# Project 3

## *Computational Physics and Machine Learning*

Wenxue Cao, Fahimeh Najafi, and Tómas Zoëga

**Abstract**

A 185 year long timeseries of monthly mean surface air temperature from the town of Stykkishólmur in western Iceland is fitted using Gaussian processes regression. We construct three different kernels and analyse the results arrived at by each of them. One kernel is constructed in such a way it should capture all the main characteristics of the timeseries, namely the annual cycle, long-term trend, and noise or short term variability. This kernel we term $k_{tseries}$. The other two kernels are similar to each other but differ slightly in their structure. They are only meant to capture the main feature of the timesries, namely the periodicity, and we term them $k_{period}$ and $k_{basic}$. We find that the most complex of the kernels, $k_{tseries}$, gives consistent results for a wide range of experimental set-up but is also computationally expensive, taking up to six times longer to finish as task as compared to the most simple kernel, $k_{basic}$. $k_{basic}$ is also able to perform well but depends highly on external parameters such as noise in the training data and the train/test split. In fact, $k_{basic}$ is able to reach the same levels of correlation coefficients and root-mean-square-error as $k_{tseries}$ in certain cases. Of the three, $k_{period}$ was found to be the least suited to our problem. Based on our results we recommend that unless the underlying data is known all the better, the best choice of a kernel for Gaussian processes regression is a simple, non-problem specific one.

# Contents

# 1 Introduction

During terrible events in the cyberspace, the Icelandic Meteorological Office lost all its data. This includes timeseries of observations spanning decades. Fortunately, meteorologists are not the only type of people interested in the weather and around the country countless farmers have kept good records of the weather throughout the centuries. However, paper can be lost as easily as digital data and old observations are now hard to come by. It therefore came as a surprise when temperature records from the town of Stykkishólmur in western Iceland were found among old insurance papers at the bottom of a drawer in a fish meal factory. And not only that, there were two versions. One was in the form of a vinyl record with a complete timeseries of monthly mean temperatures between 1830 and 2015 in Stykkishólmur, officially registered by the Icelandic Met Office. Why someone stored the data in this format is a mystery. The other item was a journal from a nearby farm. It included notes on the weather and from those notes it was possible to derive the monthly mean temperature. Unfortunately parts of the journal were missing and hence some periods are missing data. Journals like this have now been found in various places around the country but they also only provide fractional knowledge about the weather of the past. No other vinyl records have been found.

This finding is important since it gives us the opportunity to see if we can somehow reconstruct the lost timeseries temperature using fragmented notes from farmers. We have decided to apply Gaussian processes regression on the Stykkishólmur data and see if this method can be used to reconstruct the lost temperature records. If successful, similar models can be applied to data from other old journals and hence help us reclaim some of the knowledge which now is lost. We choose to use Gaussian processes in our regression since that way we can make some educated assumptions about the data we want to fit. By constructing a plausible prior distribution for the data in the available temperature timeseries, we hope to arrive at reliable results.

In section 2 we briefly introduce the main concepts of Gaussian processes, in section 3 we describe the data and the methods used to fit, in section 4 we present and discuss our results, and finally we conclude this report by summarizing our main findings in section 5.

# 2 Theory

## 2.1 Priors and posteriors

In Bayesian statistics, a *prior probability*, or just a prior, is the probability of a hypothesis ($H$) to be true given some prior information ($I$). This information could for example be the assumption that a die is fair and when it is thrown it is equally likely to land on any of its sides. Priors therefore contain information about the assumptions we make about the function we are trying to model. Formally, a prior can be written as $p(H|I)$.

When new data ($D$) has been acquired, for example by throwing the die and observing the result, the prior can be updated to a *posterior probability*. It can be expressed as $p(H|I, D)$. In other words, a posterior is a combination of data and a prior [1]. Based on this relationship between priors and posteriors we can see that the choice of the prior is very important when it comes to the final prediction of a model.

## 2.2 Gaussian processes

According to Rasmussen and Williams [1] (see chapter 2.2 in their book), a Gaussian process is a collection of random variable which all have a joint normal distribution. A Gaussian process is therefore defined by a joint normal distribution which again is completely defined by its mean and covariance functions. The mean function calculates the mean value of an unknown function $f(x)$ in the target space for some point $x$ from the feature space. It can be written as

$$m(x) = \mathbb{E}\left[f(x)\right]. \tag{1}$$

Similarly, the covariance function, which is also called a *kernel*, can be written as

$$k(x_i, x_j) = \mathbb{E}\left[(f(x_i) - m(x_i))(f(x_j) - m(x_j))\right] \tag{2}$$

where $x_i$ and $x_j$ are some pair of $x$'s from the feature space.

