Analysis of AGN Variability Using Gaussian Process Regression in TESS Data

Kirsten Cowling
Department of Physics, Texas Tech University

Spring 2025

Abstract

This paper applies Gaussian Process Regression (GPR) to investigate the variability properties of Type 1.5–2 Active Galactic Nuclei (AGNs). Using light curve data from the Transiting Exoplanet Survey Satellite (TESS) three AGNs were analyzed. NGC 1566 (Type 1.5), ESO 253-003 (Type 2), and PGC 2816183 (Type 2) were chosen either due to their variable or noisy light curve. To quantify the results, metrics like Root Mean Square (RMS), Signal-to-Noise Ratio (SNR), fractional variability, and characteristic timescale (τ) were applied. NGC 1566 exhibited rapid variability with a characteristic timescale of 4.28 days, an amplitude of 42096.75 counts, and an RMS of 46690.82 counts. ESO 253-003 demonstrated more gradual variability with a 37.59-day timescale and an RMS of 24833.76 counts. PGC 2816183, despite having the highest fractional variability of 11.6024, was dominated by noise, resulting in a negative SNR of -0.1588. The GPR modeled the light curve behavior with R^2 values ranging from 0.3370 to 0.9914. However, high reduced χ^2 values between 2.4714–4.0403 suggest the presence of variability not captured by the model or overestimated uncertainties.

1 Introduction

Active Galactic Nuclei (AGNs) are among the most energetic and luminous phenomena in the universe. They are powered by the accretion of matter into supermassive black holes at the centers of galaxies. AGNs are classified into two main types based on their orientation relative to Earth. They are also classified by presence or absence of a dusty torus, which is a dense cloud of dust and gas surrounding the black hole's accretion disk.

Type 1 AGNs are viewed face-on with no obscuration from the dusty torus. This provides a relatively clear view of the central engine and broad-line region. This orientation allows astronomers to directly observe the high-velocity gas clouds, which produce both broad and narrow emission lines in their spectra.

In contrast, Type 2 AGNs are viewed edge-on, where the central engine and broad-line region are obscured by a dense torus of gas and dust. As a result, only narrow emission lines are observed. It is assumed that both types of AGNs have the same properties but due to the orientation, inner processes remain a mystery for the vast majority of Type 2s. Intermediate types, such as Type 1.5, exhibit characteristics of both, indicating partial obscuration or variability when observed.

The Transiting Exoplanet Survey Satellite (TESS) was launched with the prime mission to discover exoplanets. However, full-sky coverage also makes it suited for studying variable sources such as AGNs. The light curves from TESS offer extensive data about short to intermediate timescale variability. This allows the opportunity to study the dynamic and physical processes in the accretion disk of supermassive black holes.

This project aims to analyze the variability patterns of Type 1.5–2 AGNs using Gaussian Process Regression (GPR) applied to light curves NGC 1566, ESO 253-003, and PGC 2816183. GPR is chosen for its ability to model complex time series data and to account for noise that is typical for Type 2 AGN. The primary focus is to evaluate the variability using the statistical metrics Root Mean Square (RMS), Signal-to-Noise Ratio (SNR), characteristic timescale (τ), and fractional variability.

2 Methods and Analysis

2.1 Gaussian Process Regression

Gaussian Process Regression is a non-parametric method that models the light curve as a collection of random variables connected by a joint Gaussian distribution. The form of the GPR model can be expressed as:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{1}$$

where $m(x)$ is the mean function. In this case is zero due to the normalization of the data, and $k(x, x')$ is the covariance function or kernel.

