Gaussian Processes for Probabilistic Electricity Price Forecasting

Gabriel Arpino[a,b], Howard Shih[c], Wessel Bruinsma[d,b], Eric Perim Martins[b]

[a]*ETH Zürich*
[b]*Invenia Labs*
[c]*Cornell University*
[d]*University of Cambridge*

Abstract

Probabilistic electricity price forecasting (PEPF) has become a crucial component for energy systems planning and decision making in this day and age. Point predictions are unable to quantify the growing uncertainty around the introduction of renewable energies and smart technologies, so PEPF has become an integral step in the decision making pipeline of utilities, generators and other market participants. We empirically motivate the Gaussian Process model as principled, interpretable, and flexible probabilistic electricity price forecasting model for the short-term (1-2 hours) and medium term (1-2 days) prediction regimes. Following novel guidelines for PEPF described in [1], we construct a Gaussian Process model that performs competitively compared to current state-of-the-art approaches on the GEFCom2014 competition dataset, all while preserving interpretability and with proper uncertainty quantification in mind.

Keywords: Probabilistic, Forecasting

1. Introduction

Competitive electricity markets today are the main government-unregulated platform for reliable electricity trading worldwide. Electricity price forecasting is a fundamental tool for the operation of grid producers, consumers, and investors, and are essential components in the pipeline of energy companies' decision making processes. Surveys of the literature by De Gooijer and Hyndman

[2], concluded that point forecasts present serious limitations to the usefulness of electricity price and load forecasts, because the uncertainty in prediction is unaccounted for. They highlight the recent popularity of *probabilistic forecasting* as a new avenue for risk-aware electricity price forecasting. The work was followed by Nowotarski and Weron's review of PEPF (*probabilistic electricity price forecasting*) [1], offering guidelines for the rigorous evaluation of probabilistic models of electricity prices, also distinguishing probabilistic forecasts as a promising reliable method of forecasting in the field going forward. The review emphasizes that the recent introduction of smart technologies and renewable generation into the grid has played a role in increasing the uncertainty in future demand, supply, and prices of electricity. Therefore, it is expected that electricity price forecasting with improved uncertainty assessment will play an increasingly important role in energy systems decision making.

Consequently, there is an increasing interest in producing electricity price forecasts that account for uncertainties in the prediction. *Probabilistic Forecasting* involves predicting future points along with either prediction intervals or densities. There is a growing body of literature offering probabilistic solutions to the electricity price forecasting problem but, to the best of our knowledge, work evaluating probabilistic medium-term forecasting models on the reference dataset and metrics outlined in [1] is lacking. Moreover, most works simplify the framework and simply predict 24 hourly marginal distributions, rather than their joint distribution [1], which eliminates information on price correlations across different hours. It is important to note, however, that this problem has already been addressed in other areas of the energy systems probabilistic forecasting literature, such as wind forecasting [3]. Our work allows for the expressive and intuitive construction of prior covariance structure among data points, along with calibrated uncertainty analysis past the prediction interval regime. This is thanks to the chosen Bayesian reasoning framework in statistics which allows for probabilistic model design with built-in methods for quantifying uncertainty, a framework we further explore in the *Bayesian Formalism for Prediction* section. We propose Gaussian Processes (GPs), principled Bayesian probabilistic models with which one can do inference on future data, as a solution to the short (1-2 hours ahead) and medium (1-2 days ahead) term probabilistic forecasting of electricity prices. This is, to the best of our knowledge, the first presentation of such a solution to the probabilistic EPF problem. In what follows, we motivate GPs as simple and principled probabilistic forecasting models that are able to : a) return calibrated uncertainties through *Bayesian inference*, b) allow for intuitive model interpretation through a *decomposition of prediction*, c) easily incorporate expert knowledge through *kernel design*. Our code is publicly available at `codeisnotpublicyet`. We begin by outlining relevant work in the field, and proceed with an overview of the Bayesian Formalism for probabilistic prediction. This is followed by an introduction to Gaussian Processes, including their application in a decision making example. After this exposition, we outline the dataset and evaluation metrics used to evaluate our GP models, along with a guide for designing our GP model in the PEPF setting. We conclude with experimental results, and a further look ahead into the applications of GPs to PEPF.

2. Related Work

Numerous methods for probabilistic EPF have been proposed, with impressive empirical performance on arbitrary datasets as measured by specific evaluation metrics. Already existing models range from ARMA models to computational neural networks and random forests, which demonstrate notable performance on short-term (1-2 hours ahead) and medium-term (1-2 days ahead) forecasting. We start with a review of the state-of-the-art in probabilistic electricity price forecasting.

The leading survey on probabilistic EPF [1] outlines the literature's state-of-the-art probabilistic EPF techniques and motivates the need for an all-encompassing solution to the problem. It also establishes the first empirically motivated guidelines for the evaluation of probabilistic EPF models, favouring models which "maximize sharpness subject to reliability" (see *Evaluation Metrics*). The paper constructs much needed guidelines for PEPF model evaluation based on the Global Energy Foreacasting Competition (GEFCom2014) [4] data, which focused on probabilistic load, price, wind, and solar forecasting. These guidelines are the ones we will follow in this paper.

The survey in [1] also highlights two models that performed best in terms of the Unconditional Coverage and Average Coverage Error metrics (UC and ACE respectively). These models are considered baselines for comparison in this paper. They are autoregressive and produce prediction intervals: ARX [5], and m-ARX [6].

Probabilistic medium-term forecasting of energy prices is a less explored problem compared to the short-term analogue. García-Martos et al. [7] propose a dimensionality reduction model based on heteroskedastic common factors which is able to predict one-day-ahead electricity prices with Gaussian prediction intervals, but do not evaluate their prediction. This work uses nine years of electricity market price data to generate heteroskedastic common factors for one-day-ahead prediction, and hence we believe it requires some adaptation in order to be applied to our two year reference dataset containing load information. Similarly, Alonso et al. [8] propose a prediction interval-based PEPF method using Dynamic Factor Analysis, but only evaluate their prediction for a single week on a six year Spanish electricity price dataset. The work of Brusaferri et al. [9] proposes a Bayesian deep learning forecasting method (which produces probabilistic intervals), but they do not optimize the neural network hyperparameters for prediction since the objective was to contrast the Bayesian and frequentist frameworks in neural networks for EPF, hence this method could require some adaptation in order to be evaluated on the reference dataset considered in this paper ([1]).

Gaussian Process (GP) modelling was first introduced to the electricity forecasting community in the form of load forecasting models such as those outlined in [10]. Mori and Nakano [11] indicated the potential of Gaussian Processes as a PEPF model, but only demonstrated its effectiveness for short-term forecasts (one hour ahead). The work also does not demonstrate the constructible and interpretable nature of GP modelling through kernel composition to the community. This follows from other kernel methods outlined in literature, such as in [12], which are also short-term forecasting methods (2 hours ahead). In this work we present a competitive Gaussian Process model for medium-term (1 day) PEPF evaluated on the benchmark dataset from the GEFCom2014 conference mentioned in [1]. This is an important forecasting scenario due to the day-ahead and real-time two settlement structure of wholesale electricity markets present in countries such as the United States and Canada [13]. A one day ahead forecast (medium-term) is essential for estimating the difference between day-ahead and real-time market value, a difference that can have significant effects not only on market efficiency but also market reliability. Grid operators benefit from the knowledge of near-future spikes in market price, and policymakers benefit from understanding how predictable such spikes are and how they affect energy demand. Medium term market forecasts are therefore essential to the operations of grid staff and policy makers in competitive wholesale markets around the globe [14].

Efforts for medium-term to long-term probabilistic forecasting exist in the PEPF literature. The work of Ziel and Steinert [15] provide a first model for probabilistic forecasting of prices on the scale of days to years. This was the first paper to demonstrate the possibility of long-term forecasting, and the results are promising based on common error measures. The paper, however, did not evaluate their forecasts on a reference dataset, but instead served as a first approach to tackling the medium-term to long-term PEPF problem. Gaussian Processes for long-term probabilistic forecasting have not been analyzed, and could be a fruitful line of study.

3. The Bayesian Formalism for Prediction

As outlined in the introduction, uncertainty quantification has become an increasingly important component to electricity market planning and operation, motivating the need for a statistical reasoning framework for which uncertainty quantification is a central concern. With this in mind, our statistical framework is expected to output probability distributions over our predictions, and not just point predictions. We begin by introducing the Bayesian formalism for reasoning about uncertainty: we encode our assumptions about our data as probability distributions, and automatically obtain probabilistic predictions by applying Bayes' rule. We encourage viewing the Bayesian formalism as a machine where the inputs are (1) prior assumptions about our data (what we call the "prior") and (2) the dataset itself, and our outputs are uncertainty quantifications as distributions over yet unobserved data, such as market prices for the next day. In the scenario of PEPF, our prior is a distribution indicating how the market price is expected to behave (with features such as periodicities, regular spikes and dips), our dataset consists of the past market price and load data, and our prediction is a distribution over the future market price (ideally displaying similar features we added in our prior such as periodicities). This direct relationship between input uncertainty and output uncertainty (Bayes' rule) allows the user to properly quantify and control uncertainty in their prediction, and is hence an attractive statistical framework for conducting inference over the electricity market. Specifically, the uncertainty in our predictions arises directly from the assumptions applied during model building. Bayesian models as these are ubiquitous in the machine learning and statistics communities, as they allow users to be explicit about their lack of knowledge regarding the data generating process, and to automatically quantify the probabilities of their model outputs [16].