If we assume that there exists a function $f(x)$ which describes the underlying data and that function has been trained on the datapoints $\mathbf{x}_{tr}$, $f_{tr}$, then a Gaussian prior for a noiseless data can be written as

$$p(f_{tr}, f_{te}|\mathbf{x}_{tr}, x_{te}) = \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}_{tr}) \\ m(x_{te}) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_{tr}, \mathbf{x}_{tr}) & k(\mathbf{x}_{tr}, x_{te}) \\ k(x_{te}, \mathbf{x}_{tr}) & k(x_{te}, x_{te}) \end{bmatrix}\right) \tag{3}$$

and the posterior for a testing datappoint $x_{te}$

$$p(f_{te}|f_{tr}, \mathbf{x}_{tr}, x_{te}) = \mathcal{N}(\mu_{te}, Cov(f_{te})) \tag{4}$$

where $f_{te}$ is the value of the function at a training point $x_{te}$. That is, the Gaussian prior is equal to the normal distribution with mean $\mu_{te}$ and covariance $Cov(f_{te})$ which can be expressed as

$$\mu_{te} = m(x_{te}) + k(x_{te}, \mathbf{x}_{tr})k(\mathbf{x}_{tr}, \mathbf{x}_{tr})^{-1}(f_{tr} - m(\mathbf{x}_{tr})), \tag{5}$$

$$Cov(f_{te}) = k(x_{te}, x_{te}) - k(x_{te}, \mathbf{x}_{tr})k(\mathbf{x}_{tr}, \mathbf{x}_{tr})^{-1}k(\mathbf{x}_{tr}, x_{te}). \tag{6}$$

If we add normally distributed noise $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ to the training data $f(\mathbf{x}_{tr})$ then Eqs. 5 and 6 become

$$\mu_{te} = m(x_{te}) + k(x_{te}, \mathbf{x}_{tr})[k(\mathbf{x}_{tr}, \mathbf{x}_{tr}) + \sigma^2 I]^{-1}(f_{tr} - m(\mathbf{x}_{tr})), \tag{7}$$

$$\mathrm{Cov}(f_{te}) = k(x_{te}, x_{te}) - k(x_{te}, \mathbf{x}_{tr})[k(\mathbf{x}_{tr}, \mathbf{x}_{tr}) + \sigma^2 I]^{-1}k(\mathbf{x}_{tr}, x_{te}) \tag{8}$$

where $I$ is the identity matrix. This process of calculating the prediction are repeated for all testing points $x_{te}$. Notice that the equations above include a matrix inversion which can make a regression with Gaussian processes computationally expensive when the number of training datapoints is high.

To summarize, when doing a regression using Gaussian processes we assume a prior probability distribution over our target function. We then train the model on some data and arrive at a posterior distribution for the same target function. The mean of this posterior is the most likely value for the target function at given coordinates and the width of the posterior, often presented as standard deviation or variance, the uncertainty related to the function value at this specific point [2]. Note that this uncertainty is not the same as including uncertainty on the data used for training. That comes in addition to the uncertainty related to the width of the posterior distribution.

# 3 Methods

## 3.1 Data

The data we look at in this report is timeseries of monthly mean surface air temperature from Stykkishólmur in Iceland. The timeseries extends from January 1830 to December 2015 and includes the mean temperature for every month in between. It is available from the website of the Icelandic Met Office [3]. In all, the timeseries includes 2232 months. In our following analysis we always use all of this data. We only vary which data is assigned to training and which to testing.

## 3.2 Gaussian processes in Scikit-Learn

The machine learning library Scikit-Learn [4] includes a wide variety of functions useful for data analysis. Among them are tools for performing regression by Gaussian processes. When training our Gaussian processes regression models we use the default optimizer provided by Scikit-Learn. It is called `fmin_l_bfgs_b` which stands for L-BFGS-B or *limited-memory Broyden–Fletcher–Goldfarb–Shanno-B*. The L-BFGS-B optimizer was first introduced by Byrd et al. in 1995 [5] and is an extension of earlier versions of the limited-memory BFGS algorithm. The limited-memory BFGS algorithm was in turn introduced by Liu and Nocedal in 1989 [6] and is characterized by its low computational cost.

We use the function `GaussianProcessesRegressor` from Scikit-Learn for our regressions and it in turn uses the L-BFGS-B optimizer to maximise the logarithm of the marginal likelihood, namely $\log(p(\mathbf{y}|\mathbf{x}))$ where $\mathbf{x}$ and $\mathbf{y}$ are inputs and targets respectively. The optimization process can start from any allowed parameter values and from there the nearest optimum is found. This optimum might, however, be a local one and therefore we it is wise to repeat the optimization process several times, starting from different initial parameter values. We approach this problem by assigning the parameters values which we think are the most likely ones and then iterate from there. Since the initial values are chosen at random for every optimization iteration apart from the first one, this practise serves no other purpose than to check if we are able to assume good enough parameters by ourselves.

So far we have spoken about model parameters without explaining them properly. In order to get a better feeling for those parameters we have to look at the kernels used by Gaussian processes.