I used a composite kernel consisting of a Matern kernel with $\nu = 0.5$, which is a damped random walk. For AGN light curves the damped random walk is ideal for determining its stochastic behavior. A white noise kernel was also used to account for observational uncertainties. The kernels are related below:

$$k(x,x') = \sigma^2 \exp\left(-\frac{|x-x'|}{\tau}\right) + \sigma_n^2 \delta(x,x') \tag{2}$$

Where amplitude (σ) represents the variability amplitude of the signal, characteristic timescale (τ) measures how rapidly the light curve fluctuates, and noise level (σ_n) accounts for observational uncertainties and measurement noise. The GP model is trained using maximum likelihood estimation, optimizing these kernel parameters to fit the data.

2.2 Fractional Variability

Fractional variability (F_{var}) is a metric for measuring the variability relative to the mean flux of a light curve. It is defined as:

$$F_{var} = \frac{\sqrt{S^2 - \langle \sigma_{err}^2 \rangle}}{\langle y \rangle} \tag{3}$$

Where S^2 is the variance of the sample light curve, $\langle \sigma_{err}^2 \rangle$ is the mean of squared error of the light curve data points and $\langle y \rangle$ is the mean flux.

In this assessment, fractional variability is calculated for each AGN to determine how significant the observed variations are relative to the noise level. This metric is useful for distinguishing true variability from noise, especially when datasets with low signal-to-noise ratios are observed.

2.3 Root Mean Square and Signal-to-Noise Ratio

Root Mean Square and Signal-to-Noise Ratio are two metrics for assessing the variability and reliability of a dataset.

2.3.1 Root Mean Square (RMS)

RMS quantifies the overall variability of a light curve by measuring the dispersion of flux values around the mean as shown below.

$$\text{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \langle y \rangle)^2} \tag{4}$$

Where y_i is the flux at time i, $\langle y \rangle$ is the mean flux, and N is the number of data points.

Higher RMS values indicate greater variability in the light curve, making it useful for identifying more spasmodic AGNs.

2.3.2 Signal-to-Noise Ratio (SNR)

SNR provides a measure of the significance of the detected variability relative to the measurement noise. It is calculated as:

$$\text{SNR} = \frac{\langle y \rangle}{\langle \sigma_{err} \rangle} \tag{5}$$

Where $\langle y \rangle$ is the mean flux and $\langle \sigma_{err} \rangle$ is the mean uncertainty in the flux measurements.

SNR can distinguish between true variability and noise, especially in datasets with low flux amplitudes or high observational uncertainties.

2.4 R-Squared and Chi-Squared

R-squared (R^2) and Chi-squared (χ^2) are metrics used to evaluate the quality of the Gaussian Process Regression model fit to the data.

2.4.1 R-Squared (R^2)

The R^2 value quantifies the ratio of variance in the observed data that is explained by the GPR model. It is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \langle y \rangle)^2} \tag{6}$$

Where y_i is the observed flux value at time i, $\hat{y}_i$ is the predicted flux value at time i, $\langle y \rangle$ is the mean flux value, and N is the number of data points.

An R^2 value close to 1 indicates a strong fit between the model and the observed data, whereas values near 0 suggest a poor fit.

2.4.2 Chi-Squared (χ^2)

Chi-squared evaluates how well the model's predictions match the observed data, accounting for uncertainties. It is calculated as:

$$\chi^2 = \sum_{i=1}^{N} \frac{(y_i - \hat{y}_i)^2}{\sigma_{err,i}^2} \tag{7}$$

Where y_i is the observed flux value at time i, $\hat{y}_i$ is the predicted flux value at time i, and $\sigma_{err,i}$ is the uncertainty in the observed flux value at time i.

2.4.3 Reduced Chi-Squared

The reduced chi-squared (χ_ν^2) accounts for the degrees of freedom in the model and is given by:

$$\chi_\nu^2 = \frac{\chi^2}{N-p} \tag{8}$$

Where N is the number of data points, and p is number of model parameters. This is typically two for GPR: amplitude and characteristic timescale.

A reduced chi-squared value close to 1 indicates a good fit, while values significantly greater than 1 suggest a poor fit or the uncertainties are underestimated.