The Bayesian formalism measures model goodness of fit with the marginal likelihood. For input data $\mathcal{D} = \{x_1 \dots x_n\}$ and model M, the marginal likelihood is defined as $P(\mathcal{D}|M)$. In words, this describes the probability of the data if we assume that the data generating process is given by model M. The natural preference for simpler models built into this quantity is known as Bayesian Occam's razor [17, 18]. Because more complex models are able to describe larger families of data, the distribution $P(\mathcal{D}|M)$ naturally flattens out for models with increasing complexity. The automatic flattening of complex distributions occurs due to the fact that distributions must sum to one while having increasing support. This form of implicit regularization allows for automatic model selection based on the marginal likelihood: select models with greatest marginal likelihood over your data. Examples of such models range from simpler Bayesian linear regression models to Bayesian neural networks (BNNs) with many parameters. Nonlinear Bayesian models are preferred for modelling PEPF data as the market price time series often displays a non-linear relationship with time and load. BNNs, although nonlinear, only allow for approximate inference past the prediction interval regime. Gaussian Processes, however, are non-linear Bayesian models for which prior assumptions can be easily specified through the use of mean and covariance functions. We are able to conduct exact inference over these models, they produce analytical and closed-form distributions over our outputs, and they are maximally uncertain about all other moments of the predictive distribution after we have specified the distribution mean and covariance. Gaussian Processes (GPs) are hence the preferred Bayesian model for conducting probabilistic inference over electricity market data, and we outline the details of the model in what follows.

A Gaussian Process illustration of the natural penalization for complexity captued by the Bayesian Occam's razor is demonstrated in Figure 1, where Gaussian Processes of increasing complexity (covariance functions encoding polynomials of higher degree) are fit to a degree 4 polynomial and selected based on highest log marginal likelihood ($\log P(\mathcal{D}|M)$, abbreviated as LML). It is important to note that the Gaussian Process fit with the highest LML value (highest marginal likelihood) corresponds to the function space of order 4 polynomials. Bayesian implicit regularization in this case helps rule out models of degree 5 and of degree 6 for example, which are truly overly complex for our degree 4 polynomial data. Hence, Bayesian models are attractive models for probabilistic electricity price forecasting because they naturally output predictive distributions and contain this characteristic implicit regularization which helps prevent model overfitting.

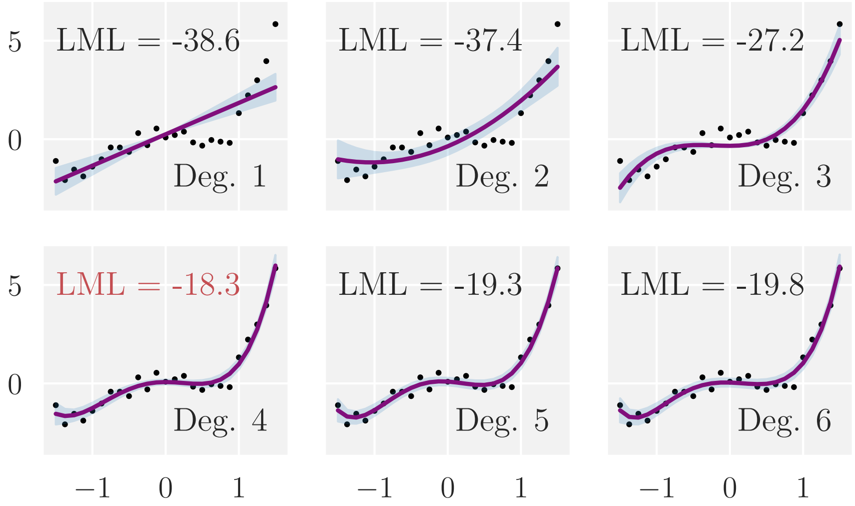

Figure 1: Gaussian Processes with covariance functions of degree ranging from 1 to 6 fit onto data sampled from the data generating process $\mathbf{f}$, where $\mathbf{f}(x) = x^4 + x^3 - 1.2x^2 + 0.2x + \eta$ and $\eta \sim \mathcal{N}(0, 0.4)$. Log Marginal Likelihood (LML) displayed, with lower value indicating better model fit. Gaussian Process posterior mean in purple, and 95% uncertainty interval in shaded blue are plotted.

4. Introduction to Gaussian Processes

The aforementioned Gaussian Process model is a principled, interpretable, and intuitively specified Bayesian model for conducting inference and producing probabilistic predictions. It can be viewed as the generalization of a Gaussian distribution in $\mathbb{R}^n$ to a space of functions. It allows us to place a prior (a distribution encoding our prior assumptions) over the space of functions which describe our signal, condition on an observation, and obtain a posterior distribution over new function points, following the Bayesian machine intuition described in the previous section. Fitting data with a GP entails specifying a prior mean and covariance function or kernel (which may contain unspecified parameters called *hyperparameters*), conditioning on observed data, and sometimes conducting an optimization over the hyperparameters. After conditioning a GP on data, the resulting Gaussian Process can be viewed as a weighted average of trained points according to the weights specified by the kernel. The conditioned Gaussian Process evaluated at an arbitrary input x is therefore an average of the observed points near x, weighted by the kernel, and forms a sort of *smoothing* between points. The kernel is therefore in practice commonly used as an intuitive and flexible way to encode our prior assumptions over the function space we are trying to model. These models come with a clear procedure for prediction: specify prior beliefs, condition on observations, and obtain predictive distributions [19].

Proceeding with a more formal introduction useful in future sections, Gaussian Process are defined as a collection of random variables any finite collection of which follows a joint multivariate Gaussian distribution. In the context of this paper, Gaussian Processes modelled over time are stationary stochastic processes (time-dependent random variables) whose values at a finite number of time points jointly obey a Gaussian distribution. In the machine learning literature, GPs have more practically been described as distributions over functions $f(\mathbf{x})$. The technical and mathematical underpinnings of Gaussian Processes are outlined in Appendix 10. The principal design component of a GP is its *kernel* or *covariance function*, which describes the assumed correlation structure between points in the function space. GPs fit into a family of Bayesian models called non-parametric Bayesian models [20], where assumptions on the data generating function space are encoded through the choice of kernel. Kernels typically take the form of functions over two time points (x_1, x_2), and tend to possess a few *hyperparameters*. Philosophically, these serve to index the family of functions described by that kernel. Examples of kernels include the *Squared Exponential (EQ)* and *Periodic (Per)* kernels defined by the following expressions:

$$\kappa_{EQ}(x_i, x_j) = \sigma^2 \text{exp}\left(-\frac{\|x_i - x_j\|}{2\ell^2}\right) \tag{1}$$

$$\kappa_{Per}(x_i, x_j) = \sigma^2 \text{exp}\Bigg(- \frac{2\sin^2(\pi(x_i - x_j)/p)}{\ell^2}\Bigg) \tag{2}$$

where σ^2, ℓ, p are positive variance, lengthscale, and period hyperparameters respectively. Variance and lengthscale are examples of typical hyperparameters present in covariance functions. The former encodes the amplitude of functions in our function space, roughly how much our function can deviate from its mean. The latter encodes the frequency of variation in our function space: the smaller the lengthscale, the smaller the correlation between two points will be, and hence, visually, the *wigglier* the function space. The EQ kernel as a function encodes an overall smoothness on the space of possible functions that describe our data, and combined with its two hyperparameters σ^2 and ℓ defines a family of smooth functions of bounded variation and wiggle. The Periodic kernel with its added p parameter encodes a smooth periodicity in the underlying data with period p [21].

Samples from a Gaussian Process with an EQ kernel for different settings of hyperparameters are shown in Figure 2, along with samples of other commonly used kernels, such as the Periodic, Squared Exponential, and Rational Quadratic kernels, used in our experiments and further outlined in Appendix 10. Notice the difference in wiggliness of functions sampled with small lengthscale ℓ, and others sampled with large lengthscale ℓ.

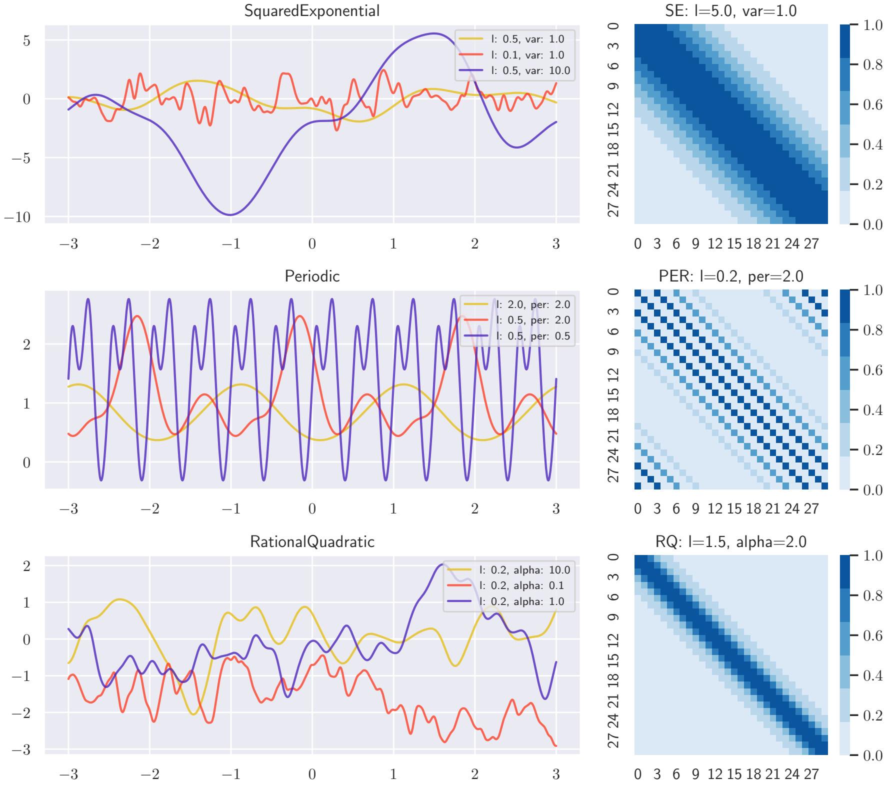

Figure 2: Sampling functions from kernel with different hyperparameters. Heatmaps on the right display the covariance matrix that is output by the kernel.

