Electricity Demand Forecasting with Gaussian Process Regression

Runar Skagestad

Master of Science in Physics and Mathematics
Submission date: July 2018
Supervisor: Bob Ohara, IMF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Sammendrag

For aktører i energiindustrien er det svært viktig å ha tilgang til pålitelige prognoser for fremtidig energietterspørsel. Disse bør nødvendigvis klare å tilpasse seg lokale, flyktige forandringer så vel som vesentlige endringer i selve forbrukslandskapet. I denne masteroppgaven foreslås en løsning for predikering av energiforbruk som er basert på gaussisk prosess-regresjon. For å gjøre rede for de ulike valgene som er tatt i utforming av de prediktive modellene, gis en grundig bakgrunnsteori for langtidsprediksjon av tidsrekker ved bruk av gaussiske prosesser. Modellene testes på virkelige data for energiforbruk innenfor de tidsrammene som er pålagt aktører i det norske energimarkedet, og dens prognoser overgår en metode som i dag benyttes kommersielt til samme formål. I tillegg har modellen mange nyttige egenskaper, slik som fravær av brukerdefinerte parametere, automatisk estimering av usikkerhet og rask kjøretid.

Abstract

For participants in the energy industry, it is vital to have access to reliable forecasts of future energy demands. The predictive routines should necessarily cope with local, transient fluctuations as well as considerable changes in the electricity consumption landscape. In this thesis, a solution for electricity demand forecasting based on Gaussian process regression is presented. To account for the choices taken in the process of designing functional predictive models, a thorough background theory for long-term time-series forecasting with Gaussian processes is established. The models are tested on a real-world data set, and while complying with the limitations that apply for participants in the Norwegian energy market, the results were found superior to a commercially employed model designed for the same task. The models have, in addition, many useful properties such as no user-defined parameters, a quantification of predictive uncertainties, and low time complexity.

Preface

This Master's thesis is submitted to the Norwegian University of Science and Technology (NTNU), and completes my Master's of Science at the Department of Mathematical Sciences (IMF). This thesis is written in cooperation with Nord-Trøndelag Elektrisitetsverk (NTE) under supervision of Christian Oshaug. They have kindly provided me with the necessary data and a very rewarding task. The work has been carried out during the spring semester of 2018 under the official title TMA4900, and builds on a specialisation project of autumn 2017. Bob O'Hara has been my supervisor at IMF, and I am grateful for him always having his office door open and for open-mindedly listening to my thoughts. I want to express a deep gratitute towards Siri Fatnes, who has devoted much of her time helping and encouraging me during the final weeks.

Runar Skagestad
Trondheim, July 2018

Contents

List of Figures

1 Introduction

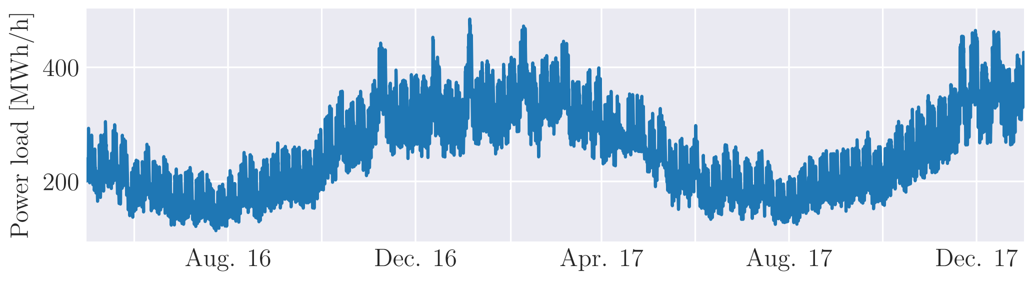
In the control and scheduling of power systems, having proper estimates of future electricity demands are of utmost importance. The same holds true for brokers in the wholesale energy market, who typically buy or sell energy at future contracts. For technical reasons, supply and demand must be at close equilibrium at any time. If failing to accurately forecast future demand, the balance must be upheld by trading the difference. Both in case of energy deficit and surplus, prices are typically unfavorable compared to the wholesale market. Inaccurate predictions can thus cause severe financial penalties, but predicting electricity demands at long time-horizons is a problem of advanced statistical nature. In this theses, we will discuss relevant aspects of time-series theory in the field of Gaussian processes, and apply the theory to a data set provided by *Nord-Trøndelag Elektrisitetsverk* (NTE), to test the performance of the suggested models.

NTE is one of the largest electric utilities in Norway, operating mainly in the county of Nord-Trøndelag. Participants in the Norwegian energy market, such as NTE, must report to Nord Pool[1] every day before 10:00 and account for the quantities of electricity that they wish to trade at each hour of the full next day. Developing forecasting models which function optimally within this time frame is therefore of great interest to NTE, and to this aim, they have provided historical data of energy consumptions in Nord-Trøndelag from 2011 up to and including 2017. Figure 1.1 shows sections of this data.

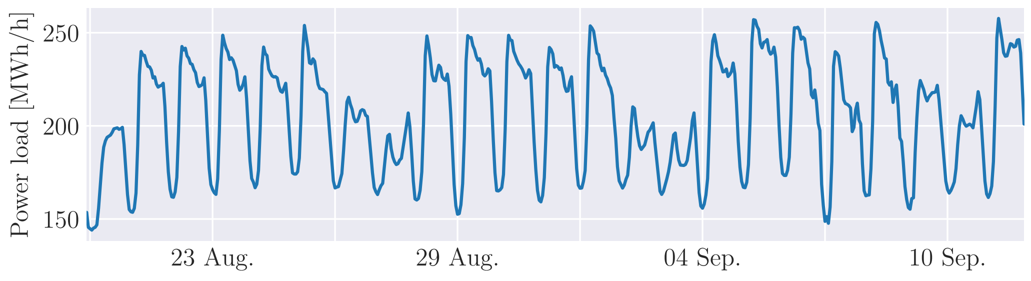
The electricity demand is primarily dependent on temporal factors such as daily and weekly cycles, but are also strongly influenced by meteorological conditions [1]. For short time-frames, univariate forecasting models have been successfully applied, including exponential smoothing methods [2] and (S)ARMA models [3]. For lead times beyond one day ahead, it is crucial to directly allow for weather-induced variations [4]. Such methods include multiple regression [5] and Kalman filters [6], to mention a few. Recently, there has been a surge of interest in artificial-intelligence methods such as recurrent neural networks [7] and support-vector regression [5], among a host of others. For an extensive review of classical as well as modern approaches, see Srivastava et. al. [8].

The current forecasting routine used by NTE and other electric utilities is a commercial Kalman-filter-based model, of which details are unavailable to the public. However, the provided dataset contains—in addition to true values—historical predictions from the model that can be compared with other alternatives. In this thesis, we will consider a solution to the forecasting problem by Gaussian process (GP) regression, which belongs to a class of methods whose learning is based on probabilistic inference. Simply put, these methods take a group of hypotheses and weights them based on how well their predictions match some given data, yielding a more refined group. The specification of the hypothesis group, which makes up a *model*, depends on how much *a priori* information that can be inferred from the system under analysis. A recent rise in interest for GPs was

[1]A major European energy exchange market. Web: `www.nordpoolgroup.com/`



(a) From May 2016 up to January 2018: the latest avaliable data.



(b) A few weeks of 2017.

Figure 1.1: Sections of the electricity demand time-series data provided by NTE.

sparked through research on neural networks [9], which GPs share intimate connections with. They have later been applied to a wide range of different problems, also within the context of electricity demand forecasting [10, 11].

The thesis is divided into two main parts. In the first part, section 2 through 5, we seek to establish a thorough background theory for long-term time-series modelling with Gaussian processes which can utilise information from large datasets and predict for long time-horizons. Some introductory theory and useful concepts are presented in section 2, before a more thorough introduction to Gaussian processes is given in section 3. The use of different covariance functions and how they impose useful model assumptions is reviewed in section 4, and the consecutive section allows for a scalable GP inference. We will assume that the reader has no prior experience with GP regression, but are acquainted with standard statistical theory and nomenclature.

The second part, sections 6 through 9, treats the application of the presented theory in the more narrow context of predicting future energy demands in Nord-Trøndelag within the time-frames and limitations that are prescribed by the Norwegian energy exchange. The dataset is reviewed in section 6, and a family of predictive models are presented in section 7. A more in-depth analysis of the general method is performed in section 8. Further discussion is presented in section 9, and by some closing remarks in section 10 we bring the thesis to a conclusion.

Remark. In the relevant literature, there is seemingly no distinction made between the terms “power load”, “electricity demand”, “energy consumption” and variations thereof.

In this thesis, these terms will be used interchangeably. The same holds true for the terms "covariance function" and "kernel", which will be introduced in section 3.

2 Introductory theory and concepts

2.1 The Gaussian distribution

2.1.1 The univariate case

If the random variable y follow a Gaussian distribution with mean μ and variance σ^2, the probability density function (pdf) of y has a simple form given by

$$\mathcal{N}(y \mid \mu, \sigma^2) \triangleq \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right). \tag{2.1}$$

When the argument is clear from context it is often omitted: $y \sim \mathcal{N}(\mu, \sigma^2)$.

The Gaussian is, for a composite set of reasons, the most widely used probability distribution throughout the statistics and machine learning communities [12, p. 38]. Firstly, the two parameters of the pdf conveniently coincide with the mean and the variance, which for the Gaussian unambiguously determine the entire distribution. Secondly, it can be proven that the Gaussian, among every conceivable distribution, is the one that makes the least assumptions about the data subject to the constraint of having a definite mean and variance [12, p. 289]. Thirdly, the central limit theorem ensures that the sum of m independent random variables can be approximated arbitrarily well by a Gaussian distribution as long as m is sufficiently big. The latter is an important property justifying the use of Gaussian distributions to model error residuals for an extensive range of data sets.

2.1.2 The covariance matrix

A considerable amount of the upcoming discussion will be concerned with covariance, so the definition will be stated here for reference and notational clearity, after which a theorem useful in later derivations will follow.

Definition 1 (The covariance matrix)**.** Let $\boldsymbol{v} = (v_1, \dots, v_d)^\top$ be a random vector with elements of finite variance, and let $\boldsymbol{\mu} = (\mu_1, \dots, \mu_d)^\top$ be the corresponding vector of means. Each entry of the *covariance matrix* $\boldsymbol{\Sigma} = \text{cov}(\boldsymbol{v})$ is then defined by

$$\Sigma_{i,j} = \text{cov}(v_i, v_j) \triangleq \mathbb{E}\left[(v_i - \mu_i)(v_j - \mu_j)\right] \tag{2.2}$$

for all $i, j \in \{1, \dots, d\}$. The relation can equivalently be formulated with the more compact vector notation

$$\boldsymbol{\Sigma} = \mathbb{E}\left[(\boldsymbol{v} - \boldsymbol{\mu})(\boldsymbol{v} - \boldsymbol{\mu})^\top\right]. \tag{2.3}$$

Quantitatively, the covariance measures to the degree of which two random variables are linearly related. It is easily seen from the above definition that $\boldsymbol{\Sigma}$ is symmetric, and along the diagonal, where $i = j$, we observe the covariance collapsing to $\text{var}(v_i)$. The next result fully determines which properties must hold for an arbitrary matrix $\boldsymbol{A}$ in order for it to be a valid covariance matrix.

Theorem 1. *A matrix $\boldsymbol{A}$ is the covariance matrix of a random vector $\boldsymbol{y}$ if and only if $\boldsymbol{A}$ is symmetric and positive semi-definite.*

Proof. Assume first that $\boldsymbol{y}$ is a random vector of dimension d and let $\boldsymbol{A} = \operatorname{cov}(\boldsymbol{y})$. Symmetry is apparent from equation (2.3). For any $\boldsymbol{u} \in \mathbb{R}^d$,

$$\boldsymbol{u}^\top \boldsymbol{A}\boldsymbol{u} = \mathbb{E}\left[\boldsymbol{u}^\top (\boldsymbol{y} - \mathbb{E}[\boldsymbol{y}])(\boldsymbol{y} - \mathbb{E}[\boldsymbol{y}])^\top \boldsymbol{u}\right] = \mathbb{E}(\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}^\top), \tag{2.4}$$

where $\tilde{\boldsymbol{y}} := \boldsymbol{u}^\top(\boldsymbol{y} - \mathbb{E}[\boldsymbol{y}])$ is a random vector of zero mean. Thus, $\mathbb{E}(\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}^\top) = \operatorname{var}(\tilde{\boldsymbol{y}}) \geq 0$. As $\boldsymbol{u}^\top \boldsymbol{A}\boldsymbol{u} \geq 0$ for all $\boldsymbol{u} \in \mathbb{R}^d$, $\boldsymbol{A}$ must be positive-semidefinite.

Conversely, suppose that $\boldsymbol{A}$ is a symmetric positive semi-definite $d \times d$ matrix. We must show that there exists a random vector having $\boldsymbol{A}$ as covariance matrix. From the spectral theorem of linear algebra we know that we can find a (unique) symmetric positive-semidefinite square root of $\boldsymbol{A}$ and thus it makes sense to define the matrix $\boldsymbol{B}$ through the relation $\boldsymbol{B}^2 = \boldsymbol{A}$. Further, let $\boldsymbol{y}$ be a d-dimensional random vector whose covariance is the $p \times p$ identity matrix. We then have

$$\operatorname{cov}(\boldsymbol{B}\boldsymbol{y}) = \mathbb{E}\left[(\boldsymbol{B}\boldsymbol{y} - \mathbb{E}[\boldsymbol{B}\boldsymbol{y}])(\boldsymbol{B}\boldsymbol{y} - \mathbb{E}[\boldsymbol{B}\boldsymbol{y}])^\top\right] = \boldsymbol{B}\operatorname{cov}(\boldsymbol{y})\boldsymbol{B}^\top = \boldsymbol{A} \tag{2.5}$$

after applying some fundamental rules from linear algebra. Hence, it is shown that $\boldsymbol{A}$ is indeed the covariance matrix of a random vector, namely $\boldsymbol{B}\boldsymbol{y}$. □

2.1.3 The multivariate Gaussian distribution

The random vector $\boldsymbol{y} = (y_1, \dots, y_d)^\top$ is said to follow a *multivariate Gaussian* distribution if and only if the linear combination $\boldsymbol{a}^\top\boldsymbol{y}$ have a univariate Gaussian distribution for all $\boldsymbol{a} \in \mathbb{R}^d$ [13]. The corresponding pdf models the joint stochastic relationship between the variables, and are as in the univariate case defined through the first two moments, namely the mean $\boldsymbol{\mu} := \mathbb{E}(\boldsymbol{y})$ and the covariance $\boldsymbol{\Sigma} := \operatorname{cov}(\boldsymbol{y})$. The equation for the joint Gaussian pdf is

$$p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \triangleq \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})\right), \tag{2.6}$$

where $|\cdot|$ determines the determinant of the argument. The valuable properties stated for univariate Gaussians hold likewise in the multivariate case.

2.1.4 Conditional Gaussian distributions

It can (and will) be exceedingly constructive to perform the partition $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2)^\top$ and then proceed to find the conditional distribution of one vector given the other, often refered to as the *posterior*.

Theorem 2 (Marginal and conditional Gaussian distributions)**.** *Assume the random vector $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2)$ is jointly Gaussian with mean and covariance parameters defined as*

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}. \tag{2.7}$$

Then the marginal distributions are given by

$$\begin{aligned} p(\boldsymbol{y_1}) &= \mathcal{N}(\boldsymbol{y_1} \mid \boldsymbol{\mu_1}, \boldsymbol{\Sigma}_{11}) \\ p(\boldsymbol{y_2}) &= \mathcal{N}(\boldsymbol{y_2} \mid \boldsymbol{\mu_2}, \boldsymbol{\Sigma}_{22}), \end{aligned} \tag{2.8}$$

and the posterior conditional distribution $p(\boldsymbol{y}_2 \mid \boldsymbol{y}_1) = \mathcal{N}(\boldsymbol{\mu}_{2|1}, \boldsymbol{\Sigma}_{2|1})$ have parameters

$$\begin{aligned} \boldsymbol{\mu}_{2|1} &= \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\boldsymbol{y}_1 - \boldsymbol{\mu}_1), \\ \boldsymbol{\Sigma}_{2|1} &= \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\Sigma_{11}^{-1}\boldsymbol{\Sigma}_{12}. \end{aligned} \tag{2.9}$$

The proof is somewhat involved and will be skipped here, but can be found in most textbooks on multivariate statistics, e.g. Murphy's [12, p. 118].

The idea of theorem 2 is visualised in figure 2.1, where the following example is considered: let $(x_1, x_2)^\top$ be a bivariate Gaussian random vector whose joint pdf is depicted in figure 2.1 as elliptical contour lines. The corresponding marginal distributions $p(x_1)$ and $p(x_2)$ are shown as projections on the horizontal and vertical axes respectively. Assume now that we observe x_2 at the location of the dashed vertical line. The distribution of the resulting random variable x_1 consitioned on x_2, denoted by $x_1 \mid x_2$, is visualised with a dashed curve. The variance of $x_1 \mid x_2$ is considerably lower than that of x_1. Thus, by observing one variable we are no longer in our initial state of ignorance about the other and can infer more information about it, analytically so by employing theorem 2.

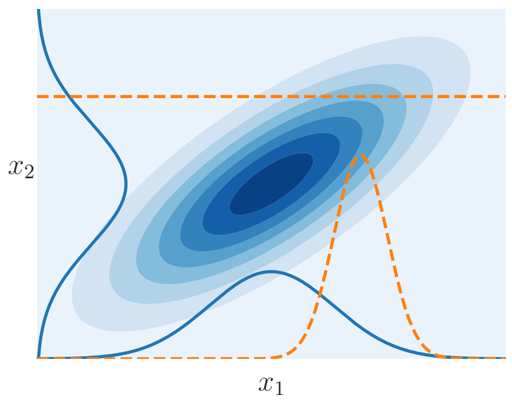


Figure 2.1: Probability density contours of a bivariate Gaussian, from which marginal distributions (blue solid lines) are formed. Initially both marginals are wide, signifying vague prior beliefs, but after observing x_2 (straight dashed line) we become simultaneously much more certain about x_1 (dotted curve).

2.2 Selected mathematical topics

Before continuing, we will review two mathematical concepts that will later be useful.

2.2.1 Cholesky decomposition

The Cholesky decomposition will have several applications in this thesis: matrix inversion, calculation of determinants and sampling from Gaussian distributions. A symmetric positive-definite matrix $\boldsymbol{A}$ of size $n \times n$ will (uniquely) decompose into a product of a lower triangular matrix $\boldsymbol{L}$, called the *Cholesky factor*, and its transpose,

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^\top. \tag{2.10}$$

The factor itself can be found using elemetary linear algebra, a procedure considered numerically very stable and solved in $n^3/6$ time [14, p. 202].

When the cholesky factor is already calculated, the determinant of $\boldsymbol{A}$ can very efficiently be computed as

$$|\boldsymbol{A}| = \prod_{i=1}^{n} L_{ii}^2, \quad \text{or equivalently,} \quad \log|\boldsymbol{A}| = 2\sum_{i=1}^{n} \log L_{ii}. \tag{2.11}$$

To find the inverse of $\boldsymbol{A}$ we solve the linear system $\boldsymbol{L}\boldsymbol{L}^\top\boldsymbol{A}^{-1} = \boldsymbol{I}$ in two steps: first by applying forward substitution to solve $\boldsymbol{L}\boldsymbol{u} = \boldsymbol{I}$ for $\boldsymbol{u}$, and then follow up with back subsitution, solving $\boldsymbol{L}^\top\boldsymbol{A}^{-1} = \boldsymbol{u}$ for $\boldsymbol{A}^{-1}$. The process requires n^2 operations in total.

Lastly, to sample from the d-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, having at our disposal a method to draw samples from $\mathcal{N}(0,1)$, we first find the cholesky factor $\boldsymbol{L}$ of $\boldsymbol{\Sigma}$ and let $\boldsymbol{y}$ be a vector of size d whose elements are independent samples from $\mathcal{N}(0,1)$. Then $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{L}\boldsymbol{y}$ will have the desired distribution as $\mathbb{E}(\boldsymbol{x}) = \boldsymbol{\mu}$ and

$$\operatorname{cov}(\boldsymbol{x}) = \boldsymbol{L}\mathbb{E}(\boldsymbol{y}\boldsymbol{y}^\top)\boldsymbol{L}^\top = \boldsymbol{L}\boldsymbol{L}^\top = \boldsymbol{\Sigma} \tag{2.12}$$

by the independence of the elements of $\boldsymbol{y}$.

2.2.2 KL divergence

The Kullback-Leibler (KL) divergence is a measure of how far an approximated probability distribution q lies from a query distribution p. It can when considering continuous probability densities be expressed as

$$\mathcal{KL}(p \mid\mid q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x. \tag{2.13}$$

It is also often referred to as a measure of entropy.

2.3 Bayesian time-series analysis

One of the motivating factors that drive researchers to consider time-series analysis, is to gain a deeper understanding of the underlying forces and structures that govern the behaviour of some quantity under observation. Another is to fit a model and proceed to forecast, monitor or control said quantity. Naturally, the latter factor builds on the first: the more we understand of the inherent forces that govern a system, the better we can model it.

Thus, time-series analysis can be formulated as a *regression problem* where we seek a function f that explains the observed quantity y through the relationship $y(\boldsymbol{x}) = f(\boldsymbol{x}) + \eta$, where η is additive noise. Often the goal of inference is to understand how the input variables interact to produce the observations we have of f. It is especially useful if we are also able to quantify the uncertainty involved in the model. For example, by evaluating the probability distribution of an unseen point y_* given the model and input points $\boldsymbol{x}_*$, it can be quantified how confident the model is in, say, predicting new values $f(\boldsymbol{x}_*)$.

2.3.1 Bayesian model selection and Occam's razor

This ability to quantify uncertainty is provided when considering inference under the Bayesian formalism. In short, we seek an answer to the following: what is the probability of a hypothesised regression model based on observed data? To understand why it is useful to ask this question, consider the example of two-dimensional data points $\mathcal{D} = \{t_i, y_i\}_i$ shown in figure 2.2. A family of curves run through these points. Each of the curves is able to explain the data identically, but their behaviour differs in regions far from data points, significantly so on the extrapolated edges. Some curves have higher variation in the form of larger curvature, while others do not spread out as much and are in that sense *simpler* explanations of the data. If we were are to follow the reasoning of *Occam's razor*, where the simpler of two adequate explanations are to be favoured, we might be better of using the least complex of the curves to model the system.

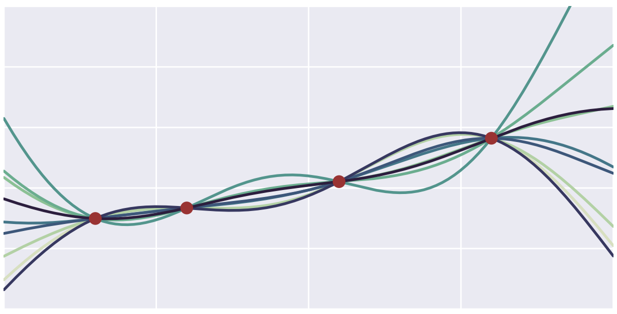


Figure 2.2: Eight different models explaining the same four data points. Some explanations are more complex than others.