3 Results

Gaussian Process Regression was applied to three AGNs: NGC 1566 (Type 1.5), ESO 253-003 (Type 2), and PGC 2816183 (Type 2). The light curves, GP fits, and associated statistical metrics are presented below.

3.1 NGC 1566 (Type 1.5)

NGC 1566 is a well-documented Type 1.5 AGN known for its pronounced variability across multiple wavelengths [1]. In this analysis, the Gaussian Process Regression model reveals a characteristic timescale of 4.28 days. The short timescale suggests rapid, stochastic fluctuations in the accretion disk or the broad-line region. The amplitude of the variability is large, reaching 42096.75 counts, indicative of significant flux deviations from the mean of the data. This degree of variability is consistent with previously observed patterns in Type 1 AGNs, where the direct line-of-sight to the central engine allows the detection of rapid, high-amplitude variations.

Despite the strong model fit, as indicated by the R^2 value of 0.9826, the reduced χ^2 of 2.4714 suggests that the GP model does not fully account for all sources of variability. Additionally, the elevated χ^2 value may result from overestimated uncertainties or the presence of data outliers contributing disproportionately to the overall variance. Figure 1 shows the GPR of this light curve. Looking at the x-axis between 1460-1470 BTJD, it was observed that there could be a data bleed that survived the sigma clipping when creating this curve. Another explanation involves the complex spectral nature of NGC 1566. As a known changing-look AGN, NGC 1566 has exhibited documented spectral transitions, particularly during its 2017–2018 outburst [1]. The data used in this analysis is from TESS's first year, which coincides with the period when these transitions occurred. The fractional variability (F_{var}) of 6.9498 is quite high, indicating that the flux variations are significant relative to the mean flux.

The GP model's predictions, shown as the shaded region in Figure 1, encompass the majority of observed data points. However, a few points lie outside the 95% confidence interval, indicating variability not well captured by the current GPR damped random walk kernel.

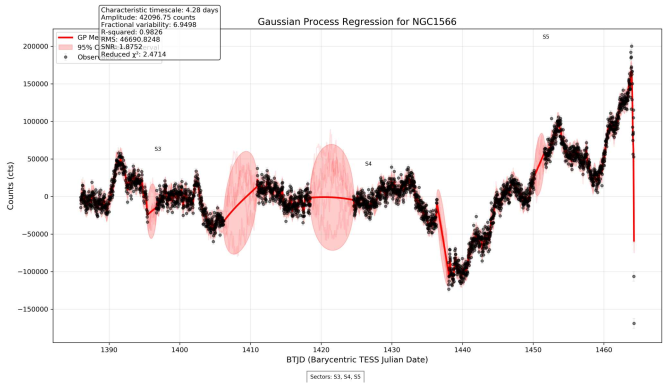

Figure 1: Gaussian Process Regression for NGC1566 showing GP mean, 95% confidence interval, and observed data points.

3.2 ESO 253-003 (Type 2)

ESO 253-003 is a Type 2 AGN which is typically characterized by a high degree of obscuration, limiting direct observation of the central engine by definition. Despite this, the light curve data reveals a prominent flux peak, indicative of significant optical variability. This could be due to a gap in the dusty torus giving a rare opportunity to view the broad line emission region. The Gaussian Process Regression model applied to this source yields a characteristic timescale of 37.59 days, which is longer than that observed in NGC 1566. This suggests slower variations and potentially different behaviors of its inner processes when compared to well-studied AGNs.

The amplitude of variability, measured at 16891.04 counts, combined with an RMS of 24833.76 counts, suggests that the flux deviations from the mean are statistically significant. These high values are particularly notable given the classification of ESO 253-003 as a Type 2 AGN, where variability is often more difficult to detect. The elevated RMS confirms the conclusion that this object exhibits intrinsic variability rather than being dominated by measurement noise.

The GP model produced a strong fit to the overall light curve structure, with the majority of observations lying within the 95% confidence interval. Despite the elevated χ^2, the GPR method recovers the characteristic timescale and amplitude of the variability.

