Assignment 8: Gaussian Process for Electrical System Demand Load Forecasting

Alex Hostick

Department of Electrical and Computer Engineering, University of New Mexico

ECE 517: Machine Learning

Dr. Manel Martínez-Ramón

Abstract

Statistical predictions are needed for regional electrical demand characteristics to assist municipalities in keeping reliable outlooks on energy demand. The non-profit organization “ISO New England” keeps energy records for Connecticut, Rhode Island, Massachusetts, Vermont, New Hampshire, and Maine. ISO New England collects data to overview grid operation, market administration, and power system planning. For this assignment, the Gaussian Process is exemplified to understand how the prediction algorithm is used and to predict energy demand load requirements based on data collected statistically. A GP method will test the data as samples from a multivariate Gaussian distribution method for prediction. The model will take 24 hours and predict 48 to 168 hours using a horizon outlook of 1, 6, and 12 hours. The energy load operations examples will be from data recorded from 2020 to 2021 in Connecticut. Other state-predicted load data is available upon request.

GP System Energy Demand Load Forecasting

This assignment will utilize raw data from the United States Northeast region to statistically predict energy load demand based on empirical data. Data for this study is retrievable from "ISO New England," a non-profit organization that assists with planning transmission systems, administers the region's wholesale markets, and operates the power system for wholesale electricity for 6 Northeastern American states. Energy load data for this prediction had been collected from the years 2020 to 2021 by ISO New England. The example I intend to show encapsulates the 2020 data from the state of Connecticut, uses it for training, and predicts specific 2021 energy load forecasts in January and July.

Discussion

Problem Statement

The test will use data from January to December 2020 to train a Gaussian Process to forecast energy demand load for specific time series 2021. The GP will provide a window view of 24-hour sample sets (W) for the hours of prediction horizon (M). The GP will return a 95% confidence interval and mean prediction when analyzing the observed training data, and we can check the performance with observed 2021 data. The GP will use a linear and square exponential (radial basis function 'RBF') kernel with 1000 training samples. The test will use a horizon of 1 hour, 6 hours, and 12 hours for testing in the first week of January and July of 2021 and plot the mean and confidence intervals based on the GP model's 2020 training.

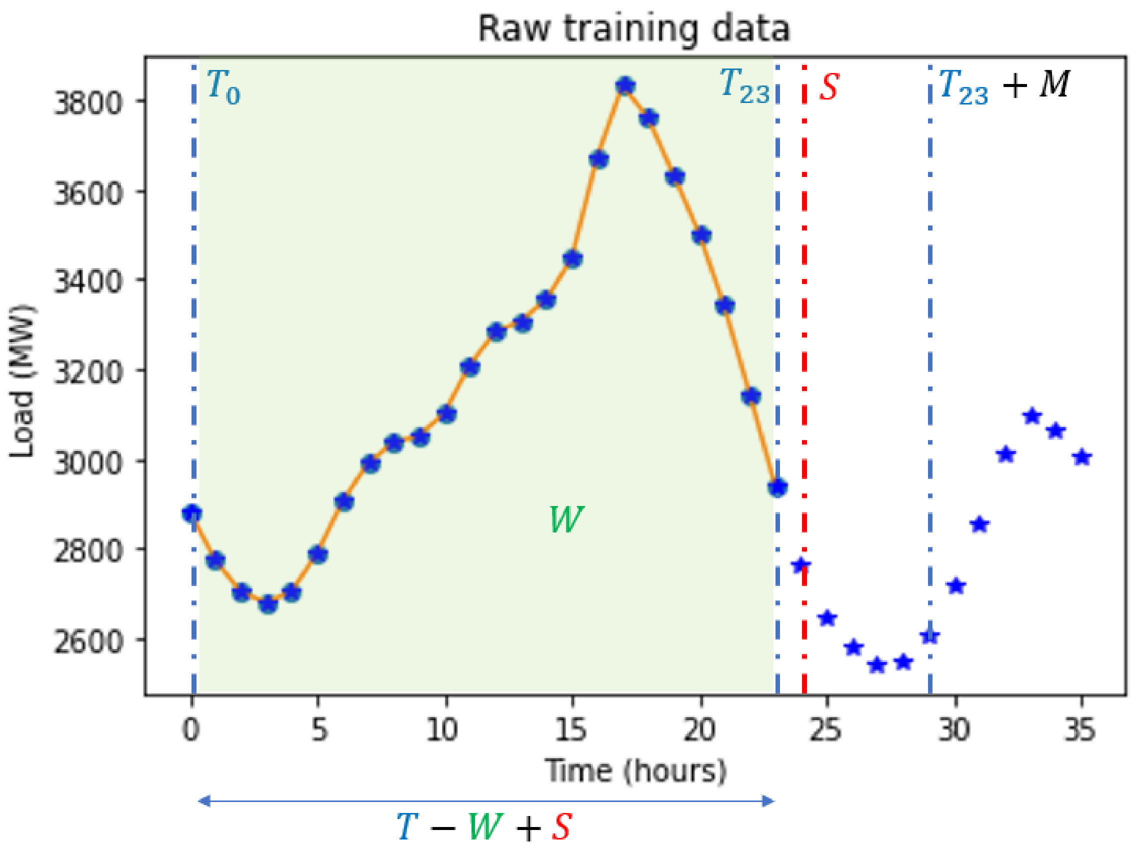
Energy load data from the ISO New England website has been provided via UNM Canvas by Professor Manel Martínez-Ramón. Python functions that include a "sliding_window" and graphing tool are also available through this course's Google Colab repository. Furthermore, Python scripts were provided to read data, convert it to arrays, normalize the data with a vector scaler, and use the sliding window function for training and testing data. The plot_graph function is slightly modified for cosmetic additions.

Sliding Window Function

After the data is converted to a time series array, we will need to use the window test training samples to predict a sample in the future. Figure 1 shows Connecticut's first 48 hours [0:48] of energy load in Mega Watts (MW) in 2021. Figure 1 denotes how the sliding window function intends to utilize 24 samples (hours) of data to predict an energy demand forecast horizon 6 hours ahead. We can see a general increase in energy demand in the 8th hour of the day (morning). The 17th hour, for this day (evening), is the highest demand and trends downwards towards the evening and night. We will use a 24-hour window for the test and iterate it by 1 hour, predicting the nth-hour sample point based on the M horizon value. We will use 24 hours of the window for the test and iterate it by 1 hour, predicting the nth hour sample point based on the M horizon value. We will use 1 and 6 as 12-hour M horizon reference points and analyze the machine's uncertainty as it attempts to predict trends farther into the future, as noted with $T_{23} + M$.

FIGURE 1

First 36 Hours of CT's Load Data, with 24 hours of training data to predict the horizon M



We can also plot the 2021 data for the first day of January and 01-07 July 2021 as the test comparison data. Figure 2 denotes the 48 hours of 01 January 2021. Figure 3 represents the 182nd day of the year, 01 July 2021, and the next seven days stacked on the plot. 01 July 2021 was a Thursday, and the plot notes that energy demand had been higher on that Thursday and Friday, trending lower for the weekend and increasing sharply at the beginning of the following week. The information in the plots is essential as the observed data points the GP attempts to predict with its training data from the 24-hour window. We can narrow down the hours on the 182nd day of 2021 and iterate the test through the first week of July by manipulating the S value and creating the stacked plot.

