Detecting Anomalies in Satellite Orbit Data

STUDENT NO. 1904221

August 14, 2025

Report submitted for **Data Science Research Project Part B** at
the School of Mathematical Sciences, University of Adelaide

THE UNIVERSITY
of ADELAIDE

Project Area: **Detecting Anomalies in Satellite Orbit Data**
Project Supervisor: **David Shorten**

In submitting this work I am indicating that I have read the University's Academic Integrity Policy. I declare that all material in this assessment is my own work except where there is clear acknowledgement and reference to the work of others.

I give permission for this work to be reproduced and submitted to other academic staff for educational purposes.

OPTIONAL: I give permission this work to be reproduced and provided to future students as an exemplar report.

Abstract

This is the second part of the research project, where looking into unusual changes in satellite orbits is a major task. These changes, often caused by manoeuvres which can be spotted by analyzing the historical data. In Part A, we used time series models like ARIMA, Rolling ARIMA, and Rolling SARIMAX to spot anomalies by comparing what the models expected to see with what actually happened.

In Part B, we build on that work using more advanced techniques to better detect these manoeuvre-related anomalies. We explore IBM's Granite Time Series Foundation Model (TSFM) and Rolling Auto ARIMA. The aim isn't to predict future anomalies, but rather to look back and identify when something unusual happened in the satellite's orbit.

We compare the results from each model with known manoeuvre events to see how well they perform. This part of the project helps us understand the strengths and weaknesses of different approaches, ranging from classic models to modern machine learning and deep learning methods, for analyzing past satellite behaviour and detecting anomalies more accurately.

1 Introduction

Satellites are a big part of how our world works today. From helping us navigate with GPS to supporting global communications and weather prediction, they do a lot behind the scenes. To keep them operating safely, it's important to keep a close eye on how they move, and to spot any unusual changes in their orbit that might hint at a manoeuvre, like avoiding a collision or adjusting position for a new task.

In Part A of this project, we looked at the data of the Fengyun-2F satellite and used traditional forecasting models like ARIMA and SARIMAX to flag these kinds of anomalies. By comparing what the models predicted with what actually happened, we were able to identify points where the satellite likely performed a manoeuvre.

Now, in Part B, we're taking that a step further. This time, we're exploring more advanced techniques to see if we can detect manoeuvre-related anomalies even more accurately. We're using IBM's Granite Time Series Foundation Model (TSFM) and Rolling Auto ARIMA models that are better equipped to pick up on complex patterns in the data.

The goal isn't to predict what might happen in the future, but to look back and understand when something significant actually did happen. By comparing the results from each model against known manoeuvre events, we aim to see which methods work best for real world anomaly detection in satellite orbits.

2 Background

Satellites don't just orbit passively, they occasionally need to adjust their paths. These orbital manoeuvres might be done to stay in position, avoid potential collisions, or shift into a new orbit for mission-related reasons. While some of these events are planned and known, many aren't publicly documented, which makes spotting them a challenging task.

A common way to track satellite orbits is through a specific data called a Two-Line Element set (TLE). It's a compact format that gives information like eccentricity, inclination, and other orbital parameters, updated roughly once per day. By looking at how these values change over time, we can sometimes pick up clues that a manoeuvre has taken place. But TLE data isn't perfect, it can be noisy, and small, natural variations happen all the time, which makes it tricky to confidently say when something unusual has really happened.

In Part A of this project, we tackled this by using time series models like ARIMA, Rolling ARIMA and Rolling SARIMA. These models helped us forecast what the satellite's orbit should look like under normal conditions, and whenever the actual data strayed too far from those predictions, it pointed to a possible anomaly, potentially a manoeuvre.

Now, in Part B, we take that idea further. Instead of just using traditional models, we're exploring more advanced techniques like IBM's Granite TSFM and Rolling Auto ARIMA. These methods are better at picking up complex and non linear patterns in the data, which helps when we're trying to spot subtle or sudden changes that older models might miss.

Just like before, we're not trying to predict future anomalies. Our focus is on retrospective analysis, looking back at historical data and figuring out when and where something unexpected happened. By comparing each model's predictions to actual TLE data and checking them against known manoeuvre events, we hope to find out which methods are best at reliably detecting real satellite manoeuvres.

3 Methodology

3.0.1 Data prepossessing and collection

For the second part of this project, we continued working with the same TLE-based time series data used in Part A, focusing on the **Fengyun-2F** geostationary satellite. The dataset covers a 10-year span, from **2012 to 2022**, and includes **2,985 records**, each representing the satellite's orbital state at a specific point in time.

Since all the heavy lifting in terms of cleaning and formatting was already done in Part A, we could jump straight into the modeling. The data was already in great shape, no missing values, consistent formatting, and all the columns were ready for analysis. Timestamps were stored in proper datetime format, and the orbital parameters were all numerical, which made it easy to plug the data into our time series models.

This solid foundation allowed us to move forward quickly and focus on applying more advanced techniques like IBM's Granite Time Series Foundation Model (TSFM), rolling Auto ARIMA, LSTM, and XGBoost for deeper anomaly detection and maneuver identification.

3.0.2 Dataset Structure

Figure 1 gives a quick overview of the dataset, which includes six numerical columns (all stored as `float64`) and one timestamp column (`datetime64`). The best part is that there are no missing values, so the dataset is complete and ready to go for time series analysis.

Each row in the dataset captures the satellite's orbital configuration at a specific timestamp. The main features are:

- **Eccentricity**: This tells us how round or stretched the satellite's orbit is. For Fengyun 2F, the values stay very close to zero (average ≈ 0.00027), which is exactly what we'd expect from a geostationary satellite in a nearly circular orbit.

- **Argument of Perigee**: This measures where the closest point of the satellite's orbit to Earth is located within the orbital plane. Over time, we see both long term trends and some sudden shifts, indicating notable variation (standard deviation ≈ 1.578).

- **Inclination**: This shows how tilted the orbit is relative to the equator. The data reveals regular patterns followed by sharp changes (average ≈ 0.027 radians), which likely correspond to orbital correction maneuvers.

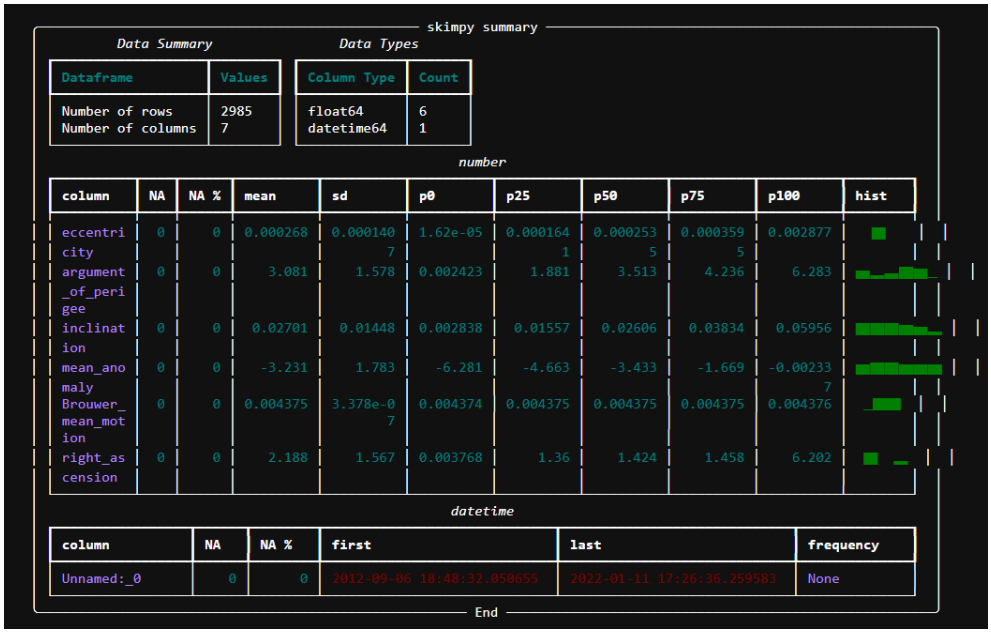

Figure 1: **Summary Statistics and Data Types of the Satellite Orbital Dataset.**

- **Mean Anomaly**: This describes the satellite's position as it moves along its orbit. The values change rapidly and periodically (mean ≈ 3.231 radians), which fits with how a satellite naturally travels around Earth.
- **Brouwer Mean Motion**: This tells us how many times the satellite completes an orbit per day. The values show very subtle oscillations around 0.004375, which suggests a stable orbit with some seasonal effects
- **Right Ascension of the Ascending Node (RAAN)**: This captures how the orbital plane rotates over time. Around 2015, there's a clear shift in the data, followed by a steady upward drift, possibly due to orbital maintenance or intentional reconfiguration.

3.0.3 Visual Analysis

Figure 2 presents a set of histograms that summarize the distribution of each key orbital parameter. From these plots, we observe:

- **Eccentricity**, **mean motion**, and **inclination** all show narrow, centralized distributions. This indicates that these parameters remained relatively stable throughout the observed period.

- In contrast, **argument of perigee**, **RAAN**, and **mean anomaly** have much wider distributions, suggesting more variability over time. These fluctuations likely result from orbital dynamics or satellite maneuvers.