Moreover, the addition and multiplication of kernels also results in a kernel. Therefore, covariance functions can be combined through multiplication and addition to produce more complex covariance structures. The addition of two kernels (κ_1 and κ_2) is analogous to an OR operation: two points x_1 and x_2 are considered highly correlated if they are highly correlated in either κ_1 OR κ_2 ($\kappa_1(x_1, x_2)$ or $\kappa_2(x_1, x_2)$ is large in magnitude). Conversely, the multiplication of two kernels is analogous to an AND operation: two points x_1 and x_2 are considered highly correlated if they are highly correlated both in κ_1 AND κ_2 ($\kappa_1(x_1, x_2) \cdot \kappa_2(x_1, x_2)$ is large in magnitude). With a composite kernel, the variance hyperparameter, σ^2, of each component can also be interpreted as the relative strength of its signal.

Kernels are, therefore, a flexible and intuitive way to encode assumptions about our function space, and Gaussian Processes are simple probabilistic models that allow us to conduct inference given these assumptions. Gaussian Process posteriors are typically visually demonstrated by plotting the mean and 95% uncertainty interval around their predictive points, as demonstrated in Figures 1 and 3.

5. Decision Making Example

In this section we present a GP-aided decision making toy problem which motivates the need for correctly calibrated predictive uncertainties in probabilistic models. It is often not enough to just output predictive distributions, but instead predictive distributions which correctly encompass the uncertainty in the underlying quantity to be modelled, as errors in uncertainty calibration can lead to significantly different decisions from automated systems, even if the expected value is the same. This helps motivate the need for improvements over past probabilistic electricity price forecasting models and the uncertainties they output. In this toy example, we consider the problem of optimizing (maximizing) the *Sharpe Ratio* at some future point $\mathbf{x}^*$ of a financial portfolio with three independent elements. Assuming a risk-free rate of zero for this example, the

Sharpe ratio of a portfolio element at time t is defined as the expected return divided by the volatility (measured by the standard deviation) of the portfolio. The greater the value of the Sharpe ratio, the more attractive the risk-adjusted portfolio return [22]. We define a data generating process, then compare the Sharpe ratio obtained by a GP with correct kernel specification (and hence correct uncertainty calibration) to a GP with mismatched kernel specification. Ideally we expect to observe that the Sharpe ratio, the quantity to be maximized, is greater for the GP with correct uncertainty calibration. This is meant to reflect the impact that uncertainty calibration can have in decision making tasks, translating to decision making scenarios in the electricity market.

Firstly, we condition three GPs on external randomly generated data to define our data generating process, which represents three elements in a portfolio. The external data can be obtained from our experiment code linked in the introduction. The data generating process consists of three conditioned independent GPs with the following covariance functions for GPs $1, 2, 3$ respectively:

$$\begin{aligned} \kappa_1 &= 5 \cdot \mathtt{P}(\text{MA32}(\ell = 1), T = 1.0) \\ \kappa_2 &= 1 \cdot \mathtt{P}(\text{MA32}(\ell = 1), T = 1.0) + L \\ \kappa_2 &= 0.01 \cdot \mathtt{P}(\text{MA32}(\ell = 1), T = 1.0) \cdot L \end{aligned} \tag{3}$$

where $\mathtt{P}$ is the periodicity operator outlined in [23], MA32 refers to a stationary kernel called the Matern–3/2 kernel, and L refers to the linear kernel (outlined in Appendix 10). All data generating GPs are defined with likelihood variance 0.01 (see Appendix 10 for an outline of the role of likelihood variance in GPs). The GP covariance functions for the data generating process were created in such a way to have inherently predictable periodicities but non-trivial uncertainty properties (the linear kernel multiplied by another kernel, for example, causes the uncertainty of the product process to increase linearly with time). We set our training window to be three days, and sample 15 points from each of the aforementioned data generating GPs within this 3 day window to create our dataset. Our goal is then to produce predictions for the prices of each portfolio element at future time $\mathbf{x}^* = 3.2$ (hour 5 of day 4). We then perform Markowitz Portfolio Optimization [24] on these three independent portfolio element predictions to determine the best weight allocation for the elements in our portfolio so as to maximize the Sharpe ratio of the entire portfolio. We contrast two probabilistic decision making systems that attempt to solve the prediction portion of this problem for each portfolio element: one GP conditioned on the 15 sample points using the same covariance function as that of the data generating process in (3) (properly calibrated uncertainties), which we call the Ground Truth GP (GT), and another GP conditioned on the 15 sample points involving covariance functions slightly different from those of the data generating process (uncalibrated uncertainties) which we call the Mismatched GP (MM).

The covariance functions chosen for the MMs $1, 2, 3$ are:

$$\begin{aligned}
\bar{\kappa}_1 &= 5 \cdot \texttt{P}(\text{EQ}(\ell = 1), T = 1.0) \\
\bar{\kappa}_2 &= 1 \cdot \texttt{P}(\text{EQ}(\ell = 1), T = 1.0) + L \qquad (4) \\
\bar{\kappa}_3 &= 0.01 \cdot \texttt{P}(\text{EQ}(\ell = 1), T = 1.0) \cdot L
\end{aligned}$$

respectively. Notice that the only difference between the MMs and the GTs is that the former's base kernel inside of the periodicity operator `P` is the EQ kernel, and the latter's the MA32 kernel. The MA32 kernel in the data generating process encodes a space of continuous time functions that are not necessarily smooth, whereas the EQ kernel induces functions with a strong degree of smoothness throughout. We will see that such an apparently small change in the underlying smoothness of the function space will cause a significant difference in our final automated Markowitz Portfolio Optimization decision.

It is observed in Figure 3 that, although the visual differences in posterior fits are small, the GTs and MMs produce significantly different Markowitz Portfolio decision weights. This difference in uncertainty quantification, although visually tolerable, is enough to cause the MM to output a decision which goes against that of the GT: allocate more weight to element 1 and less to element 3. In a stock investment scenario this would correspond to the decision of drastically switching investment confidence from one stock to another.

We conclude this example with numerical results from the Markowitz Portfolio Optimization shown in Table 1. We display the actual Expected Returns, Volatility, and Sharpe Ratio for the resulting weights optimized according to the predictions of the GTs and MMs. Since the GTs are defined according to the true covariance functions of the underlying data generating process, they are our measure of the prediction accuracy given the most accurate uncertainty calibration possible. It is clear that the Mismatched GPs performed worse compared to the Ground Truth GPs in terms of the Sharpe Ratio, and hence resulted in a significantly less attractive risk-adjusted return. We see that the MMs underestimates the risk for the first portfolio element due to improper uncertainty calibration, resulting in a higher weight attribution to this element in the portfolio. Rigorous uncertainty calibration in probabilistic prediction models is therefore of high importance as it could lead to drastic changes in decision output and worse results on our optimization objective.

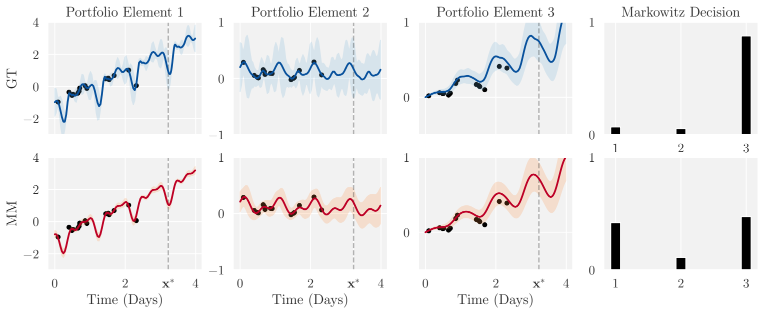

Figure 3: Comparison between the data generating process and the Ground Truth/Mismatched GP predictions for portfolio elements 1, 2, and 3. Y-axis: Price. Right: Markowitz allocation of weights between the three portfolio elements for the GT and MM processes.

	GT	MM
Expected Return	52.5%	57.9%
Volatility	15.3%	24.5%
Sharpe Ratio	3.4	2.4

Table 1: Results from evaluating the Markowitz Portfolio Optimization decisions on time x^* for the Data Generating GP (DGP), the Ground Truth GP (GT), and the Mismatched GP (MM). It is observed that the Sharpe ratio of the MM process is the lowest of the three.

6. Dataset and Evaluation Metrics

6.1. Data

The electricity price dataset from GEFCom2014 [4] recorded 2.5 years of hourly electricity prices, from 2011 to 2013. For each hourly price, the data contains the corresponding zonal load and system load, for an unspecified zone in the power grid. The Gaussian Process model we applied to this problem uses 3-dimensional inputs, namely the time (represented by hour since first hour), the provided Zonal load forecast (one hour ahead), and the provided System load forecast (one hour ahead), to forecast future prices. For the competition, the data from June 19th 2011 to December 17th 2012 is used as training set, and the data from December 18th 2012 to December 17th 2013 is used as testing set for evaluating the submitted models. The time series of price, system loads, and zonal loads are shown in Figure 4.

6.1.1. Data Transformation

In many previous studies outlined in the reference survey [1], before fitting the model to the data, the price is first log-transformed. The log-transformation compresses the extreme values and makes the underlying noise more symmetric and stable. The log-transformation is also applied to the zonal load and to the total load. When making predictions and calculating evaluation metrics, the predicted value is transformed back to the exponential scale. The predictive

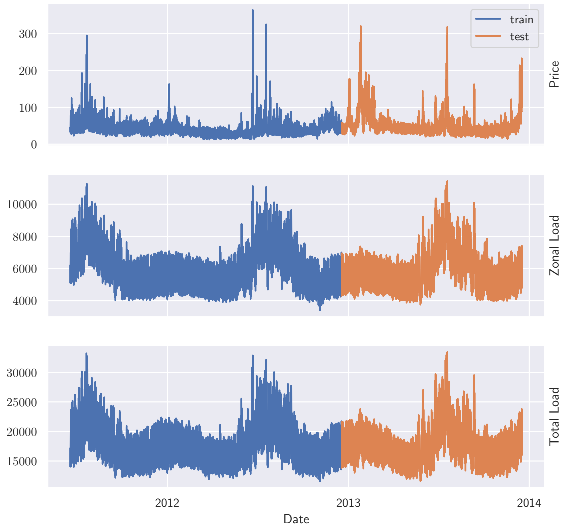

Figure 4: Raw data of Zonal price, Zonal load, and Total load from the GEFCom2014 Dataset

distribution in the non-transformed space thus becomes a Log-Normal distribution, which has positive support with heavier right tail (upwards, to account for the large spikes in the time series). We will discuss how this transformation is incorporated into our model in the Model Design section.