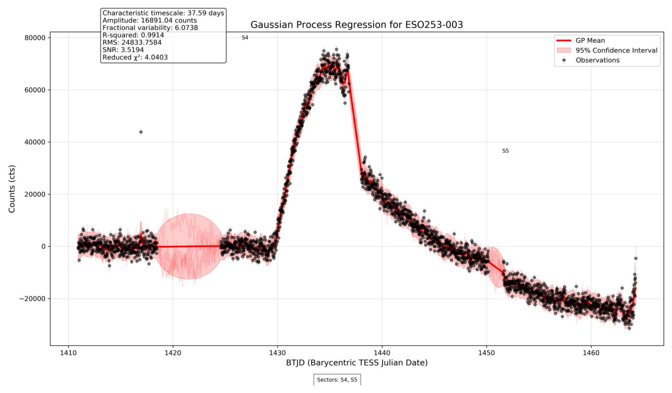

Figure 2: Gaussian Process Regression for ESO 253-003 showing GP mean (red), 95% confidence interval (shaded), and observed data points (black).

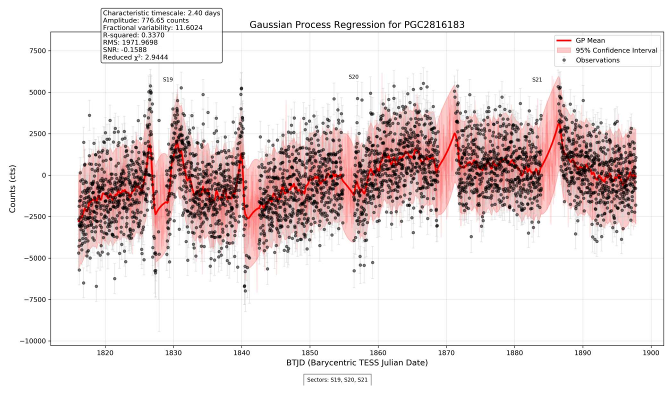

Figure 3: Gaussian Process Regression for PGC2816183 showing GP mean (red), 95% confidence interval (shaded), and observed data points (black).

3.3 PGC 2816183 (Type 2)

PGC 2816183 exhibits relatively low-amplitude variability compared to the other AGNs in this study, with a measured amplitude of 776.65 counts and a characteristic timescale of 2.40 days. Despite the modest amplitude, the calculated fractional variability of 11.6024 is the highest among the three AGNs analyzed. This implies that the variations, though small, are significant relative to the mean flux of the source.

However, the reliability of this variability is questioned by the negative signal-to-noise ratio of -0.1588. A negative SNR arises when the mean flux is near zero or when the noise level is comparable to, or greater than, the measured signal. In this case, it strongly suggests that the measured variability is due to noise rather than genuine AGN behaviors. This interpretation is further supported by the low R^2 value of 0.3370.

The reduced χ^2 value of 2.9444 does not necessarily point to a poor model fit in the usual sense. Instead, it may reflect the difficulty of fitting a smooth model to data that is predominately noise. The GPR model attempts to map over this noise, resulting in a fit that captures general trends but fails to determine the behavior of this particular AGN. Given PGC 2816183's classification as a Type 2, the rather featureless and noisy light curve is not unexpected. The obscured accretion disk limits the variability that can be detected in optical wavelengths.

Additionally, the GPR model's predictive intervals incorporates a broad region of uncertainty. Many of the observed data points fall within this band, inferring that the signal is consistent with a noise-dominated dataset. This case is an example of the limitations of light curve-based variability in faint or obscured Type 2 AGN.

3.4 Summary of Results

The three AGNs analyzed in this study exhibit distinct variability patterns, as summarized in Tables 1 and 2. NGC 1566 shows the most rapid and high-amplitude variability, while ESO 253-003 presents more gradual changes. PGC 2816183, despite exhibiting the highest fractional variability, is predominantly noise-dominated, as reflected by its negative SNR.