FIGURE 2

MW Load Usage in CT by ISO New England, January 1-7, 2021, horizon annotated for 1, 6, and 12.

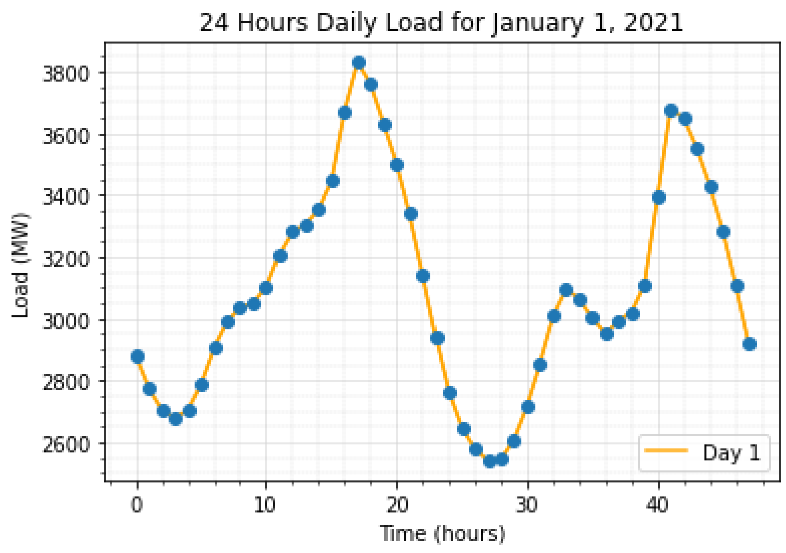
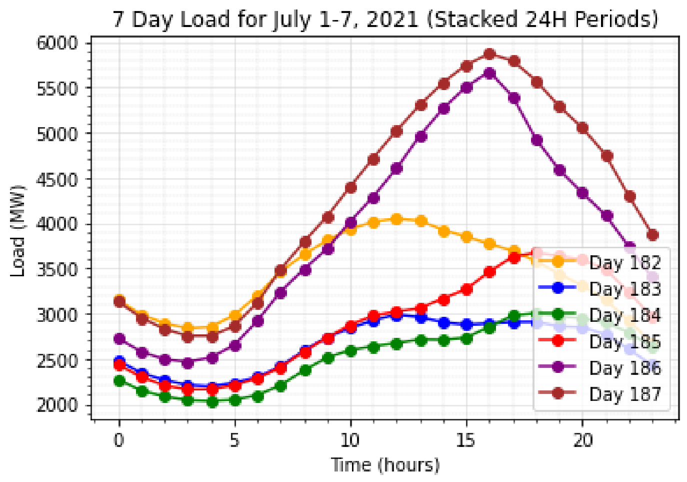


FIGURE 3

MW Load Usage in CT by ISO New England, July 1-7, 2021, horizon annotated for 1, 6, and 12.



Expected Results

Expected results from the project predict 95% confidence intervals based on the predicted performance of the horizon. For example, we take 24 hours, iterate through the raw data in 1-hour increments for 48 hours, and forecast the next 1 hour (M=1) beyond the window. The slider_function is responsible for the iteration of data to the next hour. The GP should do well predicting 1 hour, while the uncertainty should increase as the GP machine attempts to predict 12 hours (M=12). The GP trains on Connecticut's 2020 data to predict the 2021 horizons of 1 hour, 6 hours, and 12 hours for the selected date(s) and time.

Reviewing the raw data, the hourly differentials in energy demand appear primarily non-linear; however, we will see how the training data can handle both winter and summer season test inputs based on the first 1000 training samples of the year. The RBF kernel should be best suited to accurately predict the horizon value based on the non-linear nature of the data. However, the maximum window is 24 hours, so the complexity of the RBF kernel may overfit the data with poor generalized outputs. The choice of the kernel will ultimately affect decision-making on energy load forecasting.

More advanced or various GPR models may yield more precise training for the model. A study published on 08 July 2023 (Yadav et al., 2023) used Exponential GBR kernels to predict energy load demand for cities in India and Australia. The study included short-term load forecasting (1 hour to 1 week) and a GPR model like the one used in this test. An excerpt of results for an Australian city MAPE% using a GBR Linear kernel bias is available in Table 1. The average of the MAPE% were all close for a year's worth of data, with January (winter) and May through August (summer) having the highest error margins.

TABLE 1

MAPE (%) obtained by different Kernel functions and 'Linear' basis function.

Kernel function →	Exponential	Squared exponential	Matern3/2	Matern5/2	Rational quadratic	Ard-exponential
JAN.	0.43	0.2	0.47	0.54	0.1	0.53
FEB.	0.2	0.74	0.34	0.44	0.06	0.38

Kernel function →	Exponential	Squared exponential	Matern3/2	Matern5/2	Rational quadratic	Ard-exponential
MAR.	0.7	0.28	0.13	0.3	0.65	0.01
APR.	0.17	0.01	0.38	0.92	0.28	0.2
MAY	0.33	0.28	0.24	0.77	0.29	0.17
JUN.	0.51	0.84	0.42	0.44	0.41	0.21
JUL.	0.38	0.15	0.55	0.31	0.61	0.6
AUG.	0.39	0.03	0.09	0.01	0.54	0.39
SEP.	0.03	0.13	0.01	0.08	0.53	0.06
OCT.	0.07	0.13	0.22	0.02	0.26	0.41
NOV.	0.28	0.65	0.62	0.18	0.28	0.36
DEC.	0.1847	0.76	0.46	0.64	0.29	0.5
Average	**0.30**	**0.35**	**0.32**	**0.38**	**0.35**	**0.31**

Kernel Training

Data Preprocessing

First, we standardize the data due to the raw data having a mean and covariance. This preprocessing standardization of the data will first result in the mean at 0 and the variance maximum magnitude of 1 before passing the information to the kernel. The predictions will also be standardized but need an inverse scaler transformation applied to the mean and standard deviation after the test. Using the inverse scaler transformation will not only scale the MW Load to its raw data values to reference the observed data but also provide the mean and confidence interval. We can check the GP's performance with a selected kernel versus observed data.

GP Training: Linear Kernel

A linear kernel is implemented with the GP machine for this test. The performance will be analyzed, as tweaks to the machine could yield better results with a low-complexity algorithm. The code used for the assignment utilized the Gaussian Process regression prediction function and a dot product kernel function with noise added to the diagonal of the kernel matrix.