6.2. Evaluation Metrics

The performance of the model is evaluated by a rolling window procedure. For every day in the test set, the preceding days are used to forecast the 24 hourly prices on that day. The number of preceding days used for training, or the calibration window, is unrestricted, with some submissions using only the previous 13 days and some using up to 365 days. The prediction window is rolled forward by one day until the prices for all 365 days in the test set are predicted and scored by the evaluation metrics.

To evaluate the quality of a probabilistic forecast, the metrics should account for both the accuracy of the prediction and the width of the prediction intervals.

Since the true distribution of the underlying data generating process is never observed, we cannot directly compare the predictive distribution to the true distribution. Instead, we use metrics that assess the properties of the predictive distribution, such as quantile predictions and the prediction interval widths. Following the survey, we use Average Coverage Error (ACE), Pinball Loss, and Winkler Loss to evaluate the performance of our model, because these metrics accurately capture both point prediction accuracy and uncertainty around prediction requiring only prediction intervals as input [1]. This also allows us to compare our model to the state of the art and other submissions which only output prediction intervals as opposed to explicit predictive distributions distributions. The metrics are defined and summarised below.

6.2.1. Average Coverage Error (ACE)

The ACE is derived from the *Unconditional Coverage (UC)*, which measures the proportions of actual values captured by the prediction interval.

$$I_t = \begin{cases} 1 & \text{if } P_t \in [\widehat{L}_t, \widehat{U}_t] \\ 0 & \text{if } P_t \notin [\widehat{L}_t, \widehat{U}_t] \end{cases}$$

$$UC = \frac{1}{|T|} \sum_{t \in T} I_t$$

where P_t is the actual value at time t, and I_t is the indicator of whether the prediction interval $[\widehat{L}_t, \widehat{U}_t]$ contains P_t. In other words, this metric asserts that a $X\%$ prediction interval should contain $X\%$ of the observed points. Therefore, ACE is the discrepancy between the percentage of points contained by the PI and the confidence level of the PI, or $UC_\alpha - \alpha$, where α is the confidence level of the PI. However, because the ACE can take on both negative and positive values, we noted that an averaged ACE score can be close to zero even though individual scores are not. Therefore, the average ACE score is only a measure of *reliability*, or the *unbiasedness* of the prediction interval. In this paper we will consider both the $\alpha = 50\%$ PI ACE and the $\alpha = 90\%$ PI ACE.

6.2.2. Pinball Loss

The Pinball loss is a metric on the *sharpness* of the predictive distribution, which evaluates the precision of the prediction and how tightly the prediction interval wraps around the true value. It is defined as follows:

$$\text{Pinball}(q,t) = \begin{cases} (1-q)(\widehat{Q}_t(q) - P_t) & \text{for } P_t < \widehat{Q}_t(q) \\ (q)(P_t - \widehat{Q}_t(q)) & \text{for } P_t \geq \widehat{Q}_t(q) \end{cases}$$

Where $\widehat{Q}_t(q)$ is the predicted q^{th} quantile at time t. The pinball loss is calculated for every t and averaged. The averaged pinball loss is then calculated by averaging the pinball loss for 99 percentiles. If the prediction is close to the true value and the prediction interval is narrow, then the Pinball loss will be low.

6.2.3. Winkler Loss

Similar to the *Pinball loss*, the *Winkler Loss* also measures the *sharpness* of the predictive distribution.

$$\text{Winkler}(t,\alpha) = \begin{cases} \delta_t & \text{for } P_t \in [\widehat{L}_t, \widehat{U}_t] \\ \delta_t + \frac{2}{\alpha}(\widehat{L}_t - P_t) & \text{for } P_t < \widehat{L}_t \\ \delta_t + \frac{2}{\alpha}(P_t - \widehat{U}_t) & \text{for } P_t > \widehat{U}_t \end{cases} \tag{5}$$

where δ is the width of the interval $= \widehat{U_t} - \widehat{L_t}$, and α is equal to $1 - confidence$. The second term in the Winkler score penalises prediction intervals that fail to contain the actual values. The term α scales the penalty term by its confidence level. The Winkler loss is averaged across all time t, and a lower Winkler loss indicates a more favourable prediction.

7. Model Design

In this section, we outline the design of our Gaussian Process model for forecasting electricity price time series data. We recall that our model takes as inputs time, zonal load, and total load, stored as a 3-dimensional matrix. The prior mean of our GP is taken to be the zero function, and we begin our exposition by describing our kernel design. We then perform a sanity check that our model design makes sense through an analysis on model decomposition, along with properly validating our model with a chosen cross-validation scheme. Finally, as outlined in the previous section, we describe how we transform the GP back into the real (not log transformed) space to output our final Log-Normal process over time and load on which we calculate our final test metrics.

7.1. Kernel Design

As discussed in Section 4, the kernel of a GP determines the properties of the functions that it generates, and allows us to encode assumptions about the underlying data in the form of expert knowledge. After fitting, kernel design also plays a large part in rendering our final GP model interpretable, as we will see in the Model Decomposition section. Kernel design is therefore an important step in Gaussian Process model design, and a crucial component of the design process where one can inject domain knowledge into the model. We first introduce a few kernels that are useful in modelling electricity price data, and then we explain how to use a composition of these kernels to build the kernel for our model.

7.1.1. Squared Exponential Kernel

The Squared Exponential Kernel, κ_{EQ}, is a kernel which encodes a high degree of smoothness in the function space, and hence can be used to design a GP which produces function samples that are smooth.

$$\kappa_{EQ}(x_i, x_j) = \sigma^2 \text{exp}\left(-\frac{\|x_i - x_j\|}{2\ell^2}\right) \tag{6}$$

7.1.2. Locally Periodic Kernel

A locally periodic kernel, κ_{LPer}, is a product of Squared Exponential (κ_{EQ}) and periodic kernel (κ_{Per}).

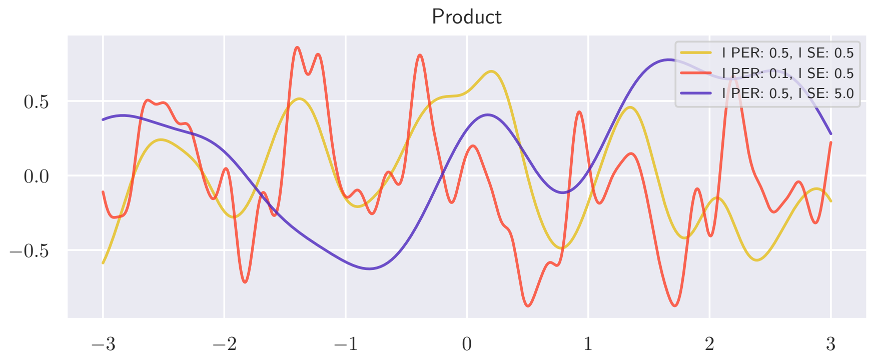

Figure 5: Examples of Locally Period kernels with different periodic and squared exponential lengthscales

$$\kappa_{LPer} = \sigma^2 \exp\left(-\frac{2\sin^2(\pi(x-x')/p)}{\ell_{per}^2}\right)\exp\left(-\frac{(x-x')^2}{\ell_{eq}^2}\right) \tag{7}$$

A GP with this kernel can generate functions that change their periodic structure over time. The periodic component correlates points that are far away from each other but still in the same phase of a cycle, whereas the squared exponential component decorrelates them. The rate at which this structure changes is determined by the hyperparameter ℓ_{eq}. Smaller ℓ_{eq} corresponds to faster changing periodic structures. This kernel is especially useful when modelling electricity price time series, as they have repeating daily cycles that change shape over time. Samples of LPER kernel with different ℓ_{eq} and ℓ_{per} are illustrated in Figure 5.

Another important hyperparameter in this kernel is the period. We will show how we can see the autocorrelation function as an empirical estimate of our kernel, and use this to guide our choice of hyperparameters. Indeed, the autocorrelation of a stochastic process $\mathbf{X}_t$ indexed at t is equal to the expectation of the process at two different time points: $R_{\mathbf{X}}(t_1, t_2) = \mathbb{E}[\mathbf{X}_{t_1}\mathbf{X}_{t_2}]$. For a stationary stochastic process, the autocorrelation can be equivalently parametrized by a lag quantity $\tau = t_2 - t_1$ for all t_1, t_2 in the index set (as, by definition, the

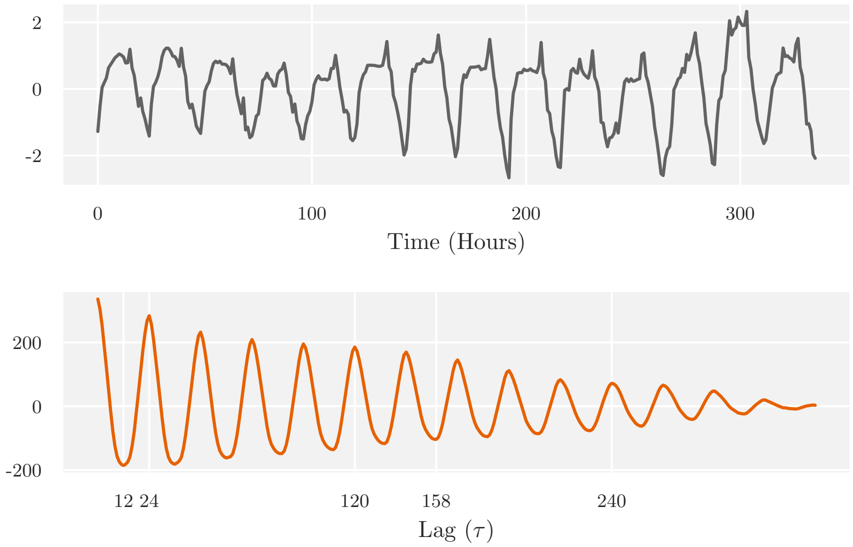

Figure 6: Top: Time Series plot, time in units of hours. Bottom: Autocorrelation plot for different lags τ in units of hours.

autocorrelation of this process only depends on this lag quantity) and reduces to: $R_{\mathbf{X}}(\tau) = \mathbb{E}[\mathbf{X}_{t+\tau}\mathbf{X}_t]$ for all t. We display the autocorrelation plot in Figure 6, where we plot the autocorrelation of a random two week sample of training data with respect to the lag parameter τ. From the analysis of the peaks and troughs of the autocorrelation plot, we notice that peaks and troughs occur at regular 12 and 24 hour intervals, indicating a 12 and 24 hour periodicity. Specifically, we see a positive autocorrelation at lag multiples of 24 hours, and negative autocorrelation at the remaining lag multiples of 12 hours. Additionally, we noticed the decay in autocorrelation as lag τ increases, which reaffirms the aptness of the locally periodic kernel as oppose to just a periodic kernel. Therefore, we incorporate the 12 and 24 hour locally periodic kernels with fixed periods as the first component of the proposed composite kernel.