As it turns out, Bayesian modelling incorporates this principle elegantly and allows us to choose functions that are well balanced between data fit and complexity. We assume that the curves are drawn from a probability distribution $p(\mathcal{M} \mid \mathcal{D})$, where $\mathcal{M}$ represents a model space. By Bayes we calculate the *posterior*,

$$p(\mathcal{M} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M})p(\mathcal{M})}{p(\mathcal{D})}, \tag{2.14}$$

which has naming convention

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}. \tag{2.15}$$

Now we may perform *Bayesian model selection* and pick the model that maximises the posterior distribution, $\hat{m} = \text{argmax}_{m \in \mathcal{M}}\, p(m \mid \mathcal{D})$, which is frequently referred to as the MAP model [12]. However, if $\mathcal{M}$ can be parametrised, and we impose a uniform prior distribution $p(\mathcal{M}) \propto 1$ on the model (i.e., we do not specifically favour any of the models), then the probability of the data given the model class is

$$p(\mathcal{D} \mid \mathcal{M}) = \int p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})p(\boldsymbol{\theta} \mid \mathcal{M})d\boldsymbol{\theta}, \tag{2.16}$$

where $\boldsymbol{\theta}$ is the set of parameters for $\mathcal{M}$. This is proportional to the marginal likelihood $p(\mathcal{D})$ from equation (2.14). Thus, the Bayesian model selection amounts to choosing the model that maximises the marginal likelihood [12]. The approach has an built-in safeguard against overcomplex models: the marginal likelihood quantity itself is the probability that the data set is generated from *randomly selected* parameter values of the model class. Very simple models (relative to the complexity of the data set $\mathcal{D}$) will be very unlikely to have generated $\mathcal{D}$, whereas too complex models can generate vast arrays of different data sets and are therefore unlikely to generate $\mathcal{D}$, in particular, at random [15]. This is called the *Bayesian Occam's razor* effect [16].

2.3.2 Parametric and non-parametric models

As there exist a bewildering amount of functions that can equally well explain the data we observe, it is often wise to attempt to reduce the complexity of the model space before any inference is performed. In particular, if we have any strong prior belief about the underlying process responsible for a set of observations, we can impose this knowledge by directly specifying a subset of the model space; for the data set in figure 2.2, a family of exponential functions might be a strong contender. Such models are called *parametric* since a finite set of parameters must be determined in order to fit the model.

In many cases, however, we have limited prior knowledge regarding appropriate models to use. At the same time, it might be possible to pick out certain aspects of the underlying process that are seemingly less definite. For example, when considering the

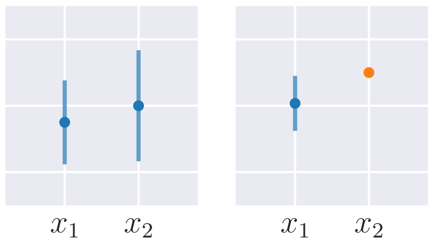


Figure 2.3: The conditional distribution of the variables from figure 2.1 presented in a different format. The left panel shows x_1 prior to observing x_2. In the right panel, both the mean and variance of x_1 is influenced by x_2 being observed. The bars illustrate two standard deviations.

power consumption data set provided by NTE, we see from figure 1.1 that it contains several periodic components, from which we can pick out some typical amplitudes, and the signal is seemingly continuous in the time scale that we are interested in. Working mathematically with infinite function spaces of which their members share such features are (remarkably!) entirely feasible. Moreover, we can turn the process of explaining, modelling and extrapolating data into a matter of contemplating *probability distributions* over such spaces, and then later refine the distributions so that samples from it agree with the observed data [17]. Such models have no parameters that directly govern the resulting function values, and so they are referred to as (Bayesian) *non-parametric* models, of which one interesting flavour is the *Gaussian process* (GP).

2.3.3 A conceptual basis of Gaussian processes in a time-series context

Before we delve into the mathematical framework that allows for probabilistic GP inference we will introduce a conceptual basis through a direct appeal to joint Gaussian distributions, and see how they can be naturally extended to model continuous time-series data. The simple but revealing interpretation is inspired by Roberts et al. [17].

Consider again the simple example displayed in figure 2.1. The information given by the marginal distributions may, with a slight change of perspective, instead be presented as in figure 2.3. As before, subsequent to observing one variable (x_2), the variance and most probable location of the other (x_1) is affected.

Now we extend the example to higher dimensions. Let $(x_1, \ldots, x_n)$ be a vector of random variables, the relationship between which are defined by an $n \times n$-sized covariance matrix, and associate with each variable an integer *time label*. The left tile of figure 2.4 depicts the conditional distributions for the specific case when $n = 20$ and two values have been observed. Calculations are performed using equation (2.9). Keeping the observations fixed, the right tile shows the effect of increasing n dramatically over the same time interval. Note that the conditional distribution in the left tile perfectly fits within the "dense" counterpart in the right tile. That is, from the random vector depicted in the right tile we can *marginalise out* the points which are not included in the original 20-dimensional vector, and the result will be *identical* to the original “sparse” random vector.

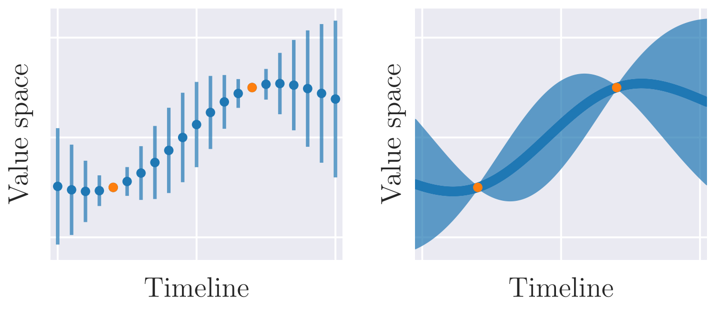


Figure 2.4: Associating each element of the random vector $(x_1, \ldots, x_n)$ with a time-label allows us to illustrate the means (blue dots) and variances (vertical bars) along a timeline. There are two observations (orange dots). In the left pane $n = 20$, while the right pane employs $n = 500$.

Lastly, we let $n \to \infty$ and in this limit see the joint distribution of all the points become equivalent to a distribution over functions. Yet, even in this dense space, we are able to work mathematically and computationally with the distribution—it is sufficient that we are able to evaluate the pdf at *any location in time*, and that we can do so with any finite number of simultaneous query points. This amounts to working with continuous time-series—a Gaussian process. As the process is conceptually equivalent to an infinite extension of the Gaussian distribution, it benefits from properties inherited from multivariate Gaussians.

3 Gaussian processes

Formally, a Gaussian process (GP) can be defined as the following:

Definition 2 (Gaussian process)**.** For an index set $\mathcal{X}$ the collection of random variables $\{f(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{X}\}$ is called a *Gaussian process* if for each finite subset $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\} \subset \mathcal{X}$ the random vector $f(\boldsymbol{X}) = (f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_n))$ have a joint Gaussian distribution.

The index set $\mathcal{X}$ will henceforth be assumed to be $\mathbb{R}^d$ with input dimension $d \in \mathbb{N}$, and referred to as the *input space*. Let us consider the case where we are provided a set of input points $\boldsymbol{X} \in \mathcal{X}$ with corresponding observations $\boldsymbol{y} \in \mathbb{R}^n$ from the function—possibly corrupted by noise—and our objective is to estimate the latent function $f \colon \mathcal{X} \mapsto \mathbb{R}$. Through domain knowledge and analysis of observed samples we may have obtained some idea of properties possessed by the latent function; it may show a certain degree of smoothness, for example, or tend to repeat itself at some periodicity. Assumptions like these are reflected in the choice of the *mean function* $m \colon \boldsymbol{x} \mapsto \mathbb{E}[f(\boldsymbol{x})]$ and the *covariance function* $\kappa \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, defined for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ as

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\Big[\Big(f(\boldsymbol{x}) - m(\boldsymbol{x})\Big)\Big(f(\boldsymbol{x}') - m(\boldsymbol{x}')\Big)\Big]. \tag{3.1}$$

Just as a Gaussian distribution is entirely determined by its first two moments, the mean and covariance functions—the latter commonly referred to as the *kernel*—uniquely specifies a GP. That is, we may compactly convey all our prior beliefs about the function f by the expression

$$f \sim \mathcal{GP}\Big(m(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}')\Big), \tag{3.2}$$

signifying that f is *distributed* according to a GP. Note that it is over the input space $\mathcal{X}$ that the mean and covariance are defined, as opposed to over the set of input points $\boldsymbol{X}$, allowing the covariance and mean to be evaluated for any finite combinations of input elements. Specifically, for the set of vectors $\{\boldsymbol{x}_1, \dots \boldsymbol{x}_n\}$ from $\mathcal{X}$, we build a covariance matrix $\boldsymbol{K}$ by evaluating each combination individually as

$$\boldsymbol{K} := \kappa\Big((\boldsymbol{x}_1, \dots, \boldsymbol{x}_n), (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)\Big) = \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) & \kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) & \kappa(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x}_2, \boldsymbol{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x}_n, \boldsymbol{x}_1) & \kappa(\boldsymbol{x}_n, \boldsymbol{x}_2) & \dots & \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}. \tag{3.3}$$

Here the off-diagonal elements define the covariance between two distinct variables. From theorem 1 it follows that in order for $\boldsymbol{K}$ to be a proper covariance matrix it must be symmetric and positive semi-definite, hence a valid covariance function κ must return matrices having this property for each and every input combination from $\mathcal{X}$. Conversely, we know from the same theorem that any kernel with this property will return only legitimate covariance matrices.

One example function for which this property holds is the *squared exponential*[2] (SE), defined as

$$\operatorname{cov}(f(\boldsymbol{x}), f(\boldsymbol{x}')) = \kappa_{\text{SE}}(\boldsymbol{x}, \boldsymbol{x}') := \sigma_\kappa^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}''\|^2}{2l^2}\right), \tag{3.4}$$

where $\|\cdot\|$ denotes the Euclidean distance. The parameters l and σ_κ^2 governs the shape of the kernel. The squared exponential encodes, among other things, the assumption that if two inputs are similar, then their corresponding function values ought most likely also to be similar. It is indubitably the most frequently used covariance function in the GP literature, and will be discussed further in section 4.1.2.

3.1 Prior and posterior distributions

To simplify notation, we will throughout this thesis assume $m(\boldsymbol{x}) \equiv 0$ for all inputs $\boldsymbol{x}$, which is a fairly common assumption in the GP literature.[3] As soon as we have selected and kernel κ, a prior distribution on the function f have been imposed from equation (3.2). To sample from this distribution, we need first to pick a collection of points $\boldsymbol{X}_*$ from $\mathcal{X}$ at which to evaluate the function, and calculate the covariance matrix $\boldsymbol{K}_* := \kappa(\boldsymbol{X}_*, \boldsymbol{X}_*)$ using equation (3.3). Then, from the resulting random vector $\boldsymbol{f}_* \sim \mathcal{N}(\boldsymbol{m}_*, \boldsymbol{K}_*)$, whose Gaussian distribution is ensured by definition 2, it is a fairly standard exercise to draw samples following the procedure described in section 2.2.1. Several prior samples of f are shown in figure 3.1a for one-dimensional input points and a SE covariance function with unit length-scale and variance.

As we are ultimately not interested in naïvely drawing samples from the prior until we obtain a function that agrees with observed data, a natural next step is to refine the distribution so that *every* sample will interpolate the observed target values. To achieve this, let $\mathcal{D} := (\boldsymbol{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_i,\ y_i)\}_{i=1}^n$ be a set of input vectors and corresponding output values, respectively, and denote by $\boldsymbol{f}$ the latent function values at points that are included in $\mathcal{D}$ and $\boldsymbol{f}_*$ the function values at some finite set of query points $\boldsymbol{X}_*$ not in $\mathcal{D}$. Under the GP prior, the joint distribution $p(\boldsymbol{f}, \boldsymbol{f}_*)$ is Gaussian by definition, specifically,

$$p(\boldsymbol{f}, \boldsymbol{f}_*) = p\left(\begin{bmatrix} f(\boldsymbol{X}) \\ f(\boldsymbol{X}_*) \end{bmatrix}\right) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K}_*^\top \\ \boldsymbol{K}_* & \boldsymbol{K}_{**} \end{bmatrix}\right), \tag{3.5}$$

with $\boldsymbol{K} := \kappa(\boldsymbol{X}, \boldsymbol{X})$, $\boldsymbol{K}_{**} := \kappa(\boldsymbol{X}_*, \boldsymbol{X}_*)$ and lastly $\boldsymbol{K}_* := \kappa(\boldsymbol{X}_*, \boldsymbol{X}) = \kappa(\boldsymbol{X}, \boldsymbol{X}_*)^\top$. To make for a slightly more realistic scenario we will impose one final assumption, namely that

[2] It is supposedly difficult to agree on a name for this function in the GP community. Many call it a "radial basis function" or simply "Gaussian" but more recently a name change has been proposed to the more demonstrative "exponentiated quadratic". However, the SE name still seems to be prevalent so we will stick to that.

[3] The rationale behind this seemingly severe restriction will be made clear in section 4.

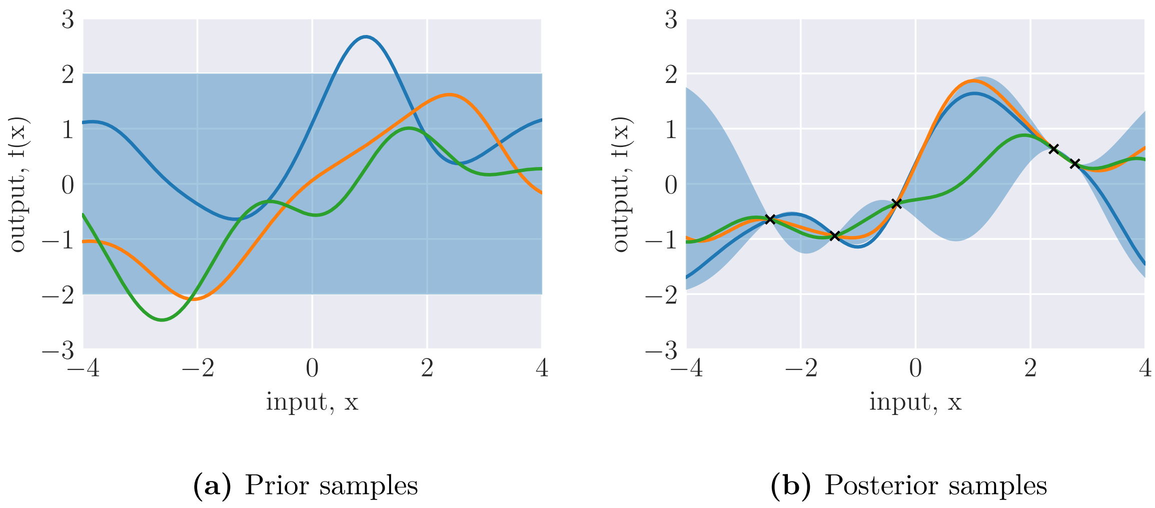


Figure 3.1: Panel (a) depicts functions sampled from a GP prior distribution. Panel (b) shows random samples drawn from the GP posterior, i.e. samples generated after conditioning the prior on the five noiseless data points shown in black crosses. Shaded areas illustrate two standard deviations.

we do not have access to exact values of the latent function f, only to noisy observations thereof. In the simplest setting, and the one within the scope of this thesis, the noise is Gaussian and assumed to be additive, independent and identically distributed with variance σ_n^2, i.e.,

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i, \quad \text{where} \quad \epsilon_i \sim \mathcal{N}(0, \sigma_n^2). \tag{3.6}$$

The above is called a Gaussian likelihood assumption because it leads a Gaussian $p(\boldsymbol{y} \mid \boldsymbol{f})$, which is recognizable as the likelihood from equation (2.14). Still subject to the GP prior, this assumption leads to the joint distribution $p(\boldsymbol{y}, \boldsymbol{f}_*)$ also being Gaussian. It follows from the independency of the noise that a diagonal matrix must be added to the covariance of the observed target values, and by comparing to the noiseless case in equation (3.5), we have

$$p(\boldsymbol{y}, \boldsymbol{f}_*) = p\left(\begin{bmatrix} f(\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I} \\ f(\boldsymbol{X}_*) \end{bmatrix}\right) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^\top & \boldsymbol{K}_{**} \end{bmatrix}\right). \tag{3.7}$$

On the above distribution we can apply theorem 2 and readily obtain the posterior predictive mean and variance for the function evaluations $\boldsymbol{f}_*$, which are the key predictive equations in GP regression:

$$\begin{aligned} p(\boldsymbol{f}_* \mid \mathcal{D}, \boldsymbol{X}_*) &= \mathcal{N}\left(\bar{\boldsymbol{f}}_*,\ \operatorname{cov}(\boldsymbol{f}_*)\right) \\ &= \mathcal{N}\left(\boldsymbol{K}_*\boldsymbol{K}_n^{-1}\boldsymbol{y},\ \boldsymbol{K}_{**} - \boldsymbol{K}_*\boldsymbol{K}_n^{-1}\boldsymbol{K}_*{}^\top\right). \end{aligned} \tag{3.8}$$

Here we have, to save a drop of ink, introduced the notation

$$\boldsymbol{K}_n := \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}. \tag{3.9}$$

The predictive variance in equation (3.8) is the difference between the prior covariance and a term representing the information gained about the function f from observing $\boldsymbol{y}$. Note that the predictive variance is not dependent on the actual observed values, only on the input points $\boldsymbol{X}$ where they are located. The predictive mean, on the other hand, follows the observations as we would expect. In figure 3.1b, a few random samples from the posterior GP are shown, with a SE covariance function having unit variance and length-scale hyperparameters.

Remark. Instead of evaluating the distribution of the latent variables $\boldsymbol{f}_*$, we wish often in practice to compute the predictive distribution of the targets $\boldsymbol{y}_*$ at the test points, i.e., $p(\boldsymbol{y}_* \mid \mathcal{D}, \boldsymbol{X}_*)$. This is easily achieved by adding $\sigma_n^2 \boldsymbol{I}$ to the expression for $\operatorname{cov}(\boldsymbol{f}_*)$ in equation (3.8).

3.2 Determining hyperparameters

Most kernels depend on a set of parameters that regulate their behaviour. They are typically called *hyperparameters* because the role they play with respect to the underlying function f under analysis is an indirect one: they specify the distribution of the function's *parameters*.[4] Before any inference can be performed about f, the hyperparameters must be adjusted to suitable values. Luckily, the GP framework allows for closed-form computation of the *marginal likelihood*, enabling fast and reliable model selection.

In the following, let the kernel hyperparameters be aggregated by $\boldsymbol{\theta}$, on which, although not stated explicitly, GP covariance matrices always depend on. Under the GP prior and the noise assumption in equation (3.6), it follows readily that the marginal likelihood is Gaussian, specifically, $p(\boldsymbol{y} \mid \boldsymbol{X}) \sim \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{0}, \boldsymbol{K}_n)$. This quantity is so called because it implicitly integrates, or *marginalises*, over all function values at all the locations where we have no observations. Using equation (2.6), it follows that

$$\begin{aligned}\mathcal{L} := \log p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) &= \log\left(\frac{1}{\sqrt{(2\pi)^n |\boldsymbol{K}_n|}} \exp\left(-\frac{1}{2}\boldsymbol{y}\boldsymbol{K}_n^{-1}\boldsymbol{y}\right)\right) \\ &= \underbrace{-\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_n^{-1}\boldsymbol{y}}_{\text{data fit}} - \underbrace{\frac{1}{2}\log|\boldsymbol{K}_n|}_{\text{complexity penalty}} - \frac{n}{2}\log 2\pi.\end{aligned} \tag{3.10}$$

As expected from the discussion surrounding figure 2.2, the marginal likelihood incorporates the prinsiples of Occam's razor. The data fit term and complexity penalty term elegantly balances the capacity of a model and how well it fits to the data. A small value

[4]For reference, the hyperparameters may themselves also have priors—they are referred to as *hyperpriors*

of the complexity penalty means that a restricted number of different datasets can be accomodated by the model, while a large data fit term indicate that the model adapts well to the data.

Many optimization routines benefit from gradient information. The gradient of the log marginal likelihood with respect to $\boldsymbol{\theta}$ is most easily derived component-wise, so we write $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$ and calculate for $i = 1, \ldots, k$,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta_i} &= \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_n^{-1} \frac{\partial \boldsymbol{K}_n}{\partial \theta_i} \boldsymbol{K}_n^{-1} \boldsymbol{y} - \frac{1}{2} \operatorname{tr}\left(\boldsymbol{K}_n^{-1} \frac{\partial \boldsymbol{K}_n}{\partial \theta_i} \right) \\ &= \frac{1}{2} \operatorname{tr}\left((\boldsymbol{\beta}\boldsymbol{\beta}^\top - \boldsymbol{K}_n^{-1}) \frac{\partial \boldsymbol{K}_n}{\partial \theta_i} \right), \end{aligned} \tag{3.11}$$

where $\boldsymbol{\beta} = \boldsymbol{K}_n^{-1}\boldsymbol{y}$, and we have used the following identities from matrix calculus:

$$\frac{\partial \log |\boldsymbol{A}|}{\partial a} = \operatorname{tr}\left(\boldsymbol{A}^{-1} \frac{\partial \boldsymbol{A}}{\partial a} \right), \quad \text{and} \quad \frac{\partial \boldsymbol{A}^{-1}}{\partial a} = -\boldsymbol{A}^{-1} \frac{\partial \boldsymbol{A}}{\partial a} \boldsymbol{A}^{-1} \tag{3.12}$$

for an arbitrary matrix $\boldsymbol{A}$. Given a GP prior, we can pass the expressions for the log marginal likelihood and its derivative to a continous optimisation routine. We informally say a hyperparameters is “learned” when it is optimised to a particular value. This does not, however, imply that this value is optimal in the global sense, as the log likelihood space is generally highly non-convex.

3.3 Efficient and numerically stable implementation

Computing $\boldsymbol{K}_n^{-1}$ is in general a computationally demanding and unstable operation, scaling cubically with the number of data points n in the input set. Luckily, as the covariance matrix is positive-semidefinite, it can be decomposed into a product of Cholesky factors which enables numerically stable inverse and determinant evaluations, as seen in section 2.2.1.

In algorithm 1, a complete and numerically stable GP regression routine is given. For improved optimisation, one ought in addition to include in line 7 the analytical gradient of $\mathcal{L}$, equation (3.11), but this is not a prerequisite for a stable implementation. Calculating the gradient requires an additional n^2 time per kernel hyperparameter, but the expected amount of time saved from potentially needing less evaluations of $\boldsymbol{K}_n^{-1}$, which is required at every interation of the optimisation algorithm, is substantial.

Finding the Cholsky factor of $\boldsymbol{K}_n$ requires $O(n^3)$ operations, so algorithm 1 will not scale well to datasets containing thousands of entries. A method to deal with the issue is established in section 5.

Algorithm 1: Gaussian process regression with a Gaussian likelihood assumption.

Input:

$\boldsymbol{X} \in \mathcal{X}^n$ (input points), $\boldsymbol{y} \in \mathbb{R}^n$ (correponding observations), $\sigma_n^2 \in \mathbb{R}$ (noise variance), $\boldsymbol{X}_* \in \mathcal{X}^{n_*}$ (test input points), $\kappa\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (prior covariance function with hyper-parameters $\boldsymbol{\theta}$).

