Progress_Report

March 5, 2025

1 Progress Report

1.1 Summary

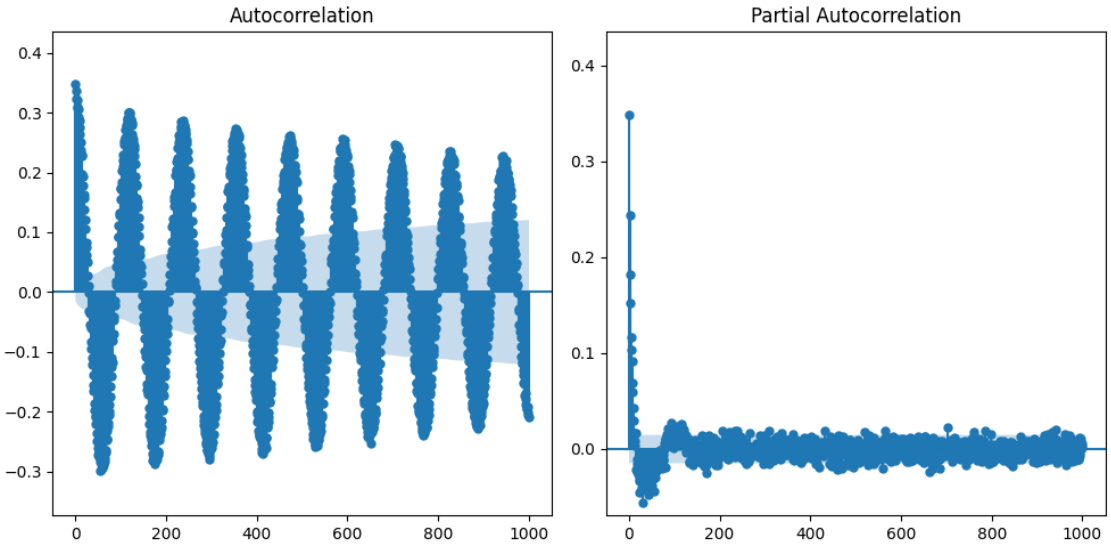
To date, I have completed EDA, attempted ARIMA imputation of missing values, applied STL decomposition, and successfully run an Isolation Forest model. The **Exploratory Data Analysis** was productive, giving insight into the general form of the time series and potential model parameter values to initiate tuning. The **ARIMA imputation** was intended to impute larger gaps in the time series, creating a more stable and representative input for anomaly detection. While functional, the current results are not optimal. However, I've decided to prioritize other tasks before revisiting this step. That being said, as a learning experience, the experiment was valuable. Details are provided below. The **STL Decomposition** applied to the imputed time series captured seasonality well, though the trend component was not perfectly linear. Most recently, the **Isolation Forest** model building was straightforward and yielded promising results. Further tuning is required. Additionally, the anomaly scores could be useful for flare mapping, which I plan to explore if time permits. As for **next steps**, I plan to focus on flare simulation to perform model evaluation. If time permits, I'll run a basic sigma-clipping model for comparison.

1.2 Exploratory Data Analysis (Updates)

After starting my ARIMA imputation, I realized a significant error made during my EDA. I'd presented and analyzed autocorrelation plots for the time series using only $s = 30$ lags for an approximately $n = 17,000$ observation dataset. Consequently, the autocorrelation plots did not meaningfully reflect patterns in the data and my analyses were flawed. Updated plots and comments are presented here.

[1]:
```
## Autocorrelation
from IPython.display import Image
Image("../1.Figures/EDA_autocorrelation.png")
```

[1]:



1.2.1 Comments

The autocorrelation function (ACF) on the left exhibits a strong periodic pattern with significant spikes at regular intervals. The oscillatory behaviour suggests the presence of a seasonal component. The partial autocorrelation function (PACF) on the right shows a strong spike at lag 1 and a few subsequent smaller lags before tapering off. This suggests that an Auto-Regressive (AR) process might be applicable. For ARIMA imputation, we'd need to explore a method that can handle seasonality, such as SARIMA or using STL to detrend/deseasonalize prior to imputation.

1.3 ARIMA Imputation

As mentioned in the summary, the intention of ARIMA imputation was to impute larger gaps in the time series in order to provide a more stable and representative input to the anomaly detection model. However, time series forecasting is difficult and unreliable at the best of times, but I wanted to use this as an opportunity to try my hand at running a forecasting model in an applied setting.