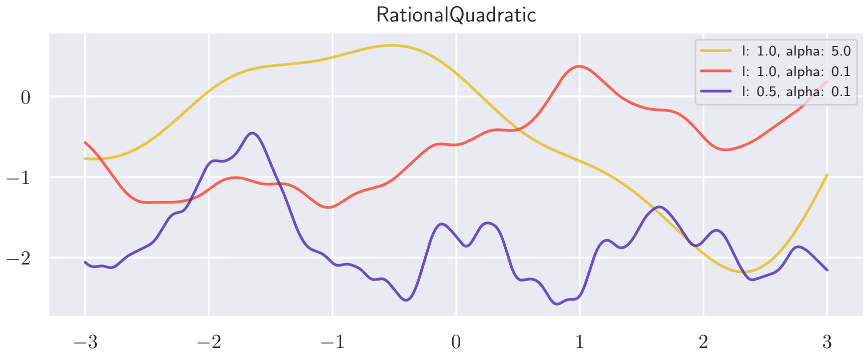

Figure 7: Examples of Rational Quadratic kernels with different lengthscales and scale mixture

7.1.3. Rational Quadratic Kernel

The Rational Quadratic kernel can be interpreted as the sum of many EQ kernels of different lengthscales.

$$\kappa_{RQ} = \sigma^2 \text{exp}\left(1 + \frac{(x-x')^2}{2\alpha\ell^2}\right)^{-\alpha} \tag{8}$$

Similar to an SE kernel, the RQ kernel encodes smoothness in the function space, but with the additional flexibility of having both local variations and long term variations. The hyperparameter α, also known as the scale mixture, determines how much local variations (from the smaller lengthscales) contribute to the overall variation. We use this kernel to model the non-periodic trends within the data. For example, we can interpret the large lengthscale variation as non-periodic weekly trend and the small lengthscale variation as non-periodic daily trend. Samples of RQ kernel with different α and ℓ are illustrated in Figure 7.

7.1.4. Exponential Kernel

The exponential kernel is also very similar to the SE kernel, but without squaring the norm.

$$\kappa_{Exp} = \sigma^2 \text{exp}\bigg(- \frac{|x - x'|}{2\ell^2} \bigg) \tag{9}$$

We use this kernel to model any remaining trends missed from the locally periodic kernels and RQ kernel.

7.1.5. Kernels for Load

Lastly, we use a two dimensional EQ kernel on the loads, with independent lengthscales on each load. This is also known as the ARD kernel.

$$\kappa_{ARD}(\mathbf{x}, \mathbf{x}') = \sigma^2 \text{exp}\bigg(- \frac{1}{2}(\mathbf{x} - \mathbf{x}')^T \Sigma^{-1} (\mathbf{x} - \mathbf{x}') \bigg) \tag{10}$$

Where Σ is diagonal. The correlation between the loads and the price is evident, so this kernel will be very important in the predictive performance.

7.1.6. Kernel Composition

Adding the kernels allows us stack their properties together. In addition to the two locally periodic kernels with period 12 and 24, we have a Rational Quadratic (RQ) kernel and an Exponential (Exp) kernel to model any local and non-periodic trends in the data. By having κ_{LPer}, κ_{RQ}, κ_{Exp}, we have the elements to build a GP that could model different periodic, non-periodic, short term and mid/long term temporal trends in the electricity price data. Lastly, the EQ kernel over load models the relationships between the price and the two different loads, which we selected by observing the plots of total and zonal load against price that hint at a smooth underlying function. We combine all of these kernels through the addition operation, which can be seen as establishing a correlation between two time and load points if any of the component kernels indicate a high correlation at those points. This gives us the following kernel:

$$\kappa_{LPer24}(t) + \kappa_{LPer12}(t) + \kappa_{RQ}(t) + \kappa_{Exp}(t) + \kappa_{EQ}(\text{load}) \tag{11}$$

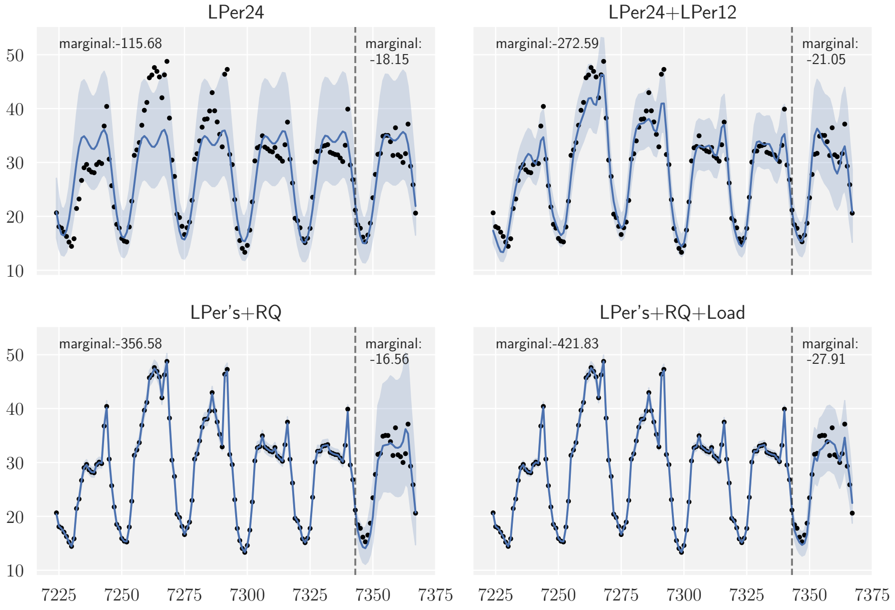

Figure 8: Iterative kernel design fits. The dotted line separates the training data from the validation data. The fit of the training data and the predictive performance is measured by log marginal likelihood(LML).

To illustrate the effect each kernel component has on the composite kernel, we fit the data with cumulatively more complex kernels as shown in Figure 8. Starting with the κ_{LPer24} kernel, the model is able to capture the daily repeating patterns with slight variations in from cycle to cycle. However, its lack of complexity is reflected by the large uncertainty in the training data fit, as well as the inability to fit more dramatic daily variations. When κ_{LPer12} is added, the uncertainty decreases significantly as a result of having more parameters and thus greater flexibility. Nevertheless, the prediction intervals (uncertainty to the right of dotted line) is still very wide, indicating poor extrapolation. Adding κ_{RQ} and κ_{Exp} reduces the uncertainty significantly, as they capture the non-periodic trends and irregularities in the data. Finally, the SE kernel on load input proves to be an crucial addition to the composite kernel, especially in prediction, as it significantly reduces the LML and the width of the prediction interval.

In general, increasing the complexity of the model improves both the model fit and the predictive performance. However, more complex kernel design may also lead to overfitting and improper uncertainty estimation. To verify the model fitness and refine the model, we analyze the predictions by decomposing each kernel's contribution, and conduct cross-validation experiments on different model designs.

7.2. Sanity Check for Model Design

Since the kernel of the GP model is composed of additive parts, the posterior distribution after conditioning can be decomposed into sums of individual Gaussian Processes. This allows us to visualize the contribution of each kernel component to the prediction and interpret the predictive components of our model, highlighting the interpretability of the Gaussian Process as a predictive model. The mathematical formulation of kernel decomposition is outlined in Appendix 10.2. We show how we can use this property to diagnose overfitting and improve upon the proposed model.

In Figure 9, we observe the breakdown of the posterior distribution into each

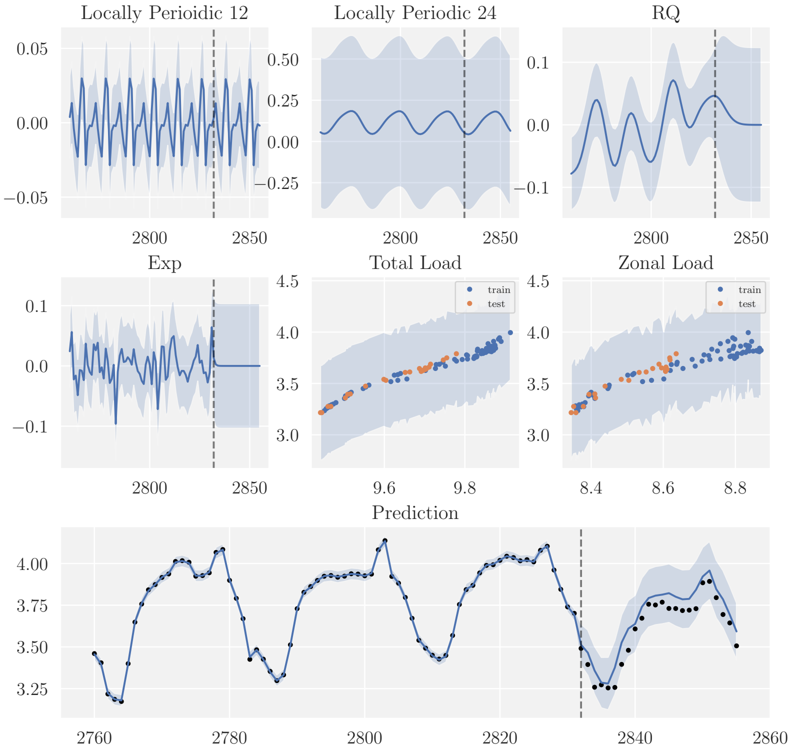

Figure 9: Decomposition of kernels into individual components. This plot shows that posterior distribution of GPs with individual component of kernel conditioned on the observed data(log-transformed). For the load kernels, the testing inputs are scattered among the training inputs. Hence, in prediction, prediction with this component GP is interpolation instead of extrapolation.