## 3.3 Designing kernels

Kernels, also called covariance functions, are an integral part of Gaussian processes. They define the prior on the function space we are working with and hence the prediction of the model we are applying. It is therefore vital to choose the kernel well in order to get good results from a regression using Gaussian processes. Kernels can be very wildly but one popular way to construct them is to combine simple, general kernels in such a way they fit the problem at hand better. Here we will introduce a few elementary kernels and describe how we construct kernels we think might work well for a regression of a meteorological

temperature timeseries. A good guide to constructing kernels can be found in chapter 2 in David Kristjanson Duvenaud's PhD thesis [7].

### 3.3.1 Some simple kernels

A kernel or a covariance function measures the similarity between two input points. We expect that a model produces similar output for similar inputs. But is the kernel which defines which inputs are similar and which are not. Therefore it is basically the choice of kernel that controls which functions will be sampled from the function space. That shows the importance of the choice of kernel function in using Gaussian processes method.

There are several kernel functions that have been used and have resulted in good models. The kernel functions generally have one or more hyperparameters which must be tuned to find the best model. The following kernels are some simple kernels we used in Gaussian processes methods.

Radial Basis Function (RBF) kernel is a commonly used kernel in Gaussian processes and it is also known as the 'squared exponential' kernel. The kernel is given by:

$$k(x, x') = \exp\left(-\frac{|x - x'|^2}{2\sigma^2}\right) \tag{9}$$

where $\sigma$ is a length-scale parameter of this kernel.

White kernel is one of the basic kernels, which is normally used as part of a sum-kernel where it explains the noise of the signal. The parameter noise level equals the variance of this noise.

Exp-Sine-Squared kernel is a classic periodic function. It is parameterized by the hyperparameter $\sigma$ and a periodicity parameter $p$. The kernel is given by:

$$k(x, x') = \exp\left(-\frac{2}{\sigma^2}\sin^2\left(\frac{\pi(x - x')}{p}\right)\right) \tag{10}$$

where $\sigma$ is the length scale of the kernel and p is the periodicity of the kernel.

The Dot-Product kernel is non-stationary and can be obtained from line regression. It is parameterized by a parameter $\sigma_0$ which controls the inhomogeneity of the kernel.Normally it is combined with exponentiation such as RBF kernel or Exp-Sine-Squared kernel. The kernel is given by

$$k(x, x') = \sigma_0^2 + x \cdot x' \tag{11}$$

To ensure a valid kernel, the kernel matrix $\sum = k(x, x')$ should be positive definite, which means the matrix should be symmetric and invertible.

### 3.3.2 Kernels for a temperature timeseries

Without looking at any data we can make some educated guesses about the behaviour of the monthly mean temperature timeseries form Stykkishólmur. First of all we can assume that the temperature follows a strict annual cycle, lower temperatures in winter and higher

in summer. The magnitude of this cycle is unknown but we can expect it to be quite similar form year to year, although not exactly the same. We can also expect some sort of a long-term trend. Finally we can expect noise in the data. This noise can have many sources. Those can for example be uncertainty in measurements and instruments, errors in reading the instruments and record the data, and natural variability of the climate system. A general setup for a kernel describing this kind of temperature timeseries could be on the form

$$kernel = trend + period + noise \tag{12}$$

where *kernel* is the final kernel, *trend* is a kernel accounting for a long-term trend, *periodic* accounting for the annual cycle, and *noise* taking noise and uncertainties into account. For the *trend* kernel we have several options. Her we assume a linear trend and suggest:

$$k_1(x, x') = \theta_0^2 \exp\left(-\frac{(x - x')^2}{2\theta_1^2}\right) \cdot (\theta_2^2 + x \cdot x') \tag{13}$$

where $(x, x')$ is some pair of coordinates and the $\theta$'s parameters to be tuned. $\theta_0$ is the magnitude of an expected change in mean temperature over the whole timeseries in °C and $\theta_1$ is a lengthscale corresponding to the length of the timeseries in years. For the *period* part we have

$$k_2(x, x') = \theta_3^2 \exp\left(-\frac{(x - x')^2}{2\theta_4^2}\right) \exp\left(-\frac{2}{\theta_5} \sin^2(\pi(x - x'))\right). \tag{14}$$

Here the period has been fixed to one year, $\theta_3$ is the periodic amplitude, $\theta_4$ the lengthscale controlling the decay of the period, making sure the cyclic behaviour does not repeat it self exactly but evolves in time, and $\theta_5$ controls how smooth each cycle is. Finally we have the *noise* part:

$$k_3(x, x') = \theta_6^2 \left[\exp\left(-\frac{(x - x')^2}{2\theta_7^2}\right) + \theta_8 \delta_{x,x'}\right] \tag{15}$$

where $\theta_6$ is the magnitude of the noise present between datapoint, $\theta_7$ its lenghtscale as earlier, and $\theta_8$ the magnitude of the noise related to each individual datapoint. The sub-kernels described in Eqs. 13 to 15 are inspired by chapter 5.4.3 in Rasmussen and Williams [1].