The linear kernel is defined in Eq. 1, where: $K_f(x_1, x_2)$ is the kernel function between x_1 and x_2, and is noted as the DotProduct() with scikit learn. Noise is added to the diagonal of the kernel matrix with $\sigma_n^2 I(x_1 - x_2)$, contributing to the non-zero covariance matrix only on the diagonal. The WhiteKernel() function adds variance to each observation as the kernel of training samples. With the identity statement, the result will be zero if the samples differ. If the samples are the same, the result will be 1.

$$K(x_1, x_2) = K_f(x_1, x_2) + \sigma_n^2 I(x_1 - x_2) \qquad \text{Eq. 1}$$

This is defined in the code below, with a scaler of 1. The scaler will also change as the GP adapts to the parameters with the GaussianProcessRegressor(). This occurs because the kernel dot product is the covariance of y, as expressed in Eq 2.

```
kernel = 1*DotProduct() + WhiteKernel()
```

Utilizing gp.predict(), we can calculate the test sample's predictive mean and standard deviation at new points using the function below and the gp.fit() function for the kernel's best-fit hyperparameters. In Eq 2., the $K(x_*)$ is the kernel dot product between all training and test samples.

$$\bar{f}(x_*) = y^\top (X^\top X - \sigma_n^2 I)^{-1} X^\top X \xrightarrow{yields} y^\top (K + \sigma_n^2 I)^{-1} K(x^*) \qquad \text{Eq. 2}$$

For this linear case, we can compute the linear expectations of two samples, y_1 and y_2. For the linear kernel of $y = w^\top x + error$, we can write the proof of what the DotProduct() + WhiteKernel() kernel can be derived from the following expectation expression and noted in Eq. 3:

$$\mathbb{E}(y_1, y_2) = \mathbb{E}[(w^\top x_1 + E_1)(w^\top x_2 + E_2)]$$

$$= \mathbb{E}[(x_1^\top w)(w_2^\top)] + \mathbb{E}(E_1 E_2) + \mathbb{E}(w^\top x_1 E_2)$$

$$= x_1^\top \mathbb{E}(ww^\top) x_2 + \sigma_n^2 I(x_1 - x_2)$$

$$= x_1^\top \sum p x_2 + \sigma_n^2 I(x_1 - x_2)$$

$$= K(x_1, x_2) + \sigma_n^2 I(x_1 - x_2)$$

$$\therefore\ \mathbb{E}(yy^\top) = K + \sigma_n^2 I \qquad \text{Eq. 3}$$

GP Training: Gaussian RBF Kernel

For the Gaussian RBF kernel, sci-kit learn (sklearn, 2023) uses the covariance function in Eq. 4:

$$k(x, x') = v^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right) \qquad \text{Eq. 4}$$

This function fits appropriately with our data model for predicting future energy load requirements. Eq. 5 expresses how the GP will predict the data, with $f(x)$ representing the training data $(f(x_1), \dots f(x_n)$.

$$p(y|x, f) \sim Normal(f(x), \sigma_n^2 I) \qquad \text{Eq. 5}$$

We can use a covariance function with the DotProduct() kernel and the WhiteKernel() to express the model in Eq 6. The $v, \ell,$ and σ are the hyperparameters.

$$k(x, x') = v^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right) + \sigma_n^2 I \qquad \text{Eq. 6}$$

The code required to run this kernel is stated below and allows the GP to find the best hyperparameters for the model:

```
kernel = 1.0 * RBF(length_scale=1.0, length_scale_bounds=(1e-2, 1e2)) +
WhiteKernel(noise_level=1.0, noise_level_bounds=(1e-10, 1e1))
```

GP Prediction

Within the class GaussianProcessesRegressor(), we train the optimizer with "n_restarts_optimizer = 9". The machine will optimize nine times randomly and choose the best DotProduct() scaler amplitude and best covariance. Next, the gaussian_process.fit() method will train the GP with 1000 training inputs and 1000 samples of the training regressor. The training is accomplished with the code:

```
gp.fit(X_train1[:1000,:], y_train1[:1000,:])
```

Next, we can use the guassian_process to predict the mean and standard deviation. We use a subset of the test data. A prediction interval will define the length of test data where the prediction will be made relative to the window (W) parameter. The GP will predict the mean and standard deviation for the test set's first 'L' window. Due to the slicing of L with "[:L,:]," the prediction is made of a subset of the test data and defines the extent of the test data that is used for forecasts after the model is trained.

Experiment Results

The following section is the experiment's results. As stated, the first test will be to predict the energy load demand in Connecticut for the first day in January 2021 using data from 2020. The second set of figures will denote how well the GP can predict Connecticut's first week of July using horizons 1, 6, and 12.

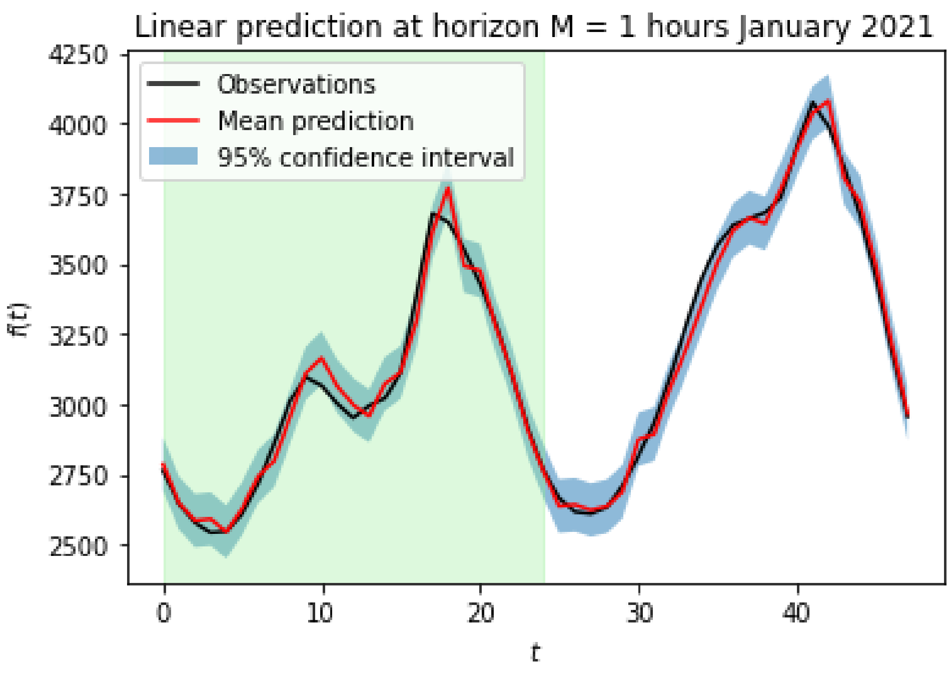
01 January 2021 Linear Kernel Results

The black observation line is the 48-hour data from 01-02 January 2021. When the GP predicts the horizon using the sliding_function, we receive the mean_raw and std_raw parameters. The mean_raw and std_raw plot the mean prediction and the 95% confidence interval for each test.

We can review the plots and determine how well the GP prediction method utilized CT's 2020 training for the 2021 test data. Figure 4 denotes the 24-hour window (light green) to predict 1 hour through each sliding iteration interval (L) in the time series. For the linear kernel, predicting 1 hour for each sliding iteration results in low uncertainty; thus, the mean differential between observed and predicted data is low. We can observe the 95% confidence interval bandwidth as narrower for predicting a point 1 hour ahead of the window.

FIGURE 4

01 January 2021 Predicted Load, Horizon of M=1, Window W=24



Next, we can analyze the GP's prediction efficiency in Figure 5 when the horizon is 6 hours ahead of the window. By only visually inspecting the plot, we see that the mean prediction and its resulting confidence interval contain higher uncertainty. Figure 6 represents the horizon 12 hours ahead of the window and thus denotes the even higher uncertainty. However, the amount of uncertainty is still close to the 6-hour horizon.