Table 1: Variability Timescale and Amplitude for Selected AGNs

Object	Type	τ (days)	Amplitude (cts)
NGC 1566	1.5	4.28	42096.75
ESO 253-003	2	37.59	16891.04
PGC 2816183	2	2.40	776.65

Table 2: RMS, SNR, and R^2 for Selected AGNs

Object	RMS (cts)	SNR	R^2
NGC 1566	46690.82	1.8752	0.9826
ESO 253-003	24833.76	3.5194	0.9914
PGC 2816183	1971.97	-0.1588	0.3370

4 Future Work and Study

This study focused on three light curves taken from the first year of TESS data. Expanding the analysis to include approximately 1300 Type 2 AGNs in the data set could provide a more statistically robust assessment of their variability. Applying the Gaussian Process Regression model to the complete dataset is necessary to gather clues about the true characteristics of Type 2 Active Galactic Nuclei.

Comparative analysis plots, such as RMS versus SNR and characteristic timescale versus fractional variability, could further elucidate trends and outliers that may not be obvious by focusing on a small set of individual light curves. Analyzing variability across multiple years could also provide insights into behavior or long-term trends. Moreover, correlating variability metrics with AGN properties such as black hole mass, accretion rate, infrared flux, redshift and other attributes could provide a deeper understanding of the mechanisms driving this variability.

5 Discussion and Conclusion

This study applied Gaussian Process Regression with a damped random walk kernel to analyze the variability patterns of three Active Galactic Nuclei: NGC 1566 (Type 1.5), ESO 253-003 (Type 2), and PGC 2816183 (Type 2). The data was acquired by using light curves from the TESS mission. The analysis aimed to quantify variability through metrics, including Root Mean Square, Sound-to-Noise Ratio, fractional variability, and characteristic timescale.

The results show distinct behaviors across the three AGNs. NGC 1566 exhibited the highest amplitude variability and a short characteristic timescale. This indicates rapid and pronounced flux changes. In contrast, ESO 253-003 displayed a longer characteristic timescale and a moderate amplitude, which suggests a more gradual variability. PGC 2816183, despite showing the highest fractional variability relative to its mean flux, had the lowest amplitude and a negative SNR. This indicates that the noise dominated the signal, which cannot provide reliable detection of trends that might be present in the light curve.

Future work should focus on extending the analysis to a larger dataset, such as, the full first year of TESS. This will allow for more comprehensive statistical assessments. Incorporating a broader range of AGNs will enable a more robust exploration of variability across Type 2 Active Galactic Nuclei.

References

[1] P. Trakhtenbrot, G. Dewangan "AstroSat view of spectral transition in the changing-look active galaxy NGC 1566 during the declining phase of the 2018 outburst," *arXiv:2111.08271*, 2021.

Appendix: Gaussian Process Regression with a Damped Random Walk and Other Statistics Analysis Code

A portion of the code utilized in this project was originally written for another analysis. The section specifically related to Gaussian Process Regression, starting from the fit gp model function to the main function, was written specifically for this project. The previous sections of the code were adapted from prior work but have been modified as necessary to integrate with this project.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import Matern, WhiteKernel, ConstantKernel
import logging
from sklearn.metrics import r2_score

logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

class LightCurveGPR:
    def __init__(self, root_directory):
        self.root_directory = root_directory

    def load_data_from_sector(self, obj_name, sector, cam, ccd):
        directory = f'{self.root_directory}/sector{sector}/cam{cam}_ccd{ccd}/lc_hyperleda'
        file_path = os.path.join(directory, f"lc_{obj_name}_cleaned")
        try:
            if os.path.exists(file_path):
                data = pd.read_csv(file_path, sep=r'\s+')
                data.columns = data.columns.str.strip()
                data = data.replace('-', pd.NA).astype(float)
                data['sector'] = int(sector)
                return data
            else:
                logging.debug(f"File not found: {file_path}")
                return None
        except Exception as e:
            logging.error(f"An error occurred while reading {file_path}: {e}")
            return None