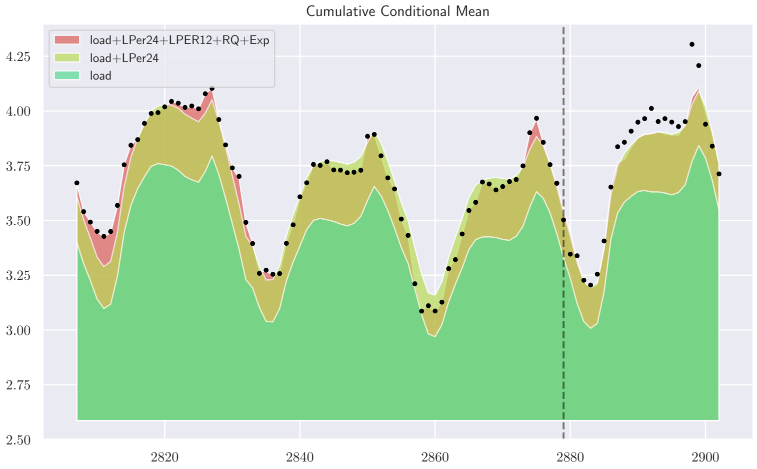

Figure 10: Cumulatively adding conditional mean from each kernel. The area below each fitted line is highlighted to show the additive effect of each kernel.

kernel component. We examine the shapes of individual posterior distributions and compare them when feeding training data and test data. The κ_{LPer24} kernel models the daily cycle while the κ_{LPer12} kernel models the more fine-grained hourly oscillation. The κ_{RQ} kernel models the overall trend and the κ_{Exp} kernel accounts for any additional irregularities. Moreover, the κ_{eq}(load) kernel takes in load data as input, which is not a time series, so the test inputs are interpolations. Furthermore, because the prediction posterior mean is the sum of the posterior means from each kernel components, as illustrated in Figure 10, we can observe the signal contribution from each component. The κ_{eq}(load) and κ_{LPer24} kernels supply the main signal, while the other kernels supply fine tuning information and periodicity.

For forecasts, the conditional mean and variance of each component can be interpreted as their respective contribution to the predictive mean and variance. For example, the mean and variance of the two κ_{LPer} kernels hardly vary between training and validation data. This is consistent with the extrapolating properties of periodic kernels, since far away points can be considered highly correlated if they are close in position within their respective cycle. The EQ kernel for load is also a main contributor for prediction, since the validation inputs are scattered among the training inputs. This allows the kernel to interpolate for predictions, which is a task that the EQ kernel excels at due to it inducing a smooth function space. On the other hand, the uncertainty from κ_{RQ} kernel becomes greater the further the test data is away from the training data. Although for training input it captures a clear trend with small variance, in prediction the posterior slowly reverts to zero mean with large variance. Lastly, we look at the κ_{Exp} kernel. The κ_{Exp} kernel demonstrates extreme behaviours in both training data and testing data. In training it is very wiggly with small uncertainty, but it does not capture any recognizable trend. In prediction it very quickly reverts to zero mean and large variance. This indicates that the exponential kernel has minimal contribution to the prediction besides increasing uncertainty. Therefore, we conclude that the inclusion of this kernel does not improve predictive precision, and its contribution to the variance can be replaced by adding additional noise to the GP. Indeed, when we add a fixed variance of $\sigma^2 = 0.005$ on the GP, we observe the new decomposition in Figure 11, in which the exponential kernel is simply white noise.

7.3. Optimization and Cross Validation

The optimization of a GP model is done through maximizing the marginal likelihood $(P(\mathcal{D}|M))$ with respect to the hyperparameters. We use ADAM [25] as an optimizer for this objective function. While optimizing the marginal likelihood allows us to determine the best-fitting hyperparemeters given a kernel, to compare different kernel design and model design, we resort to conducting cross-validation experiments.

Since the data is a time series, the sequence must be maintained and thus K-fold cross validation cannot be applied. Furthermore, since the training size is fixed, we also cannot use a nested cross validation scheme. Instead we use a bootstrap scheme to sample training and validation data, randomly sampling a date for the start of the validation data and taking the preceding n_{train} en-

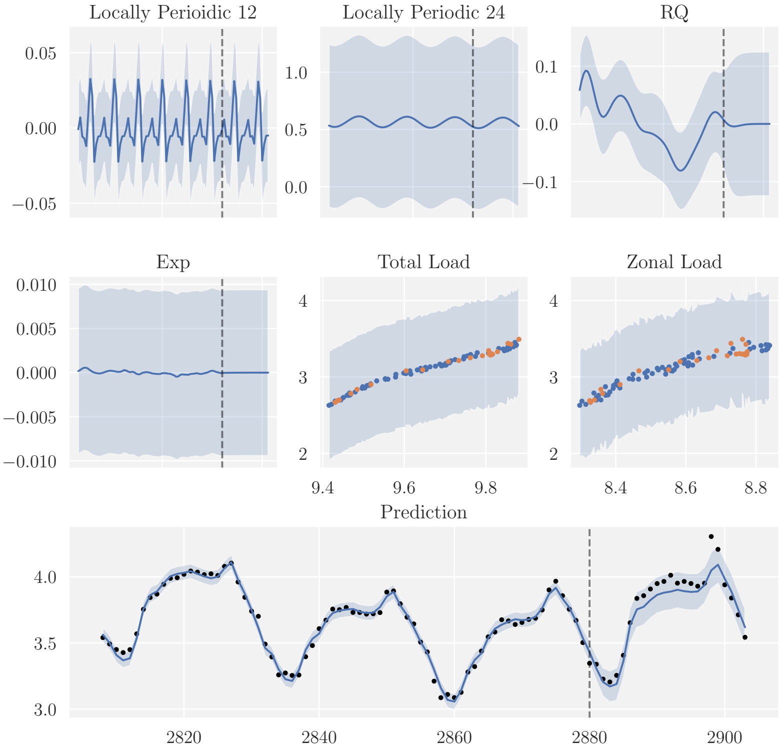

Figure 11: Decomposition of GP model with fixed variance $\sigma^2 = 0.005$. The exponential kernel in this model provides no information to the model and was contributing to overfitting and poor uncertainty calibration.

tries as training data. We use the evaluation metrics defined in section 6.2 as well as the Mean Square Error on the test set to evaluate performance. For the experiments, we test 1) different combinations of kernels and 2) different calibration windows. For kernel design, we found that the proposed composite kernel with κ_{exp} replaced by noise achieves the best performance. Our final kernel and Gaussian Process model can then be summarized as follows:

$$\kappa = \kappa_{LPer24}(t) + \kappa_{LPer12}(t) + \kappa_{RQ}(t) + \kappa_{EQ}(\text{load}) \tag{12}$$

$$\mathbf{y} \sim \mathcal{N}\Big(\boldsymbol{\mu}(X),\ \kappa(X,X) + \sigma_\epsilon^2 \mathbb{I}\Big),\quad \sigma_\epsilon^2 = 0.005 \tag{13}$$

Where our prior mean $\boldsymbol{\mu}$ is taken to be the zero function. For the calibration window, we tested short (7 days), medium (14 days), long (28 days). We found that a 14-day training window yields the best results.

7.4. Model Transformation

While previous works have applied transformation to the data before fitting the model, we describe the equivalent perspective of applying a transformation to the model instead of the data. Our time series price data contains infrequent, very large spikes in price which cause a visual asymmetry in the empirical data distribution. Our use in applying a transformation to the regular GP model is to "squash" the high peaks so that we can fit a vanilla GP in this more symmetric space. We thus continue the Bayesian paradigm by treating transformation as another way to apply prior knowledge on the model. The model can be expressed as follows:

$$\mathbf{y} = e^{f(X)}, \quad f(X) \sim GP(\boldsymbol{\mu}, \mathbf{K}) \tag{14}$$

where $\mathbf{y}$ represents our observed data in the real space, and X our input space (in our case, time and load). That is, we place a GP prior over a latent function $f(X)$, and apply an exponential function to it. As derived in Appendix, in optimization, the marginal likelihood of 14 is equivalent to that of a GP with $\mathbf{y}$ log-transformed. The equations for this closed form reverse transformation are outlined in Appendix 10.3.

8. Experiment Results

With a finalized model, we evaluate its performance on the test set, compare the achieved metrics to the other submissions in the survey paper [1], and inspect the optimized hyperparameters to demonstrate the interpretability of our Gaussian Process model, as mentioned in the introduction.

8.1. Experiment Setup

For each day in the 365 days of the test set, the preceding 14 days are used as input for the model. The hyperparameters of the model are optimized for each window, and the predictive posterior distribution is used to calculate evaluation metrics - ACE score with 50% confidence interval (ACE50), ACE score with 90% confidence interval (ACE90), Pinball Loss, Winkler loss with 50% confidence interval (Winkler 50), and Winkler loss with 90% confidence interval (Winkler 90). The metrics for the 365 test days are then averaged to calculate the final average metrics.