FIGURE 5

01 January 2021 Predicted Load, Horizon of M=6, Window W=24

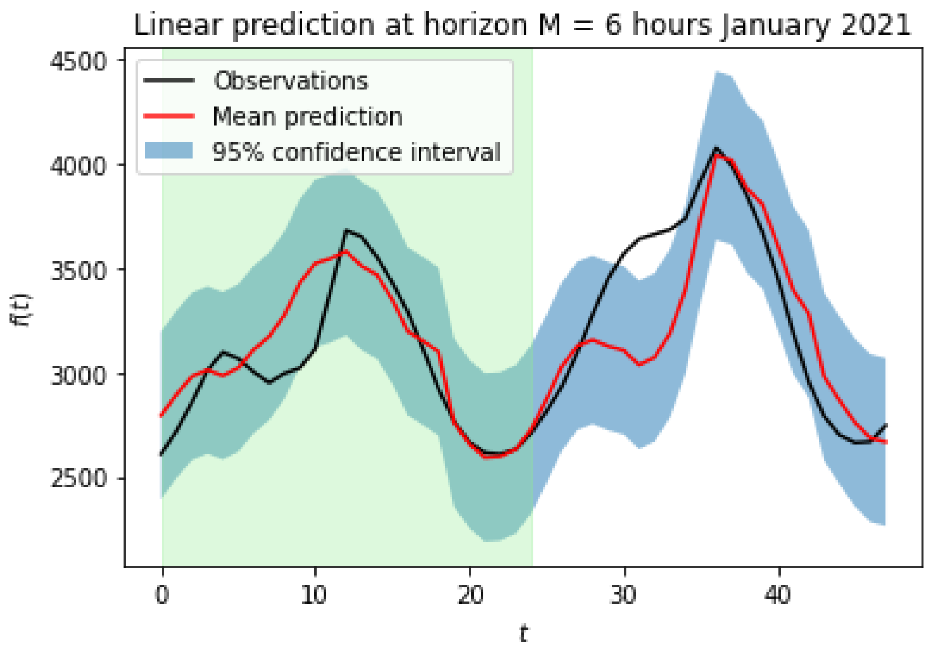
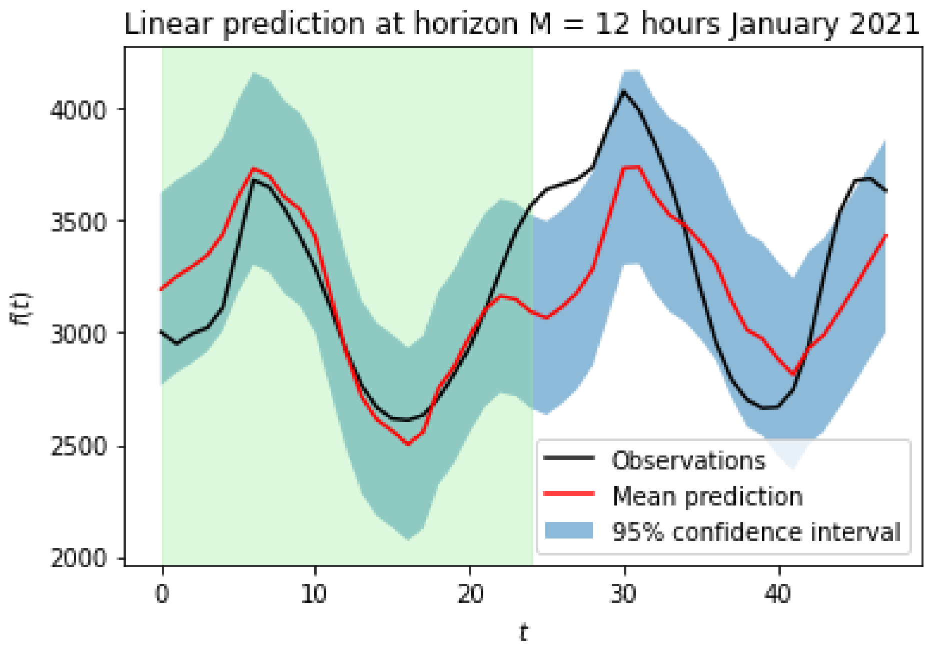


FIGURE 6

01 January 2021 Predicted Load, Horizon of M=12, Window W=24



01-07 July 2021 Linear Kernel Results

In contrast to providing a day of prediction, we can use the GP prediction for the first week of July 2021. 01 July 2021 is the 182nd day of the year; however, due to Python arrays starting at 0, we will need to treat the day as the 181st for the test data and the sliding function. Multiplying 181 by 24 hours results in the understanding that we will need to slide the test data and the sliding function to the 4344th hour of the year.

The week will contain 168 hours of test day, and we will take 24 hours of test data. Again, the sliding function will iterate by 1 hour and predicts points nth horizon hours ahead of the window until it reaches the L interval value. The GP will predict a 1-hour horizon ahead of the 24-hour window, as illustrated in Figure 7. Figure 8 and Figure 9 show how well the process works with 6- and 12-hour points beyond the 24-hour window. Like the January predictions, uncertainty rises as the horizon outlook increases. Indeed, the prediction would still provide an outlook for 6-hour future load planning and mitigation as the confidence intervals are closer to the observed data.