Result:

$\bar{\boldsymbol{f}}_* \in \mathbb{R}^{n_*}$ (predicted mean at test points), $\mathrm{cov}(\boldsymbol{f}_*) \in \mathbb{R}^{n_* \times n_*}$ (covariance of predictions)

Procedure:

```
 1: while optimality condition not satisfied do
 2:     $\boldsymbol{K} \leftarrow \kappa(\boldsymbol{X}, \boldsymbol{X})$, $\boldsymbol{K}_* \leftarrow \kappa(\boldsymbol{X}, \boldsymbol{X}_*)$, $\boldsymbol{K}_{**} \leftarrow \kappa(\boldsymbol{X}_*, \boldsymbol{X}_*)$
 3:     $\boldsymbol{L} \leftarrow$ Cholesky factor of $(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})$
 4:     $\boldsymbol{\Upsilon} \leftarrow$ solution of $\boldsymbol{L}\boldsymbol{\Upsilon} = \boldsymbol{I}$                        $\triangleright$ by forward substitution
 5:     $\boldsymbol{\Phi} \leftarrow$ solution of $\boldsymbol{\Upsilon} = \boldsymbol{L}^\top \boldsymbol{\Phi}$                        $\triangleright$ by back substitution
 6:     $\mathcal{L} \leftarrow -\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{\Phi} \boldsymbol{y} - \sum_i \log L_{ii} - \frac{n}{2}\log 2\pi$     $\triangleright$ a func. of $\boldsymbol{\theta}$; from equation (3.10)
 7:     $\boldsymbol{\theta} \leftarrow \mathrm{argmax}_{\boldsymbol{\theta}}(\mathcal{L})$                                 $\triangleright$ by a cont. opt. routine
 8: $\bar{\boldsymbol{f}}_* \leftarrow \boldsymbol{K}_*^\top \boldsymbol{\Phi} \boldsymbol{\zeta}$                                        $\triangleright$ from equation (3.8)
 9: $\mathrm{cov}(\boldsymbol{f}_*) \leftarrow \boldsymbol{K}_{**} - \boldsymbol{K}_*^\top \boldsymbol{\Phi} \boldsymbol{K}_*$                          $\triangleright$ from equation (3.8)
10: return $\bar{\boldsymbol{f}}_*$, $\mathrm{cov}(\boldsymbol{f}_*)$
```

4 Kernels for time-series modelling

Correlations between values of the unknown function f evaluated at given input locations are expressed by kernels, the choice of which can have a profound impact on the predictive performance of the resulting GP models. The task of choosing suitable kernels is, however, not simple and has been referred to as “black art” even for experts [18]. Moreover, the space of available choices is extensive [19, 20]. In this section, we will discuss selected kernels relevant to the present forecasting context and see how they can express structural assumptions and be combined as required to build rich non-parametric models for time-series.

4.1 Standalone kernels

4.1.1 Elemental kernels

The most basic kernels are, although extremely simple, very useful in the context of modelling time-series. The *constant* or *bias* kernel $\kappa_{\mathrm{c}}(\boldsymbol{x}, \boldsymbol{x}') := c$ where c is a positive constant, is often used in place of a mean function m to add an uncertain offset to the model. If the time-series are expected to have a linear trend, or if there are uncertainty whether this might be the case, the linear kernel is a suitable option. It is defined as

$$\kappa_{\mathrm{lin}}(\boldsymbol{x}, \boldsymbol{x}') := \sigma_\kappa^2(\boldsymbol{x} - \boldsymbol{c})(\boldsymbol{x}' - \boldsymbol{c}) \tag{4.1}$$

where σ_κ^2 determines the magnitude of the slope of the line and $\boldsymbol{c}$ specifies the point on the input axis at which all all the lines in the posterior intersect, i.e., where the contribution to covariance from this kernel is zero. One-dimensional kernel evaluations and functions drawn from the resulting prior distribution $\mathcal{GP}(0, \kappa_{\mathrm{lin}}(x, x'))$ are displayed in figure 4.1a.

As discussed in section 3.1 the Gaussian noise assumption can be used to incorporate uncertainty from the observed data into the GP. If the noise is known in advance or can be bounded from above, e.g. if the data source is a sensor with known resolution, then it may be desirable to treat the variance σ_n^2 as a property of the *data* and keep it fixed. Adding to the prior covariance a *white noise* kernel allows us to additionally entertain uncertainty in the *model*; the definition is very simple:

$$\kappa_{\mathrm{WN}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_\kappa^2 \delta(i, j), \quad \text{where } \delta(i, j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

4.1.2 Stationary kernels

In equation (3.4) the squared exponential kernel was defined. Note that this function only depends on the relative distance between two points—it is a part of the *stationary* function class whose members are translation invariant, meaning $g(\boldsymbol{x}, \boldsymbol{x}') \equiv g(\|\boldsymbol{x} - \boldsymbol{x}'\|)$.

Stationary kernels are applied in situations were one expect values of f to be similar when evaluated at points close to each other. In a time-series setting where the input space is the timeline, for example, a prior with stationary kernels encodes the belief that the informativeness of past observations in forecasting future data is dependent on how long ago the observations were made.

The length-scale l in a stationary kernel regulates the input (length, or time) scale: a larger value yields smoother functions and vice versa, because the contribution to covariance is then shared across more of the input points. The *signal variance* hyperparameter σ_κ^2 defines the output (i.e. covariance) scale; it determines the variability of samples around the GP mean function m. A reasonable hyperprior for this value should encode how much, on average, we expect the latent function to deviate from its mean.

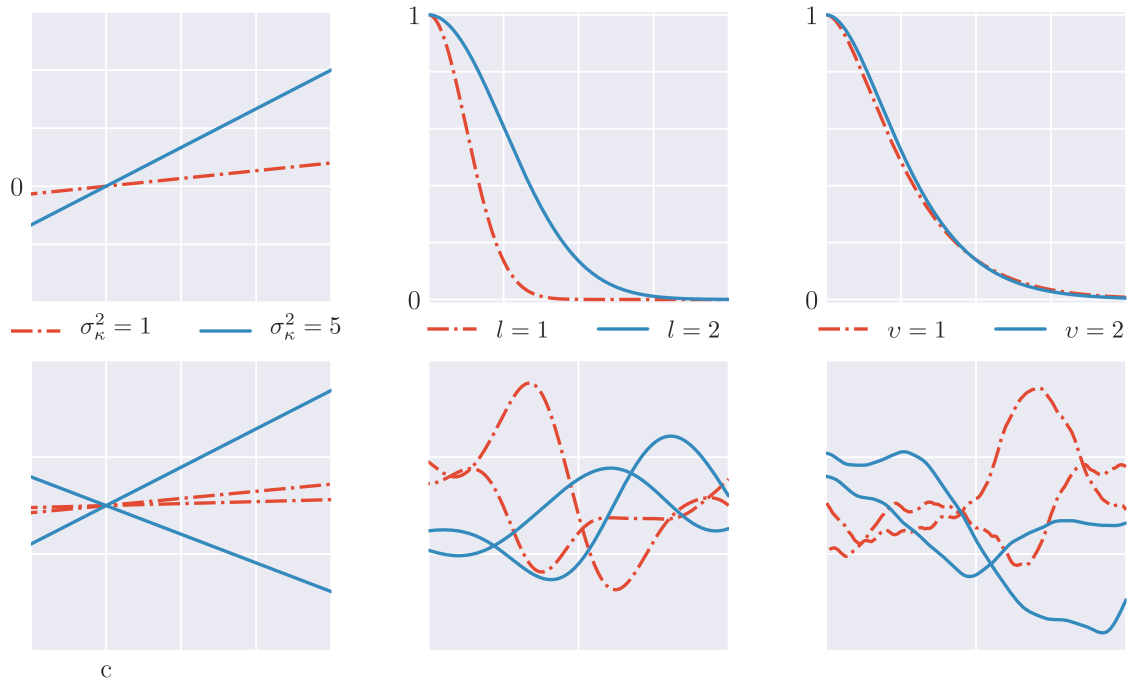
The squared exponential kernel can be shown to correspond to a Bayesian linear regression model with an infinite number of basis functions [14, sec. 4.3.1], Moreover, a GP with a SE kernel can approximate arbitrarily well any function given enough data [21], making it the default setting in many GP applications. It is also infinitely differentiable and thus admits functions drawn from the GP prior that are very smooth. In some situations, however, this assumption is unrealistic for f[5] and it is more suitable to adopt a kernel from the *Matérn* family of functions, of which a useful branch follows the definition

$$\begin{aligned}\kappa_{\text{Mat}}^{(\upsilon)}(\boldsymbol{x}, \boldsymbol{x}') := \exp\left(-\sqrt{2\upsilon+1} \cdot \frac{\|\boldsymbol{x}-\boldsymbol{x}'\|}{l}\right) \frac{\Gamma(\upsilon+1)}{\Gamma(2\upsilon+1)} \cdot \\ \sum_{i=0}^{\upsilon} \frac{(\upsilon+i)!}{i!(\upsilon-i)!}\left(\sqrt{8\upsilon+4} \cdot \frac{\|\boldsymbol{x}-\boldsymbol{x}'\|}{l}\right)^{\upsilon-i}\end{aligned} \tag{4.3}$$

where Γ denotes the gamma function. The non-negative integer υ determines and equals the order of differentiability of resultant function samples.

The Matérn kernel as defined above is interesting in the context of time-series modelling, since for a one-dimensional input space (e.g. time) a GP with this kernel is equivalent to a continuous-time AR(υ)-process [21]. Rasmussen and Williams [14, p. 85] argue that $\upsilon = 1$ and $\upsilon = 2$ constitutes the most interesting cases for machine learning, because for $\upsilon = 0$ function samples becomes very rough—they are not differentiable at all—and for higher υ it can, from a basis of noisy training data, be very hard to distinguish between different values of υ. Indeed, it can be hard to differentiate even between finite values of υ and the limit $\upsilon \to \infty$, in which the Matérn kernel becomes the squared exponential. One-dimensional kernel evaluations and corresponding samples are for the SE and the two Matérn variants displayed in figures 4.1b and 4.1c respectively.

[5] See Stein [22] for examples.



(a) Linear, x' fixed **(b)** SE **(c)** Matérn, $l = 1$

Figure 4.1: *Upper row*: plot of $\kappa(x, x')$ for the respective kernel types. *Lower row:* sample functions from the corresponding GP priors.

4.1.3 Expressing periodicity

Following Mackay [23] we can, given any covariance function $\kappa_a(\boldsymbol{u}, \boldsymbol{u}')$, introduce the mapping $\boldsymbol{x} \mapsto \boldsymbol{u}(\boldsymbol{x})$ and retrieve a new covariance function by defining

$$\kappa_b(\boldsymbol{x}, \boldsymbol{x}') := \kappa_a\Big(\boldsymbol{u}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}')\Big). \tag{4.4}$$

If κ_a is a stationary kernel the above mapping will, in general, be non-stationary with uniform variance. The mapping $\boldsymbol{u}$ is arbitrary—it need not be linear, invertible or have equal dimensionality of domain and codomain; consider for instance a one-dimensional domain on which we define the mapping $\boldsymbol{u}(x) = [\sin(x), \cos(x)]$ and choose κ_a to be the SE kernel in $\boldsymbol{u}$-space. Then we attain a covariance function for a *periodic* random function,

$$\kappa_{\text{SE;per}}(x, x') := \sigma_\kappa^2 \exp\left(\frac{-2\sin^2\left(\frac{1}{2}(x - x')\right)}{l^2}\right), \tag{4.5}$$

after making use of some trigonometric identities. The hyperparameter l of the periodic kernel determine the *period* of the resulting function samples. Our choice of kernel is not limited to the squared exponential: any stationary covariance functions will work, incorporating different assumptions of smoothness.

4.2 Compound kernels

Two kernels will, when composed together additively or multiplicatively, yield a new valid covariance function in the sense that the resulting covariance matrix is symmetric positive-semideinite [14, sec. 4.2.4]. Hence we may add or multiply known kernels freely and doing so allows us to build priors expressing many different qualities.

For instance, multiplying q linear kernels results in a prior for polynomials of degree q. A product of the periodic kernel in equation (4.5) and the linear kernel can model functions with growing amplitude, where the marginal standard deviation of the function grows linearly away from the location c in the kernel definition [18]. Adding to the covariance function a linear or a white noise kernel incorporates assumptions of global trend or additive noise, respectively. The possibilities are extensive and a more in-depth examination can be found e.g. in the works of Duvenaud [18, ch. 2].

4.2.1 Local periodicity

One interesting combination that will become exceedingly useful is the product of a periodic and a stationary covariance function. Whereas the family of standalone periodic kernels models *exact* periodic structure globally across all the training data, they have in combination with a stationary kernel the ability to model *local* periodic structure. To expand on this, assume that we are provided a periodic kernel $\kappa_{\text{per}}(t, t')$ where t and t' are elements on the timeline. If we multiply it by a stationary kernel $\kappa_{\text{stat}}(t, t')$, the resulting covariance function models a periodic structure that can change as time goes by, at a rate of change that is governed by the length-scale of κ_{stat}.

If we relax the condition that κ_{stat} is defined on the timeline and allow instead inputs from any real-valued input space $\mathcal{X}$, we can model functions with a periodic structure that change over this general space. In a pertinent example, we choose $\mathcal{X}$ to be a space of weather features and thus achieve a GP prior for periodic functions whose exact periodicity is dependent on the current weather conditions.

4.3 Remarks on the use of multi-dimensional input

Most time-series are not purely dependent on its former temporal states, but changes additionally with respect to exogenous factors. Of the kernels defined so far, most of them accept multi-dimensional input and can be used as-is, but there are certain aspects to keep in mind when deciding how to best combine kernels over multiple dimensions.

4.3.1 Multiplicative kernels allow for automatic relevance determination

Multiplying together many one-dimensional stationary kernels, each defined on an individual input dimension, leads to flexible GP models with *Automatic relevance determination (ARD)*. As an example, a product of SE kernels, each with variance σ_j^2 and length-scale

l_j, becomes

$$\kappa_{\text{SE-ARD}}(\boldsymbol{x}, \boldsymbol{x}') := \prod_{j=1}^{d} \sigma_j^2 \exp\left(-\frac{1}{2}\frac{(x_j - x_j')^2}{l_j^2}\right) = \sigma_\kappa^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{d}\frac{(x_j - x_j')^2}{l_j^2}\right), \tag{4.6}$$

where d is the dimensionality of the input. The length-scales l_d implicitly determines how relevant each input dimension is in explaining the variation in the function being modelled. A short length-scale implies that small variations along the corresponding input dimension can still impact the function values to some degree. The opposite is true for long length-scales, where variations in the input dimension is essentially neglected.

Pure multiplicative covariance structures allow models to learn distinct features for each combination of input values, potentially leading to very flexible distributions over functions. There are, however, some problems involved with using multiplicative kernels like the multi-dimensional SE defined above: they are *smoothing interpolators*, and so for any dimension d we can typically not extrapolate more than l_d units away from the closest known training point on this dimension before the GP mean function takes over as the most dominant correlating factor. In high-dimensional input spaces, the problem can manifest itself as an instance of the more general *curse of dimensionality*, to which multiplicative kernels are especially susceptible [24]. The phenomenon occurs when there are large amounts of possible combinations the input can take, and exponentially many training examples are required in order to sufficiently cover the relevant parts of the input space.

4.3.2 Additive kernels aid extrapolations away from training data

Having additive structure present in the kernels yield distributions that are more restricted, but which allow us to confront the curse of dimensionality with the *blessing of abstraction*: the more structure we explicitly account for, the less data we need. We have already covered one method of adding structure, namely by employing covariance functions of the (locally) periodic type. Adding kernels together over different dimensions also automatically encodes more structure, because it results in models for which it is sufficient that *some* of the inputs correspond with historical data in order to confidently predict into new, unseen situations [25]. Not all situations are fit for additive kernels, however—time-series have additive dependence on covariates only occasionally.

4.4 A recursive kernel to mimic deep architecture

As previously stated, the connection between single-layer neural networks and GPs have long been known. More recently—motivated by the wish to combine the intriguing recent successes of deep architectures, such as multilayer neural nets, with the "elegance of kernel methods"—Cho and Saul [26] formulated a family of kernels that "mimic the computation on large neural nets". They did so through a *recursive* kernel definition, which was later used in the GP literature with empirical success [27], and will be tested in this thesis in relation to power demand forecasting.

The recursive kernel class is expressed in terms of the angle ϑ between two vectorial inputs, which may be written as

$$\vartheta(\boldsymbol{x}, \boldsymbol{x}') := \cos^{-1}\left(\frac{\boldsymbol{x} \cdot \boldsymbol{x}}{\|\boldsymbol{x}\|\,\|\boldsymbol{x}'\|}\right). \tag{4.7}$$

This relation leads to the following covariance function being referred to as the *arc-cosine* kernel:

$$\kappa_{\text{acos}}^{(\eta)}(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{\pi}\,\|\boldsymbol{x}\|^{\eta}\,\|\boldsymbol{x}'\|^{\eta}\,J_{\eta}(\vartheta), \tag{4.8}$$

where $\eta \in \mathbb{N}$ denotes the *order* of the recursion and J_η is a family of functions that captures all the angular dependence. The arguments of ϑ have been ignored for simplicity. Although J_η can be defined for arbitrary order η, we will confine ourselves with the first and second orders, following

$$\begin{aligned} J_1(\vartheta) &= \sin\vartheta + (\pi - \vartheta)\cos\vartheta \\ J_2(\vartheta) &= 3\sin\vartheta\cos\vartheta + (\pi - \vartheta)(1 + \cos^2\vartheta). \end{aligned} \tag{4.9}$$

Analyses regarding positive-semidefiniteness and an examination of the properties of the arc-cosine kernel can be found by consulting the original paper [26].

4.5 The spectral mixture kernel

As discussed in section 4.2 we can achieve sophisticated kernels by combining a few basic ones. However, these compositions often involve tight restrictions and the kernels must be hand-crafted for specialised applications. Moreover, operations other than simple addition in many cases entail a lack of interpretability and change the resulting GPs in ways that are difficult to identify.

From a machine learning perspective, a requirement of human intervention in discovering patterns is not desirable, especially when it due to lack of interpretability involves imposing unknown inductive biases. To be able to use GPs as expressive statistical tools that can automatically discover hidden patterns in the data and also extrapolate intelligently away from the training examples, we will need to consider a more flexible class of kernels. However, it is not trivial to ensure positive-definiteness in kernel constructions that go beyond combining simple analytical forms while allowing simultaneously for truly adaptive basis functions. Besides, we still ought to maintain a requirement of interpretable results.

One method to bypass these obstacles was proposed by Wilson and Adams [28] and considers instead of direct kernel construction the design of the *spectral density* (or *power spectrum*) of the process, which is tied to an appurtenant covariance function through the inverse Fourier transform. A sufficient condition for kernel positive-semidefiniteness is strict positivity of the spectral density, which is much easier to enforce. The method embeds an assumption of *stationarity*, but no further inductive biases are imposed.

The derivation here will follow the original treatment of the authors [28], in which the following theorem is employed as theoretical support:

Theorem 3 (Bochner)**.** *A complex-valued function on $\mathbb{R}^P$ is the covariance function of a weakly stationary mean square continous complex-valued random process on $\mathbb{R}^P$ if and only if it can be represented as*

$$\kappa(\boldsymbol{\tau}) = \int_{\mathbb{R}^P} e^{2\pi i \boldsymbol{s}^\top \boldsymbol{\tau}} \psi(d\boldsymbol{s}), \tag{4.10}$$

where ψ is a positive finite measure.

A stationary kernel is shift-invariant, depending only on $\boldsymbol{x} - \boldsymbol{x}'$; hence if we denote $\boldsymbol{\tau} = \boldsymbol{x} - \boldsymbol{x}'$, Bochners theorem states that every stationary kernel can be expressed by an integral. Moreover, if ψ has density $S(\boldsymbol{s})$, then S is the spectral density of k and fully determines the properties of k through the Fourier duality relation

$$\kappa(\boldsymbol{\tau}) = \int S(\boldsymbol{s}) e^{2\pi i \boldsymbol{s}^\top \boldsymbol{\tau}} \boldsymbol{ds}, \tag{4.11}$$

$$S(\boldsymbol{s}) = \int \kappa(\boldsymbol{\tau}) e^{-2\pi i \boldsymbol{s}^\top \boldsymbol{\tau}} \boldsymbol{d\tau}. \tag{4.12}$$

Next, we must consider how to design the power spectrum of a process in terms of functions that we can easily evaluate. From the universal function approximation theorem that considers Gaussian mixtures we can approximate the power spectrum to arbitrary precision given enough mixture components. Thus, we may assume that $S(\boldsymbol{s})$ is a Q-component mixture of P-dimensional Gaussian distributions of which the qth component has mean $\boldsymbol{\mu}_q = (\mu_q^{(1)}, \ldots, \mu_q^{(P)})$ and covariance matrix $\boldsymbol{\Sigma}_q = \operatorname{diag}(\sigma_q^{(1)}, \ldots, \sigma_q^{(P)})$. The integral in equation (4.10) is tractable for this composition of S and evaluates to

$$\kappa_{\text{SM}}(\boldsymbol{\tau}) = \sum_{q=1}^{Q} w_q \cos(2\pi \boldsymbol{\tau}^\top \boldsymbol{\mu}_q) \prod_{p=1}^{P} \exp(-2\pi^2 \tau_q^2 \sigma_q^{(p)}), \tag{4.13}$$

where τ_p denotes the pth component of $\boldsymbol{\tau}$. The above is referred to as the *spectral micture (SM)* kernel. The hyperparameters are relatively simple to interpret: w_p weights the contribution of each mixture component, $1/\mu_q$ are the componenent periods and $1/\sqrt{\sigma}$ are length-scales determining the input scale, analogous to the variance hyperparameter of other kernels.

The SM kernel is from theorem 3 in theory able to approximate any stationary covariance function to arbitrary precision as long as the number of mixture components Q in the spectral representation is sufficiently large. It was applied to time-series in the original paper, successfully so in regards to both pattern discovery and extrapolation, and will in this thesis be tested on the power load data set with the same rationale in mind.

5 Scalable Gaussian process regression

Although the GP framework is elegant and powerful, it involves both in the training and prediction phases the inversion of a potentially gigantic $n \times n$ covariance matrix. This limits the usefulness whenever large data sets are involved: the memory and processing requirement scales quadratically and cubically with the number of data points, respectively. In the training step, especially, the covariance matrix must be inverted at every single iteration of the optimisation routine.

Many methods exist that allows sacrificing a bit of accuracy for much greater computability, many of which utilise the fact that it is generally unnecessary to evaluate the *full* covariance matrix and hence perform *exact* GP inference. Inverting instead an approximation may still yield adequate results. Having lately gained much attention and approval [29, 30], we will consider one approach which approximates the GP posterior by a variational method. But first, we discuss briefly a much simpler approach.

5.1 Sparse GP formulations

5.1.1 Choosing a subset of data

Contemplating different methods of scaling down the computational demand, the most intuitive approach is likely to simply split the data into subsets and perform inference on each set separately. In addition to reducing the computational demand this can have beneficial side effects; as an example, consider that we split the electricity demand data with respect to each hour of the day and applies exact GP regression on each hour separately. This procedure will avoid the need to specify complex intraday patterns, commonly called load profiles, that a full model would require in any useful inference scheme. In a review of load forecasting papers, trouble involved in modelling load profiles were observed to be very common [31].