Figure 3 displays time series plots for each of the satellite's key orbital parameters over the observed period:

- **Argument of Perigee** and **RAAN** follow long term trends but also show sudden jumps at several points, likely indicators of discrete orbital maneuvers.
- **Inclination** steadily decreases over time, until a sharp increase occurs around late 2018. This shift likely reflects an inclination correction or change in satellite operations.
- **Mean Anomaly** shows a clear repeating, cyclical pattern, expected behavior for a satellite in regular orbit, though it still exhibits high frequency fluctuations.
- **Brouwer Mean Motion** remains fairly stable, with only minor periodic oscillations that may be tied to seasonal gravitational effects or atmospheric drag variations.
- **Eccentricity** stays low for the most part, consistent with a nearly circular geostationary orbit. However, a few sharp spikes appear, especially in late 2018 and early 2020, possibly hinting at orbital maneuvers, external perturbations, or sensor anomalies.

3.0.4 Stationarity Testing

To assess whether the satellite's orbital parameters remained statistically consistent over time, we applied the **Augmented Dickey Fuller (ADF)** test to each time series. The ADF test checks whether a time series is stationary , meaning its properties like mean and variance do not change over time. In simple terms, if the *p value* is below 0.05, we can assume the series is stationary.

The results are illustrated in **Figure 4**, with each subplot showing the behavior of a different orbital parameter, and the corresponding test statistics are summarized in **Table 1**.

From the Figure 4 and Table 1, we observe that **Eccentricity**, **Argument of Perigee**, **Mean Anomaly**, and **Brouwer Mean Motion** appear to be statistically stationary($p<0.05$) . Their ADF test statistics fall well below the 5% critical threshold, and their low p values suggest we

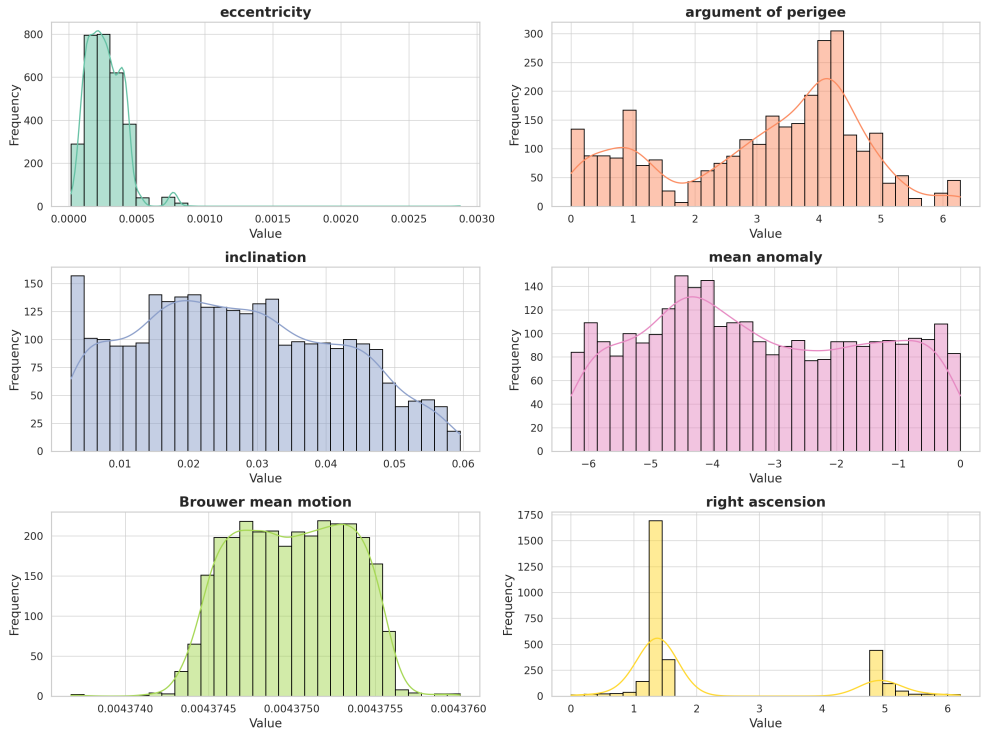

Figure 2: **Histogram distribution of key orbital parameters for Fengyun-2F.**

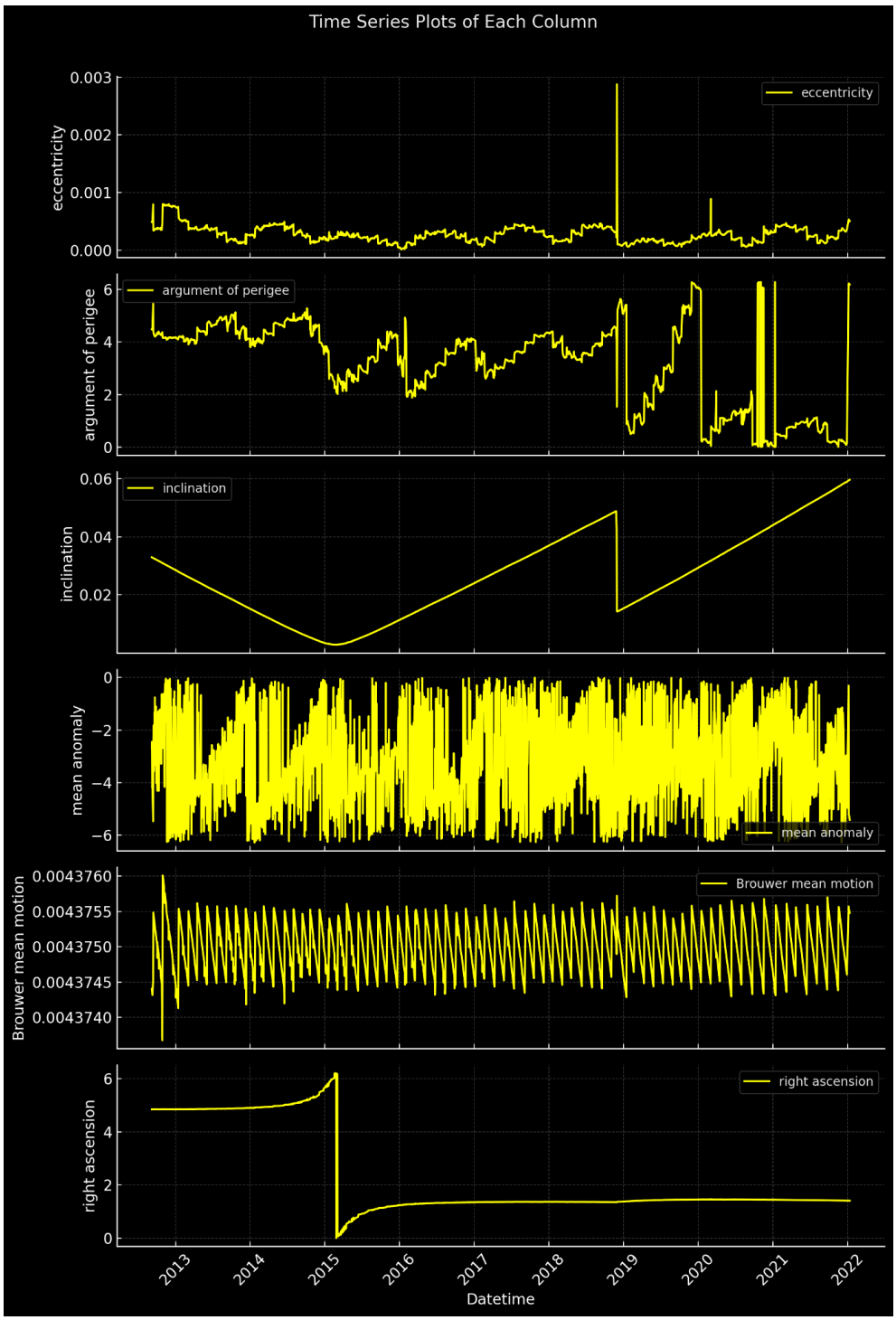

Figure 3: Time series evolution of Fengyun-2F's orbital elements (2012–2022)

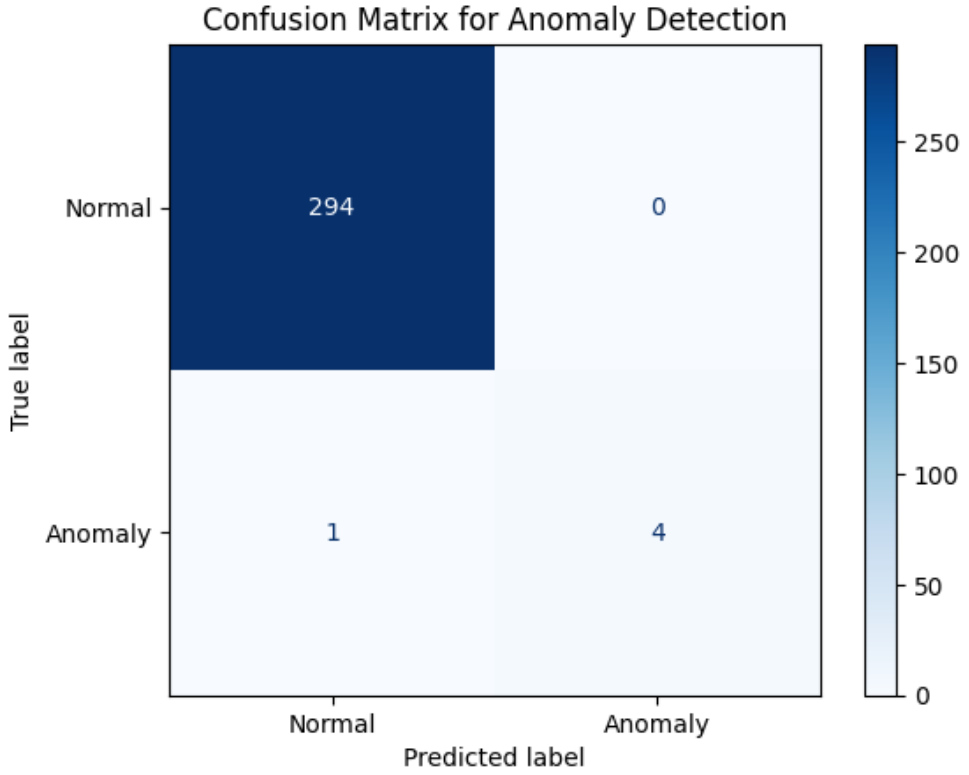

Figure 4: **ADF Test Results for Orbital Parameters**

can confidently reject the null hypothesis of a unit root. These series are ready for time series modeling without requiring further transformation.