    def sigma_clip_data(self, data, sigma=3, maxiters=5):
        try:
            if data is None or data.empty:
                return None
            data_values = data['cts'].values
            mask = np.ones(len(data_values), dtype=bool)
            for _ in range(maxiters):
                mean = np.mean(data_values[mask])
                std = np.std(data_values[mask])
                new_mask = np.abs(data_values - mean) < sigma * std
                if np.all(new_mask == mask):
                    break
                mask = new_mask
            clipped_data = data.iloc[mask]
            return clipped_data
        except Exception as e:
            logging.error(f"An error occurred during sigma clipping: {e}")
            return None

    def combine_sector_data(self, obj_name):
        combined_data = []
        sectors = ['02', '03', '04', '05', '07', '19', '20', '21']
        for sector in sectors:
            for cam in range(1, 5):
                for ccd in range(1, 5):
                    data = self.load_data_from_sector(obj_name, sector, cam, ccd)
                    if data is not None and not data.empty:
                        clipped_data = self.sigma_clip_data(data)
                        if clipped_data is not None and not clipped_data.empty:
                            combined_data.append(clipped_data)

        if not combined_data:
            logging.warning(f"No data found for {obj_name} in any sector")
            return None
        combined_df = pd.concat(combined_data, ignore_index=True)
        combined_df = combined_df.sort_values('BTJD')
        return combined_df

    def fit_gp_model(self, data, n_restarts=10):
        X = data['BTJD'].values.reshape(-1, 1)
        y = data['cts'].values
        y_err = data['e_cts'].values

        self.X_mean = X.mean()
        self.X_std = X.std()
        self.y_mean = y.mean()
        self.y_std = y.std()
        X_norm = (X - self.X_mean) / self.X_std
        y_norm = (y - self.y_mean) / self.y_std
        y_err_norm = y_err / self.y_std

        amplitude = ConstantKernel(constant_value=1.0, constant_value_bounds=(0.1, 10.0))
        length_scale = 10.0
        drw_kernel = amplitude * Matern(length_scale=length_scale, length_scale_bounds=(0.1, 100.0), nu=0.5
        noise_kernel = WhiteKernel(noise_level=np.mean(y_err_norm**2),
                                   noise_level_bounds=(np.min(y_err_norm**2), np.max(y_err_norm**2)*10))
        kernel = drw_kernel + noise_kernel
        gp = GaussianProcessRegressor(kernel=kernel, alpha=y_err_norm**2,
                                      n_restarts_optimizer=n_restarts, normalize_y=False)
        gp.fit(X_norm, y_norm)
        logging.info(f"Optimized kernel: {gp.kernel_}")
        return gp, X_norm, y_norm, y_err_norm

    def extract_kernel_parameters(self, gp_model):
        kernel_params = gp_model.kernel_.get_params()

        amplitude = None
    length_scale = None
    noise_level = None

    for param_name, param_value in kernel_params.items():
        if 'constant_value' in param_name and not 'bounds' in param_name:
            amplitude = param_value
        elif 'length_scale' in param_name and not 'bounds' in param_name:
            length_scale = param_value
        elif 'noise_level' in param_name and not 'bounds' in param_name:
            noise_level = param_value

        if isinstance(length_scale, (list, tuple, np.ndarray)):
            tau_drw_days = length_scale[0] * self.X_std

        else:
            tau_drw_days = length_scale * self.X_std

    else:
        tau_drw_days = None


        if isinstance(amplitude, (list, tuple, np.ndarray)):
            amplitude_counts = amplitude[0] * self.y_std

        else:
            amplitude_counts = amplitude * self.y_std

    return {
        'amplitude': amplitude_counts,
        'tau_drw_days': tau_drw_days,
        'noise_level': noise_level
    }