8.2. Inspection of Learned Hyperparameters

Analyzing the optimized value of the hyperparameters (kernel parameters), we can further infer the properties of the model. Namely, the lengthscales of the EQ part of the locally periodic kernels reflect how quickly the locally-repeating structure changes. Comparing the EQ lengthscales ℓ_{eq} between the κ_{LPer12} kernel ($\ell = 448$ hours), and the κ_{LPer24} kernel ($\ell = 2475$ hours), the latter has lengthscales more than 5 times larger than the former. This indicates that the locally repeating patterns change more quickly for the κ_{LPer12} kernel, matching our intuition, as this kernel models the fine-grained variations of the price within a day, which varies more frequently. Another easily interpretable parameter is the variance of each kernel. As discussed previously, the variance is an indicator of the kernel's strength of signal relative to the other kernels. In this model, the main signal is supplied by the κ_{EQ}(load) kernel ($\sigma^2 = 2.29$), followed by the κ_{LPer24} kernel ($\sigma^2 = 0.093$), then κ_{RQ} ($\sigma^2 = 0.0023$) and κ_{LPer12} ($\sigma^2 = 0.0006$). This indicates that the load information is very important for the prediction, the most important of the features. It also indicates that the GP relies more heavily on the 24 hour periodicity than the 12 hour periodicity in the time series data. Note that the hyperparameters converge to different values depending on the training data, so the values provided are only representative examples from a particular run. However, the scale and relative order of the variances is mostly robust.

	Best Result	GP Model
ACE 50% PI	0.08%	-0.34%
ACE 90% PI	-2.56%	-4.25%
Average Pinball Loss	2.634	**2.276**
Winkler Score 50% PI	23.108	**20.386**
Winkler Score 90% PI	50.657	**44.887**
MSE	-	141.78

Table 2: Performance Comparison across Evaluation Metrics. The best results come from the survey paper [1], where mARX-B attains the best ACE50 and ACE90 scores, and QRA(3) attains the best Pinball, Winkler50 and Winkler90 scores.

8.3. Performance Metrics

We compared the performance of the Gaussian Process model using our kernel to the top results from the survey paper [1], summarized in Table 2. The Gaussian Process Model achieves superior performance compared to the the state-of-the-art in Pinball Loss, Winkler Score with 50% prediction interval, and Winkler Score with 90% prediction interval by considerable margins. In reliability metrics (ACE50 and ACE90), the model is competitive, managing to beat the next best scores in ACE50(-0.62% from ARX-B). A negative ACE score suggests underestimation of uncertainty. In this case, a better ACE50 score than ACE90 score could also indicate that the underlying error distribution has fatter tails, which could be mitigated by more sophisticated data transformations such as in [26].

9. Conclusion

In this work we have proposed the Gaussian Process (GP) as a solution to the PEPF problem that is: a) simple and principled with calibrated uncertainties through the Bayesian framework, b) interpretable through its intuitive kernel design and decomposition, c) flexible to include expert knowledge through kernel design. We have demonstrated the above features of the GP model through the use of a decision making toy example, a walkthrough of our design procedure, and finally through experiments on real data. Our constructed model outperformed the previous state-of-the-art with respect to most metrics on the benchmark GEFCom2014 dataset (medium-term forecasting), outlined in [1], and the model is publicly available at `urlnotavailableyet`. Future directions of this work include experimentally veryfing model accuracy on short-term and long-term benchmark datasets in the field. In regards to the latter, it could be fruitful to consider scalable GP approximations such as Sparse GPs [27, 28]. Another avenue of research could be to consider the modelling of power grids in a holistic fashion, where multiple time series such as load and price are modelled together with Gaussian Processes. Such dependent GPs could then be combined into one more informed forecasting for price using techniques such as GPAR [29]. In the case of modelling and outputting multiple variables of the grid, one could consider a multi-output GP which could learn correlations between different input components such as load and price and output multiple component predictions in a scalable way [30]. All in all, we find Gaussian Processes to be a promising avenue not just for probabilistic EPF but probabilistic energy systems modelling as a whole.

References

[1] J. Nowotarski, R. Weron, Recent advances in electricity price forecasting: A review of probabilistic forecasting, Renewable and Sustainable Energy Reviews 81 (2018) 1548 – 1568.

[2] J. G. D. Gooijer, R. J. Hyndman, 25 years of time series forecasting, International Journal of Forecasting 22 (2006) 443 – 473. Twenty five years of forecasting.

[3] P. Pinson, H. Madsen, H. A. Nielsen, G. Papaefthymiou, B. Klöckl, From probabilistic forecasts to statistical scenarios of short-term wind power production, Wind Energy 12 (2009) 51–62.

[4] T. Hong, P. Pinson, S. Fan, H. Zareipour, A. Troccoli, R. J. Hyndman, Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond, International Journal of Forecasting 32 (2016) 896 – 913.

[5] A. Misiorek, S. Trueck, R. Weron, Point and interval forecasting of spot electricity prices: Linear vs. non-linear time series models, Studies in Nonlinear Dynamics Econometrics 10 (2006).

[6] K. Maciejowska, J. Nowotarski, A hybrid model for gefcom2014 probabilistic electricity price forecasting, International Journal of Forecasting 32 (2016) 1051 – 1056.

[7] C. GarcÃa-Martos, J. RodrÃguez, M. J. SÃ¡nchez, Forecasting electricity prices and their volatilities using unobserved components, Energy Economics 33 (2011) 1227 – 1239.

[8] A. M. Alonso, C. GarcÃa-Martos, J. RodrÃguez, M. J. SÃ¡nchez, Seasonal dynamic factor analysis and bootstrap inference: Application to electricity market forecasting, Technometrics 53 (2011) 137–151.

[9] A. Brusaferri, M. Matteucci, P. Portolani, A. Vitali, Bayesian deep learning based method for probabilistic forecast of day-ahead electricity prices, Applied Energy 250 (2019) 1158 – 1175.

[10] H. Mori, M. Ohmi, Probabilistic short-term load forecasting with gaussian processes, volume 2005, 2005, p. 6 pp. doi:10.1109/ISAP.2005.1599306.

[11] H. Mori, K. Nakano, Epso-based gaussian process for electricity price forecasting, in: 2015 IEEE Congress on Evolutionary Computation (CEC), 2015, pp. 291–296. doi:10.1109/CEC.2015.7256904.

[12] M. Alamaniotis, N. Bourbakis, L. H. Tsoukalas, Very-short term forecasting of electricity price signals using a pareto composition of kernel machines in smart power systems, in: 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2015, pp. 780–784. doi:10.1109/GlobalSIP.2015.7418303.

[13] H. S. Sandhu, L. Fang, L. Guan, Forecasting day-ahead price spikes for the ontario electricity market, Electric Power Systems Research 141 (2016) 450–459.

[14] P. Damien, R. Fuentes-GarcÃa, R. H. Mena, J. Zarnikau, Impacts of day-ahead versus real-time market prices on wholesale electricity demand in texas, Energy Economics 81 (2019) 259 – 272.

[15] F. Ziel, R. Steinert, Probabilistic mid- and long-term electricity price forecasting, Renewable and Sustainable Energy Reviews 94 (2018) 251 – 266.

[16] Z. Ghahramani, Bayesian non-parametrics and the probabilistic approach to modelling, Philosophical transactions. Series A, Mathematical, physical, and engineering sciences 371 (2012) 20110553–20110553.

[17] W. H. Jefferys, J. O. Berger, Ockham's razor and bayesian analysis, American Scientist 80 (1992) 64–72.

[18] C. E. Rasmussen, Z. Ghahramani, Occam's razor, in: T. K. Leen, T. G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems 13, MIT Press, 2001, pp. 294–300. URL: http://papers.nips.cc/paper/1925-occams-razor.pdf.

[19] D. MacKay, Information theory, inference, and learning algorithms, Cambridge University Press, Cambridge, UK New York, 2003.

[20] P. Orbanz, Y. W. Teh, Bayesian nonparametric models, in: Encyclopedia of Machine Learning, Springer, 2010.

[21] D. Duvenaud, The kernel cookbook:, 2014. URL: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html.

[22] W. F. Sharpe, The sharpe ratio, Journal of portfolio management 21 (1994) 49–58.

[23] C. Rasmussen, Gaussian processes for machine learning, MIT Press, Cambridge, Mass, 2006.

[24] J. Francis, Modern portfolio theory : foundations, analysis, and new developments + website, Wiley, Hoboken, N.J, 2013.

[25] D. P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, arXiv e-prints (2014) arXiv:1412.6980.

[26] E. Snelson, Z. Ghahramani, C. E. Rasmussen, Warped gaussian processes, in: S. Thrun, L. K. Saul, B. Schölkopf (Eds.), Advances in Neural Information Processing Systems 16, MIT Press, 2004, pp. 337–344. URL: http://papers.nips.cc/paper/2481-warped-gaussian-processes.pdf.

[27] E. Snelson, Z. Ghahramani, Sparse gaussian processes using pseudo-inputs, in: Proceedings of the 18th International Conference on Neural Information Processing Systems, NIPS'05, MIT Press, Cambridge, MA, USA, 2005, p. 1257–1264.

[28] M. Bauer, M. van der Wilk, C. E. Rasmussen, Understanding probabilistic sparse gaussian process approximations, in: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 1533–1541. URL: http://papers.nips.cc/paper/6477-understanding-probabilistic-sparse-gaussian-process-approximations.pdf.

[29] J. Requeima, W. Tebbutt, W. Bruinsma, R. E. Turner, The gaussian process autoregressive regression model (gpar), in: K. Chaudhuri, M. Sugiyama (Eds.), Proceedings of Machine Learning Research, volume 89 of *Proceedings of Machine Learning Research*, PMLR, 2019, pp. 1860–1869. URL: http://proceedings.mlr.press/v89/requeima19a.html.

[30] W. P. Bruinsma, E. Perim, W. Tebbutt, J. S. Hosking, A. Solin, R. E. Turner, Scalable exact inference in multi-output gaussian processes, 2019. arXiv:arXiv:1911.06287.