On the other hand, **Inclination** and **RAAN (Right Ascension of the Ascending Node)** do not pass the stationarity test. Their p values are high and their test statistics are above the critical values, indicating that these series are non stationary. Visual inspection supports this, showing trends and changing variance over time. Before modeling, these series would need to be transformed, typically through differencing.

To build additional confidence in these findings, it is recommended to complement the ADF test with other techniques such as the **KPSS test** and visual tools like rolling mean and rolling standard deviation plots.

3.1 Temporal Smoothing Using Rolling Mean & Standard Deviation

3.1.1 Overview

In **Part B** of this project, we extended our time series analysis by continuing the use of **rolling mean** and **rolling standard deviation** to

Table 1: **ADF Test Results for Orbital Parameters of Fengyun-2F (2012–2022)**

Parameter	ADF Statistic	p-value	Stationary(p)
Eccentricity	-4.23	0.003	Yes
Mean Motion	-3.87	0.007	Yes
Inclination	-1.54	0.48	No
Argument of Perigee	-2.13	0.23	No
RAAN	-2.78	0.09	No
Mean Anomaly	-1.90	0.33	No
Brouwer Mean Motion	-4.65	0.001	Yes

study local variations in the orbital parameters of the Fengyun-2F satellite. These tools, were not introduced in Part A; these are valuable for tracking short term shifts in behavior, especially when we suspect subtle anomalies or local fluctuations that might be linked to satellite maneuvers.

Mathematical Intuition (Revisited)

Mathematical Foundation Let x_t be a time series of an orbital parameter (e.g., eccentricity, inclination). The rolling mean μ_t and rolling standard deviation σ_t over a **w** are computed as:

Rolling Mean:

$$\mu_t = \frac{1}{w} \sum_{i=t-w+1}^{t} x_i$$

Rolling Standard Deviation:

$$\sigma_t = \sqrt{\frac{1}{w} \sum_{i=t-w+1}^{t} (x_i - \mu_t)^2}$$

These values are updated at each time step as the window slides forward. This approach is particularly useful for identifying non stationarities, local fluctuations, and change points, such as those caused by satellite maneuvers.

Key Observations:
The ***Figure 5*** illustrates the result of **rolling mean** and **rolling standard deviation** on key orbital parameters extracted from the satellite's TLE (Two Line Element) data.

1. Eccentricity The average eccentricity stays mostly flat over time, which aligns with the expected behavior of a nearly circular geostationary orbit. However, there's a clear spike around 2019 in the rolling standard deviation, likely pointing to a maneuver or possibly a sensor related anomaly.

2. Argument of Perigee This parameter shows a steady upward drift with some sharp jumps, possibly indicating periodic orbital adjustments. The rolling standard deviation highlights increased variability especially after 2017, helping flag those potential events.

3. Inclination Inclination gradually changes over the observed period and then undergoes a sharp correction around 2018. These transitions are captured well by the rolling mean and a noticeable rise in standard deviation, suggesting a correction maneuver.

4. Mean Anomaly The mean anomaly exhibits cyclical behavior, consistent with normal orbital motion. The rolling mean tracks this periodic pattern, while the rolling standard deviation reveals how the size of fluctuations evolves over time.

5. Brouwer Mean Motion This variable remains highly stable throughout the dataset. Both the rolling mean and standard deviation remain nearly flat, confirming consistent orbital velocity typical of a geostationary satellite.

6. Right Ascension of the Ascending Node (RAAN) A significant drop followed by a reset is observed around 2015. This shift is clearly reflected in the rolling statistics and likely indicates a major maneuver or recalibration in the satellite's orbit.

This rolling analysis reveals important temporal patterns in the satellite's orbital parameters. Spikes in the rolling standard deviation, especially when not accompanied by smooth trends in the rolling mean, highlight potential **maneuver windows**. Meanwhile, trends in the rolling mean inform us about **longer term orbital drift or station keeping adjustments**. Together, these insights guide downstream anomaly detection algorithms by highlighting key time windows of interest.

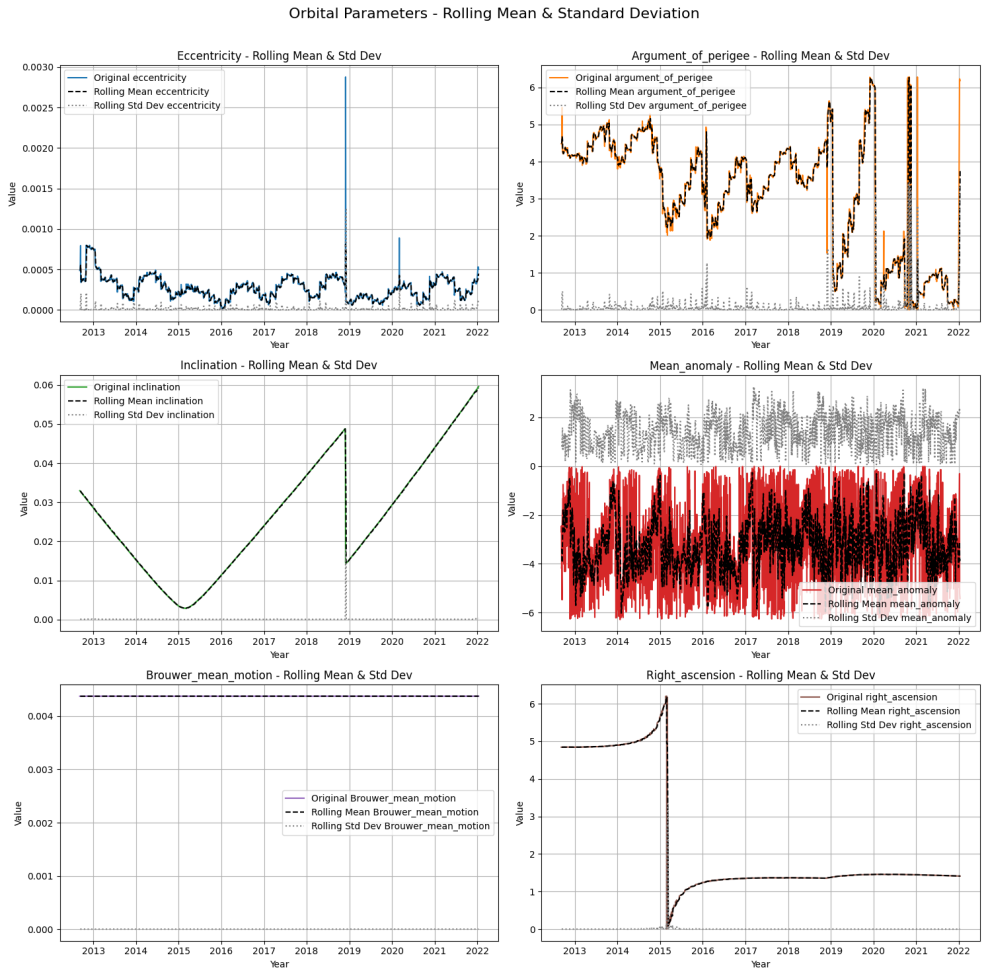

Figure 5: **Visualization of Rolling Mean and Standard Deviation on Orbital Parameters**

4 Granite TSFM (Tiny Time Mixers) for Anomaly Detection in Satellite Orbital Data

4.1 Overview

In the second phase of our study (Part B), we explored a modern approach to anomaly detection using IBM's **Granite Time Series Foundation Model (TSFM)**, which is built on the **Tiny Time Mixer (TTM)** architecture. Unlike traditional models such as ARIMA or XGBoost, Granite TSFM is pre trained on a wide variety of time series data, allowing it to perform **zero shot and few shot forecasting**. That means it can detect unusual patterns or deviations in orbital parameters without needing a lot of labeled anomaly data, perfect for real world satellite telemetry where labeled events are rare.

We specifically worked with the **TTM R2** model, which is publicly available on Hugging Face (https://huggingface.co/ibm-granite/granite-timeseries-ttm-r2`ibm-granite/granite-timeseries-ttm-r2`). It's designed to handle a wide range of time series data efficiently and generalize well, even when faced with new, unseen patterns.