def calculate_statistics(self, data, gp_model, X_norm, y_norm):
    X = X_norm * self.X_std + self.X_mean
    y = y_norm * self.y_std + self.y_mean
    time_span = X.max() - X.min()
    mean_flux = np.mean(y)
    std_flux = np.std(y)
    min_flux = np.min(y)
    max_flux = np.max(y)
    amplitude = max_flux - min_flux
    rms = np.sqrt(np.mean(np.square(y - mean_flux)))
    mean_err = np.mean(data['e_cts'].values**2)
    snr = mean_flux / mean_err if mean_err > 0 else np.nan
    variance = np.var(y)
    mean_err_squared = np.mean(data['e_cts'].values**2)
    excess_variance = variance - mean_err_squared

    if mean_flux != 0 and excess_variance > 0:
        frac_var = np.sqrt(excess_variance) / np.abs(mean_flux)
        frac_var_err = np.sqrt(
            (1/(2*len(y)*frac_var)) *
            (mean_err_squared/mean_flux**2) +
                (2*mean_err_squared**2/(len(y)*mean_flux**4))
            )
        else:
            frac_var = np.nan
            frac_var_err = np.nan
        y_pred_norm = gp_model.predict(X_norm)
        y_pred = y_pred_norm * self.y_std + self.y_mean

        r2 = r2_score(y, y_pred)
        log_likelihood = gp_model.log_marginal_likelihood_value_

        residuals = y - y_pred
        errors = data['e_cts'].values
        chi2 = np.sum((residuals / errors) **2)
        reduced_chi2 = chi2 / (len(y) - 2)
        kernel_params = self.extract_kernel_parameters(gp_model)
        stats = {
            'time_span': time_span,
            'mean_flux': mean_flux,
            'std_flux': std_flux,
            'min_flux': min_flux,
            'max_flux': max_flux,
            'amplitude': amplitude,
            'rms': rms,
            'snr': snr,
            'variance': variance,
            'excess_variance': excess_variance,
            'frac_var': frac_var,
            'frac_var_err': frac_var_err,
            'r2': r2,
            'log_likelihood': log_likelihood,
            'chi2': chi2,
            'reduced_chi2': reduced_chi2,
            'kernel_params': kernel_params
        }
        return stats

    def predict_gp(self, gp, X_norm, X_pred=None, n_samples=20):
        if X_pred is None:
            X_min, X_max = X_norm.min(), X_norm.max()
            X_pred_norm = np.linspace(X_min, X_max, 1000).reshape(-1, 1)
        else:
            X_pred_norm = (X_pred - self.X_mean) / self.X_std
        y_pred_norm, sigma_norm = gp.predict(X_pred_norm, return_std=True)

        X_pred = X_pred_norm * self.X_std + self.X_mean
        y_pred = y_pred_norm * self.y_std + self.y_mean
        sigma = sigma_norm * self.y_std

        samples_norm = gp.sample_y(X_pred_norm, n_samples=n_samples)
        samples = samples_norm * self.y_std + self.y_mean

        return X_pred, y_pred, sigma, samples
```

```
def plot_gp_results(self, data, X_pred, y_pred, sigma, samples, obj_name, stats, save_dir=None):
    plt.figure(figsize=(14, 8))

    plt.errorbar(data['BTJD'], data['cts'], yerr=data['e_cts'],
                fmt='o', color='black', ecolor='lightgray',
                elinewidth=1, capsize=2, markersize=4, alpha=0.5, label='Observations')
    plt.plot(X_pred, y_pred, 'r-', linewidth=2.5, label='GP Mean')
    plt.fill_between(X_pred.flatten(),
                    y_pred - 2*sigma,
                    y_pred + 2*sigma,
                    color='red', alpha=0.2, label='95% Confidence Interval')
    for i in range(min(5, samples.shape[1])):
        plt.plot(X_pred, samples[:, i], 'r-', alpha=0.1)
    sectors = data['sector'].unique()
    sectors.sort()
    for sector in sectors:
        sector_data = data[data['sector'] == sector]
        mid_point = len(sector_data) // 2
        if not sector_data.empty and mid_point < len(sector_data):
            x_pos = sector_data.iloc[mid_point]['BTJD']
            y_max = sector_data['cts'].max()
            plt.text(x_pos, y_max * 1.05, f"S{sector}",
                    horizontalalignment='center', fontsize=8)