When subsets of the training data are considered separately we may, however, severely restrict the ability of the regression model to pick up correlations between the different partitions or to get a realistic picture of the uncertainties [32]. In short: the quality of the inference may suffer. For this reason, we will also consider a more advanced approach involving inducing variables, which has to a greater degree the ability to capture rich patterns from data.

In the upcoming derivations we will neglect the explicit conditioning on the input data—it is *always* the case that the expressions related to posterior GP models are conditional on the inputs.

5.1.2 Inducing variables

Inducing variable[6] methods introduce an additional set of $m \ll n$ latent variables $\boldsymbol{u} \in \mathbb{R}^m$ corresponding to some arbitrary *inducing input points* $\boldsymbol{Z} \in \mathcal{X}$. The inducing variables lie on the posterior GP similar to $\boldsymbol{f}_*$ and are not (necessarily) restricted to coincide with training (or test) data. Instead of considering inference in the *full* GP in equation (3.8), an *approximated* posterior $p(\boldsymbol{f}_* \mid \boldsymbol{u})$ is instead formulated, in which all the information flow through $\boldsymbol{u}$, hence the name *inducing* variable. In figure 5.1 a one-dimensional example is displayed, in which a set if inducing inputs are chosen at random from the input space—we can thus not expect them to capably approximate the posterior GP.

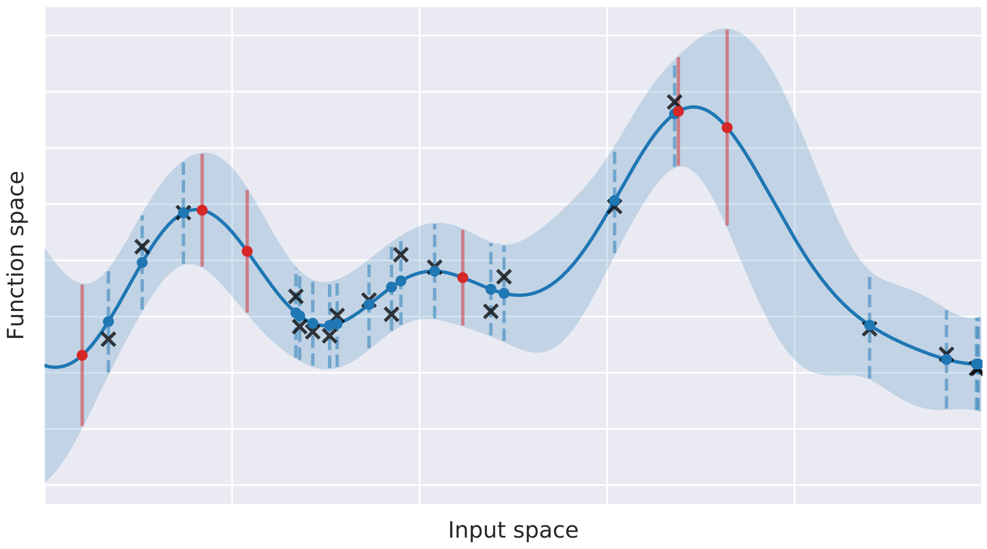


Figure 5.1: Inducing variables (red dots) and predicted function values (blue dots) are placed on the same posterior GP (blue line). The actual observations from which the GP is derived are depicted as black crosses. At inputs where we have corresponding observations, 95% confidence intervals are illustrated with vertical bars. The shaded area illustrates the same interval for inputs in-between observations.

To find a more optimal set of inducing variables we must define a suitable training procedure to select $\boldsymbol{Z}$ as well as the hyperparameters $(\sigma_n^2, \boldsymbol{\theta})$ of the kernel and the likelihood. These two problems are tightly conjoint: the set of inducing points depends on the latent function f, and f itself depends on the hyperparameters (thought the prior specification). Clearly, then, the optimal approach is to *simultaneously* pick inducing points and hyperparameters, that is, optimise some objective function that solves for $(\boldsymbol{Z}, \sigma_n^2, \boldsymbol{\theta})$ jointly.

An approximation to the exact log marginal likelihood,

$$\tilde{\mathcal{L}} = \log \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{0}, \tilde{\boldsymbol{K}} + \sigma^2 \boldsymbol{I}), \tag{5.1}$$

[6]Often also referred to as auxiliary, support or pseudo variables.

can be used to infer these quantities. Here, $\tilde{\boldsymbol{K}}$ is an m-rank approximation to the true covariance $\boldsymbol{K}$. Former state-of-the-art inducing variables methods considered different ways of approximating this matrix; this includes the projected process method [33] and the sparse pseudo-input method [34]. However, when optimising equation (5.1) with joint learning of inducing variables and hyperparemeters, there are no guarantee that the approximation will not *surpass* the true log marginal likelihood of the training data, in which case severe overfitting may occur. Titsias [35] adressed this issue by instead considering a *lower bound* of the true marginal likelihood using a variational procedure.

5.2 Variational learning of inducing variables

The approach of Titsias, which will be applied in this thesis, is, in essence, to formulate a variational distribution to approximate the true GP posterior (whose predictive variant is given in equation (3.8)). With some clever specifications, this leads to a lower bound for the marginal likelihood which may then be maximized with respecfurther workt to the inducing inputs and the hyperparameters simultaneously with continuous optimization.

To start of, imagine that the m latent function values $\boldsymbol{u}$ are associated with arbitrary inputs $\boldsymbol{Z}$.[7] The values of $\boldsymbol{u}$ reside in the same space as the test and training data points, but are otherwise independent of these. Consider a simple augmented model given by $p(\boldsymbol{f}, \boldsymbol{u})$; by standard statistical identities we may write

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{y} \mid \boldsymbol{f})p(\boldsymbol{f} \mid \boldsymbol{u})p(\boldsymbol{u}), \tag{5.2}$$

which, we notice, is equivalent to the original model $p(\boldsymbol{y}, \boldsymbol{f})$ as we can always recover the latter by marginalising out $\boldsymbol{u}$ from the former. So in essence the inducing points plays no active role at the moment (they are not model parameters, unlike σ_n^2 and $\boldsymbol{\theta}$), and there is no fear of overfitting when we specify them.

The ultimate goal is to use the auxiliary variables $(\boldsymbol{Z}, \boldsymbol{u})$ to facilitate inference about $\boldsymbol{f}$, and so they should be picked to be optimally useful for this inference. In his paper, Titsias defines the inducing variables as optimal if they result in $\boldsymbol{y}$ and $\boldsymbol{f}$ being *conditionally independent given* $\boldsymbol{u}$, in other words,

$$p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{y}) = p(\boldsymbol{f} \mid \boldsymbol{u}), \tag{5.3}$$

so that at optimality, the true augmented posterior will factorize as

$$p(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{y}) = p(\boldsymbol{f} \mid \boldsymbol{u})p(\boldsymbol{u}, \boldsymbol{f}). \tag{5.4}$$

The question is then: how can we locate these optimal inducing variables?

[7]The notation $\boldsymbol{f}_m$ and $\boldsymbol{X}_m$ are in the paper used for inducing variables and inputs, respectively, but $\boldsymbol{u}$ and $\boldsymbol{Z}$ have later become more standard in the literature.

5.2.1 A variational lower bound on the marginal likelihood

One possible optimisation procedure would be to minimise a distance metric between the true augmented posterior $p(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{y})$ and a variational distribution $q(\boldsymbol{f}, \boldsymbol{u})$ with respect to the inducing input points $\boldsymbol{Z}$. The distance metric could, as in standard variational inference, be the KL divergence between the two distributions. This is exactly the approach of Titsias, but with the additional key specification that the variational distribution $q(\boldsymbol{f}, \boldsymbol{u})$ must satisfy the factorisation that holds for optimal inducing variables, as given in equation (5.3). The two distributions to be compared under the KL divergence are then

$$\begin{aligned} \text{True:} \qquad & p(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{y}) = p(\boldsymbol{f} \mid \boldsymbol{u}, \boldsymbol{y}) p(\boldsymbol{u} \mid \boldsymbol{y}), \\ \text{Approximate/variational:} \qquad & q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f} \mid \boldsymbol{u}) \phi(\boldsymbol{u}), \end{aligned} \tag{5.5}$$

where $\phi(\boldsymbol{u})$ is a unconstrained ("free") variational Gaussian distribution over $\boldsymbol{u}$ that depend only on a mean and a covariance parameter. We call $q(\boldsymbol{f}, \boldsymbol{u})$ an *approximation* to the posterior because in general it may be difficult, and often even impossible, to locate inducing inputs that are sufficient statistics for $\boldsymbol{f}$. This holds true especially if the number m of inducing inputs is very low compared to n, in which case the flexibility of $\phi(\boldsymbol{u})$ can be severely strangled.

What remains now is to perform the actual minimisation of the KL divergence with respect to $\phi(\boldsymbol{u})$, which can be shown to be equivalent to maximising a variational lower bound on the true log marginal likelihood,

$$\mathcal{L}(\boldsymbol{Z}, \phi) \geq \int_{\boldsymbol{f},\boldsymbol{u}} p(\boldsymbol{f} \mid \boldsymbol{u}) \phi(\boldsymbol{u}) \log \frac{p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{u})}{\phi(\boldsymbol{u})} d\boldsymbol{f} d\boldsymbol{u}. \tag{5.6}$$

As the calculations involved in maximising the above with respect to ϕ are rather spaceous we will not repeat them here; they can be found in a technical report by Titsias [36]. The resulting bound are

$$\mathcal{L}(\boldsymbol{Z}) \geq \tilde{\mathcal{L}}(\boldsymbol{Z}) := \log \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{0}, \boldsymbol{Q}_{ff} + \sigma^2 \boldsymbol{I}\right) - \frac{1}{2\sigma^2} \operatorname{tr}(\boldsymbol{K}_{ff} - \boldsymbol{Q}_{ff}), \tag{5.7}$$

with

$$\boldsymbol{Q}_{ff} := \boldsymbol{K}_{fu} \boldsymbol{K}_{uu}^{-1} \boldsymbol{K}_{uf}. \tag{5.8}$$

The quantity in equation (5.7) is a lower bound for the true marginal log likelihood for any number of inducing inputs m. The first term says to fit the data $\boldsymbol{y}$, whereas the second term encourages minimisation of the variance of $\boldsymbol{f}$ given $\boldsymbol{u}$ (the term inside the trace operator is $\operatorname{cov}(\boldsymbol{f} \mid \boldsymbol{u})$). Evaluating the bound has complexity $\mathcal{O}(nm^2)$ and can be maximised with respect to $(\boldsymbol{Z}, \boldsymbol{\theta}, \sigma^2)$ using any continous optimisation tool. If the bound becomes tight, i.e., $\widetilde{\mathcal{L}} = \mathcal{L}$,[8] the trace term tends to zero and the original log marginal

[8]Which may happen if there are redundancy in the data, or if $m \geq n$, which is in practical terms pointless.

likelihood is recovered.

5.3 Variational predictive posterior

In order to perform predictions with the newly learned parameters $(\boldsymbol{Z}, \boldsymbol{\theta}, \sigma_n^2)$ we must state the approximate GP posterior that arise from the variational formulation. When ϕ is chosen to be a "free" Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, it can be shown that the augmented model (in combination with the definition of optimal inducing variables and the Gaussian likelihood assumption) leads quite straightforwardly to an approximate predictive posterior on the form

$$\begin{aligned} p(\boldsymbol{f}_* \mid \boldsymbol{u}) &= \mathcal{N}\left(\bar{\boldsymbol{f}}_*^u,\ \operatorname{cov}(\boldsymbol{f}_*^u)\right) \\ &= \mathcal{N}\left(\boldsymbol{K}_{*u}\boldsymbol{K}_{uu}^{-1}\boldsymbol{\mu},\ \boldsymbol{K}_{**} - \boldsymbol{Q}_{**} + \boldsymbol{K}_{fu}\boldsymbol{K}_{uu}^{-1}\boldsymbol{\Sigma}\boldsymbol{K}_{uu}^{-1}\right), \end{aligned} \tag{5.9}$$

with $\boldsymbol{Q}_{**}$ as in equation (5.8) for test points rather than data points. Up until now the distribution ϕ have been completely free—the lower bound was maximised without the need to specify the mean and covariance parameters—but to perform predictions using equation (5.9) they must be calculated. Titsias derives them analytically by diffentiating equation (5.6) with respect to $\phi(\boldsymbol{u})$ without imposing any further restrictions on ϕ. The optimised parameters of the distribution are

$$\begin{aligned} \boldsymbol{\mu}_{\text{opt}} &= \sigma^{-2}\boldsymbol{K}_{uu}\boldsymbol{C}\boldsymbol{K}_{uf}\boldsymbol{y}, \\ \boldsymbol{\Sigma}_{\text{opt}} &= \boldsymbol{K}_{uu}\boldsymbol{C}\boldsymbol{K}_{uu}, \end{aligned} \tag{5.10}$$

where $\boldsymbol{C} := [\boldsymbol{K}_{uu} + \sigma_n^{-2}\boldsymbol{K}_{uf}\boldsymbol{K}_{fu}]^{-1}$. The above may be substituted into equation (5.9) to perform approximate GP predictions with time complexity $\mathcal{O}(nm^2)$.

Figure figure 5.2 displays how the approximated predictive distribution is affected by the placement of $m = 15$ inducing inputs for a short slice of the power load dataset with $n = 170$ data points. In the above panel, an optimisation routine has maximised the marginal likelihood lower bound with respect to $(\boldsymbol{\theta}, \sigma_n^2)$, whereas in the lower panel we allow additionally for optimisation of $\boldsymbol{Z}$. A more evenly spaced distribution of inducing variables are here favoured, and the result is more expressive and better fit to the data. Both cases had the same prior imposed: a one-dimensional SE plus a linear kernel.

Figure 5.3 illustrates how changing the number of inducing variables affects the flexibility of the resulting GPs for a segment of power load data of length $n = 360$. In each panel the GP is equipped with a SE plus a linear kernel and each undergo the same optimisation routine over $(\boldsymbol{\theta}, \sigma_n^2, \boldsymbol{Z})$. Clearly, the flexibility of the resulting GP suffers greatly when m is set very low. The optimal m depends on the situation, and the choice constitutes a balancing argument between accuracy and computability. Note that the data points seem to always fit well within the calculated confidence intervals; in fact, overestimation of the noise is a known drawback of the method [30].

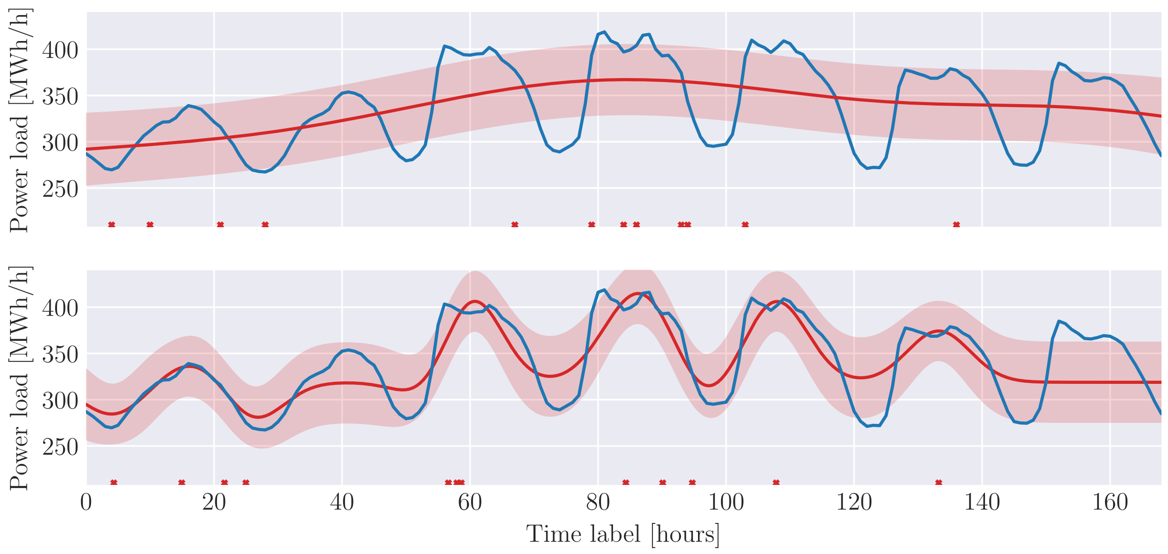


Figure 5.2: The placement of inducing inputs changes the representative power of the posterior GP. Blue and red curves represent (interpolated) true values and GP predictions, respectively. Red dots indicate the placement of the inducing inputs (their position in regard to the y-axis is coincidental). In the upper panel, $\boldsymbol{Z}$ are initialised randomly, whereas in the lower panel the same initial $\boldsymbol{Z}$-values are fitted according to the variational method. Shaded areas depict 95% confidence intervals.

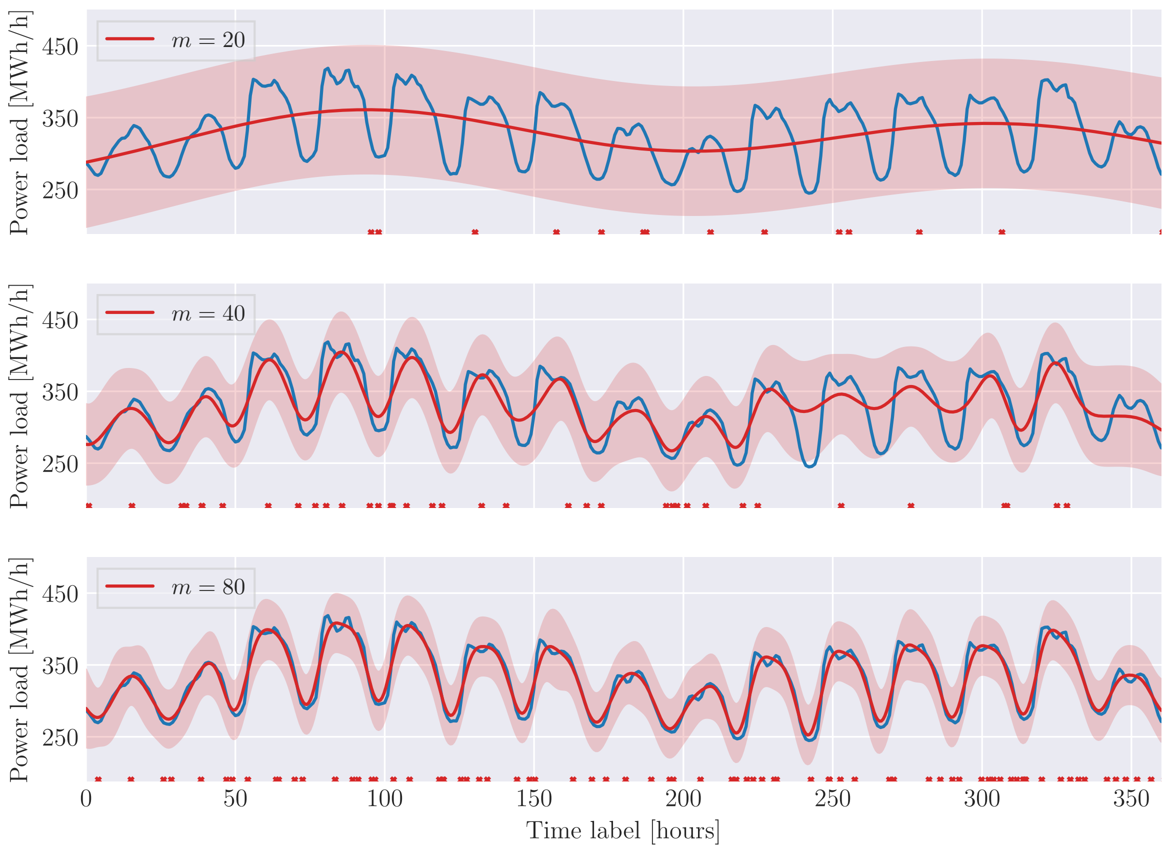


Figure 5.3: For a different number of inducing inputs m, as indicated in each respective panel, the corresponding approximated posterior is shown. Blue and red curves represent (interpolated) true values and GP predictions, respectively. Red dots indicate the placement of the inducing inputs; their position in regards to the y-axis is coincidental. Shaded areas depict 95% confidence intervals.

6 NTE's data set of electricity demand

The dataset provided by NTE includes historical values of power load and predictions thereof, weather data and weather forecasts. To establish domain knowledge, from which we can build functional GP models to forecast electricity consumption in Nord-Trøndelag, we here provide a more in-depth look at the data set, in order to search for exploitable patterns.

An overview of the time-series included in the data set is given in table 6.1. Each series consist of 61,368 measurements, spanning from the year 2011 up to and including the year 2017. A predictive counterpart complements each series; details for predictive electricity demands and weather data are found in sections 6.1 and 6.2, respectively.

Table 6.1: Complete list of time-series present in the dataset. Each series have a corresponding predictive dual.

Type	**Included time-series**	**Unit**
Electricity demand	1	MWh/h
Air temperature	3^{**}	°C
Wind speed	3^{**}	m/s
Solar irradiance	3^{**}	W/m^2

** At selected county locations.

6.1 Analysis of the power load data

The principal information contained in the dataset is the power demand time-series, each entry giving electricity demand in MWh/h (megawatt-hours per hour) averaged over the whole preceding hour relative to the time-label. The slight redundancy in the unit definition conveys the fact that we are not considering momentary demands. Moreover, the values are the aggregated demand for the whole of Nord-Trøndelag, from residential as well as industrial sources, which together constitute the primary source of electricity demand in the county. A simple plot of the values was given in figure 1.1 in the introduction.

A time-series of predicted electricity consumption is also included in the set; the values are calculated by NTE's current predictive model which utilises Kalman-filtering. The algorithm is originally developed by SINTEF[9] and commercially maintained by Powel AS[10]; it is proprietary and hence no further information are granted to us about its inner workings.

6.1.1 Linear trend

After calculating monthly means for every year in the data set, a least-squares regression line are fitted to the points as shown in figure 6.1. The slope indicates a yearly average

[9]An independent research organisation headquartered in Trondheim, Norway. Web: `www.sintef.no`
[10]A Norwegian supplier of software solutions. Web: `www.powel.com/no/`

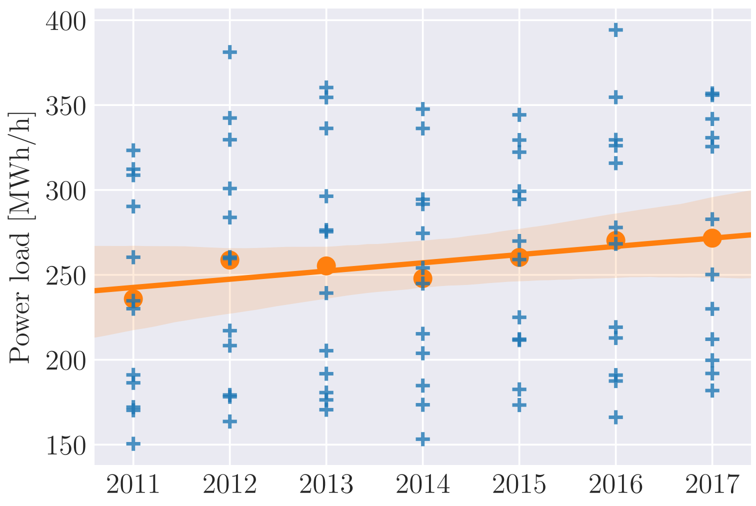
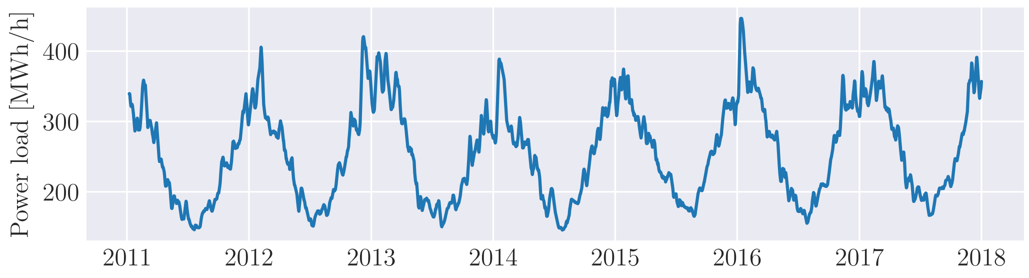


Figure 6.1: Linear regression for mean monthy values (blue crosses) grouped by year. The orange dots show yearly averages. Translucent band indicate 95% confidence in the regression line.