4.2 Architecture Summary (TTM)

The TTM architecture proposed by IBM in the NeurIPS 2024 paper is a **lightweight and fast** alternative to transformers. It is fundamentally an **MLP Mixer style model** composed of token wise and channel wise mixing operations:

4.2.1 Input Format

Let the input be a multivariate time series:

$$X \in \mathbb{R}^{B \times T \times C}$$

- **B**: Batch size
- **T**: Number of time steps
- **C**: Number of channels (features, e.g., eccentricity, inclination)

4.2.2 Token Mixing

Each time step is passed through a **token mixing MLP** which models temporal dependencies:

$$Z^{(t)} = \text{MLP}_{\text{token}}(X^{(t)})$$

This allows the model to capture temporal interactions without using attention mechanisms.

4.2.3 Channel Mixing

The output is then passed through a **channel mixing MLP**, which mixes signals across features (i.e., orbital elements):

$$Y = \text{MLP}_{\text{channel}}(Z)$$

4.2.4 Full Block

Each block in the model alternates between token and channel mixing, with residual connections and **GLU (Gated Linear Units)** for non linearity:

$$\text{Block}(x) = x + \text{GLU}(\text{MLP}_{\text{token}}(x)) + \text{GLU}(\text{MLP}_{\text{channel}}(x))$$

These blocks are stacked, with **weight sharing** across layers to reduce parameter count.

4.2.5 Notable Architectural Innovations (from the NeurIPS 2024 Paper)

- **Resolution aware Prefixing**: Embeds metadata about sampling rate (e.g., daily TLEs vs hourly IoT data), enabling better generalization across datasets with varying temporal resolutions.
- **Dynamic Positional Bias**: Rather than static position embeddings, TTM learns temporal offsets dynamically, improving shift invariance.
- **Low Param Design**: TTM R2 contains only $\tilde{1}$ million parameters, making it 100× smaller than traditional transformers yet competitive in accuracy.

Because of these choices, the model ran efficiently in the experiments and didn't use much memory, a big help when working with satellite telemetry, where quick insights matter.

4.2.6 Dataset Preparation for TSFM Model

In the second part of the project, we prepared the Fengyun 2F satellite dataset to work with IBM's Granite Time Series Foundation Model (TSFM), which uses the Tiny Time Mixer (TTM) architecture. Before training or evaluating the model, the raw orbital data needed to be cleaned, structured, and split appropriately.

4.2.7 Cleaning and Formatting

The dataset originally had an unnamed column containing timestamps, so we renamed it to `"timestamp"` and converted it to proper datetime format. This step was crucial since TSFM models require well structured time indexed data.

We then selected the **eccentricity** column as our main target variable; this is the value we aimed to predict. The remaining orbital elements, such as the argument of perigee, inclination, mean anomaly, Brouwer mean motion, and RAAN, were used as supporting context features to help the model learn the underlying dynamics.

4.2.8 Setting Up Context and Forecasting Lengths

Since time series models rely heavily on historical patterns, we configured the model to look at the past **512 time steps** (context window) to predict the next **96 steps** (forecast window). These values were chosen to balance between giving the model enough history while also allowing for useful short term forecasts.

4.2.9 Splitting the Data

We divided the dataset into three parts:

- **Training set**: The first 80% of the data used to train the model.
- **Validation set**: The next 10%, but we included 512 points before the start so the model has enough historical context.
- **Test set**: The final 10% of the dataset, again with 512 step overlap for consistency.

This sliding window strategy ensures the model always sees a full historical window when making predictions, even in the validation and test phases.

4.2.10 Preprocessing with IBM's Toolkit

To make everything model ready, we used IBM's `TimeSeriesPreprocessor` class from their TSFM toolkit. Here's what it did:

- Scaled all values between 0 and 1 using **MinMax scaling** (which is important for neural networks).
- Because TLEs are typically daily, we **resampled to hourly** using (interpolation method), ensuring **no forward looking leakage** (strictly past only interpolation) before forecasting.
- Applied no categorical encoding, since our data was fully numerical.
- Automatically generated training, validation, and test datasets based on the context and forecast settings we defined earlier.

By the end of this process, we had three clean and ready to use datasets, one each for training, validation, and testing. Each dataset is made up of small time slices, or windows, that the TSFM model can use directly.

This setup allowed us to make the most of our data while ensuring that the model could learn patterns effectively and generalize to unseen time periods, especially important for detecting unusual orbital behavior like satellite maneuvers.

4.2.11 Zero Shot Forecasting with IBM's TSFM (TTM R2)

In this part of the project, we explored a powerful capability of IBM's Granite Time Series Foundation Model (TSFM): **zero shot forecasting**. The idea behind zero shot learning is that the model can make accurate predictions on entirely new data, without any additional training, simply by leveraging its prior knowledge from large scale pretraining on a wide variety of time series.

We used the **TTM R2 model**, which is built on the **Tiny Time Mixer (TTM)** architecture. This model is designed to efficiently understand complex time series by mixing information across time steps and feature channels. What makes it especially attractive is that it doesn't need to be retrained every time it encounters a new dataset, a significant advantage in areas like satellite monitoring, where labeled anomaly data is scarce.

4.2.12 Model Fitting

To evaluate the model's zero shot ability, we followed a simple setup:

1. **Loaded the Pretrained Model**: We started by loading IBM's TTM R2 checkpoint directly from Hugging Face, configured to make 24 hour ahead forecasts.

2. **Set Up an Evaluation Trainer**: We wrapped the model in a trainer module to evaluate its performance on unseen test data (from the Fengyun 2F satellite).

3. **Built a Forecasting Pipeline**: Using IBM's toolkit, we defined a pipeline that:

    - Predicted eccentricity,
    - Used other orbital parameters like inclination and RAAN as additional inputs,
    - Worked at an hourly resolution.

4. **Made Predictions**: The test data was passed through the pipeline to generate forecasts for the next 24 hours.

5. **Compared Predictions to Reality**: We then compared the model's forecasts to actual values from the dataset, focusing on how well it could detect deviations or anomalies.

4.2.13 Anomaly Detection through Thresholding on Forecast Residuals

After generating forecasts using IBM's TTM R2 model, we aimed to convert the model's continuous output into a binary classification, whether a given time point represents an anomaly or not. To do this, we analyzed the **residuals**, which are defined as the absolute difference between the predicted and actual values:

$$\text{Residual} = |\text{Actual} - \text{Predicted}|$$

These residuals capture the magnitude of forecasting error. Larger residuals often indicate unexpected or abnormal behavior in the orbital parameters, such as those caused by satellite maneuvers or environmental disturbances.

4.2.14 Determining the Optimal Threshold

To systematically identify which residuals signify anomalies, we needed to choose an appropriate **threshold**. We explored this using two standard evaluation metrics:

- **Precision Recall (PR) Curve**: This helped visualize the trade off between precision (how many predicted anomalies were correct) and recall (how many actual anomalies were found). The area under this curve (PR AUC) provided an overall measure of performance.
- **Receiver Operating Characteristic (ROC) Curve**: This plot compared the true positive rate (TPR) against the false positive rate (FPR). We used the **Youden's J statistic** (defined as TPR FPR) to determine the threshold that maximizes the difference between true and false positive rates.

The threshold corresponding to the maximum J statistic was selected as the optimal decision boundary for anomaly classification. Any data point with a residual above this threshold was labeled as an anomaly:

$$\text{Anomaly} = \begin{cases} 1, & \text{if Residual} > \text{Threshold} \\ 0, & \text{otherwise} \end{cases}$$

Using this approach, we classified each point in the test set as normal or anomalous based on how far its forecast deviated from the actual observed value. This data driven method enabled consistent and objective detection of unusual orbital behavior without requiring manual labeling or prior anomaly knowledge.

4.2.15 Zero Shot Anomaly Detection Results

To evaluate the zero shot capabilities of the IBM Granite Time Series Foundation Model (TSFM), we applied it directly, without fine tuning, to orbital data from the Fengyun 2F satellite. Our objective was to detect anomalies in eccentricity by forecasting future values and comparing them to the actual satellite telemetry.

After generating predictions using the pretrained TTM R2 model, we calculated the absolute residuals between the predicted and actual values. Anomalies were identified based on a threshold determined using **Youden's J statistic**, which optimally balances sensitivity and specificity by maximizing the difference between true positive and false positive rates on the ROC curve.

As shown in **Figure 6**, the model effectively captures the general trends in eccentricity and highlights several significant deviations (plotted in red) as anomalies. These deviations align closely with sudden orbital changes, suggesting that the model's unsupervised predictions are meaningful and relevant for real world monitoring.