We use functions from the machine learning library Scikit-Learn [4] throughout this report. The kernels in Eqs. 13 to 15 are constructed from a few functions each, all of which are available in Scikit-Learn. In terms of Scikit-Learn's function, the above kernels can be written as:

$$k_1 = \theta_0^2 \cdot \texttt{RBF}(\theta_1) \cdot \texttt{DotProduct}(\theta_2), \tag{16}$$

$$k_2 = \theta_3^2 \cdot \texttt{RBF}(\theta_4) \cdot \texttt{ExpSineSquared}(\theta_5), \tag{17}$$

$$k_3 = \theta_6^2 \cdot [\texttt{RBF}(\theta_7) + \texttt{WhiteKernel}(\theta_8)]. \tag{18}$$

The terms written in the `monospace` font are functions from Scikit-Learn and correspond exactly to the analytical functions in Eqs. 13 to 15. For more information about these

functions, see Scikit-Learn's documentation [4]. For the analysis of the temperature timeseries we will use three different kernels. They are

$$k_{tseries} = k_1 + k_2 + k_3, \tag{19}$$

$$k_{period} = k_2, \tag{20}$$

$$k_{basic} = \texttt{ExpSineSquared}(\text{default params.}). \tag{21}$$

Kernel $k_{tseries}$ is supposed to capture many of the underlying processes the timeseries displays, kernel $k_{period}$ is focused on capturing the annual cycle, and kernel $k_{basic}$ is focused in capturing any periodicity, be it annual or not. The last kernel is an out-of-the-box kernel from Scikit-Learn and the simplest of the three. The point of including it is to see if a default kernel is sufficient for the kind of modelling we do in this report.

To calculate the Gaussian processes regression we use the function `GaussianProcessRegressor` from Scikit-Learn. To find the optimal values for the parameters $\theta$ we use the inbuilt `fmin_l_bfgs_b` optimizer and iterate over it a predefined number of times. The optimizer samples $\theta$ values for a log-uniform distribution for allowed values for the $\theta$'s and starts at random coordinates. It is however possible to specify the starting point of the optimization process by giving the $\theta$'s manual initial values, see Table 2 in Section 4. Here we have chosen values characteristic for our timeseries based on our prior knowledge about the climate in Stykkishólmur as it is today and the length of the timeseries.

## 3.4 Performance metrics

We use two main performance metrics in this report to compare the true and fitted data. One is the root-mean-square-error (RMSE) given by

$$RMSE = \sqrt{\frac{1}{N}(\mathbf{y}_{true} - \mathbf{y}_{fit})^2} \tag{22}$$

where $N$ is the number of datapoints, and $\mathbf{y}_{true}$ and $\mathbf{y}_{fit}$ are the true and fitted targets respectively. The other performance metric is the correlation coefficient given by

$$\rho = \frac{\text{Cov}(\mathbf{y}_{true}, \mathbf{y}_{fit})}{\sigma_{true}\sigma_{fit}} \tag{23}$$

where $\sigma_{true}$ and $\sigma_{fit}$ are the standard deviations of $\mathbf{y}_{true}$ and $\mathbf{y}_{fit}$ respectively, and $\text{Cov}(\mathbf{y}_{true}, \mathbf{y}_{fit})$ is the covariance.

# 4 Results and discussion

Fig. 1 shows the correlation coefficient for the Stykkishólmur temperature timeseries as a function of the number of times the optimizer is restarted. The training data is a random sample of 50% of the available datapoints and the testing data is the remaining 50%. We use five different random splits, represented by the different lines in each subplot, and try out three different kernels. In Fig. 1(a) we use $k_{tseries}$, in (b) $k_{period}$, and (c) $k_{basic}$. As we can see the kernels behave somewhat differently. In all cases the correlation coefficient seems to peak somewhere between 0.8 and 0.9 for high enough number of optimizer restarts. When the number of optimizer restarts is 0 only one optimization run is performed, starting from the initial parameters of the kernels. In some cases this peak correlation is already achieved during the first optimisation iteration, indicating that the choice of the initial parameters displayed in Table 2 is not too bad.

When comparing $k_{tseries}$ to the other two kernel we see that it seems to be more consistent and arrive at the highest correlation after fewer number of optimisation iterations. However, each iteration takes much longer time for $k_{tseries}$ than the other two since it is more complicated and requires many more operations to be calculated. This can be demonstrated by training the regression model on the same data with each of the three kernels and measure the time it takes. We did this on one of our laptops and set the number of optimizer restarts to five. The training times turned out to be 137 s, 26 s, and 23 s, for $k_{tseries}$, $k_{period}$, and $k_{basic}$ respectively. In this case the respective correlation coefficients were 0.858, 0.801, and 0.803. Based on these results we can see that we should use as many optimisation iterations and computationally viable.