FIGURE 7

01-07 July 2021 Predicted Load, Horizon of M=1, Window W=24

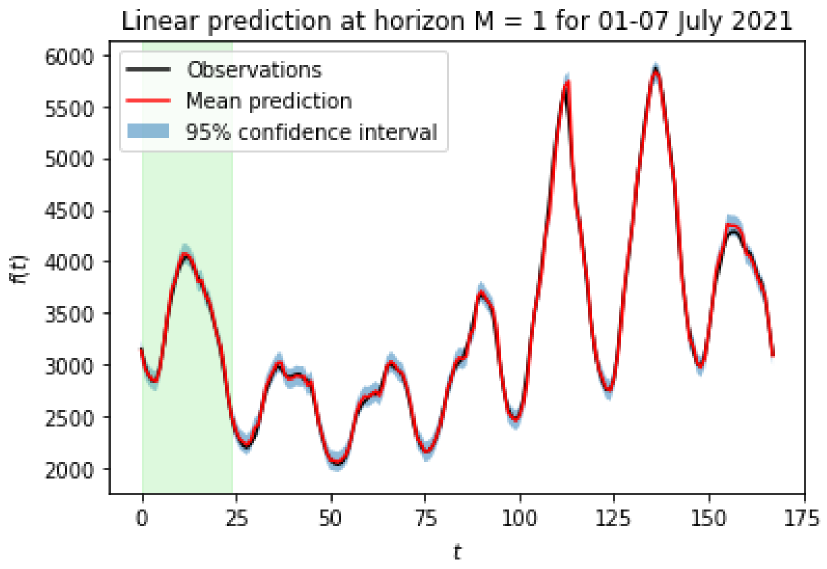


FIGURE 8

01-07 July 2021 Predicted Load, Horizon of M=6, Window W=24

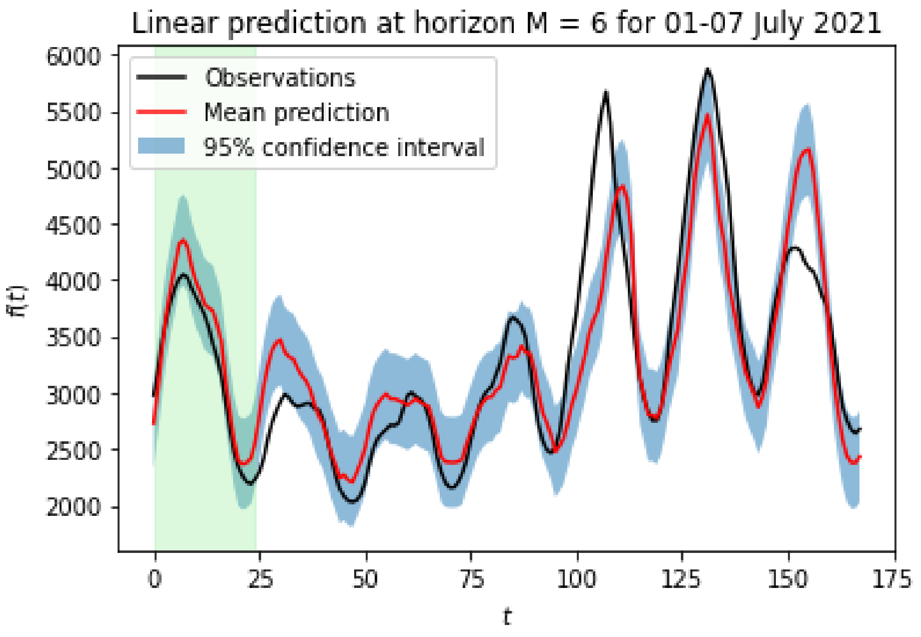
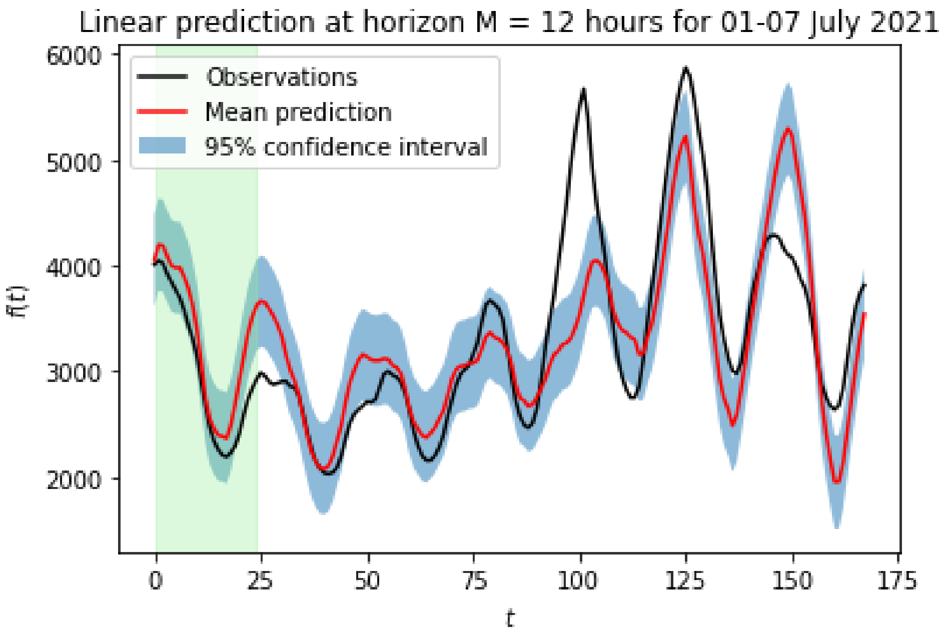


FIGURE 9

01-07 July 2021 Predicted Load, Horizon of M=12, Window W=24



01 January 2021 RBF Kernel Results

The GP GaussianProcessRegressor and the gp.fit() followed the same process as the linear kernel. The most significant difference is how the GP uses the kernel in its prediction model. Both linear and RBF kernels predict the January timeline with similar outcomes, as observed in Figure 10, Figure 11, and Figure 12, denoting horizons 1, 6, and 12, respectively.