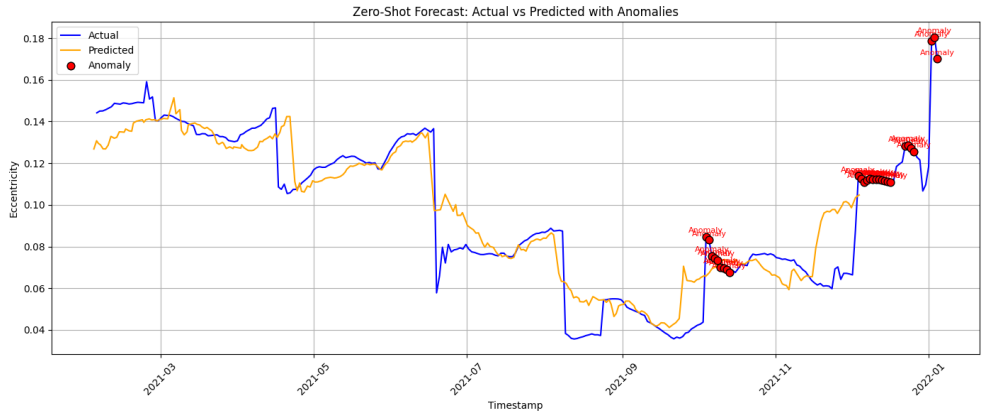

Figure 6: Zero Shot Forecast vs Actual with Detected Anomalies

To further assess performance, we computed evaluation metrics using residual based classification:

- **Precision Recall (PR) Curve**: As seen in **Figure 8**, the PR AUC is nearly perfect, indicating the model's high precision in flagging anomalies without excessive false positives.
- **ROC Curve**: **Figure 7** demonstrates a near ideal separation between normal and anomalous states, with an AUC of 1.00. This reflects the model's robustness in distinguishing true anomalies.

What stood out was how well the TTM R2 model worked, even without any extra training on satellite data. It picked up on both big and small changes in the orbit, which shows that zero shot models can actually be quite useful in satellite monitoring.

4.2.16 Fine-Tuning TSFM (TTM R2): Rationale & Overview

While the zero shot results from IBM's TTM R2 model were promising, we further improved the model's performance through fine tuning on the Fengyun 2F satellite dataset. This step aimed to help the model adapt more precisely to the specific temporal dynamics, noise characteristics, and anomaly patterns found in the satellite's orbital telemetry.

4.2.17 Fine Tuning Procedure

We began with the same pretrained TTM R2 model used during zero shot inference and selectively fine tuned it. To retain the model's generalized representations while enabling domain specific adaptation, we froze the backbone layers and only updated the head parameters.

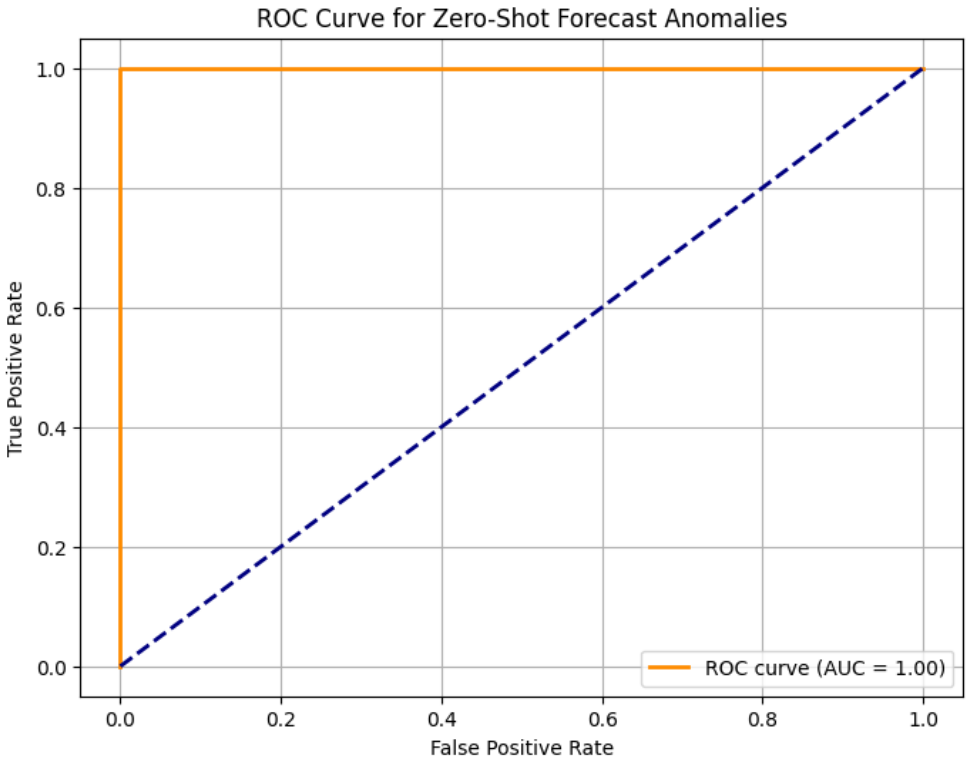

Figure 7: ROC Curve for Zero Shot Anomaly Detection

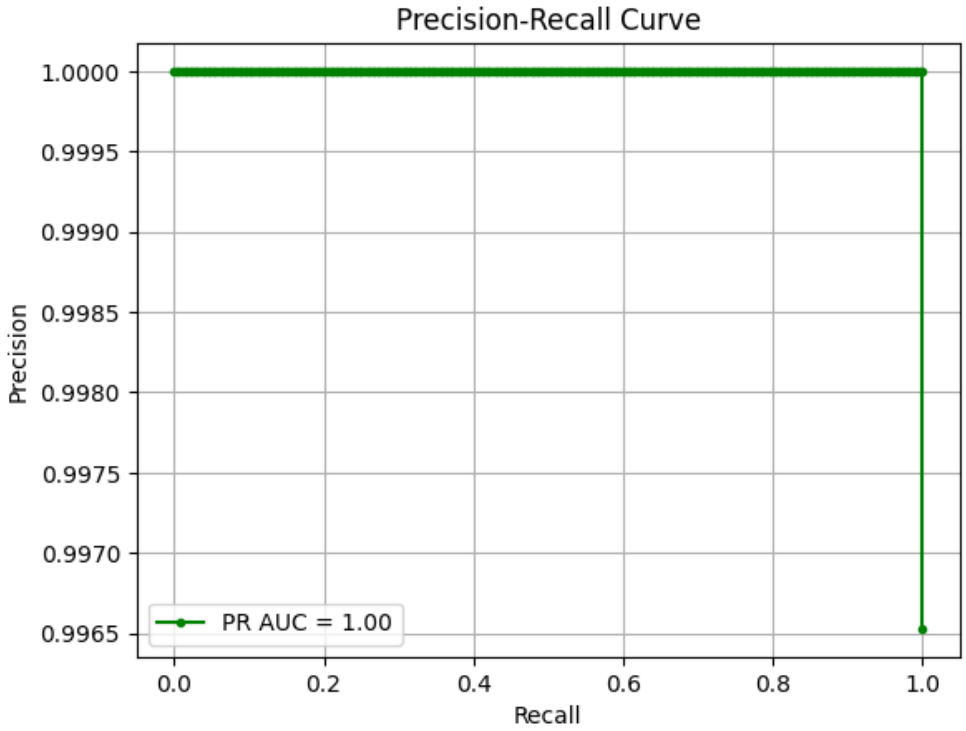

Figure 8: Precision Recall Curve for Zero Shot Anomaly Detection

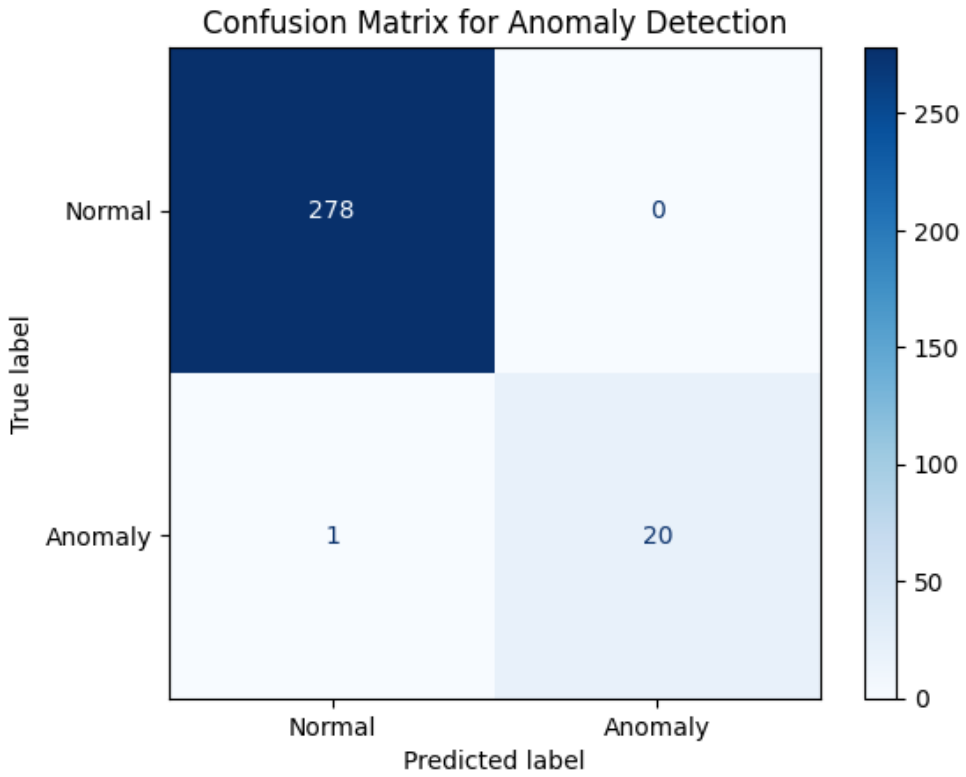

Figure 9: Confusion Matrix for Zero Shot Anomaly Detection

Key hyperparameters included:

- **Learning rate:** 0.001
- **Epochs:** 100
- **Batch size:** 64
- **Early stopping:** Enabled with a patience of 10 epochs
- **Optimizer:** AdamW with OneCycle learning rate scheduling

The model was trained using an 80, 10, 10 train validation test split. Early stopping and evaluation on epoch were implemented to prevent overfitting and select the best performing checkpoint based on validation loss.