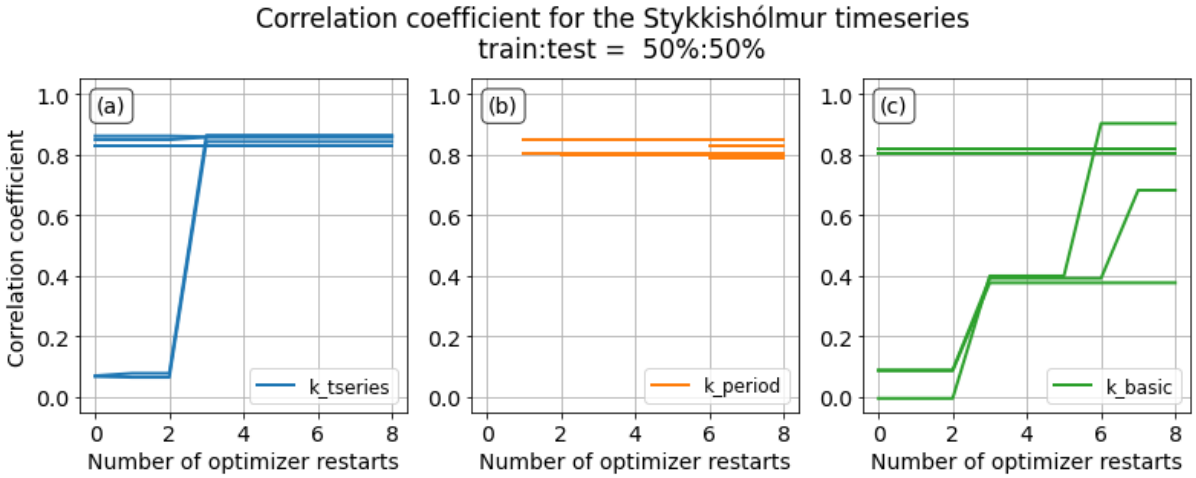


**Figure 1:** *The correlation coefficient as a function of number of optimizer restarts for the Stykkishólmur temperature timeseries. 50% of the data is used for training and the remaining 50% for testing. The correlation coefficient is calculated for the testing data. The different lines in each plot represent different random splits of the data. Three different kernels are used, namely (a) $k_{tseries}$, (b) $k_{period}$, and (c) $k_{basic}$.*

One thing to notice in Fig. 1 is the behaviour of the periodic kernel, $k_{period}$. As can be seen, the correlation coefficient is not calculated for low number of optimization iterations. The reason for this that the optimal length scale of the `RBF` sub-kernel in $k_{period}$,

namely $\theta_4$, approaches zero and as a result the solution does not converge. When this is the case, the default response of `GaussionProcessRegressor` (which the the function we use for regression) is to revert to the mean of the prior, which default value happens to be zero. This is illustrated in Fig. 2. It shows the first ten years of the timeseries (blue line) with modelled training (orange dots) and testing (green dots) values. In this case, a single optimization iteration was performed. Here we can see how the Gaussian processes regression model using $k_{period}$ fits the training data very well, and perhaps too well, but completely fails to fit any other data. This mode set-up predicted that all the testing data has a value of zero, i.e. that the testing data is constant as a function of time. This means that there is no variance in the predicted testing data and hence the covariance for the predicted and observed testing data is undefined. This is what we saw in Fig. 1(b). This indicates that if our kernel is only constructed to capture the periodic behaviour of the timeseries, simpler is better. That is, it is enough to assume a fixed cycle which does not decay away form exact periodicity. Therefore we should not use the $k_{period}$ kernel in isolation when trying to reconstruct the Stykkishólmur temperate timeseries but rather the simpler $k_{basic}$ kernel. This suggestion is reinforced when comparing the correlation coefficients displayed in Figs. 1 (b) and (c) where these two kernels give very similar results. For the remainder of this report we will drop the $k_{period}$ kernel and only focus on $k_{tseries}$ and $k_{basic}$.