FIGURE 10

01 January 2021 Predicted Load, Horizon of M=1, Window W=24

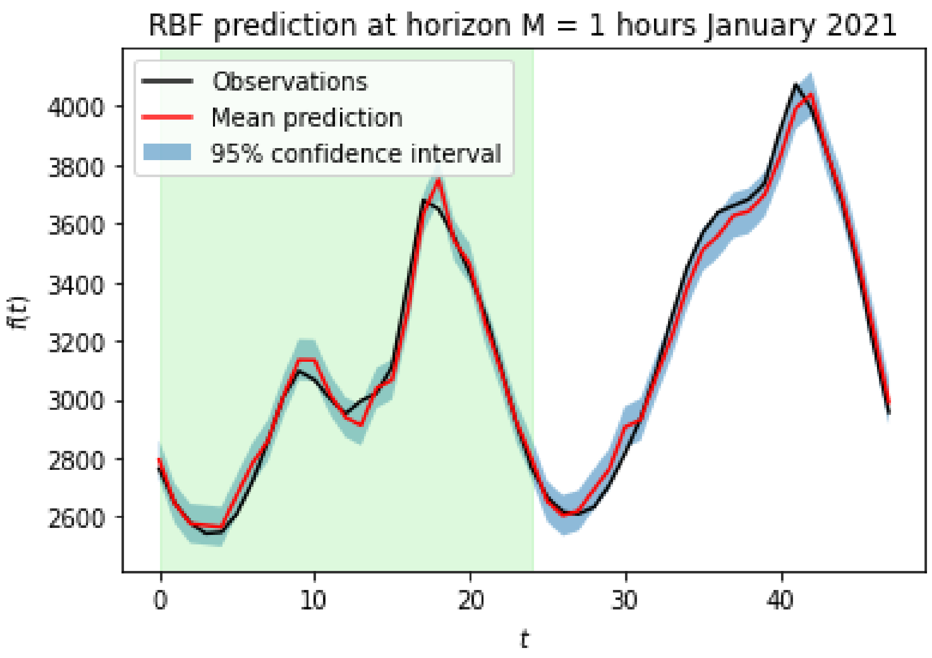


FIGURE 11

01 January 2021 Predicted Load, Horizon of M=6, Window W=24

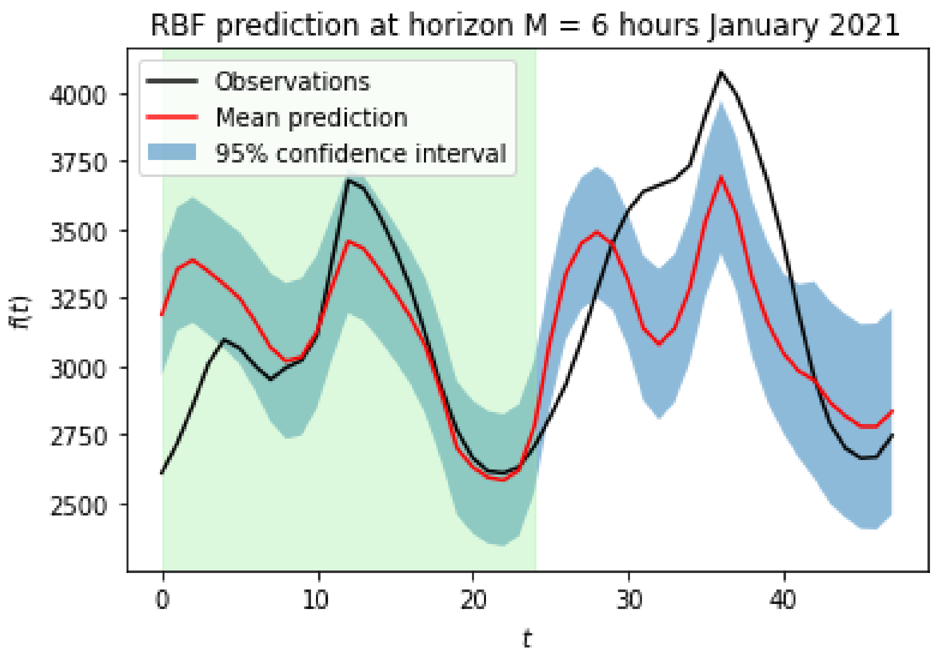
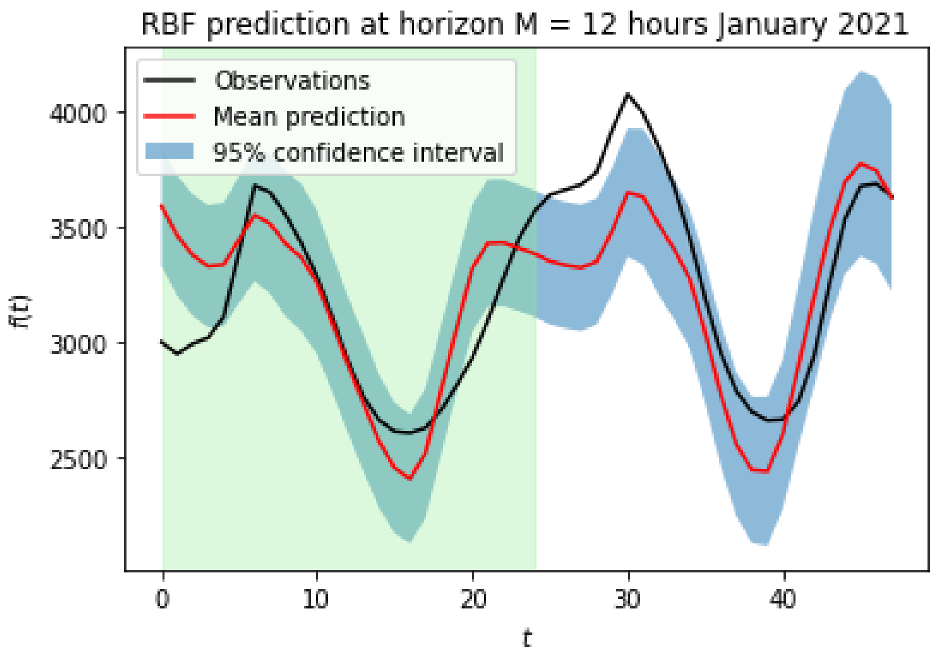


FIGURE 12

01 January 2021 Predicted Load, Horizon of M=12, Window W=24



01-07 July 2021 Linear Kernel Results

Running the GP code with the RBF kernel against the July data yielded different results than the linear kernel. We see in Figure 13 that, for a 1 hour ahead forecast, the RBF kernel can predict the data with little uncertainty. The results in Figure 14 show that with a 6-hour ahead forecast, the RBF also predicts the 2021 trend with lower uncertainty based on a 5-hour difference. In Figure 15, we see a broad increase in uncertainty with the RBF function. This may be due to the RBF overestimating the variability of linear data, and the hyperparameters chosen by the GP model may have skewed the results. The complexity of the RBF kernel is able to detect minor variances with a 1 hour horizon in Figure 13, but may have generalized the 24 hour period too heavily to see unpredictable fluctuations of data. The load use increased the uncertainty after the Thursday to Sunday training data, resulting in high uncertainty in the model. Anomalies in energy demand between weekdays-to-weekends and energy demand between seasons may be a root issue. If we start on a Monday versus a Thursday, the model may encapsulate the wide variance of energy demands. The RBF data does not appear to predict any data from the 186th day onward when the horizon is equal to and greater than six.