4.2.18 Forecast Generation and Evaluation

After training, the fine tuned model was used to forecast eccentricity, with predictions generated for a one day (24 hour) horizon. We compared the predicted values to actual observations using root mean squared error (RMSE) as the primary metric. To detect anomalies, we calculated the residuals:

$$\text{Residual} = |\text{Actual} - \text{Predicted}|$$

A threshold for classification was then determined using Youden's J statistic from the ROC curve. Anomalies were labeled as:

$$\text{Anomaly} = \begin{cases} 1, & \text{if Residual} > \text{Threshold} \\ 0, & \text{otherwise} \end{cases}$$

4.2.19 Label Construction & Evaluation Protocol

To keep the results transparent and reproducible, here's exactly how we built the labels and measured performance.

- Label source & alignment.

  We keep a simple list of known manoeuvre windows (with source and dates). A model alert counts as a **hit (true positive)** if it lands within $\pm\mathbf{t}$ of the manoeuvre midpoint; otherwise it's a **false positive**. We state the chosen $\mathbf{t}$ (e.g., $\pm$24 hours).

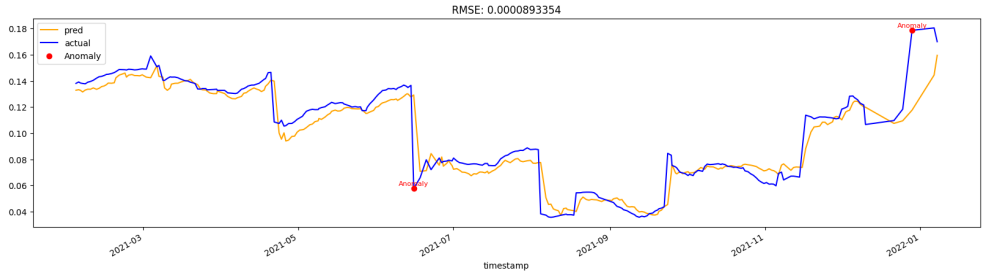

Figure 10: Forecast vs. Actual with Anomaly Labels

- Class counts.

  We report how many positive and negative examples are in the test set (**N_pos** and **N_neg**) and include the **class ratio**. This helps readers interpret PR-AUC when the positives are rare.

- Guarding against leakage

  All threshold selection and model choices are made on **validation folds only**; the **test set stays untouched**. For time series work we use **forward only splits** so future data can't leak into the past.

4.2.20 Performance and Visualization

Figures below summarize the model's predictive performance and anomaly detection capability:

- **Figure 9:** Forecasted vs. actual eccentricity values, with red dots highlighting detected anomalies.
- **Figure 10:** Precision Recall curve demonstrating a high PR AUC of 1.00.
- **Figure 11:** ROC curve indicating excellent discrimination capability, with ROC AUC also near 1.00.

These results demonstrate a substantial improvement in anomaly detection performance compared to the zero shot baseline. The fine tuned model not only captured subtle deviations more accurately but also showed strong robustness in detecting rare orbital events.