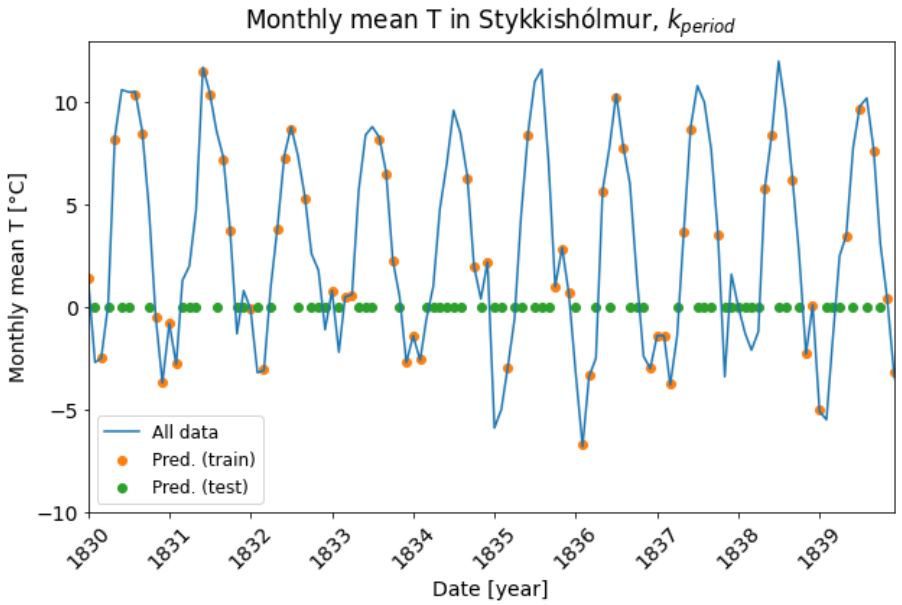


**Figure 2:** *Timeseries for the first years of the monthly mean temperature record from Stykkishólmur. The blue line indicates observed values, the orange dots are model predictions based on the training data, and the green dots are predictions based on the testing data. This is an example how the $k_{period}$ kernel can fail.*

So far we have worked with the assumption that the training data includes noise which follows a normal distribution and has a variance of 0.5°C$^2$. In Figs. 3 (a) and (b) we have plotted the correlation coefficient as a function of different levels of noise for the kernels $k_{tseries}$ and $k_{basic}$. Note that this is an artificial noise added to the data after-the-fact and an assumption on our behalf. When using $k_{tseries}$ we get a maximum correlation of about 0.83 to 0.87 for variances between 0.5 and 2.0°C$^2$. As the variance increases the

correlation decreases. For all the different random splits, the correlation coefficient drops by about 0.02 when going from a variance of 2°C$^2$ to 5°C$^2$. If we look at $k_{basic}$ we see that the maximum correlation reaches a maximum value of around 0.92. This is considerably higher than the maximum correlation for $k_{tseries}$. However, while $k_{tseries}$ seems to be relatively insensitive to changes in the noise level, the opposite holds for $k_{basic}$. In the case of the latter kernel the split into training and testing data also seems to have much greater effects on the final results. So even though $k_{basic}$ seems to be able to arrive at better results than $k_{tseries}$ under certain circumstances, $k_{tseries}$ gives more reliable results. The reason for this might be that noise is taken into account when $k_{tseries}$ is constructed, see Eq. 12.

The RMSE in Figs. 3 (c) and (d) shows very similar behaviour as the correlation coefficient but instead of getting slightly worse results as the variance of the noise is increased (lower correlation), we get better (lower RMSE). This is not surprising since as the variance of the noise added to the training data is increased, the regression method is allowed to fit the data more loosely. This means that the model does not try as hard to fit the training data and the contribution of the month-to-month variability to the final results is smaller, leading to lower RMSE for the testing data. In the case of $k_{tseries}$, the dependence on the variance is somewhat higher for the RMSE compared to the correlation coefficient. As before, the more complex and problem specific kernel has an edge over the more simple one when it comes to reliability. However, this comes at the cost of a longer computing time. Based on these results we will add normally distributed noise with 1°C$^2$ variance to the training data for the rest of our analysis.