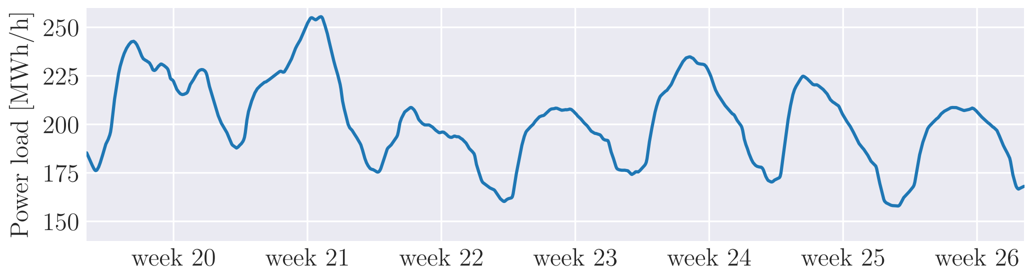
increase in power demand of 4.8 MW.

6.1.2 Seasonalities

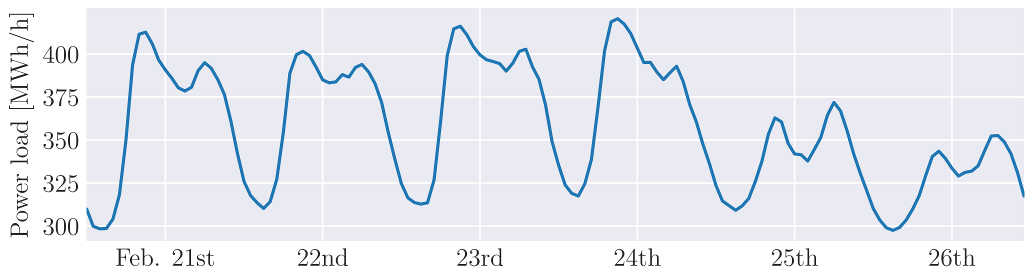
Annual, weekly and daily periodic patterns, which are very clearly present in the power load time-series, are portrayed in figure 6.2. To remove some variability, and with that more clearly demonstrating the annual and weekly periodic patterns, a rolling mean function with weekly and daily window sizes have been applied to the data in figures 6.2a and 6.2b, respectively.



(a)



(b)



(c)

Figure 6.2: Known seasonalities in the power load time-series. In (a), annual periodic patterns of the complete span of the data set is shown. In (b), a data slice containing multiple weeks from 2017 shows weekly patterns. To emphasize the periodic patterns, a rolling mean operator with a weekly and daily window has been applied to reduce data variability in (a) and (b), respectively. In (c), a data segment from February 2017 depicts daily periodic patterns. The labels on the horizontal axis are here centred at noon.

.

6.1.3 Load profiles

The intraday patterns of the data set, often referred to as *load profiles*, change depending on various latent factors, of which some can be estimated from domain knowledge. Holidays, for example, induce a pattern that is distinct from any other days. In figure 6.3 the electricity consumption values are grouped with respect to various factors and averaged over each hour. Most notably are the differences between weekdays, weekends and holidays, but some clear disparity exists also between Saturdays and Sundays. More subtle, when compared to the rest of the weekdays, Fridays exhibit a slightly lower consumption in the evenings, while Mondays show a trend of having generally lower consumption throughout the day.

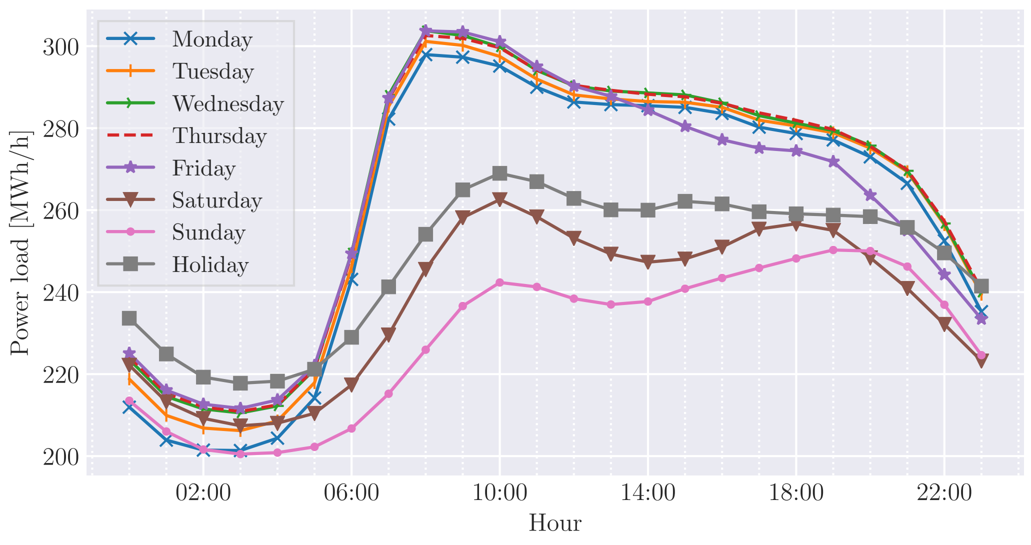


Figure 6.3: Daily profiles. The complete data set are partitioned with respect to factors as indicated by the labels, then averaged by hour.

6.1.4 Seasonal heteroscedasticity

The variance in the observations is not stable throughout the day. Figure 6.4 illustrate the hourly distribution of points by a box-plot for two different parts of the year, where we observe that the early morning hours have the greatest variability. Another interesting observation is that values recorded in March appear to follow a notably heavier-tailed distribution than corresponding observations from August.

The figure brings to light two cases of seasonal heteroscedasticity present in the data. The phenomenon could potentially cause difficulties if we attempt to formulate a model with the intention of capturing too much structure at once.

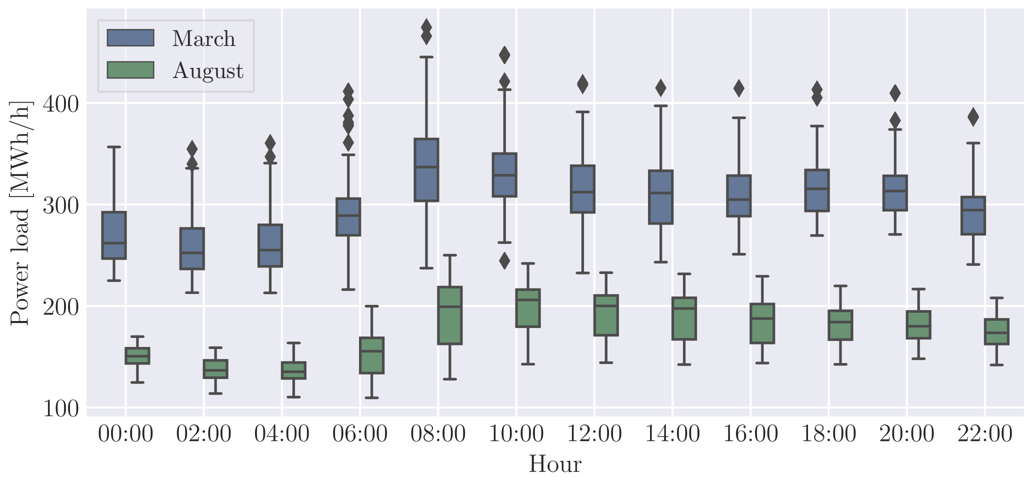


Figure 6.4: The distribution of power consumption values for each hour of the day, shown for two different months. Included data are from 2015 and newer; weekends and holidays are excluded. The streaks inside the boxes indicate the median and the box bodies extend from the lower to the upper quartiles. The whisker lengths extend an additional 1.5 times the interquartile ranges. Data points outside this range are marked with diamonds.

6.2 Weather data

Temperature, solar irradiance and wind speeds are recorded at the three most populous locations in Nord-Trøndelag: Stjørdal, Steinkjer and Namsos along with forecast variants thereof. The weather forecasts are provided by StormGeo[11] and are exactly those which NTE's currently employed model for power demand had access to at prediction time.

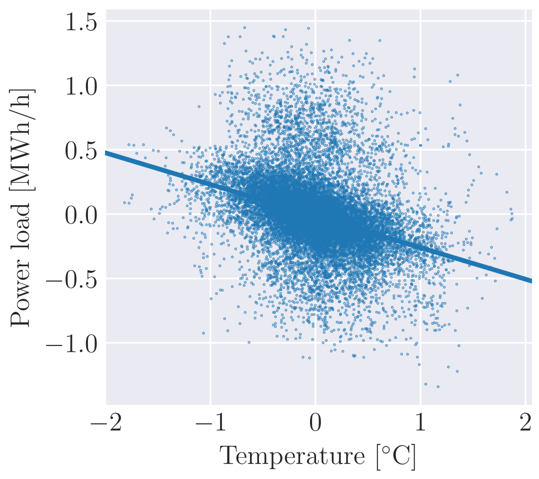
Even as Nord-Trøndelag is a sizable county, the weather conditions on the three different metering locations are strongly correlated, so to reduce the amount of redundancy a simple dimensionality reduction scheme will be applied in which for each distinct weather type a single time-series are constructed as a weighted average over the three locations. From NTE a suggestive set of location weights are provided, which are approximated based on demographic considerations and will be used here without further verification. In the following, whenever *temperature*, *solar irradiance* or *wind speed* is mentioned, it is tacitly referred to the respective weighted version.

6.2.1 Correlation of weather and power loads

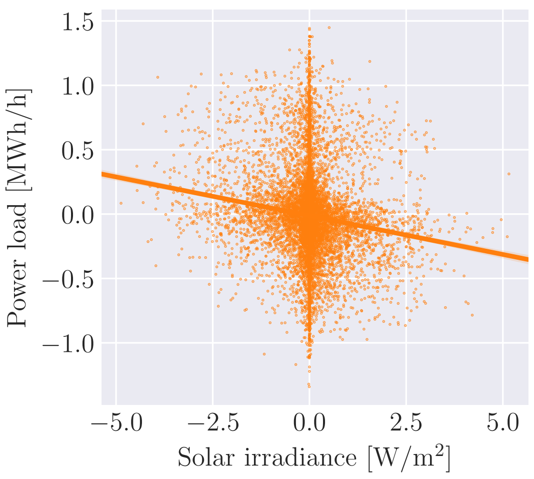
In order to reveal possible relationships between the time-series in the data set, power load values of 2016 and 2017 are plotted with respect to the coincident weather conditions in figures 6.5a to 6.5c. Each time-series is standardised to zero mean and unit variance so that values can more easily be compared across data types. Moreover, to remove

[11]Privately held supplier of meteorological services, headquartered in Bergen, Norway. Web: www.stormgeo.com

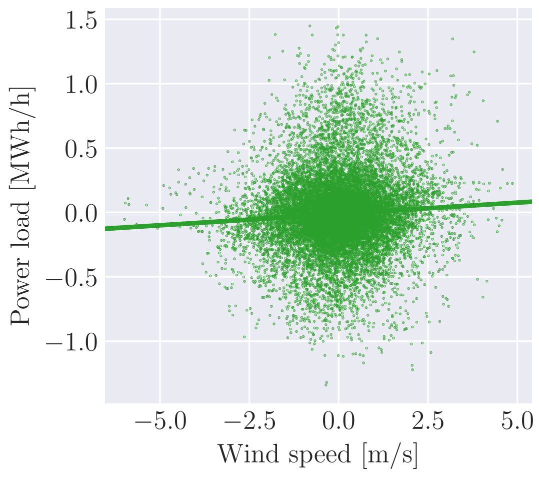
correlations due to daily cycles, a twenty-four-hour difference operator is applied to each time-series before they are plotted against each other and fitted with a least-squares regression routine that de-weights outliers.



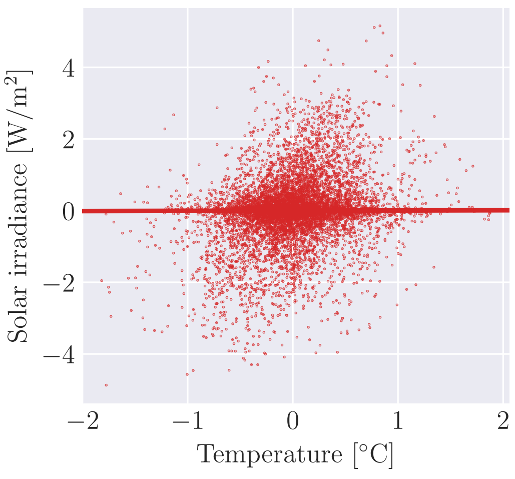
(a) Power load w.r.t. temperature. The regression slope is -0.27.



(b) Power load w.r.t. solar irradiance. The slope of the regression is -0.08.



(c) Power load w.r.t. wind speed. No significant linear relationship were found.



(d) Solar irradiance w.r.t. temperature. No linear relationship was discovered whatsoever.

Figure 6.5: Power load values from 2016 and 2017 are plotted with respect to different weather measurements. All time-series are standardised and exposed to a daily-difference transformation. A least-squares regression line is fitted to the data in each panel.

The resulting lines show that temperature is by far the most important linear psredictor of the three. The slope is -0.27, which is intuitively reasonable, since when the temperature is higher today than yesterday, then the power demands today are likely to be lower than yesterday. In regards to wind speed, no linear relationship of significance was found, whereas the solar irradiance attains a slope of -0.09. To analyse if this dependency can potentially be attributed to the temperature differences—or vice versa—due to correlations between the two weather components, one is plotted with respect to the other in figure 6.5d. Surprisingly, no relationship was found between them. As more sunny days are by most attended with warmer weather, this result might appear nonsensical. However, sunless days implies a cloud cover, which has a warming effect on temperature. These two factors, and presumably many more latent ones, appear to annul any linear relationship between the two time-series. In conclusion, solar irradiance seems to provide information about the power demands that are not contained in the temperature measurements, and are likely a useful covariate to include in a power load forecast model.

6.3 Pattern recognition with the spectral mixture model

To look for additional patterns in the data, a GP model fitted with a spectral mixture kernel is applied to a quarter-year slice of power demand data, with time as the only input feature. The resulting spectral density is depicted in figure 6.6 together with the empirical spectrum, calculated as the mean of the absolute values of the squared discrete fourier transform of the data. For a review of this algorithm, see e.g. Press et al. [37, Chp. 12]. The density peaks at daily, weekly and two-day frequencies, as is to be expected. However, the amount of useful information the kernel provides is, sadly, slightly underwhelming. We will discuss the SM kernel in the context of forecasting power demands in section 9.4.

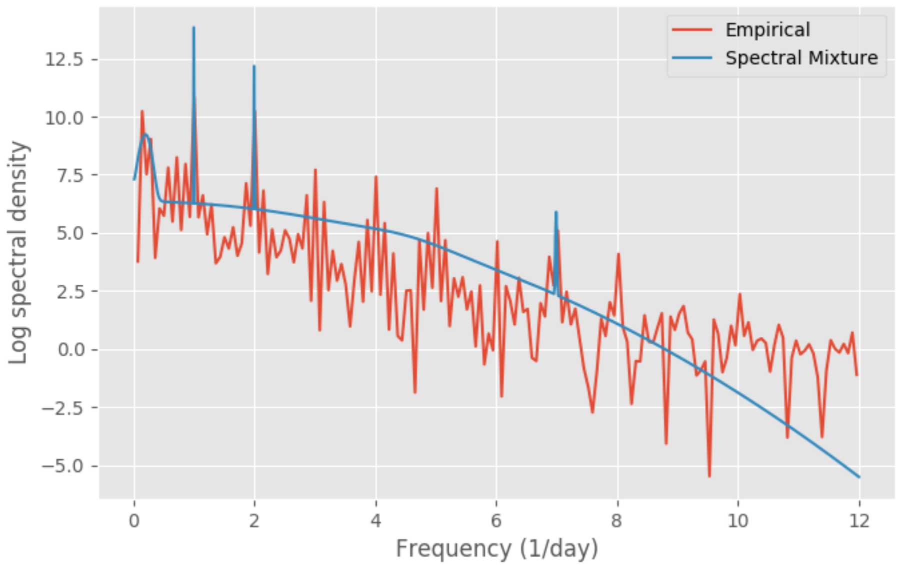


Figure 6.6: A spectral mixture kernel with $Q = 10$ components is applied to a section of power demand data, and the resulting spectral density is displayed together with the empirical spectrum.

6.4 Other considerations

6.4.1 Outliers

By inspection, there are no missing values in the dataset. Whether or not there are outliers present, however, is much harder to answer in general, and automated attempts at outlier detection have been unsuccessful. Occasionally the power consumption peaks drastically at a rate that seems unlikely, but when looking in parallel at the weather conditions the spike can often be attributed to a sudden and severe drop in temperature. For this reason, we should be reluctant to disregard abrupt spikes as outliers. On the other hand, abnormal *vallies* in the power load data often happens despite stable weather conditions. There are two evident explanations for this: measurement errors or power outages in parts of the county. Both will be considered outliers and removed from the *training* set manually. This way erroneous values will not pollute model predictions for other points, while the validity of the test results is preserved.

Figure 6.7 displays an example where points are believed to be collective outliers from a power outage. The load profile of the affected day is somewhat preserved, but the constant factor by which the power values differ from neighbouring days are extreme and could bear witness to a situation where the electricity is cut off from parts of the consumers.

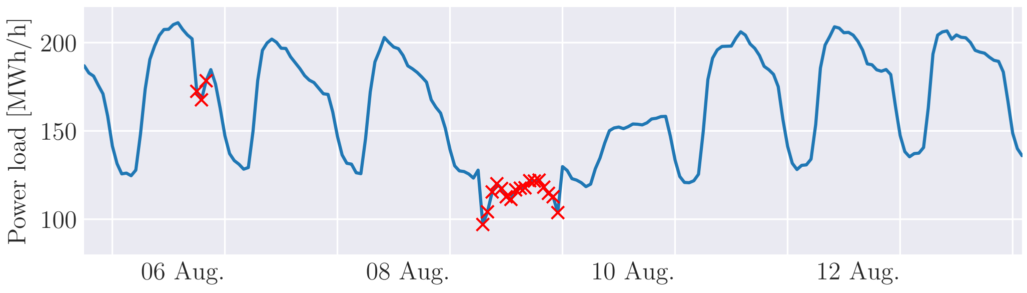


Figure 6.7: Electricity consumption of 2014, where red crosses indicate values that are believed to be outliers caused by power failure.

6.4.2 Daylight Saving Time

Due to the practice of Daylight Saving Time (DST), considerations must be taken around the transition points to avoid offsets in the model. At spring every year, usually a Sunday in March or April, a leap from 02:59 to 04:00 occurs. In the fall, usually a Sunday in October or November, the hour 02:00 appears twice. To handle the issue, certain authors correct for DST by adding an hour to the time-series at the spring transition, filling it with the mean of the two adjacent time-series values, while the extra hour occurring in the fall is simply removed. Hinman et.al.[38] argues that electricity consumers are likely to treat the original 02:00 hour as they normally would do, and simply treat the added 02:00 hour as extra sleeping time.

To avoid imposing additional assumptions, a different approach will be considered. All time-labels are converted into UTC format, and all internal procedures in the code use these instead of time-zone aware labels. Whenever user interaction is concerned, the time-labels are converted back to the current time-zone with an additional parameter added to all time-labels, indicating whether it is summer or winter time; e.g., 03:00 + 02 in summer and 03:00 + 01 in winter. With this procedure, when the values of a time-series are shifted by k hours, the lag order is automatically adjusted to $k + 1$ or $k - 1$ whenever a DST transition is crossed.

7 A family of power load forecasting models

Theory and knowledge of the preceding sections will here be unified with the motivation of obtaining a family of predictive models for electricity consumption that incorporates necessary structure while still being flexible and expressive. Various tests will in sections 7.4 to 7.6 experimentally estimate an optimal member of said family, leading to a satisfactory final model choice that can ably predict future electricity consumption. More general model properties will be tested in section 8, and a more in-depth discussion of the results is provided in section 9.

7.0.1 Precise formulation of the prediction problem

Our main goal is to formulate a set of models that can optimally forecast electricity consumption while being subject to the limitations that apply in the Norwegian power market. As previously mentioned, NTE must every day at 10:00 report to Nord Pool estimated electricity consumption values for Nord-Trøndelag for the full next day. This exact time-point is in this thesis referred to as *prediction time*. If we let hour h at a date d be denoted by t_h^d, then t_{10}^d is the prediction time for this day. The problem at hand can then be stated as to predict power loads at all the time-points in the set $\{t_0^{d+1}, \dots, t_{23}^{d+1}\}$, that is, an interval from 14 to 38 hours into the future from t_{10}^d. This set will be referred to as the *predictive window*.

7.1 Integration of periodic load components

It is apparent from the discussion in section 6.1 that any operative power load forecast method must at a bare minimum incorporate structure in the form of weekly, daily and annual periodicities. From section 4, we know how to design kernels that incorporate certain periodicities and, moreover, that assembling kernels multiplicatively will allow the resulting GP model to learn different functional characteristics for each combination of inputs. We therefore define

$$\kappa_{\text{per}}(\boldsymbol{x}, \boldsymbol{x}') := (\kappa_{\text{daily}} \cdot \kappa_{\text{weekly}} \cdot \kappa_{\text{annual}})(\boldsymbol{x}, \boldsymbol{x}'), \tag{7.1}$$

where the component kernels are of any stationary type. The kernels κ_{daily} and κ_{weekly} take as input two-dimensional vectors of sine-cosine transformed time labels, as discussed in section 4.1.3, with respective periods $p_{\text{day}} = 24$ and $p_{\text{week}} = 24 \times 7$. To learn the yearly seasonalities, κ_{annual} recieves as input a transformation of the time labels with periodicity $p_{\text{year}} = 24 \cdot 365.25 = 8766$. However, when observing that the annual periodic pattern of the data closely resembles the output of a pure periodic function, the cosine component is ommitted from κ_{annual} to reduce redundancy.

The kernel in equation (7.1) incorporates all known seasonal patterns of the electricity consumption data, but are in its current state globally periodic and unable to learn local features. Figure 7.1a shows an example where the periodic structure of july 2016 are learned and used to predict for july 2017; clearly, no departure from absolute periodicity is allowed. In section 7.2, κ_{per} will be extended to allow for local deviations.