Attempts: 1. **Reindexing** - To begin, I introduced the missing indices (time values) with PDCSAP flux null values into the time series at the designated 2-minute cadence. 2. STL Decomposition, followed by ARIMA modelling on the Noise component. - This was my first idea and in hindsight makes most sense, but I got sidetracked because STL doesn't accept null values. Will consider returning to this. 3. SARIMA (Seasonal ARIMA) - I learned that SARIMA models don't work well for large periods, such as 240 which I identified during EDA. 4. **ARIMA with Fourier Terms** (Exogenous Seasonal Dummy Variables) - This is interesting and works well, but my method of application might be flawed. Since none of these models accept null values, I've been interpolating the null values with cubic splines prior to running the models and then updating the estimations with the in-sample predictions. However, I'm not convinced that this is the best way to approach this.

The bolded steps are those that I've kept. However, given the current timeline, I'm putting this aside for now in order to focus on model evaluation. If time permits, I'd like to return to this to

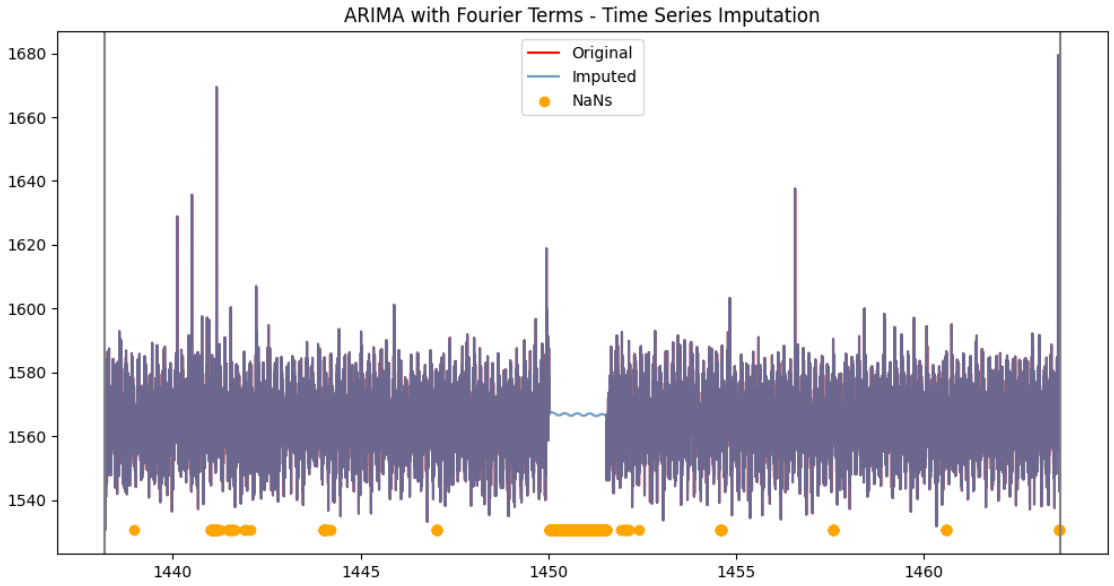
refine my approach.

1.3.1 ARIMA with Fourier Terms

As described above, the model I landed on was an ARIMA model with exogenous Fourier Terms to represent seasonal patterns. This method works well for longer periods, such as the roughly 240-lag pattern identified during EDA. I opted to use the `FourierFeaturizer` and `auto_arima` from the pyramid arima (`pmdarima`) library to create the fourier terms and ARIMA model respectively. To choose model order, `auto_arima` conducts stepwise model selection using AIC, landing on an ARIMA(12, 0 , 0) model with one of the fourier terms being statistically significant. However, since `auto_arima` doesn't handle null values, I used `pandas interpolate` with `method='pchip'`, a cubic spline method, to interpolate the null values prior to running the ARIMA model.