10. Appendix A: Gaussian Processes

Our bold notation $\mathbf{x}$ indicates that $\mathbf{x}$ is a vector, non-bold $f(\mathbf{x})$ indicates that f is a scalar output function receiving a vector input, and scalar functions receiving two vectors as input will follow the notation $\kappa(\mathbf{x}, \mathbf{x}')$ and denote the Gram matrix with entries (i, j) taking on values $\kappa(x_i, x_j)$. We proceed to describe the Gaussian Process:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), \kappa(\mathbf{x}, \mathbf{x}')), \tag{15}$$

where $m(\mathbf{x})$ is the *mean function*:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{16}$$

and $\kappa(\mathbf{x}, \mathbf{x}')$ is *kernel* or *covariance function*:

$$\kappa(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \tag{17}$$

and where any collection of function values of size n follows a joint Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{K}$. These are expanded as follows:

$$[f(x_1), f(x_2), \ldots, f(x_n)]^T \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}), \tag{18}$$

$$\boldsymbol{\mu} = [m(x_1), m(x_2), \ldots, m(x_n)]^T, \tag{19}$$

$$\mathbf{K_{i,j}} = \kappa(x_i, x_j). \tag{20}$$

A Gaussian Process with a given *mean function* m and *covariance function* κ is a prior that can generate functions. One can define the properties of the desired function $f(\mathbf{x})$ through the design of m and κ. The former is simply a function that encodes the mean of the multivariate Gaussian distribution, and in most cases is kept simple with a zero function or constant function. The latter, however, allows the user to more flexibly encode prior information about the data that can be useful for prediction and inference. More intuitively, a kernel can be seen as a similarity measure between two inputs. For example, consider the *Squared Exponential (EQ)* kernel defined by the following expression.

$$\kappa_{EQ}(x_i, x_j) = \sigma^2 \text{exp}\left(-\frac{\|x_i - x_j\|}{2l^2}\right) \tag{21}$$

where σ^2 and l are positive variance and lengthscale parameters respectively. The variance parameter encodes prior information about how wide the range of values the functions can take. It can also be interpreted as the power of its underlying signal ($\int f(x)^2 dx$), observed by noting that for $x = x'$, $\kappa_{EQ}(x, x') = \sigma^2$, and hence the diagonals of our corresponding covariance matrix for our GP (the Gram matrix of the covariance function) will be equal to σ^2. The lengthscales determines the smoothness of the functions. A larger l means that for two inputs x, x' that are further apart, the kernel still treats them as correlated points, which results in smoother functions. Conversely, a smaller l corresponds to smaller covariance between two points, which leads to more random fluctuations. Figure 2 shows the functions generated by a GP prior with κ_{EQ} with varying variance and lengthscales.

Other kernel functions can encode different prior information about the generated functions. The *Periodic Squared Exponential (Per)* kernel can provide prior information as to the periodicity of the underlying process and can be used for modelling cyclical time series data:

$$\kappa_{Per}(x_i, x_j) = \sigma^2 \text{exp}\left(-\frac{2\sin^2(\pi|x_i - x_j|/p)}{l^2}\right) \tag{22}$$

This kernel can also be seen as applying a *periodicity* operator to the regular EQ kernel (see [23]). Similarly, periodic kernels with different base structures can be created by applying the same *periodicity* operator onto another stationary kernel such as the *Matern 3/2* kernel [23]. Such a transformation endows the kernel with a periodic structure, while still maintaining the underlying smoothness qualities of the function space.

The *Matern 1/2 (MA)* kernel encodes potential discontinuities in the sample path of functions f sampled from our process and can be used for modelling functions that are less smooth. Sample functions drawn from GPs with these kernels are also shown in Figure 2.

$$\kappa_{MA}(x_i, x_j) = \sigma^2 \text{exp}\left(-\frac{|x_i - x_j|}{\ell^2}\right) \tag{23}$$

This family of kernels, the Matern kernels, come in different flavours depending on the level of desired smoothness in the function space. Variants include the Matern 3/2 kernel:

$$\kappa_{MA32}(x_i, x_j) = \sigma^2 \left(1 + \frac{\sqrt{3}|x_i - x_j|}{\ell^2}\right) \exp\left(-\frac{\sqrt{3}|x_i - x_j|}{\ell^2}\right) \tag{24}$$

which induces a function space with smoother sample paths. Simpler covariance functions also exist, such as the Linear kernel, with correlations between points decaying linearly in their separation distance:

$$L(x_i, x_j) = \sigma_a^2 + \sigma_b^2(x_i - c)(x_j - c) \tag{25}$$

where σ_a^2 and σ_b^2 are hyperparameters.

Moreover, the addition and multiplication of kernels results a valid kernel (where a kernel is valid if it is a positive-definite function of its two inputs [23]). Therefore, covariance functions can be combined through multiplication and addition to produce more complex covariance structures. The addition of two kernels (κ_1 and κ_2) is analogous to an OR operation: two points t_1 and t_2 are considered highly correlated if they are highly correlated in either κ_1 or κ_2 ($\kappa_1(t_1, t_2)$ or $\kappa_2(t_1, t_2)$ is a large in magnitude). Conversely, the multiplication of two kernels is analogous to an AND operation: two points t_1 and t_2 are considered highly correlated if they are highly correlated both in κ_1 and κ_2 ($\kappa_1(t_1, t_2) \cdot \kappa_2(t_1, t_2)$ is large in magnitude). With a compositional kernel, the variance σ^2 parameter of a each component kernel can also be interpreted as the strength of its signal.

10.1. Inference with Gaussian Process

To make predictions from input $\mathbf{x}^*$ with Gaussian Processes given observations $\{(x_i, y_i)\}$ assuming i.i.d additive Gaussian noise with variance σ_n^2, the prediction and observations are expressed as a joint distribution following the GP prior,

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}^* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{\mu}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I & K(\mathbf{X}, \mathbf{X}^*) \\ K(\mathbf{X}^*, \mathbf{X}) & K(\mathbf{X}^*, \mathbf{X}^*) \end{bmatrix}\right) \tag{26}$$

and from the joint distribution one can derive the conditional distribution,

$$\begin{aligned} \mathbf{y}^*|\mathbf{y}, \mathbf{X}, \mathbf{X}^* &\sim \mathcal{N}(K(\mathbf{X}^*, \mathbf{x})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2]^{-1}(\mathbf{y} - \boldsymbol{\mu}), \\ \bar{K} &= K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X}))\Big(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2\Big)^{-1} K(\mathbf{X}, \mathbf{X}^*)) \end{aligned} \tag{27}$$

The model's parameters(kernel parameters) can be tuned by optimizing the marginal log likelihood :

$$\log P(y|\mathbf{X}, \theta) = -\frac{1}{2} y^T (\bar{K} + \sigma_n^2 I)^{-1} y - \frac{1}{2} \log |\bar{K} + \sigma_n^2| - \frac{n}{2} \log 2\pi \tag{28}$$

Where the first term can be interpreted as a data fit term, and the second term as a complexity penalty (note the similarity with the entropy of a Gaussian distribution: $H = \frac{1}{2} \ln |2\pi e K|$). In the case where the kernel of the GP consists of multiple components, optimizing the log marginal likelihood automatically selects the importance of each kernel component by its variance hyperparameter. All kernel hyperparameters are optimized, including the likelihood variance σ_n^2, which encodes our estimate of Gaussian noise in the data generating process.

10.2. Decomposing Posterior Distribution into Kernel Components

As described in the inference section above, the posterior mean and variance of the GP is as follow,

$$\bar{\mu} = K(\mathbf{X}^*, \mathbf{x})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2]^{-1}(\mathbf{y} - \boldsymbol{\mu}) \tag{29}$$

$$\bar{K} = K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X})\Big(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2\Big)^{-1} K(\mathbf{X}, \mathbf{X}^*) \tag{30}$$

Suppose our kernel κ is composed of D components, $\kappa = \kappa_1 + \kappa_2 + \dots \kappa_D$. Then we can decompose the posterior mean and variance of the full GP into the posterior mean variance of GPs with individual kernel component. We use $\bar{\mu}_d$ and $\bar{K}_d$ to describe the GP with kernel component κ_d.

$$\bar{\mu}_d = K_d(\mathbf{X}^*, \mathbf{x})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2]^{-1}(\mathbf{y} - \boldsymbol{\mu}) \tag{31}$$

$$\bar{K}_d = K_d(\mathbf{X}^*, \mathbf{X}^*) - K_d(\mathbf{X}^*, \mathbf{X})\Big(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2\Big)^{-1} K_d(\mathbf{X}, \mathbf{X}^*) \tag{32}$$

With these expressions, we can see that $\bar{\mu} = \sum^D \bar{\mu}_d$. Therefore, the posterior mean of the full model is the sum of the posterior means of the independent models. However, the covariance matrix does not have the same linear structure, hence it is harder to infer the relationship between the full covariance matrix and its components.

10.3. Log-Normal Gaussian Process

For a random variable $\mathbf{Z}$ following a multivariate Normal distribution, if $\mathbf{Y} = e^{\mathbf{Z}}$, then $\mathbf{Y} \sim \text{LogNormal}(\boldsymbol{\mu}, \mathbf{K})$. Having trained a Gaussian Process over log-transformed data, we can view it analogously as a multivariate Gaussian $Z \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ over log-transformed data, and we can transform our model back into the real space to obtain a Log-Normal process $\mathbf{Y} \sim \text{LogNormal}(\widetilde{\boldsymbol{\mu}}, \widetilde{\mathbf{K}})$ over our data, where:

$$\widetilde{\boldsymbol{\mu}}_j = e^{\boldsymbol{\mu}_i + \frac{1}{2}\mathbf{K}_{ii}} \tag{33}$$

$$\widetilde{\mathbf{K}}_{ij} = e^{\boldsymbol{\mu}_i + \boldsymbol{\mu}_j + \frac{1}{2}(\mathbf{K}_{ii} + \mathbf{K}_{jj})}(e^{\mathbf{K}_{ij}} - 1) \tag{34}$$

where i, j represent indices in the mean and covariance matrices of the distribution.

The log marginal likelihood of a multivariate log normal distribution is simply the LML of a Gaussian transformed by log(Y) subtracting the sum of y_i.

$$\begin{aligned} \log P(y|\mathbf{X}, \theta) = &-\frac{1}{2}(\log y - \mu)^T K^{-1}(\log y - \mu) - \\ &\frac{1}{2}\log|K| - \frac{n}{2}\log 2\pi - \sum log y_i \end{aligned} \tag{35}$$

Therefore, optimizing the LML for a log normal distribution is equivalent to optimizing the LML for a normal distribution with log-transformed signal, since the last term in 35 does not involve any parameters.