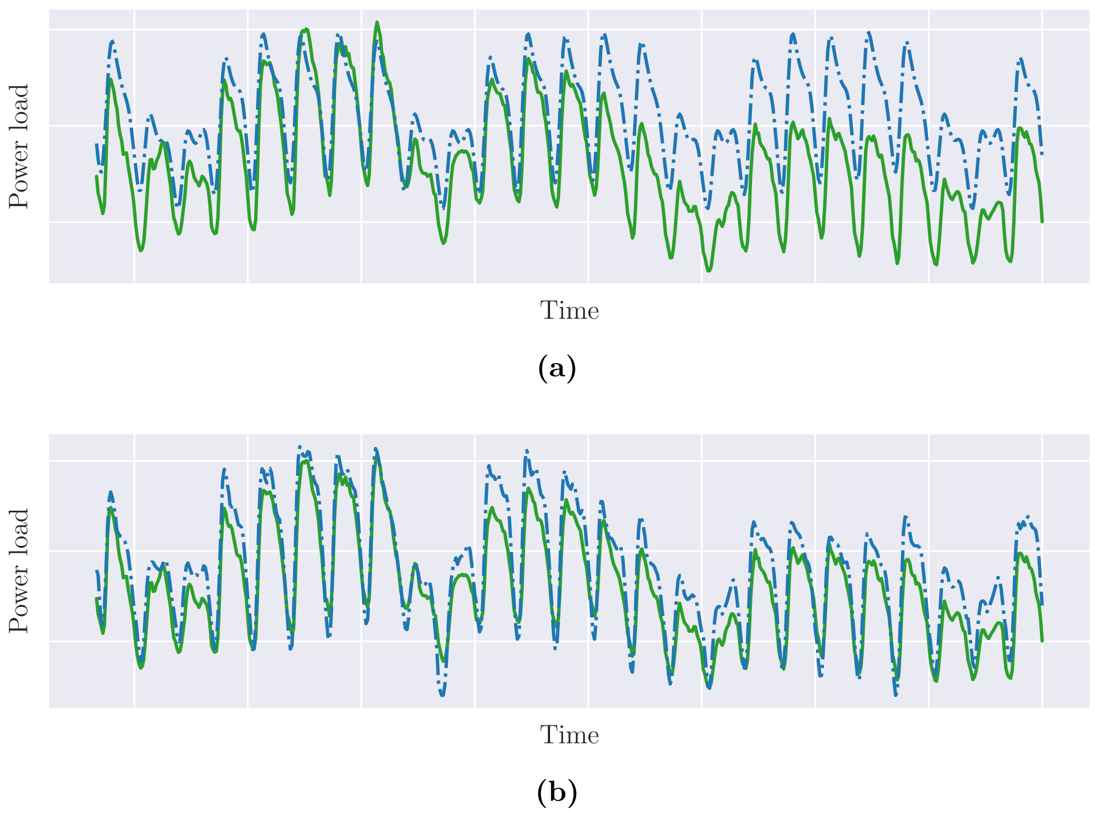


Figure 7.1: Training on data from july 2016, two periodic GP models predicts for july 2017. Dotted blue lines are predictions, solid green lines are target values. Panel (a) shows periodic patterns resulting from κ_{per}. In panel (b), a one-dimensional SE kernel, recieving the single temperature prediction time-series as input, are multiplied by κ_{per}, yielding a much more compromising periodic structure.

7.2 Incorporating recent knowledge and weather outlook

It is undeniably the case that weather conditions in general—and temperature specifically—affects the electricity consumption. At prediction time the best information we have available about the weather in the predictive window is from the weather forecasts. On the basis of the discussion in section 4.2.1, it is reasonable to incorporate these forecasts into the model as inputs to a stationary kernel, or a compound thereof, that is multiplied with κ_{per} to create a locally periodic covariance function of which the exact periodicities are weather dependent. To the best of my knowledge, this is a novel approach to the problem of electricity demand forecasting. In figure 7.1b the effect is illustrated: a single one-dimensional SE kernel, taking as input the time-series of temperature forecasts, are multiplied by κ_{per} and the resulting GP model is fit to the same strip of data as the example in the former section. We observe that the rigid periodic pattern loosen, leading to notably improved predictions.

Further improvements can be readily obtained by incorporating additional information into the localizing kernel. This can be recently observed data, whether it be electricity consumption data or historic weather data, holiday information or other potential covariates. If we model correlations due to known, observed data with κ_{obs} and let data that are based on estimated future information, e.g. the weather forecasts, be handled by κ_{outlook}, then all possible combinations of localized periodic models can be expressed by

$$\kappa_{\text{load}}(\boldsymbol{x}, \boldsymbol{x}') := [\kappa_{\text{per}} \cdot \zeta(\kappa_{\text{outlook}}, \kappa_{\text{obs}}) + \kappa_{\text{WN}}](\boldsymbol{x}, \boldsymbol{x}'), \tag{7.2}$$

where the white-noise kernel is included so that we can fix σ_n^2 if so desired, and ζ is a function that relates the observed and estimated data under the sole constraint of having to preserve positive-semideiniteness. For simplicity only two choices of ζ will be considered: a *sum* of the two input kernels, and a *product* thereof—we will refer to ζ as *additive* and *multiplicative* for these two cases, respectively. As discussed in section 4.3, purely multiplicative models are flexible but often requires much data. Incorporating some additive properties can help overcome the curse of dimensionality but does not necessarily lead to models that fits well to the data.

Equation (7.2) defines a family of covariance functions from which specific members can be formulated. This induces GP priors and, after data is presented, posterior predictive models. This follows from the main assumption that for the latent function f, responsible for the power load observations, it holds that $f \sim \mathcal{GP}(0, \kappa_{\text{load}})$ with Gaussian likelihood. It is, however, not clear at this point which composite kernels are good choices in regards to optimally forecasting power demands. In addition to choosing the functional form of the kernels, the input over which they are defined must also be specified. To reduce the option space slightly, the covariance functions κ_{obs} and κ_{outlook} are restricted to be of equal type.

With these considerations in mind, we can continue with some implementation details.

7.3 The testing environment

To arrive at test results that can most easily be compared and discussed, a common implementational framework will here be stated which applies to all the results in this thesis.

7.3.1 All tests must comply with the availability of data at prediction time

The difficulty of the prediction problem in section 7.0.1 is slightly aggravated by the fact that for any date d, the most recently available power load data at prediction time is typically from t_{05}^{d}, that is, five hours from prediction time. This gives an effective predictive horizon of 43 hours for the farthest prediction time point, t_{23}^{d+1}. For accurate results, all tests must comply with this condition.

7.3.2 Error metric and benchmark model

The mean absolute percentage error (MAPE) will be used to measure the quality of predictions. If we denote the target values by $\{y_i \colon i = 1, \ldots, n_*\}$ and the predicted time series by $\{\hat{y}_i \colon i = 1, \ldots, n_*\}$, then the corresponding MAPE is given by

$$\text{MAPE} = \frac{1}{n_*} \sum_{i=1}^{n_*} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \cdot 100\%. \tag{7.3}$$

This error measure is scale-independent, hence it is meaningful to compare MAPE-values across different datasets. It does not, however, tackle well data sets that have any values in near proximity to zero, generating indefinite loss values. The electricity consumption time series have values in the range 100-500 MW, but when transformations are considered, for example, if data is shifted to have zero empirical mean, the corresponding inverse transformation must be applied to predictions and targets alike before MAPE is calculated.

As a final benchmark of performance, the GP forecasts will be compared to the Kalman-filter based predictions included in the dataset. This model is at the time of writing used by NTE and will thus serve as an interesting comparison to a real-world, commercially employed method. The predictive errors are registered in table 7.1.

Table 7.1: MAPE values of the Kalman-filter predictions.

2011	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**
5.26	3.45	3.96	4.17	3.17	4.34	3.21

7.3.3 Remarks on implementation

Unless otherwise specified, $m = 450$ inducing inputs will be used, which are in the initialisation phase uniformly sampled from the training set. Initial kernel hyperparameters are drawn from wide distributions[12] and are, together with the inducing variables, allowed 20 optimisation iterations. This procedure will be repeated in five independent *retries*, after which the best model (with respect to marginal likelihood) are given 30 additional iterations before predictions are made. This will be referred to as the *initialisation algorithm.* All optimisation is done with the L-BFGS algorithm [39], which is well-proven and popular in the GP community.

In a pre-processing phase, all weather data, including the values in the test sets, are standardized to zero mean and unit variance with respect to the *training* sets. All power load values are shifted to zero mean, also with respect to the training set.

[12]Gamma distributions with shape 1 and scale 4 are used as hyperpriors.

Finally, every test will be subject to the *chaining* strategy, meaning that the predictions are performed in batches, and the training set for each batch is built from all the data that are historical with respect to the time-labels in the batch. That is, we train on all available data without using any data “of the future”, so to speak, which would add a possibly favourable—but unwanted—bias to the results.

All implementation is done in Python using the open libraries Numpy [40], Pandas [41] and GPFlow [42] for computations and Matplotlib [43] for generating plots.

7.3.4 Covariate identifiers

To declutter the notation, all the input covariates that will be considered are given an identifier; see table 7.2. The time-series might additionally be affected by an hourly backshift operator, the order of which is indicated by a subscript; for example, P_{48} denotes the power load time-series shifted 48 hours back in time.

Table 7.2: List of covariate identifiers. All the time-series may additionally appear subscripted with a value that indicates time-lag in hours.

Identifier	**Covariate**
P	power load
T, $\widetilde{T}$	temperature: measurement, prediction
V, $\widetilde{V}$	wind speed: measurement, prediction
S, $\widetilde{S}$	solar irradiance: measurement, prediction
D, W	daily, weekly periodic components (includes sine and cosine)
A	annual periodic component (includes sine only)
H	holiday* binary

* Norwegian public holidays and Christmas eve

Further, in a slight misuse of notation, covariate indicators appearing as input to kernels denotes ARD over the respective input dimensions. As an example, $\kappa_{\text{SE}}(P, T)$ corresponds to the SE-ARD kernel in equation (4.6) taking as input two-dimensional vectors of augmented power load and temperature values. With this little trick we can write the periodic kernel in equation (7.1) equivalently as $\kappa_{\text{per}} = \kappa(D, W, A)$.

7.4 Which weather data are relevant?

It is evidently the case that temperature largely affects the electricity consumption, at least at the residential level, but it is less obvious how long it takes for fluctuations in temperature to take effect in the demand. Another interesting question is to which degree solar irradiance and wind speeds are useful predictors. To test this further, GP models are trained on various weather covariates and tasked to predict for the whole of 2017. The model kernels will follow

$$\kappa_{(a),(b)}(\boldsymbol{x}, \boldsymbol{x}') = \kappa_{se}(D, W, A) \cdot \begin{cases} \kappa_{se}(\widetilde{T}_k + \widetilde{S}_k + \widetilde{V}_k), & \text{test case (a)} \\ \kappa_{se}(\widetilde{T}_k \cdot \widetilde{S}_k \cdot \widetilde{V}_k), & \text{test case (b)}, \end{cases} \tag{7.4}$$

where k is some arbitrary positive lag number and $\boldsymbol{x}$ takes one covariate along each dimension. The posterior SE length-scales will be used to determine the relative importance of each weather covariate. Note that the above covariance functions follows the form of equation (7.2) with additive ζ and $\kappa_{obs} = 0$.

The results are shown in figure 7.2, where a lag value of $k = 2$ stand out as the better choice in case (a) as well as (b). The temperature dimension obtains shorter length-scales than that of solar irradiance and wind speed; of these two the former is considered somewhat relevant at lower lags k, whereas the latter are deemed generally unimportant. Wind speeds will thus not be utilized in further test cases.

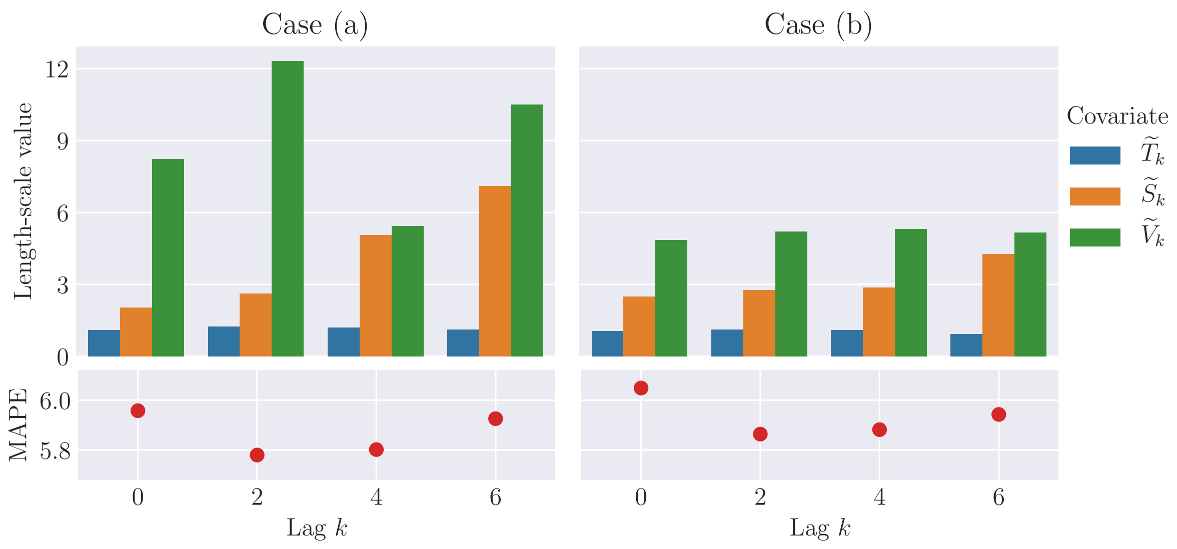


Figure 7.2: Length-scales of lagged temperature, solar irradiance and wind speed covariates, with errors of the corresponding predictions (MAPE, 2017).

7.5 Impact from increasing the number of lagged covariates

To check how an increasing number of lagged covariates affects the predictive performance, κ_{load} will receive an incrementally expanded number of covariates as input—table 7.3 states the covariates in detail. For simplicity, the components kernels of κ_{load} will be squared exponentials. Results are calculated for all test years with at least one year of training data available, so that we may examine the development of the forecast quality as the training set grows bigger.

The plot of the results, figure 7.3, shows that for the cases considered here, adding more covariates are exclusively beneficial. The additive model variants are markedly

Table 7.3: Each table row specifies inputs to equation (7.2), where κ_{per} always recieve as input (D, W, A) unless otherwise mentioned. Wheter the function ζ is additive or multiplicative will be stated. Labels are included for convenient referencing. For the model with label iv., the kernel κ_{obs} comprise two seperate kernels that are in symmetry with ζ either added or multiplied together.

Label	**Input to κ_{obs}**	**Input to κ_{outlook}**
i.	H_{48}, P_{48}	$H, \widetilde{T}_2$
ii.	H_{48}, P_{48}, T_{50}	$H, \widetilde{T}_2, \widetilde{T}_6$
iii.	$H_{48}, P_{48}, T_{50}, T_{60}$	$H, \widetilde{T}_2, \widetilde{T}_6, \widetilde{T}_{12}$
iv.	$(H_{48}, P_{48}, T_{50}, T_{60}),\ (H_{43}, P_{43}, T_{45}, T_{48})$	$H, \widetilde{T}_2, \widetilde{T}_6, \widetilde{T}_{12}, \widetilde{T}_{24}$

better, especially so when data is scarcer. The predictive error shrinks considerably from 2011 to 2014 as more data becomes available; from there the trend is less marked but still present.

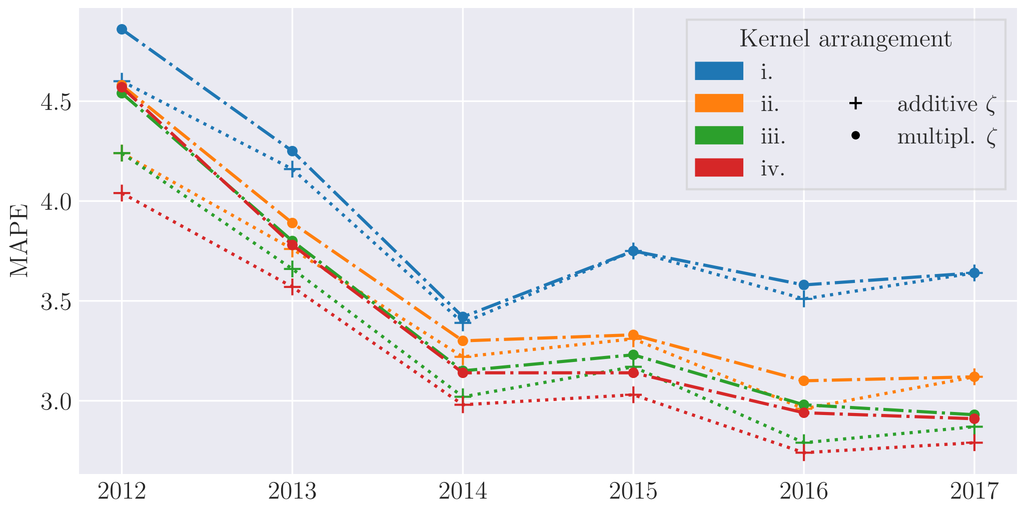


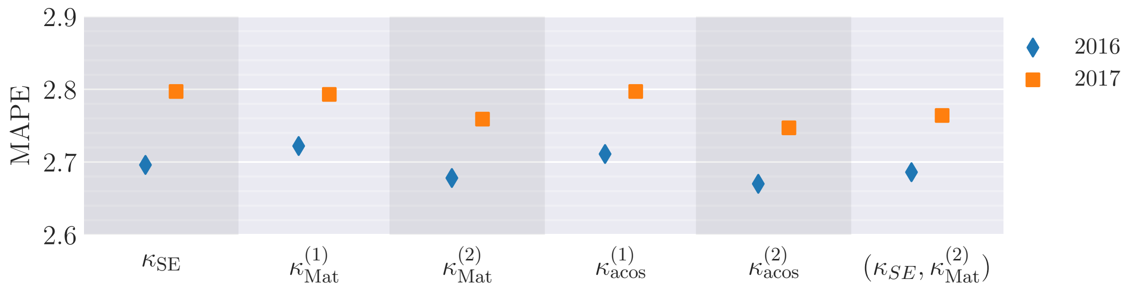
Figure 7.3: Error of predictions with respect to input size where κ_{load} are comprised of only SE kernels. The inputs corresponding to each label are detailed in the accompanying table 7.3

7.6 Finding optimal kernel compositions

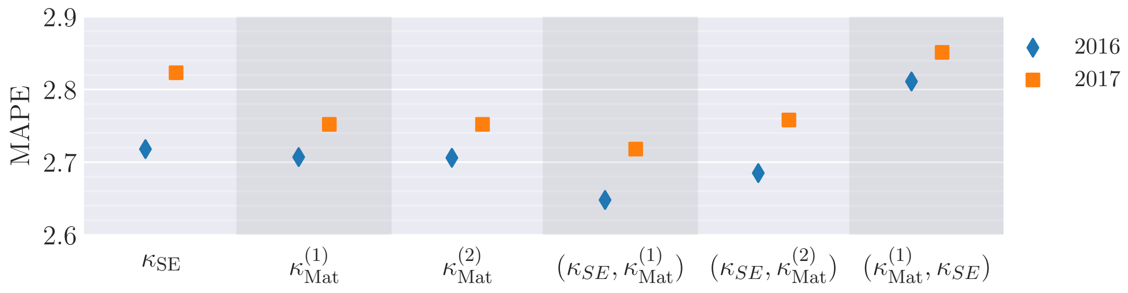
Defining κ_{load} with combinations of the SE covariance function, as have been done in the past few sections, results in distributions of very smooth functions. To test if other kernels are better suited, the input arrangement from table 7.3 which yielded the best predictive results, label iv., are here applied to different compositions of kernels.

The results are shown in figure 7.4. Here, a tuple of kernels, e.g. (κ_1, κ_2), denotes that κ_{per} are of type κ_1 while both κ_{obs} and κ_{outlook} are of type κ_2. A single kernel type is specified in the case where all three covariance functions are of equal type.

By selecting the two best-predicting models from this test, two reasonably good members of the family defined by equation (7.2) have been found. They utilize the kernel compositions $\kappa^{(2)}_{\text{acos}}$ with multiplicative ζ, and $(\kappa_{\text{SE}}, \kappa^{(1)}_{\text{Mat}})$ with additive ζ, where the latter yields slightly better results. The estimated optimal input covariates are that of label iv. in table 7.3. For future referencing, we will denote the models by $\omega_\times$ and ω_+, respectively.



(a) Multiplicative ζ.



(b) Additive ζ.

Figure 7.4: Predictive results of various combinations of κ_{per}, κ_{obs} and κ_{outlook}, where the horizontal axis labels indicate the respective types. When all three kernels are of the same type, the respective type is indicated by a single label. Tuples express by its first entry which type κ_{per} is defined as, while the second entry state the type of both κ_{obs} and κ_{outlook}.

8 Model analysis

In this section we analyse how a direct strategy affects the predictive performance of the models. Model convergence with respec to hyperparameters and the variational lower bound is covered in sections 8.2 and 8.3, respectively. The main forecasting results is presented in section 8.4.

8.1 Direct modelling routines

In section 6.1.4 we observed that the variance of the data points was distributed unevenly with respect to the hour of the day, as well as part of the year. Hence, by partitioning the data set by hour or by month, allowing for different model hyperparameters for each separate partition, this aspect can be dealt with more directly. The procedure is often referred to as a *direct* modelling strategy.

Figure 8.1 shows losses of predictions performed with the two best-performing GP models that was found in section 7: ω_+ and $\omega_\times$. Partitions are performed on the basis of hours, months and—for completion—days. The input to κ_{per} are adjusted slightly to fit each case, because there is no need to explicitly include D, W or A as input to the periodic kernel when considering direct procedures over days, weeks or months, respectively.

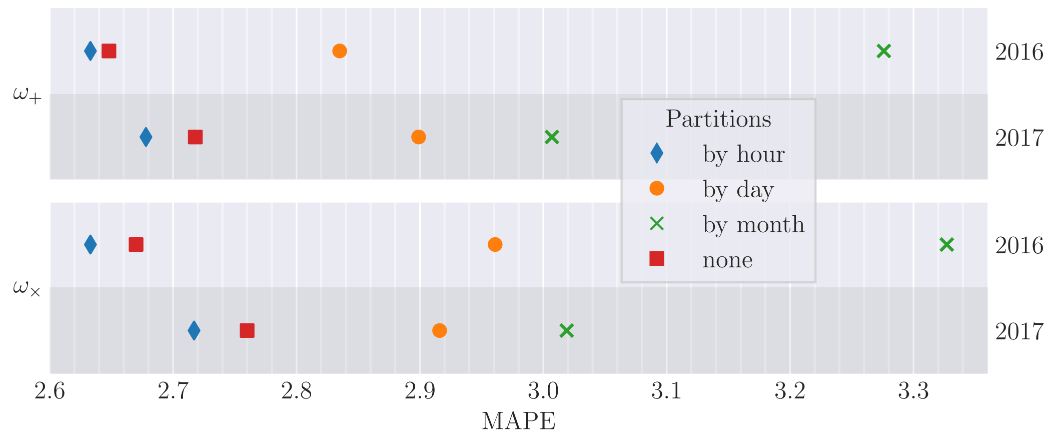


Figure 8.1: Different partition schemes are compared for ω_+ and $\omega_\times$ with respect to 2017 MAPE. The direct routine with respect to hour shows a gain in predictive results.