[2]:
```
## ARIMA with Fourier Terms
Image("../1.Figures/arima_fourier_imputation.png")
```

[2]:

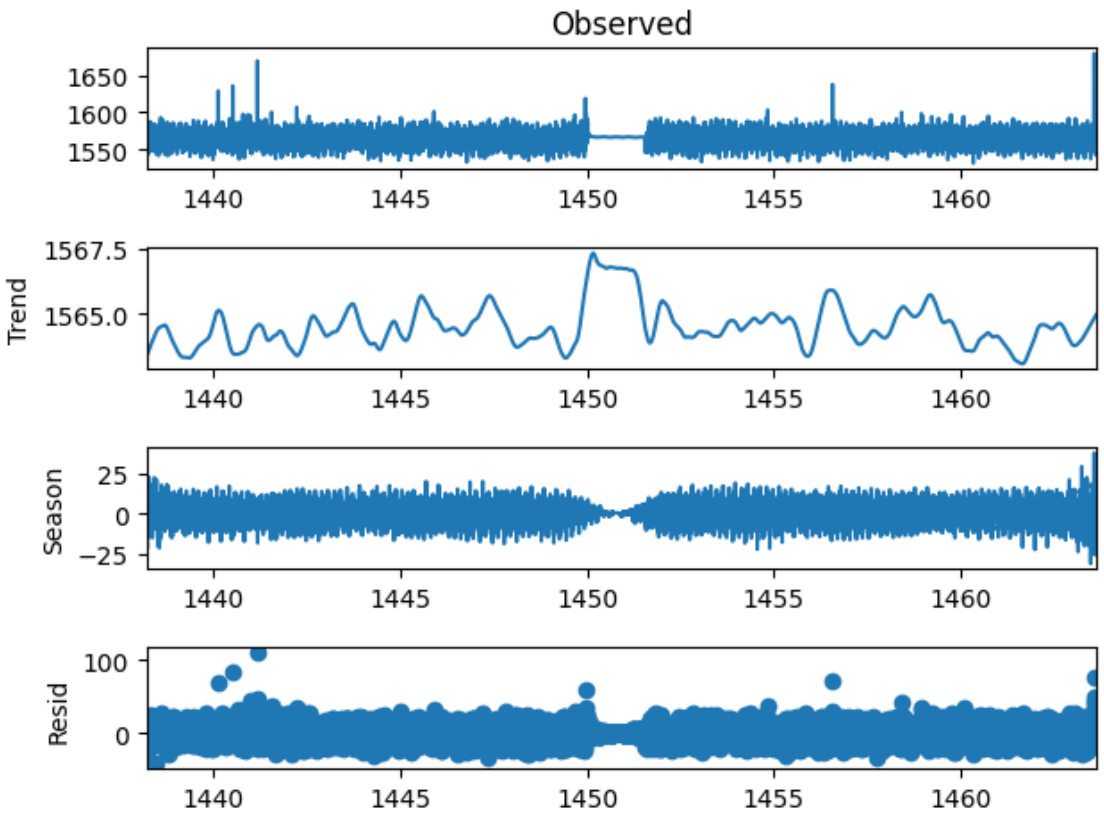


1.4 STL Decomposition

Seasonal-Trend decomposition with LOESS (STL) is a method to decompose a time series into Seasonal (S) + Trend (T) + Irregular/Noise (I) components. In particular, this means it can be used for detrending by removing the seasonal and trend components, isolating the noise. I ran the `STL` function from the `statsmodels` library with a period of `m=240` based on my EDA.

[3]:
```
## Imputed STL
Image("../1.Figures/PR_imputed_STL.png")
```

[3]:



