603. DOI: 10.1109/TAES.1970.310128. URL: https://ieeexplore.ieee.org/abstract/document/4103555.

[40] Haley E Solera, Thomas G Roberts, and Richard Linares. “AN OPEN-SOURCE ALGORITHM FOR MONITORING GEO SLOT UTILIZATION AND SATELLITE CLUSTERING”. en. In: *IAA Space Traffic Management Conference* (Mar. 2025).

[41] Matthew Tancik et al. *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*. arXiv:2006.10739 [cs]. June 2020. DOI: 10.48550/arXiv.2006.10739. URL: http://arxiv.org/abs/2006.10739.

[42] United States Space Force. *Ground-Based Electro-Optical Deep Space Surveillance*. en-US. URL: https://www.spaceforce.mil/About-Us/Fact-Sheets/Fact-Sheet-Display/Article/2197760/ground-based-electro-optical-deep-space-surveillance/.

[43] Rafael Vazquez et al. “Manoeuvre detection for near-orbiting objects”. In: *8th European Conference on Space Debris*. 2021, p. 172.

[44] Karel F Wakker. *Fundamentals Of Astrodynamics*. Delft: Institutional Repository Library Delft University of Technology Delft - The Netherlands, 2015. ISBN: 978-94-6186-419-2.

[45] E.A. Wan and R. Van Der Merwe. “The unscented Kalman filter for nonlinear estimation”. en. In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. Lake Louise, Alta., Canada: IEEE, 2000, pp. 153–158. ISBN: 978-0-7803-5800-3. DOI: 10.1109/ASSPCC.2000.882463. URL: http://ieeexplore.ieee.org/document/882463/.