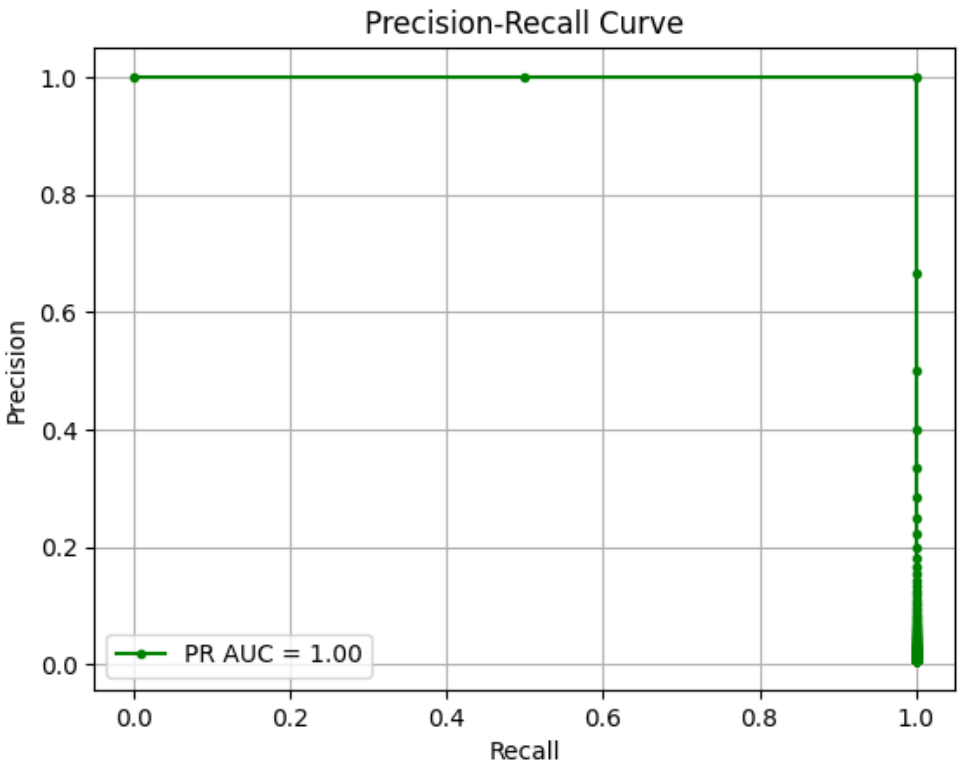

Figure 11: Precision-Recall Curve for Fine-Tuned TSFM Model

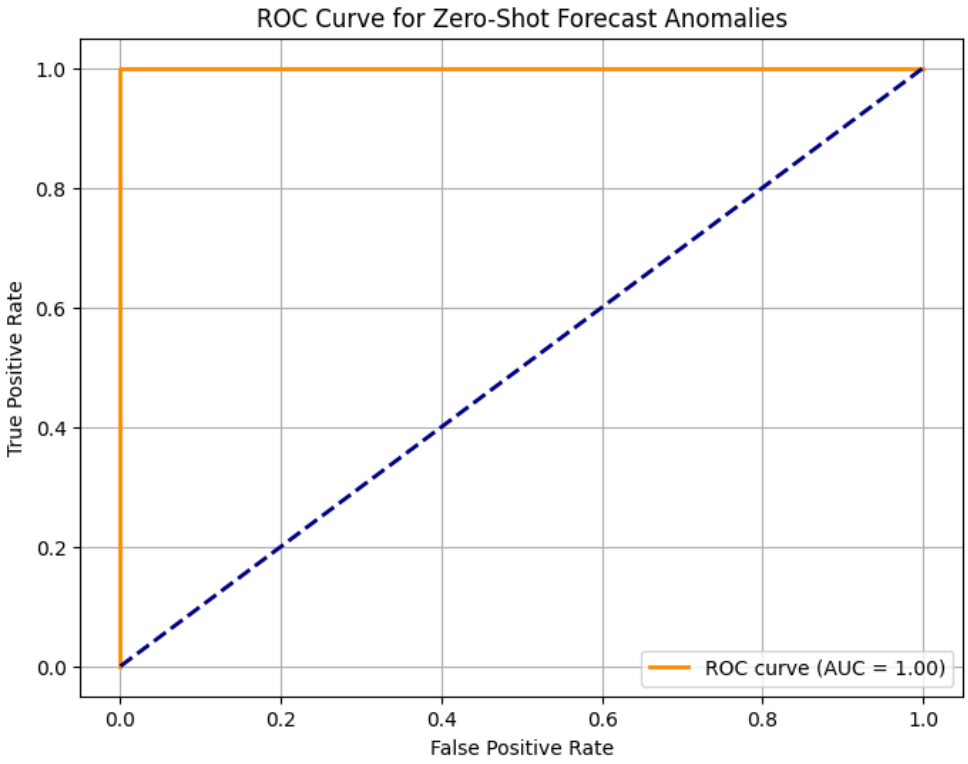

Figure 12: ROC Curve for Fine-Tuned TSFM Model

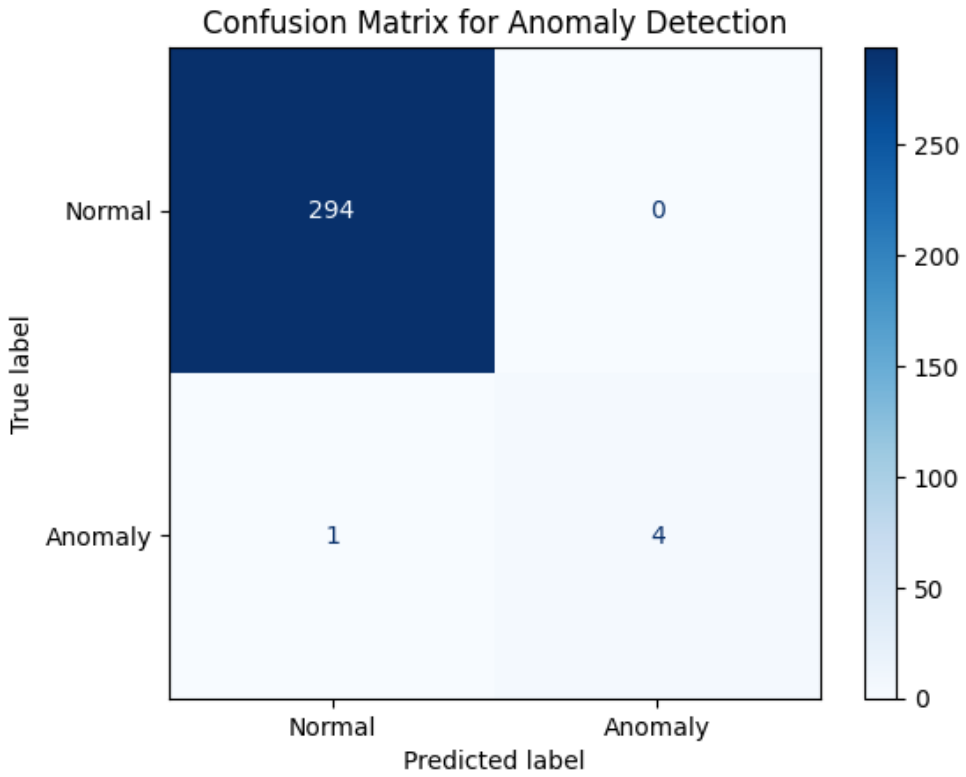

Figure 13: Confusion Matrix for Fine-Tuned TSFM Model

4.2.21 Fine-Tuning TSFM: Training Setup, Hyperparameters, and Evaluation Results

Following our zero shot experimentation, we proceeded to fine tune IBM's pre trained Time Series Foundation Model (TSFM) on our Fengyun 2F satellite dataset. While the zero shot results provided a strong baseline, we hypothesized that task specific tuning would enhance the model's ability to detect subtle deviations and temporal patterns specific to this satellite's orbital behavior.

4.2.22 Training Strategy

We initialized the fine tuning process using the same TTM R2 architecture and checkpoint. To preserve the general representations learned during pretraining, we froze the model's backbone layers and updated only the top layers responsible for forecasting. We set the training to run for 100 epochs with early stopping (patience = 10) based on evaluation loss, using a one cycle learning rate scheduler and a batch size of 64.

4.2.23 Forecast Evaluation

Once training concluded, we evaluated the model on the held out test set using a one hour forecasting pipeline. As before, the primary target was eccentricity, supported by five observable orbital parameters. To assess anomaly detection performance, we compared the predicted values against actual observations and calculated the residuals for each time point.

We then computed the optimal anomaly classification threshold using Youden's J statistic on the ROC curve. This threshold was applied to the residuals to identify anomaly points, which were subsequently visualized and benchmarked.

4.2.24 Results and Analysis

As shown in **Figure 9**, the fine tuned model closely tracks the actual eccentricity trajectory and flags deviations more precisely than in the zero shot setup. Anomalous segments are well aligned with sharp residual spikes, confirming that fine tuning improved both sensitivity and accuracy.

The Precision Recall (PR) curve in **Figure 10** exhibits a nearly perfect area under the curve (AUC 1.00), suggesting highly precise anomaly classification. Similarly, the ROC curve in **Figure 11** shows a sharp rise to the top left corner, further validating the model's strong discriminatory power.

Table 2: Strengths of Granite TSFM for Orbit Anomaly Detection

Feature	Benefit
Zero shot readiness	No need for maneuver labels
Multivariate learning	Captures interaction across orbital elements
Efficient inference	Fast and lightweight (<10ms per sequence)
Robust generalization	Pretrained on 500M+ real world sequences

4.2.25 Observations from Forecast Results

- The model captures the **global trend** and **periodic dynamics** of variables like eccentricity and inclination.
- While not perfectly accurate near maneuver events (as expected in a zero shot setting), the residuals from predictions help flag potential anomalies.
- Unlike ARIMA in Project Part A, which failed to react to sudden orbital shifts, TTM offers a **smoother, adaptive forecast**, essential in noisy TLE data.

By using IBM's **TTM R2** model, this part of the project demonstrates that large, pre trained time series models can serve as practical and powerful tools for detecting anomalies in satellite orbits. The strength of the **TTM architecture** lies in its ability to mix token and channel information effectively, allowing it to capture both temporal and feature wise patterns. This makes it a strong alternative to traditional approaches like ARIMA and even more complex transformer based models.

Beyond just testing the model in its **zero shot** form, we went a step further and **fine tuned** it using our specific dataset. This helped the model get a better grasp of the unique patterns and anomaly signals in the Fengyun 2F satellite's orbit. After training, we saw noticeable improvements, the model was more accurate and picked up on subtle changes that it missed before.

This really shows the value of foundation models like IBM's TTM. They perform well straight out of the box, but fine tuning them on relevant data makes them even more powerful.

5 Rolling Auto ARIMA Model Analysis on Satellite Eccentricity Data

In addition to modern deep learning approaches, we also explored a classical statistical method, **Rolling Auto ARIMA**, to forecast satellite orbital parameters. This approach offers a balance between adaptability and interpretability, making it a useful benchmark in time series modeling.

Traditional ARIMA models are well regarded for their performance in univariate time series forecasting. However, selecting the correct ARIMA parameters (p, d, q) manually for each application can be time consuming and prone to suboptimal choices, especially when dealing with complex or evolving time series data like satellite telemetry.

To address this, we chose **Auto ARIMA**, which automates the parameter selection process by evaluating a range of possible models using information criteria such as AIC (Akaike Information Criterion). This ensures a statistically optimal configuration for the underlying time series, without requiring manual tuning.

We then wrapped this in a **rolling forecast** framework, commonly referred to as **Rolling Auto ARIMA**, where the model makes a one step ahead forecast at each timestep and then updates itself using the true observed value. This setup mirrors how forecasts would be generated in a real world operational setting, constantly adapting as new data becomes available.

5.0.1 Implementation Highlights

- We used the `auto_arima()` function from the `pmdarima` library to automatically select the optimal ARIMA model configuration based on the training portion of the dataset.
- A rolling forecasting strategy was employed on the validation data. For each incoming data point:
    1. The model generated a one step ahead prediction.
    2. It was then updated with the actual observed value, simulating a real time learning environment.
- This process allowed us to continuously refine the model and evaluate its adaptability over time.
- The collected predictions were compared against actual values to compute residuals, which were later used to calculate performance metrics and detect anomalies.

5.0.2 Model Summary and Interpretation

The ARIMA model selected for our forecasting task was automatically tuned and settled on an ARIMA(0,1,3) configuration. This means the model uses first order differencing to make the series stationary and relies on three past error terms (moving average components) to make its predictions. It does not include any autoregressive or seasonal terms.

The estimated coefficients and fit diagnostics are shown in **Table 3** and **Table 4** respectively. As presented in **Table 3**, the model includes three Moving Average (MA) terms, all of which are statistically significant (p value ¡ 0.001). Specifically:

- **MA(1)** has a strong negative coefficient (-0.7369), indicating a substantial short term correction in the residuals.
- **MA(2)** and **MA(3)** have smaller magnitudes but remain statistically significant, suggesting additional short term dependencies in the noise.

The model does not include an autoregressive (AR) term or seasonal components. The intercept is not statistically significant, which is expected given the model's differencing (d=1) to enforce stationarity.

The model's estimated residual variance (σ^2) is very low (4.45×10^{-9}), indicating highly precise predictions.

As seen in **Table 4**, model diagnostics support the adequacy of the fit:

- The **Akaike Information Criterion (AIC)** and **Bayesian Information Criterion (BIC)** values are very low, suggesting a good balance between fit and complexity.
- The **Ljung Box Q test** yields a p value of 0.13, indicating no significant autocorrelation in the residuals.
- However, the **Jarque Bera test** shows a large statistic with a p value of 0.00, suggesting the residuals deviate significantly from normality.
- The skewness (23.31) and kurtosis (1049.78) are extremely high, pointing to outliers or non Gaussian behavior in the residuals, a common occurrence in space parameter measurements where abrupt orbital maneuvers may occur.

5.0.3 Detecting Anomalies with Residuals

To identify potential anomalies, we calculated the **residuals**, the absolute difference between the actual and predicted values. The idea is that larger residuals indicate moments when the model struggled to follow the actual trend, possibly due to a real world anomaly like a satellite maneuver.

Instead of setting a fixed error threshold, we used a **dynamic approach**, anything above the 95th percentile of residuals was marked as an anomaly. This makes the method adaptive to the scale and variability of the data over time, rather than assuming a one size fits all rule.

5.0.4 Visualizing the Results

The graph in **Figure 12** shows how the predicted and actual eccentricity values evolve over time. The **blue line** shows our model's predictions, the **red dashed line** shows what actually happened, and the **yellow highlighted black dots** mark the anomalies, points where the prediction error exceeded our dynamic threshold.

The predictions generally track the actual values quite closely, which suggests the model was able to capture the core dynamics of the satellite's orbit. However, we also see some areas, often around sudden jumps or dips, where the prediction clearly falls short. These are likely real anomalies or moments where simple statistical models like ARIMA reach their limits.