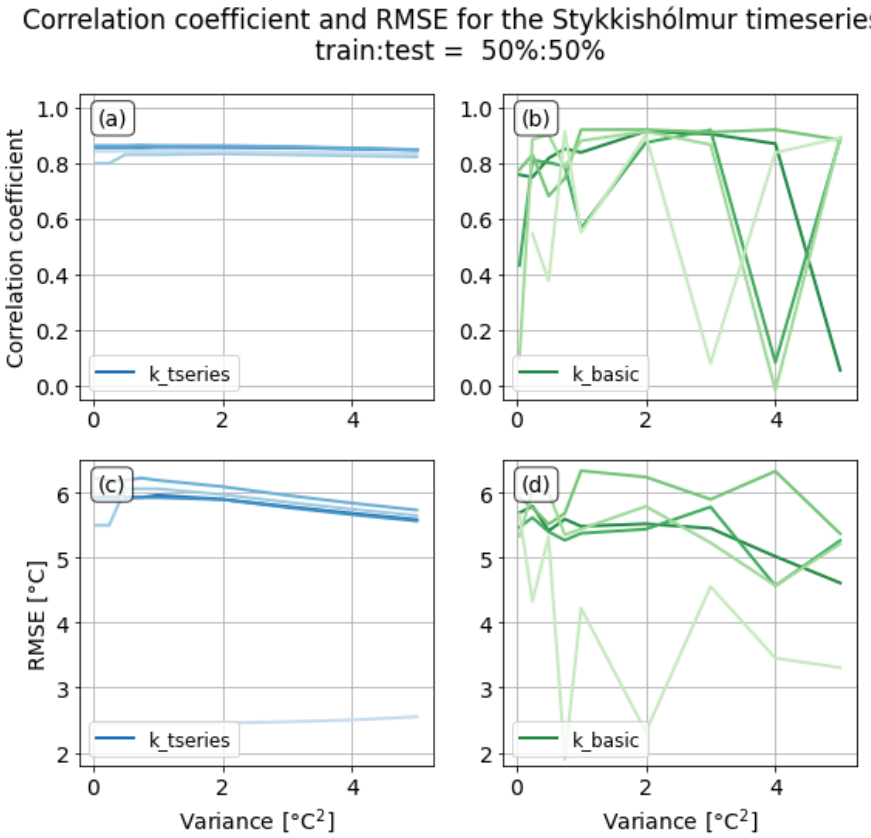


**Figure 3:** *The correlation coefficient and the RMSE as a function of the variance of the artificial noise added to the Stykkishólmur temperature timeseries. 50% of the data is used for training and the remaining 50% for testing. The two performance metrics are calculated for the testing data. The different lines in each plot represent different random splits of the data, the same splits as in Fig. 1, and two different kernels are used, namely $k_{tseries}$ for (a) and (c), and $k_{basic}$ (b) and (d).*

Figs. 4 (a) and (b) show the correlation coefficient as a function of the size of the testing data. Both in the case of $k_{tseries}$ and $k_{bsaic}$ the correlation decreases as the test size increases. This should not come as a surprise since a smaller testing size means a larger training size and hence the regression models can fit a greater amount of the existing data in more details. As before we see how $k_{tseries}$ gives much more consistent results than $k_{basic}$, both when it comes to the amount of data being assigned to training and testing, and also to different random splits of the data.

The RMSE shown in Figs. 4 (c) and (d) increases as the test size increases. This is the case for both kernels. However, these figures indicate that when the test size becomes small enough the RMSE increases slightly. This is the case for some of the random splits when 90% of the data is used for training and 10% for testing and indicates overfitting. This is because in addition to the regular cyclic behaviour, potential trend, and noise, the temperature timeseries has a level of internal variability which is not taken into account when the training data is fit to closely.

Fig. 5 shows a part of the temperature timeseries from Stykkishólmur. The blue line is observations and the orange markers are model fits for the case marked with a red

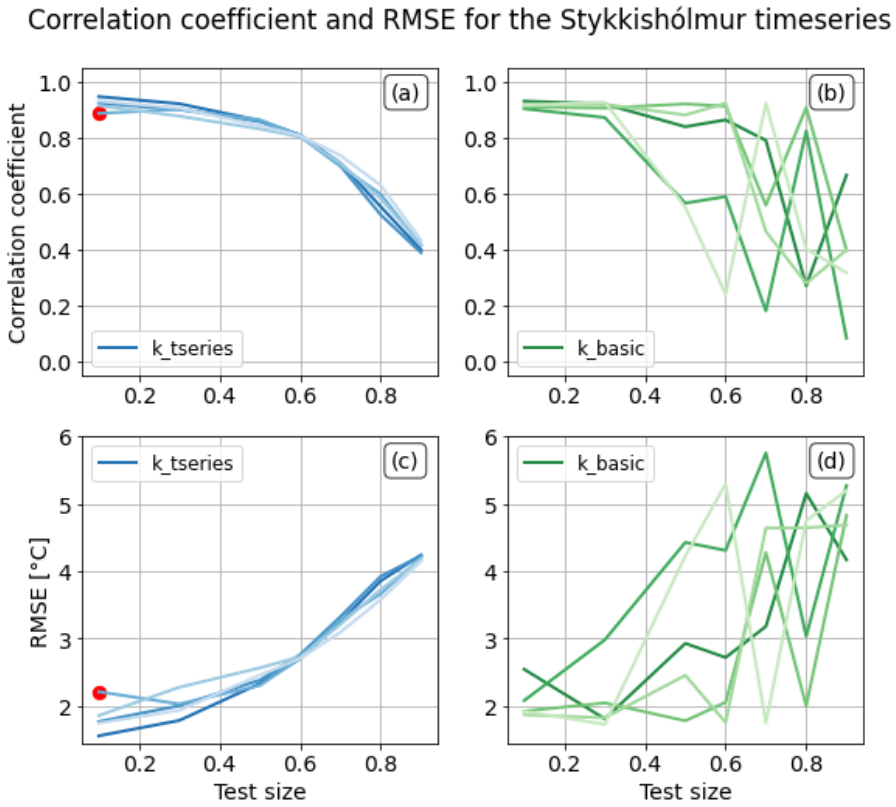


**Figure 4:** *The correlation coefficient and the RMSE as a function of the size of the testing data for the Stykkishólmur temperature timeseries. Here the test size is given in terms of fraction of the total data. The rest of the data is used for training. The different lines in each plot represent different random splits of the data, the same splits as in Fig. 1, and two different kernels are used, namely $k_{tseries}$ in (a) and (c), and $k_{basic}$ in (b) and (d). The red dots in (a) and (c) indicate the case illustrated in Fig. 5.*