Forecast performance improves slightly when a direct strategy over the hourly partitions are used, but suffers considerably when the same strategy is applied with respect to discrete days or months. Note that no distinction is made between the models in terms of inducing variable number m, even as the training sets are much smaller in the direct approaches. The effect of different m-values will be analysed further in section 8.3.

8.2 Optimality of hyperparameters

It is of primary interest to investigate whether the initialization routine in combination with marginal likelihood maximisation is an adequate enough set of tools for the task of locating optimal hyperparameters. Unfortunately, this analysis is complicated by the fact that the sparse model merely upper bounds the marginal likelihood. Thus, if we were somehow able to locate the global maximum of the marginal likelihood for the approximated posterior we could still be very far away from the true marginal likelihood. Moreover, even if it was computationally tractable to evaluate the exact GP posterior, the space over which to optimize is almost guaranteed to be highly multimodal. Consequently, even knowing if optima have been reached is a difficult and resource demanding numerical optimization problem.

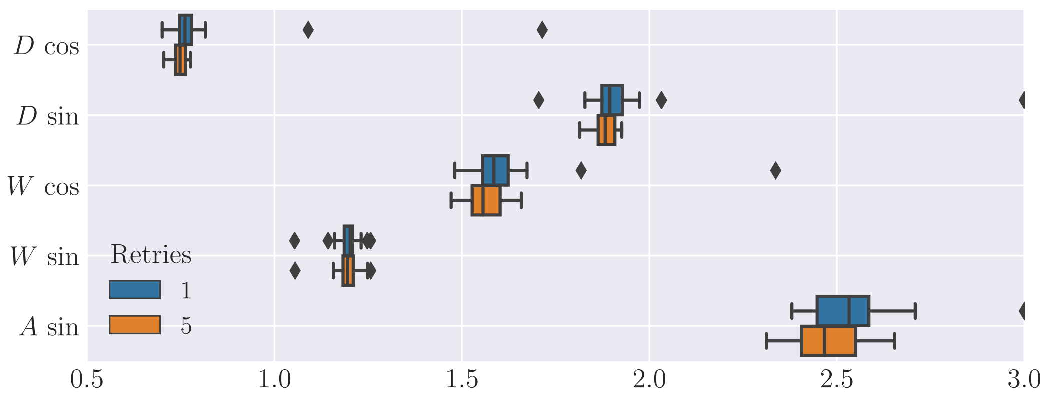
One approach to the inspection of hyperparameter optimality is based on the following statement: if the hyperparameters of a specific model, given that they are initialised from *very wide* hyperprior distributions, are *consistently* optimized to the same values in a large number of trials, then there is a high likelihood that this value is optimal for the model.

Naturally, we can only assess optimality within the limitations induced by the sparse formulation, since the space over which we optimize is merely an approximation to the true marginal likelihood space. Nevertheless, by checking if the above statement holds true for a model, i.e., if the optimisation routine provides consistent results, we may assume that it to some extent holds true also for similar models induced from the κ_{load} family. Whether the optimised values are not immensely far away from whichever values we would obtain by optimising instead the exact marginal log likelihood is hard to tell, but a convergence analysis of the marginal likelihood lower bound will be performed in section 8.3.

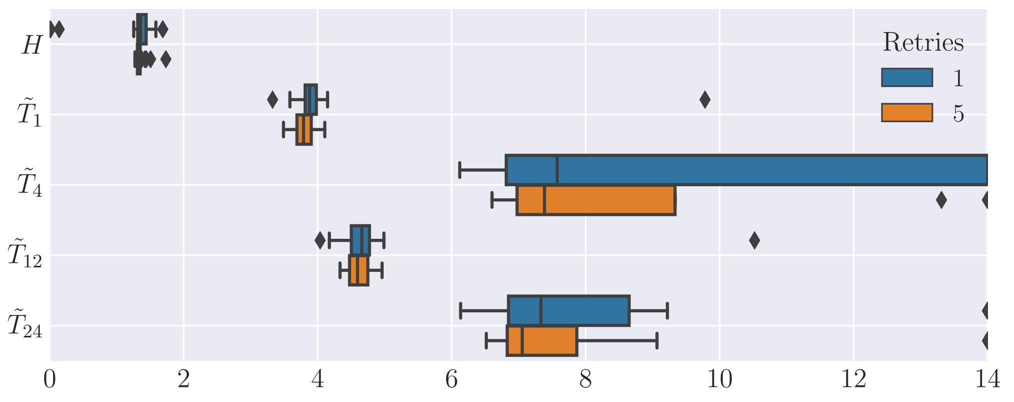
The length-scale values of the inputs to the component kernels of κ_{load}, each using the SE as parametric form with multiplicative ζ, are portrayed in figure 8.2. The hyperparameters are along with $m = 350$ inducing variables optimised with 20 independent runs of the initialisation routine. The testing was performed with one, five and ten retries in the algorithm, however, a visual comparison between the results from five and ten retries showed no substantial difference so the latter case has been left out to prevent clutter.

Figure 8.2a shows the length-scales for all the periodic components of the model. The two length-scale pairs of D and W determine the interplay between the sine and cosine components. As we observe the variance in the length-scale values are generally quite low, which presumably indicates that five retries of the initialisation routine are adequate to reach consistent values for the periodic length-scale components.

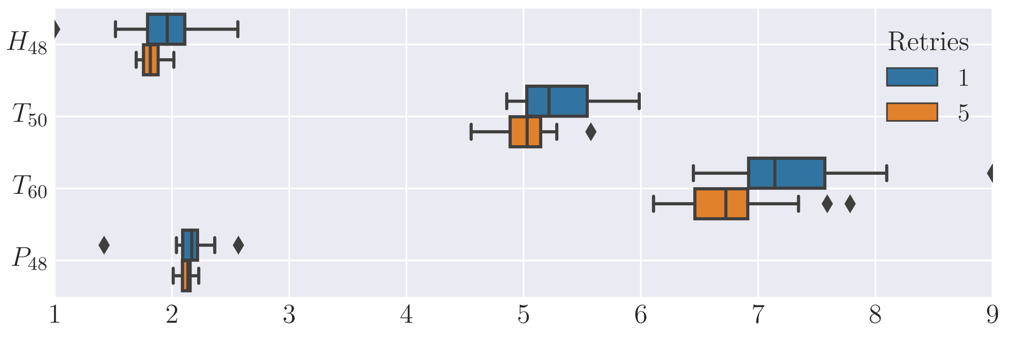
All severe outliers in the estimated hyperparameter distributions belong to the one-retry case, which we can observe from figures 8.2b and 8.2c to hold true for most of the other length-scales as well. The exceptions are $\widetilde{T}_4$, $\widetilde{T}_{24}$ and T_{60}, whose distributions are very obscure—spectacularily so for $\widetilde{T}_4$, for which the interquantile range reach far out of



(a) Length-scales of of κ_{per}.



(b) Length-scales of κ_{outlook}.



(c) Length-scales of κ_{obs}.

Figure 8.2: Boxplot of optimised length-scales (shown on the horizontal axis) of a pure SE-ARD variant of κ_{load} from 20 trials, each with the indicated number of retries in the initialisation routine. Diamond markers indicate outliers, where severe outliers are clipped and represented on the panel edges.

the visible window limits. One reason may be that $\tilde{T}_4$ strongly correlates with $\tilde{T}_1$, which are likely to be the better predictor of the two, and having short length-scales on both of them is superfluous.

8.3 Convergence assessment of the variational lower bound

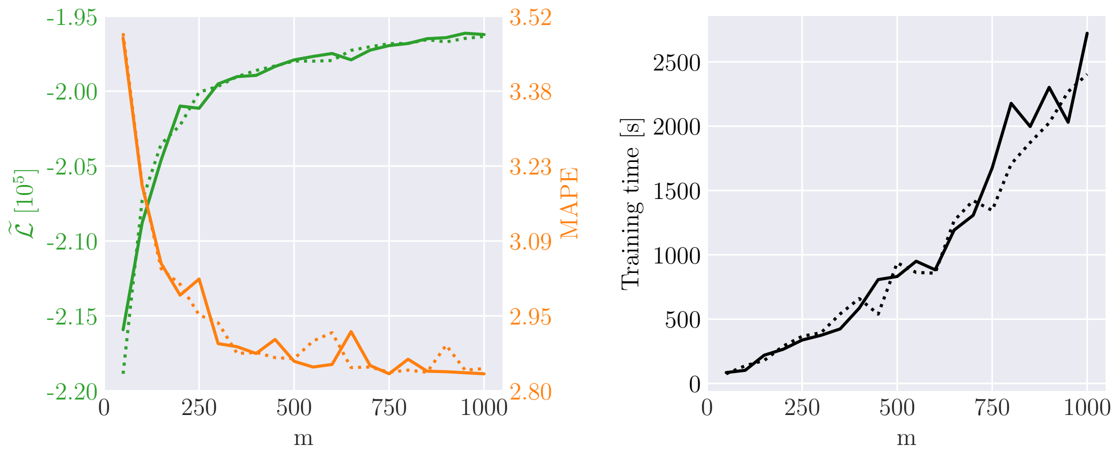
The variational marginal likelihood $\widetilde{\mathcal{L}}$ is a lower bound to the true marginal likelihood for every $m \geq 1$. However, for a given data set, GP prior and m we have no assurance that the geometry of the space described by $\widetilde{\mathcal{L}}$ even remotely resembles the geometry of the corresponing $\mathcal{L}$-space. That is, unless the inducing variables coincide exacly with the observed data, but letting $m = n$ and comparing $\widetilde{\mathcal{L}}$ with $\mathcal{L}$ is computationally intractable for any data set sizable enough to give meaningful results.

If the evaluations of $\widetilde{\mathcal{L}}$ are indeed very distant from the true marginal likelihood values, the resulting models will fit poorly to the data and consequently yield substandard predictions. However, the *primary* interest is not really to assess the convergence of the variational lower bound directly, but to find out when the approximations are *good enough.* That is, find out how large m must be in order for the load forecasting models to fit adequately to the data and yield good predictions. To analyse this we look at how the *predictive performance* varies with respect to changing the number of inducing variables. If the error in the predictions converges, we can choose an appropriate value of m accordingly.

Figure 8.3a displays MAPE values of 2017 as a function of m for predictions by the model induced by κ_{load} when component kernels are of type $\kappa_{\text{Mat}}^{(1)}$ and ζ is additive. As input we use label iv. from table 7.3. The log marginal likelihood estimates $\tilde{\mathcal{L}}$ of the chained training data are plotted in parallel. Two independent trials are performed, each with the settings described in section 7.3.3.

The error in the predictions shrinks as m increases, and the opposite is true for the marginal likelihood bound. None of the quantities has seemingly yet fully converged, which holds true in particular for the bound $\tilde{\mathcal{L}}$. Figure 8.3b displays the corresponding training times,[13] in which the five retries of the initialisation routine are included. With $m \approx 500$ the training times are below fifteen minutes while still giving decent predictive results.

[13]The running times are clocked on a single Intel® Core$^{\text{TM}}$ i7-4770 CPU at 3.40GHz $\times$ 4 processor.



(a) **(b)**

Figure 8.3: Varying m, MAPE for 2017 is calculated along with the log marginal likelihood estimate of the optimised model, and visualised in tile (a). In (b), corresponding training times are shown. The numbers include five retries of the initialisation routine. Increasing the number of inducing variables m leads to higher marginal likelihood and lower predictive error at the cost of rapidly increasing training time. The solid and stippled lines represent two independent trials.

8.4 Main forecasting results

In the experiments, ω_+ were found to be the most well-functioning model. To summarize, it is induced by the following specific member of κ_{load}:

$$\kappa_{\text{load}}^{\omega+} := \overbrace{\kappa_{\text{SE}}(D, W, A)}^{\kappa_{\text{per}}} \Big[\overbrace{\kappa_{\text{Mat}}^{(1)}(H_{48}, P_{48}, T_{50}, T_{60}) + \kappa_{\text{Mat}}^{(1)}(H_{43}, P_{43}, T_{45}, T_{48})}^{\kappa_{\text{obs}}} \\ + \underbrace{\kappa_{\text{Mat}}^{(1)}(H, \widetilde{T}_2, \widetilde{T}_6, \widetilde{T}_{12}, \widetilde{T}_{24})}_{\kappa_{\text{outlook}}} \Big] \tag{8.1}$$

through the assumption $f \sim \mathcal{GP}(0, \kappa_{\text{load}}^{\omega+})$. The covariate indicators are explained in table 7.2. As shown in section 8.1, a direct strategy over hourly partitions had a beneficial effect on the forecasts. In this case, the kernel use the same formulation as equation (8.1), but the the covariate which capture the daily periodic structure, D, is omitted from κ_{per}.

The forecasting results with a comparison to the benchmark Kalman-filter method are given in table 8.1. Five initialisation retries and $m = 500$ inducing inputs are used, found to be sufficient from the analysis in sections 8.2 and 8.3.

Table 8.1: MAPE of power load predictions using three different models. The models operate with the time horizon $\{t_0^{d+1}, \ldots, t_{23}^{d+1}\}$, where the errors shown are averaged by year for each day d in the data set.

	2012	**2013**	**2014**	**2015**	**2016**	**2017**
ω_+	3.93	3.61	**2.94**	2.91	2.64	2.77
ω_+, direct w.r.t. hour	4.19*	**3.50**	3.21	**2.87**	**2.63**	**2.68**
Kalman-filter	**3.45**	3.96	4.17	3.17	4.34	3.21

* Uses 250 inducing variables, otherwise $m > n$ for the partitioned sets.

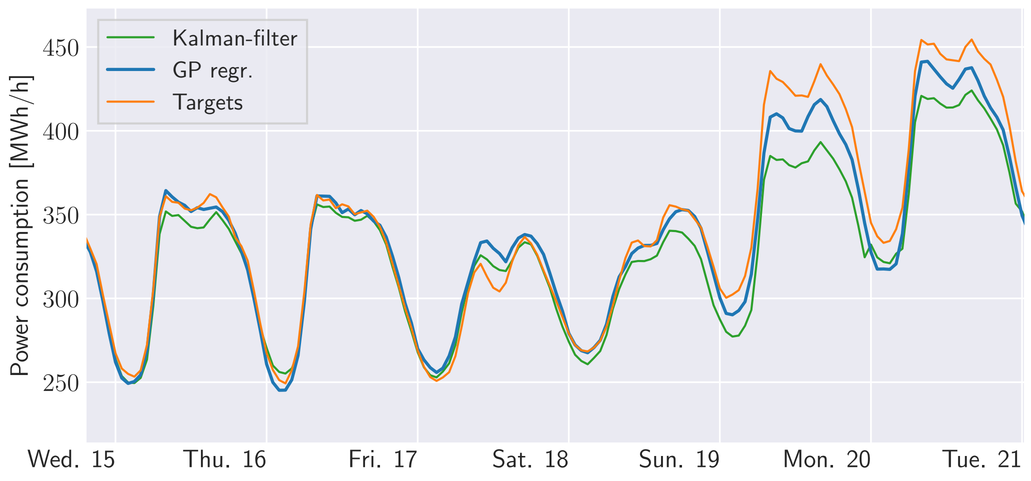


Figure 8.4: Compared to the actual demand, forecasts made with ω+ often outperforms the Kalman-filter-based predictions. The plot shows a slice of November 2017. Shaded areas illustrate the 95% confidence interval of the GP model.

9 Discussion

In this section we will draw important conclusions from the experimental results in sections 7 and 8. Afterwards, the discussion will concern properties that hold for the models induced from κ_{load}, including how they can be extended for future changes in the electricity demand landscape. Lastly, we compare the approach considered in this thesis with some closely related work.

9.1 Main conclusions from the experimental results

9.1.1 Lagged temperature values correlate well with power demands

Figure 7.2 suggested that the temperature time-series are by far the most advantageous linear predictor to include in the GP models for power demand; moreover, a time-lag of around two hours seems ideal. This indicates that it takes a few hours before temperature fluctuations start taking effect in the power demands, which is intuitively reasonable because buildings in Norway generally have high thermal inertia due to being well isolated. The differences between the tested lag values are small, but this is to be expected from a high degree of correlation between the lagged series. All in all the results of the test are on par with what we could expect from figure 6.5.

9.1.2 The GP models tackle well an increased amount of input covariates

Figure 7.3 implied that adding more covariates were exclusively beneficial to the predictive performance. Comparing with figure 8.2, we see that “irrelevant” inputs will have their optimised length-scales distributed according to vague distributions with high means and not affect the resulting model much one way or another.

However, for each new input dimension, the space in which the optimisation algorithm must search for the optimal solution grows in dimensionality and, most likely, complexity. A different aspect is the curse of dimensionality, which is known from section 4.3 to affect pure multiplicative models primarily. Thus, we ought to expect that at some point the model will get saturated with input, loosing predictive power, but the test is inconclusive in that regard and further testing must be performed to reveal more about the general case.

9.1.3 SE kernels capture the periodic structure well, but models poorly the influence of weather

The overall best-predicting model, ω_{+}, uses a SE kernel to learn the periodicities in the electricity consumption data. It is not very surprising that a relatively severe smoothness assumption works well in this case: The time-series is an aggregated demand from a whole county and are thus smooth by nature, at least in the time-scales that we are interested in.

In situations where SE kernels are tasked to model the influence of weather conditions, like for (κ_{SE}), $\kappa_{\mathrm{Mat}}^{(2)}$ in figure 7.4, the opposite situation arise: the results becomes relatively poor. Recall that the weather time-series are weighted averages of only three locations and must be expected to be much more volatile, so less severe smoothness assumptions implied by, say, Matérn kernels, stand out as a more capable choice.

9.1.4 A direct modelling strategy with respect to hours yields good results

Figure 8.1 show that training separate models for each hour of the day are indeed—by a slight margin—superior to modelling the hours jointly with a daily-periodic sine-cosine combination. This result may be due to heteroscedasticity in the daily profile, as observed in figure 6.4. Another explanation, which may be closely tied to the former, is that the degree of correlations between weather and power demands can in general not be expected to be constant throughout the day. As an example, temperature drops likely affects the power demand more if they happen *during daytime*, because buildings are in general warmer at daytime, resulting in more heat loss to the surroundings and higher electricity demands to keep up temperature levels.

With direct modelling, the discrete data sets are smaller and can doubtless be more easily approximated, when compared to the full dataset, by m inducing variables. But the effect is likely small, as monthly and daily direct approaches have the same advantage but still performs poorly.

9.1.5 Hyperparameters generally reach consistent values, at least those who matter

One central observation from figure 8.2 is that more retries in the initialisation algorithm lead to hyperparameters having fewer outliers and less diffuse distributions, despite the fact that they are initially drawn from very wide hyperprior distributions. There are at the same time input dimensions, most notably $\widetilde{T}_4$, which are attended with length-scales whose distributions are very vague and wide. Crucially, however, these all have relatively *high mean values*, meaning that the resulting GP models are not much affected by their inability to reach consistent values. Variability in these inputs is simply ignored from the ARD property.

Figure 8.2a shows the length-scales for all the periodic components of the model. As mentioned, the two length-scale pairs attended with D and W determine the relative interplay between the sine and cosine components of the transformed input. The values are themselves somewhat hard to interpret, contrary to all the other length-scales, for which the values determine relative input importance. It is, on the other hand, essential that the values for D and W are *consistent* throughout the trials, since they represent the only cardinal model components that are not correlated with any other input series. In other words, two very distinct sets of sine-cosine pairs constitute two radically different solutions. Variations in other length-scales are more forgiving because information is to a larger degree shared with other components: the power load time-series is, for example, strongly correlated with temperature—as is the annual sine component.

In conclusion, the analysis of section 8.2 provides empirical evidence that a small number of retries in the initialisation algorithm in combination with widely defined hyperpriors is, albeit very simple, a powerful combination to overcome multimodality in the log likelihood estimate so that hyperparameters of *relevant* inputs can be consistently optimised. This holds true at least for the specific model that was examined, and within the limitations brought forward by the sparse formulation.

9.1.6 A failed attempt to explore the hyperparameter posterior

To investigate the hyperparameter space and directly assess whether hyperparameters reach optimality, a Hamiltonian Monte Carlo (HMC) method was applied to different models trained to maturity[14] in an attempt to generate samples from the hyperparameter posterior $p(\boldsymbol{\theta} \mid \boldsymbol{y})$. HMC is, through the use of first-order gradient information, in theory able to ensure quicker convergence rate than the more standard Metropolis-Hastings or Gibbs sampling approaches, especially in spaces of higher dimensions. The specific algorithm can be found in a review by Neil [44, p. 14].

After considerable trial and error to tune the parameters of the algorithm (the step size and number of steps to take each iteration) the results were unfortunately rather inconclusive: either all the steps were rejected or the step size was so small that accepted steps ended up in a very close proximity to each other. This would indicate that the local maxima into which the L-BFGS routine optimises the hyperparameters are quite narrow, or that there are large areas without much support in the posterior probability space. The method did not, however, provide any satisfactory results in regards to exploring the hyperparameter posterior, so the theory and details of this method have been skipped.

9.2 On the sparse GP model approximation

9.2.1 The predictive performance is closely tied to the estimated marginal likelihood

In figure 8.3 the predictive error shrinks as m increases, and vice versa for the marginal log likelihood estimate, which does not come as a surprise. Interestingly, though, the form of the two graphs closely resembles reflections of each other, indicating that the predictive performance of the model is very closely tied to $\tilde{\mathcal{L}}$. Indeed, even small indentations in the marginal likelihood estimate seem to bring about (somewhat more pronounced) humps in the corresponding MAPE values. The reason why these bumps in $\tilde{\mathcal{L}}$ occur at all—and likewise why there is a slight discrepancy between the two trials—are not known, but the asymmetries could presumably be reduced by using more than five retries in the initialisation routine.

[14]Meaning that a strict stopping criteria in the optimisation algorithm (L-BFGS) was reached.

9.2.2 *Using 5-600 inducing variables is a well-balanced choice*

To find a balanced choice of m in regards to both complexity and accuracy, we turn again to figure 8.3b. In theory, the training times scales quadratically with m when n is fixed, and the empirical evidence in the figure agrees. The rugged form of the graph can be attributed to a varying number of extraneous processing task running in the background. Ignoring useless choices of m (with respect to MAPE), a complete training procedure takes from about six minutes up to an hour for $m = 1000$.

Updating the model hyperparameters as infrequently as once every year (as is effectively done when using chaining) is quite reasonable, and in this case, training times of one hour are insignificant. However in practice, there are few reasons not to train the model more often, perhaps also train a multitude of models and average their collective predictions to avoid overfitting. In this scenario, the training times will become more significant, and we can pick e.g. $m = 600$ with a barely noticeable loss in predictive performance as compared to higher choices of m. Indeed, the method learns very complex patterns with few inducing variables, as illustrated in figure 9.1 with $m = 450$.