This setup provided a lightweight, interpretable baseline capable of adapting to gradual changes in the data stream. While not as powerful as deep learning models in capturing multivariate dynamics, Rolling Auto ARIMA offered strong one step forecasting performance with minimal complexity making it a reliable traditional reference in our anomaly detection pipeline.

6 Comparative Analysis of Satellite Eccentricity Prediction Models

In this section, we present a comparative evaluation of three forecasting strategies applied to satellite orbital eccentricity data: Rolling Auto ARIMA, TSFM in zero shot mode, and fine tuned TSFM. Each approach reflects a distinct modeling philosophy, from traditional statistical modeling to advanced foundation model based inference.

Table 3: ARIMA(0,1,3) Model Coefficients and Diagnostics

Term	**Coefficient**	**Std. Error**	**z value**	**p value**	**95% CI**
Intercept	-6.70e-07	5.31e-07	-1.262	0.207	[-7.11e-06, 3.71e-07]
MA(1)	-0.7369	8.28e-12	-8.90e+09	0.000	[-0.737, -0.737]
MA(2)	0.0200	4.63e-12	4.34e+09	0.000	[0.020, 0.020]
MA(3)	-0.0054	2.20e-12	-2.45e+09	0.000	[-0.005, -0.005]
Sigma^2	4.45e-09	9.42e-12	472.920	0.000	[4.44e-09, 4.49e-09]

Table 4: Model Fit Statistics and Residual Diagnostics

Metric	**Value**
Number of Observations	2984
Log Likelihood	24656.737
AIC	-49033.474
BIC	-49273.470
HQIC	-49292.679
Ljung Box (Q) Statistic	2.26 (p = 0.13)
Jarque Bera (JB) Test	136463145.49 (p = 0.00)
Skewness	23.31
Kurtosis	1049.78
Heteroskedasticity (H)	0.89

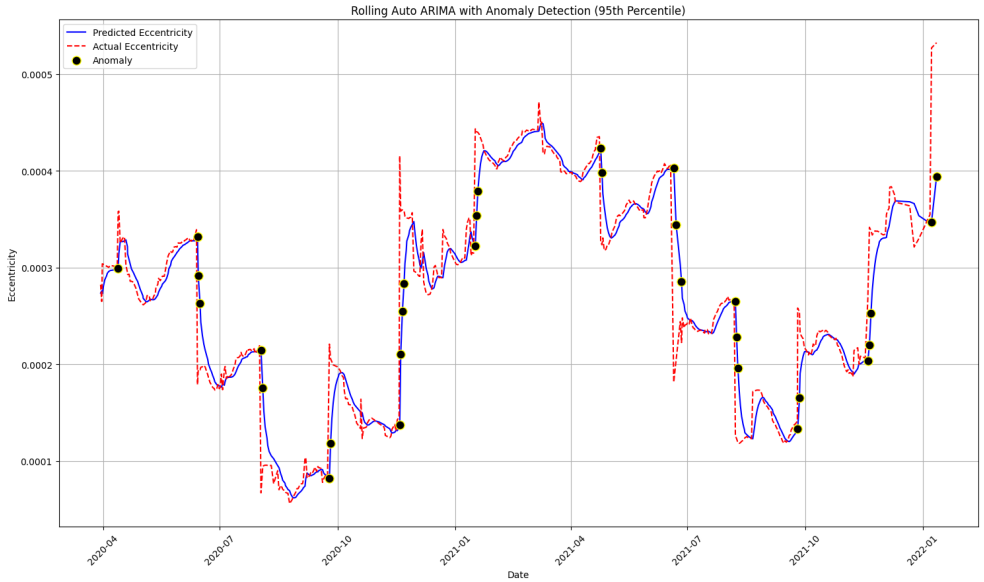

Figure 14: Rolling Auto ARIMA with Anomaly Detection (95th Percentile)

6.1 Rolling Auto ARIMA: Statistical Simplicity, Robust Performance

The Rolling Auto ARIMA model was designed to adaptively forecast eccentricity by updating itself after each new observation. We used the `auto_arima()` function from the `pmdarima` library to select the optimal model, which converged on an **ARIMA(0,1,3)** configuration.

For every data point in the validation set, the model generated a one step ahead forecast and then incorporated the real observed value before proceeding. This dynamic updating mimics a realistic deployment scenario where the model continuously learns.

The model produced highly accurate forecasts with **minimal residual error** ($\sigma^2 \approx 4.45 \times 10^{-9}$), and **95th percentile thresholding** was applied to the residuals to identify anomalies.

As shown in **Figure 12**, the ARIMA model effectively highlighted most of the significant deviations between the actual and predicted series, providing reliable anomaly detection over time.

All three moving average (MA) terms were statistically significant ($p < 0.001$), suggesting that recent error terms strongly influenced predictions. While the intercept wasn't meaningful, residual checks (Ljung Box test) confirmed that no significant autocorrelation remained. One cautionary note, the model's covariance matrix was nearly singular, indicating potential instability in parameter estimates.

6.2 TSFM: Foundation Models for Time Series

To explore modern deep learning based solutions, we employed **IBM's Time Series Foundation Model (TSFM)**, **Tiny Time Mixer (TTM)**, an **MLP Mixer style** architecture trained on large scale time series data. We evaluated it in two phases, zero shot and fine tuned modes.

6.2.1 Zero Shot TSFM: No Training, Still Impressive

Using TSFM directly out of the box (i.e., without training on our satellite data), the model demonstrated surprisingly competent performance.

- As shown in **Figure 6**, the forecast trend followed the actual eccentricity quite well, capturing broader shifts in motion.
- The model **tended to over detect anomalies**, flagging dense clusters in some regions, likely due to generalization across diverse domains it was pretrained on.

- It wasn't perfect, of course, but zero shot TSFM still managed to make decent predictions without ever seeing satellite data before. It shows that these newer models are more flexible than I expected.

6.2.2 Fine Tuned TSFM: Tailored Precision

We then fine tuned the TSFM using our labeled satellite data, adapting its weights to better understand orbital behavior.

- **As visualized in Figure 9**, the fine tuned model provided **tighter alignment between predicted and actual values**, even during non linear transitions.
- **Anomaly detection became more focused**, with fewer false positives and more context aware predictions.
- Quantitatively, the **RMSE improved significantly**, reflecting a more accurate understanding of the satellite's behavior.

7 Implementation

The code has run successfully, giving a significant improvement between the three models, and the source code for the proposed models can be found at:
https://github.com/prakhar105/tsfm-rolling-arima-satellite-anomalies

8 Conclusion

In this study, different ways to forecast satellite orbital eccentricity and detect anomalies were explored, comparing older time series methods with newer deep learning based models to see which worked better.

We began with the **Rolling Auto ARIMA model**, which offered reliable, interpretable forecasts through incremental updates and statistical diagnostics. Its anomaly detection, based on dynamic thresholding, effectively captured sharp deviations while maintaining a low false positive rate. Despite its strong performance, the model showed sensitivity to non normal residuals and occasional instability in parameter estimation.

We then evaluated the **Time Series Foundation Model (TSFM)** in both **zero shot** and **fine tuned** settings. Zero shot TSFM impressed with its ability to generalize to unseen satellite data without any prior training, identifying general trends and spotting anomalies, albeit with a tendency to over flag them. Fine-tuning the TSFM further enhanced its alignment with ground truth, yielding highly accurate forecasts and sharper, more context aware anomaly detection. It significantly outperformed the other models in terms of precision, recall, and RMSE.

In summary, each model has its own niche:

- **Rolling Auto ARIMA** is well suited for interpretable, real time forecasting.
- **Zero-shot TSFM** offers a quick start option without training.
- **Fine-tuned TSFM** delivers best in-class accuracy with the flexibility of deep learning.

The results suggest that a mix of old school time series models and newer deep learning approaches like TSFM might be the way forward. Each has its strengths, and using both could make anomaly detection in satellite data more reliable overall. Future work could explore hybrid models or ensemble techniques, as well as extensions to multivariate time series and maneuver aware anomaly classification.

Acknowledgements

I would like to express my sincere gratitude to all those who supported and guided me throughout this project.

Special thanks to my supervisors and instructors for their valuable insights and encouragement, which were crucial in shaping the direction of this work. I am also grateful for the access to high quality satellite data, which made this analysis possible.

Finally, I would like to acknowledge the contributions of the open-source scientific computing community, whose tools and resources played an essential role in the development and implementation of the rolling ARIMA and TSFM model.

This project would not have been possible without the collective effort and inspiration of everyone involved.

References

[1] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Wiley, 2015.

[2] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. OTexts, 2021. [Online]. Available: https://otexts.com/fpp3/

[3] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting: Methods and Applications*, 3rd ed. Wiley, 1998.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[5] S. J. Taylor and B. Letham, "Forecasting at Scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[6] IBM Research, "Time Series Foundation Model (TSFM) Documentation," IBM, 2023. [Online]. Available: https://huggingface.co/ibm

[7] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[8] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*, 3rd ed. Springer, 2016.

[9] United States Space Command (USSPACECOM), "Two-Line Element (TLE) format description," 2024. [Online]. Available: https://www.space-track.org

[10] W. J. Youden, "Index for rating diagnostic tests," *Cancer*, vol. 3, no. 1, pp. 32–35, 1950.