1.5 Isolation Forest

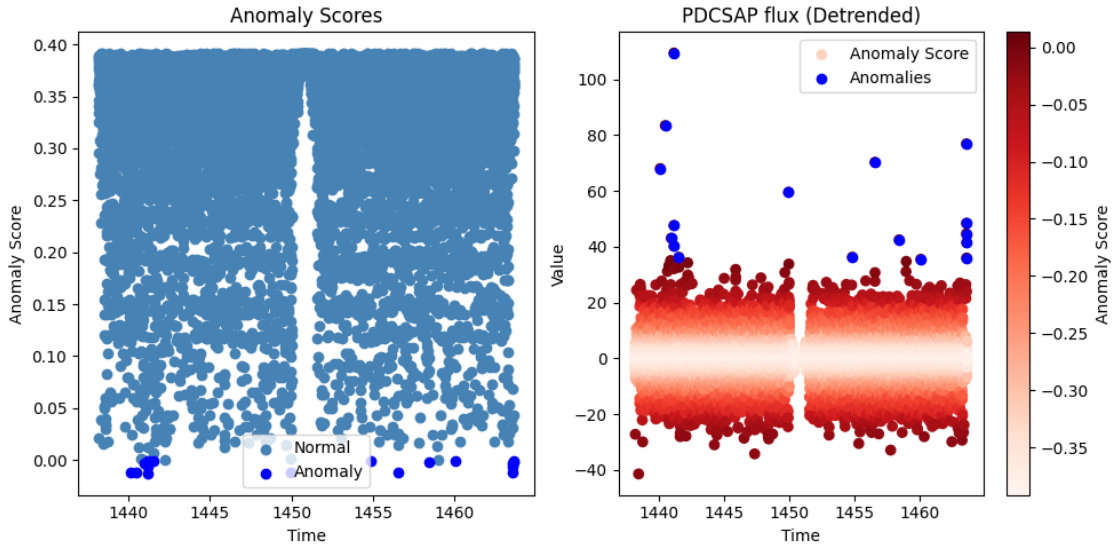
Isolation Forests refer to a tree-based unsupervised ML algorithm used for anomaly detection. They work by randomly partitioning data points via multiple decision trees and isolating anomalies, which require fewer splits to be seperated compared to normal data points. Of interest, the model assigns an **anomaly score** to each point, based on the average length requried to isolate it. Time permitting, I'd like to explore using anomaly score percentiles to map data points in close proximity to identified anomalies as part of a flare.

Hyperparameters: - `n_estimators`: Number of trees in the forest - using standard 100. - `max_samples`: Number of samples drawn to train each tree - using standard 256. - `contamination`: Expected proportion of anomalies in the data - requires **domain knowledge**, plan to tune this during model evaluation. - `max_features`: Number of features used for splitting at each node - NA for time series.

Training an initial IF model with the above hyperparameter standard values and `contamination = 0.001` produced the following results.

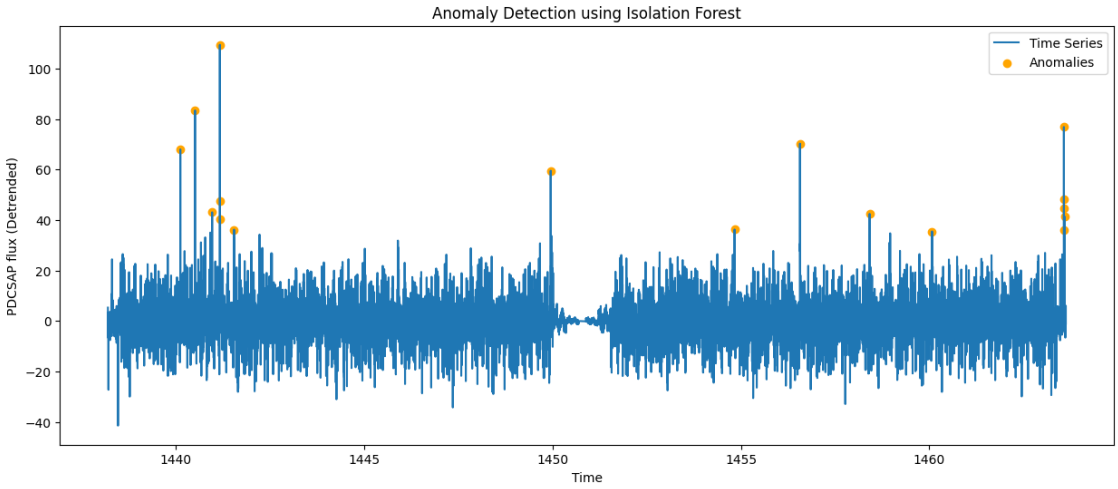
[8]:
```
## Anomaly Scores
Image("../1.Figures/PR_anomaly_scores.png")
```

[8]:



[7]:
```
## Anomaly Detection
Image("../1.Figures/PR_IF_detection.png")
```

[7]:



1.6 Next Steps

Following all of the above, I'm planning to focus my efforts on flare simulation and subsequently model evaluation. These steps are crucial and will help me tune the `contamination` parameter of the Isolation Forest model. Afterwards, I'll prioritize putting together my report and presentation before continuing new explorations.

Intended steps: 1. **Flare simulation** 2. **Model evaluation** - Tuning Isolation Forest - (Time permitting) Anomaly Model Comparison 3. **Report (and presentation)** 4. Revisit ARIMA

imputation 5. Exploration: Flare mapping with anomaly scores