FIGURE 13

01-07 July 2021 Predicted Load, Horizon of M=1, Window W=24

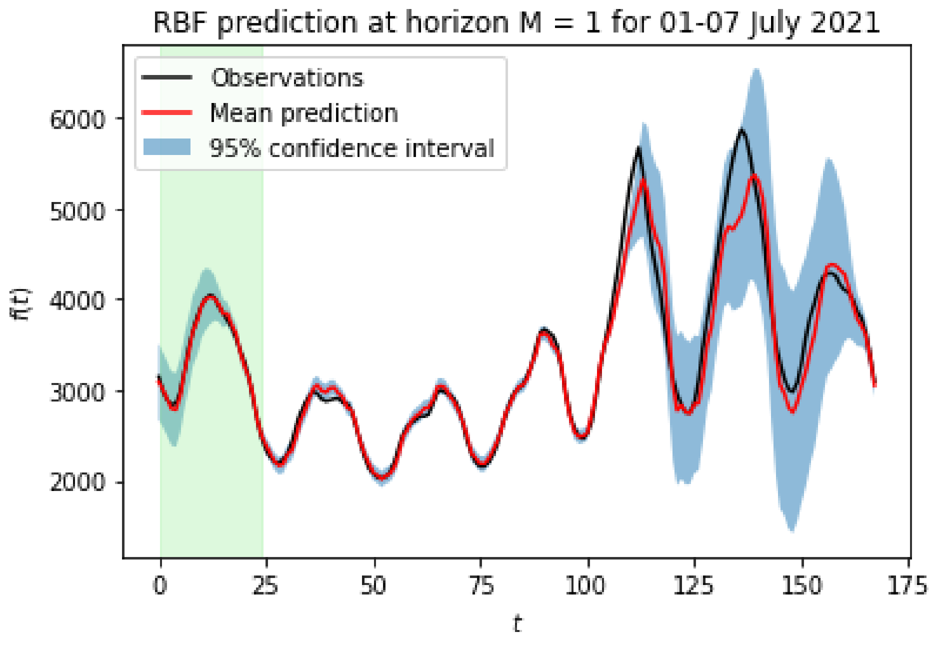


FIGURE 14

01-07 July 2021 Predicted Load, Horizon of M=6, Window W=24

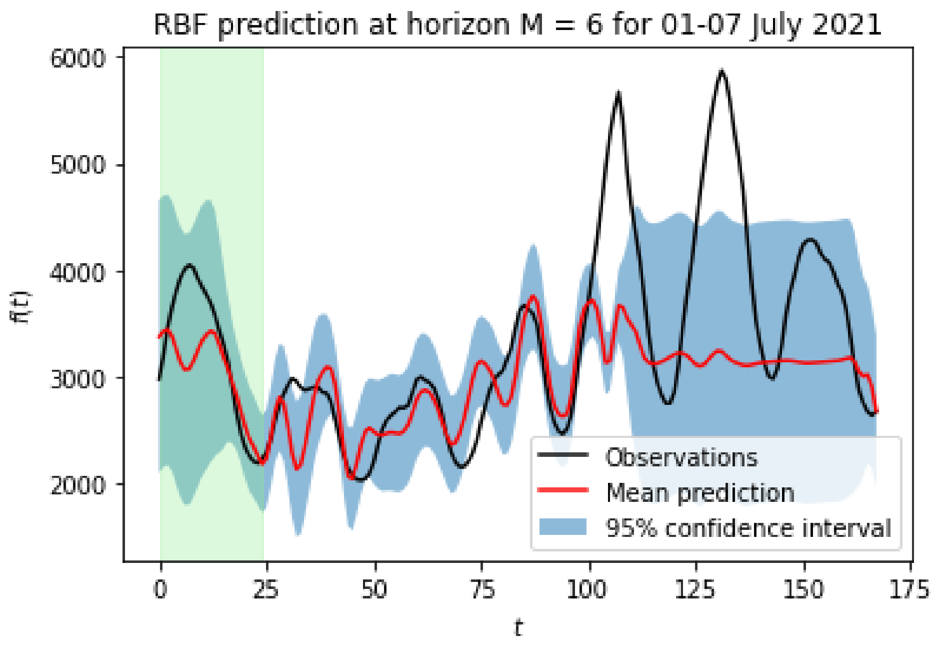
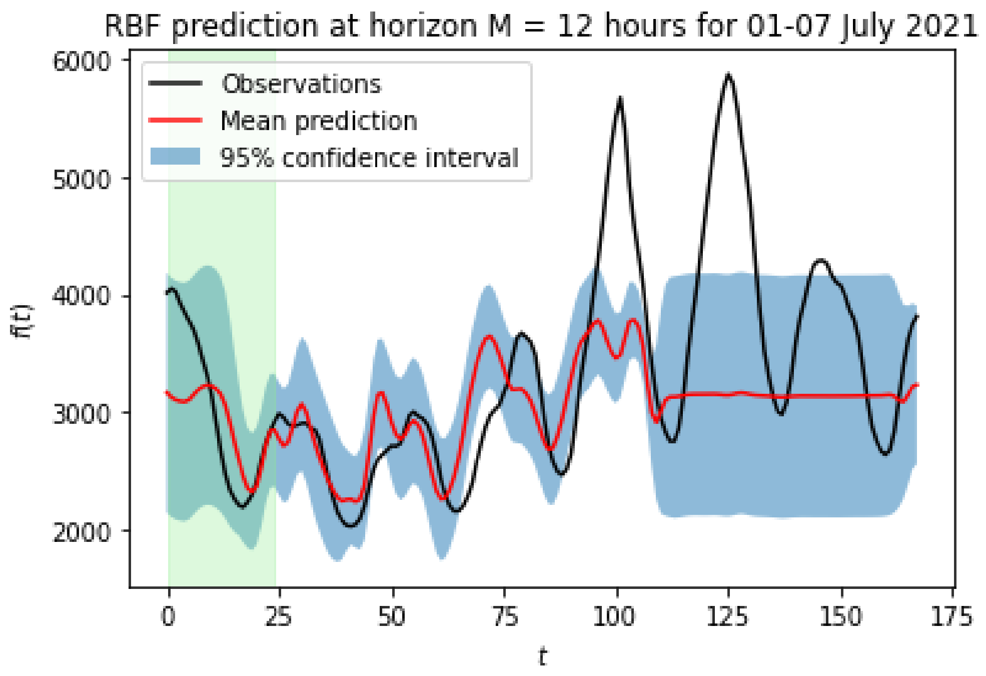


FIGURE 15

01-07 July 2021 Predicted Load, Horizon of M=12, Window W=24



Final Observations

Using a GP model provides excellent to adequate results for forecasting the energy load in this exercise. Both kernels behaved differently based on the data presented to the model, and we can undoubtedly utilize one kernel over the other based on the confidence intervals and mean predictions. In this test, we let the GPR choose the best hyperparameters based on the training data. We may find a more precise model if we use an iterative process to update the hyperparameters. Attempting to change the parameters manually did not adjust the outcome and the GBR. Separate tests with cross-validation did not improve the results, possibly due to the GBR picking hyperparameters very specific to the data and training series without deviation. Finally, starting at 1000 training hours before July 1st and encapsulating training at 3000 hours, compared to 1000 training hours for the assignment, produced less uncertainty, as training data for the specific July hours were used as opposed to January's. The Linear kernel results for July 2021 test data that trained on July 2020 data are in Figure 16, and the RBF kernel results are in Figure 17. Similar tests to increasing the sample size, prediction period, and adding additional kernels for testing could improve the model's performance for data with potential anomalies.

Other factors outside of raw data entry may help us understand how the GP trained with 2020 data to predict subsets of 2021 data. For instance, Governor Ned Lamont of CT issued an executive order for residents to stay home starting 23 March 2020 (Office of the Governor & Lamont, 2020) due to COVID-19. On 19 May 2021, Gov. Lamont lifted many restrictions for CT's residents (Office of the Governor & Lamont, 2021). The wide variance of residents and businesses using energy during January or July may have influenced the uncertainty with the population at home versus office buildings and retail outlets. Weather may have increased energy demand for air conditioning during July or decreased the demand in January if the winter was mild. This variation may have been too complex for the RBF model. Second, 01 July 2020 occurred on a Wednesday, and the weekday-to-weekend transition may have influenced the training. Data trained from the 2020 timeline will have unique results on the 2021 time series over a week versus the 48-hour timeline in January, primarily if we use winter data to predict a summer trend.