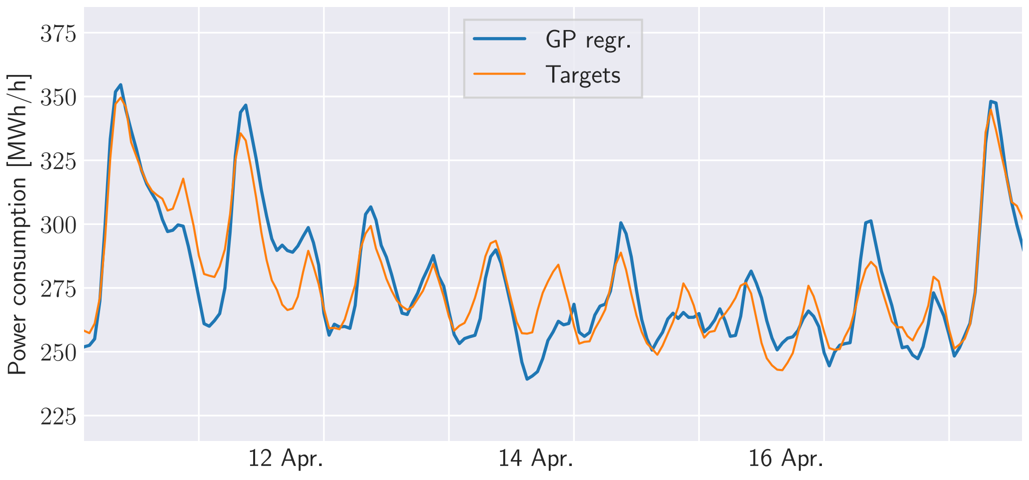


Figure 9.1: An interval of 2017, with a particularily non-regular power consumption pattern, are with $m = 450$ inducing variables predicted for. Shaded areas show two standard deviations of the GP distribution.

Note that the prediction times follow the exact same form as the optimisation times in figure 8.3b, but since predicting for a full yearly batch, after training is completed, requires only a few seconds even for $m = 1000$, the differences are insignificant.

Remark. Naturally, m should be chosen relative to the number of points n in the training data, so using the models for their designated purpose—predicting future power loads—the number of inducing variables must change accordingly. However, there is likely no simple relation that maps an optimal m to any given n. A fair assumption is that a larger m-values are required for the sparse method to handle the increasing complexity inherent in larger data sets, but a more exact relation ought to be examined at a later time when substantially more data is available.

9.2.3 Using an abundance of inducing variables will likely not affect the results considerably

The fact that $\widetilde{\mathcal{L}}$ in figure 8.3 have seemingly not yet converged are of little concern. As mentioned, convergence of $\tilde{\mathcal{L}}$ does not indicate that the lower bound is tight, and we do not possess the computational power to find out whether it is. Looking instead at the MAPE values, which we included in the analysis for this specific reason, any further increments of m seemingly result in only marginal improvements—and, if we assume that it is meaningful to extrapolate the curve, then convergence is imminent and using $m > 1000$ is not worthwhile.

Another key observation is that increasing m does *not encourage overfitting*, and so there is seemingly little hazard involved in choosing large values of m other than long-winded training times.

9.3 Useful properties of the model family

There are several useful properties which the model family induced by κ_{load} benefits from. Some are inherent to GPs, while others follow from the concrete model formulation.

9.3.1 It is intuitive and easy to modify

The covariance function κ_{load} is divided into seperate components that models temporal structures and the influence of observations and weather forecasts, respectively. This construction is intuitively simple. Depending on future needs, e.g if we gain access to additional weather forecast time-series, the relevant kernel component can easily be extended with additional input features.

As another example, we could consider as input feature the Norwegian general staff holiday (GSH) period, which typically takes place the last three weeks of July each year, during which many workplaces run on lower capacity. Ignoring this altogether may lead to slight systematic overpredictions at weekdays with GSH, as an indirect effect, underpredictions on other dates of july or August where values may be inferred from days with GSH.

9.3.2 The additive variant is robust to outliers

In the precense of outliers, the additive branch of the model family will still function: The kernel component recieving a faulty input, either κ_{outlook} or κ_{obs}, will not find any corresponding input points in the training data, effectively "turning off" the kernel, while the other kernel components will function as normal. Note that outliers can be detrimental if ζ is multiplicative, because κ_{load} as a whole will be heavily affected by a single outlier. This makes a strong argument for using models with additive ζ.

9.3.3 There are no user-defined parameters

Being able to compute marginal likelihood (estimate) of training data provides a principled way of comparing different models. Thus, there is no need for slow procedures such as exhaustive grid searches or cross-validation to optimise hyperparameters. Compared to other regression methods such as neural networks, there are very few hyperparameters, all of which are interpretable through ARD.

9.3.4 It is adaptable to considerable changes in the energy demand landscape

A model from the family will adapt to the energy consumption patterns present in a training set. In its current form it makes no distinction to the *freshness* of the data, as the provided training set is simply not large enough for such a procedure to be beneficial. This will get much more important in the future, as the introduction of *smart-grids* will, in all probability, change the current consumption patterns considerably. Also, as electrical vehicles become more common, the overall power demands are expected to rise markedly. Fortunately, it is trivial to extend the model such that older information is penalized: simply by multiplying κ_{load} with any stationary kernel taking any monotonously growing time-feature as input, such as *year*, we ensure that the resulting models correlate higher with data-points of newer date. The rate of which the model de-weights old information is automatically optimised along the other hyperparameters.

Another approach is to explicitly capture the linear of the data by adding a linear kernel, taking time-labels as input, to κ_{load}. The result will, however, not adapts as flexible to severe changes in consumption patterns as the formerly mentioned approach.

9.4 The spectral mixture model for extrapolations

Unfortunately, the spectral mixture kernel performed insufficiently with respect to forecasting electricity demands. Predictions were excessively focused on extrapolating patterns from two days before, which makes sense considering figure 6.6 and the distance to the nearest training examples in the prediction setting. One possible explanation is the following: the kernel have, unlike κ_{load}, the ability to use neighbouring hours to fit the training data, and this likely results in much higher marginal likelihood than learning to extrapolate on the basis of more distant points. In the test senario there are no immediate neighbors are present, so extrapolations to distant points are therefore poor.

A considerable effort was needed to have the kernel learn even the most obvious weekly and daily patterns. The final initialisation algorithm consisted of sampling the means μ_q uniformly from values between zero and the Nyquist frequency of the samples, i.e. once per halv hour, while initializing the weights w_q to the empirical standard deviation of the data. Lastly, the length-scales $1/\sqrt{\sigma}$ were sampled from a truncated Gaussian distribution with mean proportional to the maximum value range of the data. The effort it took to achieve the learning of any non-trivial patterns were not ideal when considering the prime motivations for introducing the kernel in the first place: using GPs as expressive statistical tools that can automatically discover hidden patterns while intelligently extrapolate away from training data.

9.5 Related work

Blum and Riedmiller [11] proposed a similar approach to the solve the power demand forecasting problem. They construct a covariance function on the form

$$\kappa_{\text{power}}(\boldsymbol{x}, \boldsymbol{x}') := (\kappa_{\text{daily}} + \kappa_{\text{weekly}} + \kappa_{\text{weather}})(\boldsymbol{x}, \boldsymbol{x}'), \tag{9.1}$$

where κ_{weekly} and κ_{daily} are periodic SE kernels with periods $p_{\text{day}} = 24$ and $p_{\text{week}} = 7 \cdot 24 = 172$, respectively. The kernel κ_{power} allows modelling of correlations between daily and weekly power usages, and while serving as a point of inspiration, the pure additive nature of κ_{power} offers many important disadvantages and would not fit our problem well.

Firstly, the daily periodic component κ_{daily} does not directly depend on which day of the week it is. As a result, when κ_{daily} is tasked to learn the daily patterns in the data, it will ignore which day of the week information is inferred from. To exemplify, the kernel will learn the patterns of, say, a Saturday by taking a smoothed average over all the days in the data set, and not, as would be more ideal, a smoothed average over all the *saturdays* in the data set. The weekly component has, by construction, a relatively long length-scale and will not be able to learn the more subtle differences present in the daily profiles that exist for a specific day *given that the day is a specific day of the week*, as was illustrated in in figure 6.3.

In equation (9.1) the weather was introduced additively through κ_{weather}, a sum of one-dimensional SE kernels, each receiving as input a time-series of relevant weather data. Yet again the additivity is troublesome: κ_{weater} will ensure correlation between function values that have similar weather conditions, without any concern about *when* the conditions occur. Assume for example that the temperature one specific night is forecast to be relatively high, just as high as it was *at daytime* for any other data points in the dataset. These values are, although having similar temperature conditions, likely to be very different as the temporal influence on the electricity consumption is, for the most part, much greater than the influence due to weather conditions. Hence, the temperature component of κ_{weather} will not learn properly. Similar cases can be constructed for other weather features as well, and in conclusion, κ_{weather} is not very useful, especially when big data sets are concerned for which a lot of these "rare cases" are likely to be present.

Note that Blum and Riedmiller were concerned with a *curve fitting* approach, where only the last few weeks of data points were used in predictions, as opposed to a *function mapping* approach we consider in this thesis, where all the avaliabl edata is used simultaneously. Constructing κ_{weather} as a pure additive kernel makes more sense it the curve fitting scenario.

10 Closing remarks

10.1 Further work

As discussed in section 9.3.4, the kernel family κ_{load} can be trivially extended to be more flexible to substantial changes in the energy demand landscape. Another extension that is doable without any significant efforts, potentially, is to consider a generalisation of the SE-ARD kernel,

$$\kappa_{\text{M}}(\boldsymbol{x}, \boldsymbol{x}') := \sigma_\kappa^2 \exp\left(-\frac{1}{2l^2}(\boldsymbol{x} - \boldsymbol{x}')^\top \boldsymbol{Q}(\boldsymbol{x} - \boldsymbol{x}')\right), \tag{10.1}$$

which is often refered to as a *Mahalanobis* covariance function. If $\boldsymbol{Q}$ is the identity matrix, the kernel collapses to $\kappa_{\text{SE-ARD}}$ in equation (4.6), but $\boldsymbol{Q}$ can in general be any positive-definite matrix. It models the covariance between input features directly by the usual marginal likelihood maximisation. As we use multiple lagged time-series as input to our models, correlations are high and κ_{M} are potentially a very useful extension. However, it introduces in general D^2 additional hyperparameters, where D is the dimension of the input vectors $\boldsymbol{x}$. Consequently, the resulting models can be much harder to train, especially when D is large, so some effort might be required to ensure consistent convergence of the hyperparameters.

One principle downside of the direct strategy we consider is that no information is shared between the input partitions or between model predictions, potentially restraining the inference quality. Potential future work could be involved in modelling these relationships in a more sophisticated way.

The spectral mixture kernel require some work before it will be a possible contender to κ_{load} as a tool in forecasting electricity demands. As the kernel was prone to using information from neighbouring hours to fit the training data, and this information is not avaliable in the test scenario, extrapolations to far-away points suffered. A possible solution to this problem is to consider *dropout*, a method for regularizing neural networks, which has recently been discussed in the context of GPs [45]. The method essensially drops random elements from the training set in an attempt to learn more general structures. An interesting future work could consider how the spectral mixture is affected by dropout. To arrive at a useful forecasting tool, however, it should also be attempted to include temperature forecasts as a second input feature in addition to time labels. Other interesting works included combining the SM kernel with κ_{load}.

10.2 Conclusion

With the motivation of constructing reliable forecasting models for electricity demand, we have established a thorough theoretical background for the use of Gaussian processes in long-term time-series forecasting with multidimensional inputs. Various properties of covariance functions have been discussed, with a focus on finding combinations that capture the seasonal structure present in the data while adapting to local conditions. We arrive at a family of Gaussian process regression models that utilise temporal information, historical data and weather forecasts to make capable predictions with quantified uncertainty. To utilise the richness of the historical data while still allowing for low time complexity, a sparse approximation was successfully applied. The predictive performance of the methods was assessed on an electricity demand data set provided by Nord-Trøndelag Elektrisitetsverk (NTE), while complying with the time-frames and limitations that apply to participants in the Norwegian energy market. Compared to a model under current commercial use, the Gaussian process models yields superior results with a 26.5 % reduction in MAPE when averaged over the last two years.

In addition to good predictive performance, many beneficial properties hold for the models. They are intuitively simple to modify with additional input features, and contain a small number of hyperparameters which are all interpretable and optimised within minutes. The models can, with a trivial extension, flexibly capture changes in the energy demand landscape.

References

[1] R. Weron, “Modeling and Forecasting Electricity Loads,” in *Modeling and Forecasting Electricity Loads and Prices*, pp. 67–100, Wiley-Blackwell, 2013.

[2] J. W. Taylor, L. M. de Menezes, and P. E. McSharry, “A comparison of univariate methods for forecasting electricity demand up to a day ahead,” *International Journal of Forecasting*, vol. 22, pp. 1–16, Jan. 2006.

[3] M. E. El-Hawary, *Advances in Electric Power and Energy Systems: Load and Price Forecasting*. John Wiley & Sons, July 2017.

[4] J. W. Taylor and R. Buizza, “Using weather ensemble predictions in electricity demand forecasting,” *International Journal of Forecasting*, vol. 19, pp. 57–70, Jan. 2003.

[5] F. Kaytez, M. C. Taplamacioglu, E. Cam, and F. Hardalac, “Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares support vector machines,” *International Journal of Electrical Power & Energy Systems*, vol. 67, pp. 431–438, May 2015.

[6] H. Takeda, Y. Tamura, and S. Sato, “Using the ensemble Kalman filter for electricity load forecasting and analysis,” *Energy*, vol. 104, pp. 184–198, June 2016.

[7] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, *Recurrent Neural Networks for Short-Term Load Forecasting: An Overview and Comparative Analysis*. Springer, Nov. 2017.

[8] A. K. Srivastava, A. S. Pandey, and D. Singh, “Short-term load forecasting methods: A review,” in *2016 International Conference on Emerging Trends in Electrical Electronics Sustainable Energy Systems (ICETEESES)*, pp. 130–138, Mar. 2016.

[9] R. M. Neal, *Bayesian Learning for Neural Networks*. Berlin, Heidelberg: Springer-Verlag, 1996.

[10] D. Leith, M. Heidl, and J. Ringwood, “Gaussian process prior models for electrical load forecasting,” in *2004 International Conference on Probabilistic Methods Applied to Power Systems*, pp. 112–117, Oct. 2004.

[11] M. Blum and M. Riedmiller, “Electricity Demand Forecasting using Gaussian Processes,” in *Trading Agent Design and Analysis: Papers from the AAAI 2013 Workshop*, University of Freiburg, Department of Computer Science, 2013.

[12] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, Aug. 2012.

[13] A. Gut, *An Intermediate Course in Probability.* Springer Texts in Statistics, New York: Springer-Verlag, 2 ed., 2009.

[14] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning.* The MIT press, 2006.

[15] C. E. Rasmussen and Z. Ghahramani, "Occam's Razor," in *In Advances in Neural Information Processing Systems 13*, pp. 294–300, MIT Press, 2001.

[16] D. J. C. Mackay, "Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks," *Network: Computation in Neural Systems*, vol. 6, pp. 469–505, Jan. 1995.

[17] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, "Gaussian processes for time-series modelling," *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, vol. 371, p. 20110550, Feb. 2013.

[18] D. Duvenaud, *Automatic Model Construction with Gaussian Processes.* PhD thesis, University of Cambridge, Nov. 2014.

[19] M. L. Stein, "Space-Time Covariance Functions," *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 310–321, 2005.

[20] P. Abrahamsen, *A Review of Gaussian Random Fields and Correlation Functions.* Norsk Regnesentral/Norwegian Computing Center, 1997.

[21] A. Wilson, *Covariance Kernels for Fast Automatic Pattern Discovery and Extrapolation with Gaussian Processes.* PhD thesis, University of Cambridge, 2014.

[22] M. L. Stein, *Interpolation of Spatial Data: Some Theory for Kriging.* Springer Series in Statistics, New York: Springer-Verlag, 1999.

[23] D. J. C. Mackay, "Introduction to Gaussian processes," in *Neural Networks and Machine Learning* (C. Bishop, ed.), pp. 133–165, Springer-Verlag, 1998.

[24] Y. Bengio, O. Delalleau, and N. L. Roux, "The Curse of Highly Variable Functions for Local Kernel Machines," *Advances in Neural Information Processing Systems*, p. 8, 2006.

[25] D. Duvenaud, H. Nickisch, and C. E. Rasmussen, "Additive Gaussian Processes," *arXiv:1112.4394 [cs, stat]*, Dec. 2011.

[26] Y. Cho and L. K. Saul, "Kernel Methods for Deep Learning," in *Advances in Neural Information Processing Systems 22* (Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, eds.), pp. 342–350, Curran Associates, Inc., 2009.

[27] K. Krauth, E. V. Bonilla, K. Cutajar, and M. Filippone, "AutoGP: Exploring the Capabilities and Limitations of Gaussian Process Models," Oct. 2016.

[28] A. G. Wilson and R. P. Adams, "Gaussian Process Kernels for Pattern Discovery and Extrapolation," in *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[29] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian Processes for Big Data," *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, 2013.

[30] M. Bauer, M. van der Wilk, and C. E. Rasmussen, "Understanding Probabilistic Sparse Gaussian Process Approximations," *Advances in Neural Information Processing Systems 29*, 2016.

[31] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Transactions on Power Systems*, vol. 16, pp. 44–55, Feb. 2001.

[32] J. Quiñnonero-Candela, C. E. Rasmussen, and R. Herbrich, "A Unifying View of Sparse Approximate Gaussian Process Regression," *Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, Dec. 2005.

[33] M. Seeger, C. K. I. Williams, and N. D. Lawrence, "Fast Forward Selection to Speed Up Sparse Gaussian Process Regression," in *In Workshop on AI and Statistics 9*, 2003.

[34] E. Snelson and Z. Ghahramani, "Sparse Gaussian Processes using Pseudo-inputs," in *Advances in Neural Information Processing Systems 18* (Y. Weiss, B. Schölkopf, and J. C. Platt, eds.), pp. 1257–1264, MIT Press, 2006.

[35] M. K. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *In Artificial Intelligence and Statistics 12*, pp. 567–574, 2009.

[36] M. K. Titsias, "Variational Model Selection for Sparse Gaussian Process Regression," tech. rep., School of Computer Science, University of Manchester, 2009.

[37] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge, UK ; New York: Cambridge University Press, 3 edition ed., Sept. 2007.

[38] J. Hinman and E. Hickey, "Modeling and Forecasting Short-term Electricity Load Using Regression Analysis," Jan. 2009.

[39] J. Nocedal, "Updating Quasi-Newton Matrices with Limited Storage," *Mathematics of Computation*, vol. 35, no. 151, pp. 773–782, 1980.

[40] S. van der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy Array: A Structure for Efficient Numerical Computation," *Computing in Science Engineering*, vol. 13, pp. 22–30, Mar. 2011.

[41] W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, vol. 51-56 (2010).

[42] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman, "GPflow: A Gaussian process library using TensorFlow," *Journal of Machine Learning Research*, vol. 18, pp. 1–6, Apr. 2017.

[43] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Computing in Science Engineering*, vol. 9, pp. 90–95, May 2007.

[44] R. M. Neal, "MCMC using Hamiltonian dynamics," June 2012.

[45] D. Duvenaud, O. Rippel, R. P. Adams, and Z. Ghahramani, "Avoiding pathologies in very deep networks," *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, p. 9, 2014.

Appendix A: Notation and abbreviations

A.1 Notation

I have tried my best to keep the notation consistent with current standards in the statistics and machine learning literature, but some disparity exist between as well as and within the communities. **Bold** Latin or Greek letters always denotes matrices when uppercase, vectors when lowercase. Scalars are lowercase and non-bold. No notational distinction is made beween stochastic and non-stochastic variables. The following list includes abbreviations and symbols used in this thesis.

A.2 List of abbreviations

acos	arc-cosine
ARD	automatic relevance determination
DST	daylight saving time
GP	Gaussian process
GSH	general staff holiday
HMC	Hamiltonian Monte Carlo
KL	Kullback-Leibler
L-BFGS	limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm
MAP	maximum a posteriori
MAPE	mean absolute percentage error
Mat	Matérn
NTE	Nord-Trøndelag Elektrisitetsverk
pdf	probability density function
SE	squared expontential
SM	spectral mixture
UTC	coordinated universal time

A.3 List of symbols

A	sine-transformation with yearly period applied to time labels
D	two-dimensional sine-cosine-transformation with daily periods applied to time labels
$\mathcal{D}$	input data
$\mathbb{E}$	expected value
$\mathcal{GP}$	Gaussian process distribution
H	holiday binary variables
$\boldsymbol{I}$	identity matrix
J_n	n-th order angle-capturing function
$\boldsymbol{K}$	GP covariance matrix

$\boldsymbol{K}_n$	covariance matrix for locations where we have observations
$\boldsymbol{K}_*$	covariance matrix for query (test) locations
$\boldsymbol{L}$	lower triangular matrix; Cholesky factor
$\mathcal{L}$	log marginal likelihood
$\widetilde{\mathcal{L}}$	log marginal likelihood lower bound
$\mathcal{M}$	model space
$\mathcal{N}(\mu, \sigma^2)$	univariate Gaussian distribution with mean μ and variance σ^2
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$
P_h	h-lagged power load time-series
Q	number of spectral mixture components
$\mathbb{R}^d$	d-dimensional Euclidian space
S_h	h-lagged solar irradiance time-series
$\widetilde{S}_h$	h-lagged solar irradiance forecast time-series
T_h	h-lagged temperature time-series
$\widetilde{T}_h$	h-lagged temperature forecast time-series
V_h	h-lagged wind speed time-series
$\widetilde{V}_h$	h-lagged wind speed forecast time-series
W	two-dimensional sine-cosine-transformation with weekly periods applied to time labels
$\mathcal{X}$	input (index) set
$\boldsymbol{X} \in \mathcal{X}$	input points
$\boldsymbol{Z}$	inducing input points
l	length-scale hyperparameter
log	natural logarithm
m	number of inducing variables
p	probability density function (pdf)
t_h^d	time point: hour h of date d
$\boldsymbol{u}$	inducing variables
$\mathrm{var}(v_i)$	variance of random variable v_i
$\boldsymbol{\Sigma}_{i\|j}$	covariance of $\boldsymbol{x}_i$ given $\boldsymbol{x}_j$
$\delta(i,j)$	the Kronecker-delta
η	additive Gaussian white noise
κ	covariance function
κ_c	constant kernel; bias kernel
κ_{lin}	linear kernel
κ_{M}	Mahalanobis kernel
$\kappa_{\mathrm{Mat}}^{(v)}$	Matérn kernel
κ_{obs}	kernel for modelling observed data
$\kappa_{\mathrm{outlook}}$	kernel for modelling predicted data
κ_{SE}	squared exponential covariance function

κ_{WN}	white noise kernel
μ	univariate mean
$\boldsymbol{\mu}$	vector of means
$\boldsymbol{\mu}_{i\|j}$	mean of $\boldsymbol{x}_i$ given $\boldsymbol{x}_j$
ω_+	specific additive model; defined in section 7.6
$\omega_\times$	specic multiplicative model; defined in section 7.6
σ^2	univariate variance
σ_κ^2	kernel variance parameter
σ_n^2	likelihood variance
θ	hyperparameter
$\boldsymbol{\theta}$	vector of hyperparameters
$\vartheta = \vartheta(\boldsymbol{x}, \boldsymbol{x}')$	Angle between $\boldsymbol{x}$ and $\boldsymbol{x}'$
υ	parameter of $\kappa_{\mathrm{Mat}}^{(\upsilon)}$
ζ	function combining κ_{obs} and $\kappa_{\mathrm{outlook}}$
$\|\cdot\|$	determinant