    plt.title(f"Gaussian Process Regression for {obj_name}", fontsize=14)
    plt.xlabel('BTJD (Barycentric TESS Julian Date)', fontsize=12)
    plt.ylabel('Counts (cts)', fontsize=12)
    plt.legend(fontsize=10)
    plt.grid(True, alpha=0.3)
    stats_text = (
        f"Characteristic timescale: {stats['GP Model Statistics']['Characteristic timescale (days)']:.2
        f"Amplitude: {stats['GP Model Statistics']['Amplitude (counts)']:.2f} counts\n"
        f"Fractional variability: {stats['Data Statistics']['Fractional variability']:.4f}\n"
        f"R-squared: {stats['GP Model Statistics']['R-squared']:.4f}\n"
        f"RMS: {stats['Data Statistics']['RMS']:.4f}\n"
        f"SNR: {stats['Data Statistics']['SNR']:.4f}\n"
        f"Reduced ²: {stats['GP Model Statistics']['Reduced Chi-squared']:.4f}"
    )
    plt.figtext(0.15, 0.85, stats_text,
               bbox=dict(facecolor='white', alpha=0.8, boxstyle='round'),
               fontsize=10)
    plt.figtext(0.5, 0.01, f"Sectors: {', '.join([f'S{s}' for s in sectors])}",
              ha="center", fontsize=8, bbox={"facecolor":"white", "alpha":0.5, "pad":5})
    plt.tight_layout(rect=[0, 0.03, 1, 0.97])

    if save_dir:
        os.makedirs(save_dir, exist_ok=True)
        save_path = os.path.join(save_dir, f"{obj_name}_gp_regression.png")
        plt.savefig(save_path, dpi=300)
        logging.info(f"GP regression plot saved to {save_path}")

    plt.show()

def run_full_analysis(self, obj_name, save_dir=None):
        logging.info(f"Starting analysis for {obj_name}")
        data = self.combine_sector_data(obj_name)

        if data is None or data.empty:
            logging.error(f"No data found for {obj_name}")
            return None
        gp_model, X_norm, y_norm, y_err_norm = self.fit_gp_model(data)
        stats = self.calculate_statistics(data, gp_model, X_norm, y_norm)
        X_pred, y_pred, sigma, samples = self.predict_gp(gp_model, X_norm)
        self.plot_gp_results(data, X_pred, y_pred, sigma, samples, obj_name, stats, save_dir)

        return {
            'data': data,
            'gp_model': gp_model,
            'X_norm': X_norm,
            'y_norm': y_norm,
            'statistics': stats,
            'predictions': {
                'X_pred': X_pred,
                'y_pred': y_pred,
                'sigma': sigma,
                'samples': samples
            }
        }

def read_object_list(file_path):
    try:
        with open(file_path, 'r') as f:
            return object
    except Exception as e:
        logging.error(f"Error reading object list from {file_path}: {e}")
        return []

def main():
    root_directory = '/home/kicowlin/SummerResearch2024'
    save_directory = f'{root_directory}/gp_regression'
    obj_list_file = f'{root_directory}/object_list.txt'
    gpr = LightCurveGPR(root_directory)
    obj_list = read_object_list(obj_list_file)

    for obj_name in obj_list:
        logging.info(f"Processing {obj_name}")
        gpr.run_full_analysis(obj_name, save_dir=save_directory)


if __name__ == "__main__":
    main()
```