FIGURE 16

GP with Linear Kernel: 3000 Test Points for 01-07 July 2021

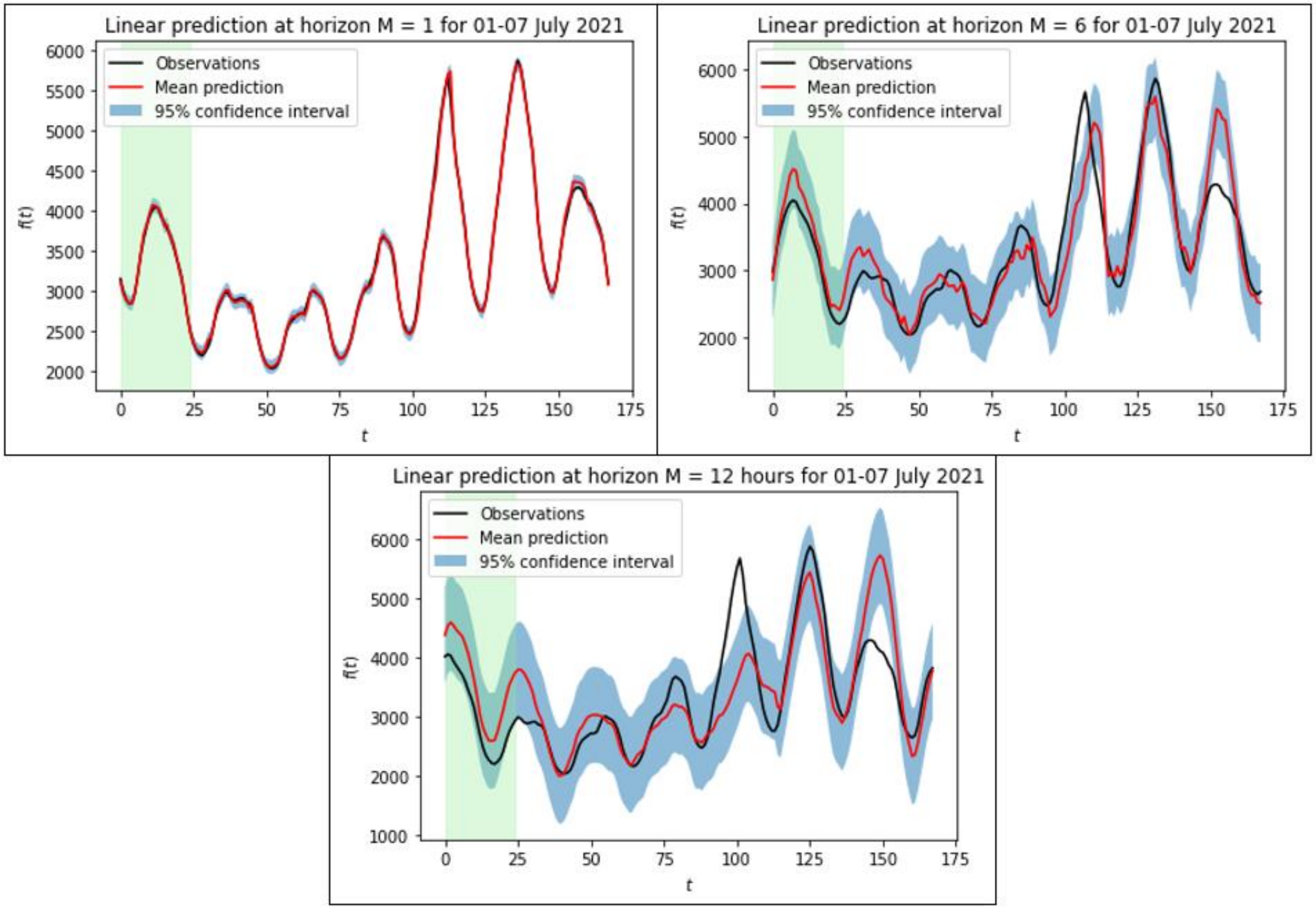
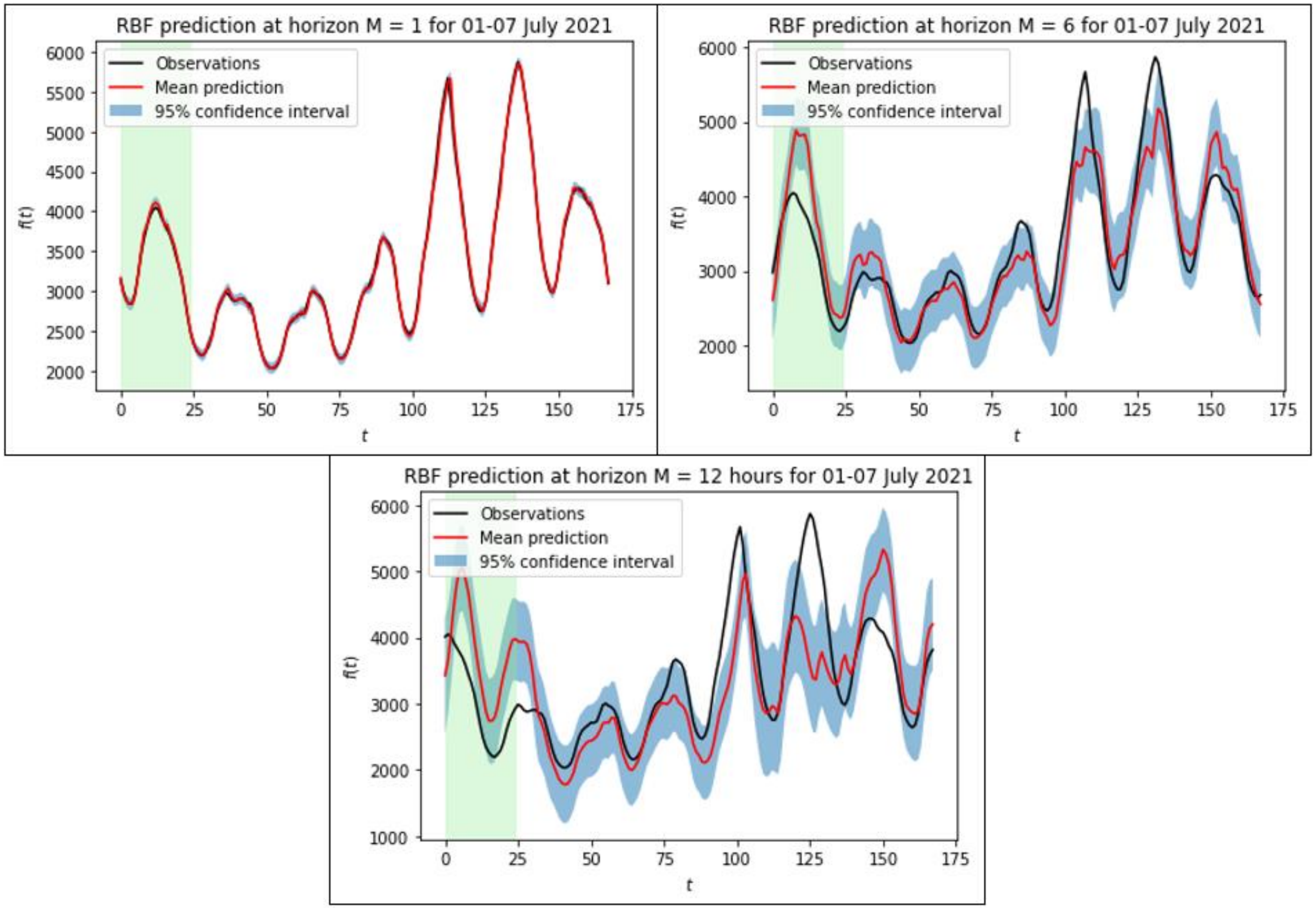


FIGURE 17

GP with RBF Kernel: 3000 Test Points for 01-07 July 2021



References

Office of the Governor, & Lamont, N., Executive order no. 7H: Protection of Public Health and safety during COVID-19 pandemic and response--restrictions on workplaces for non-essential businesses, coordinated response effort 1–5 (2020). Hartford, CT; State of Connecticut.

Office of the Governor, & Lamont, N., Executive order no. 12: Protection of Public Health and safety during COVID-19 pandemic--revised order for masks and face coverings 1–4 (2021). Hartford, CT; Connecticut.

Sklearn.gaussian_process.kernels.RBF. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.RBF.html

Yadav, A., Bareth, R., Kochar, M., Pazoki, M., & Sehiemy, R. A. (2023). Gaussian process regression-based load forecasting model. *IET Generation, Transmission &amp; Distribution*. https://doi.org/10.1049/gtd2.12926