dot in Fig. 4 (c). The $k_{tseries}$ kernel is used. The model is trained on 90% of the data (orange dots) and tested on the remaining 10% (orange triangles). The shaded area indicates a confidence range of two standard deviations or about 95%. Note that the model was fit using the entire timeseries from 1830 to 2015. For the training data the RMSE is 0.46°C and correlation coefficient 0.996. For the testing data the RMSE is 2.21°C and correlation coefficient 0.887. The performance is therefore much worse for the small testing data compared to the large training data, as could have been expected. However, when looking at Figs. 4 we see that for this 90%:10% train:test split, the results are worse than if the training size would be decreased to 70% for example. We further see in Fig. 5 that the model does not always manage to capture the finer structure of the timeseries. This is for example the case for the winter of 1972/73 where a cold January is not captured by the model. The reason is that January 1973 is a testing month and the model fits the surrounding training months too well to be able to accommodate for internal variability such as this. As we can see, the 95% confidence interval is not even close to cover the observed mean temperature in January 1973. Another example of where the regression uncertainty is underestimated is in the winter of 1970/71. There all months apart from October 1970 were used for training but still the model does not include the cold December 1970 in its 95% confidence interval, even though that month

was used for training. So what we are seeing here is both overfitting and an overconfident model.

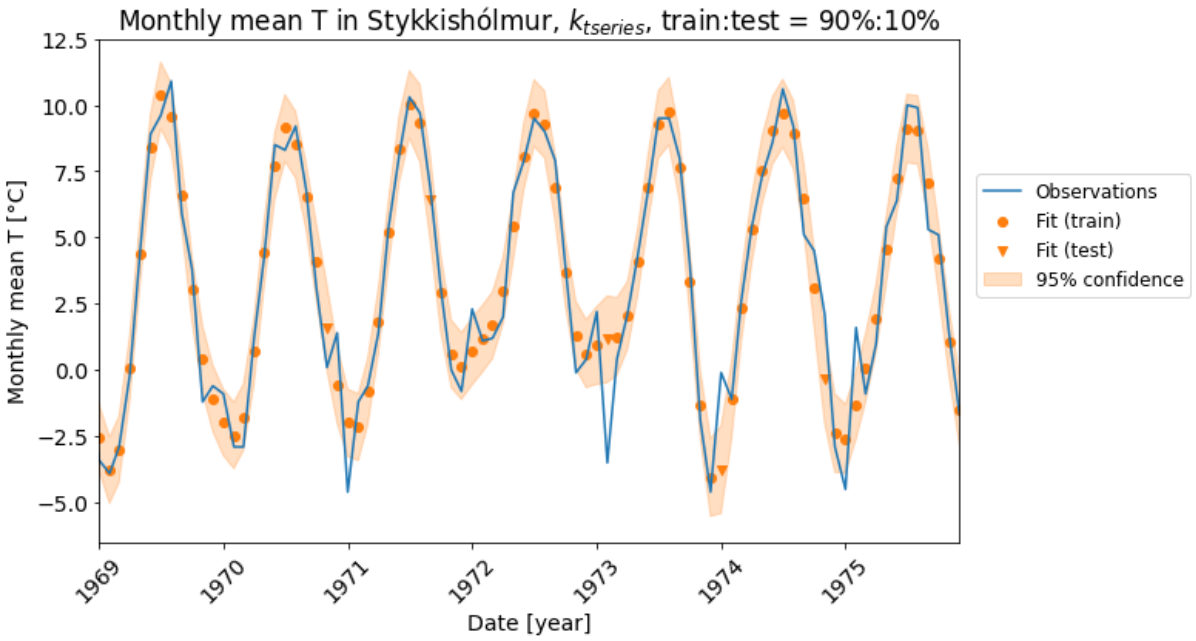


**Figure 5:** *A snippet from the Stykkishólmur temperature timeseries. The blue line represents observations, the orange dots fitted training data, the orange triangle fitted training data, and the shaded area 95% confidence of the regression.*

Fig. 6 shows a snippet of the Stykkishólmur temperature timeseries. The blue line represents observational data and the various dots (training) and triangles (testing) results from the Gaussian processes regression. The purple markers represent $k_{tseries}$ with initial parameters (see Table 2), the orange markers $k_{tseries}$ with updated parameters (again see see Table 2), and the green markers $k_{basic}$ with optimal parameters. Table 1 shows the correlation coefficients and root mean square errors (RMSE) for the results in Fig. 6. Here we are interested in looking at how well the Gaussian processes models can extrapolate into the future. We use the first 80% of the data for training and the remaining 20% for testing. As we can see, the highest training correlation and lowest training RMSE are achieved using $k_{tseries}$ with updated parameters. However, when looking at the testing data we see that the $k_{tseries}$ kernel is completely unable to capture the behaviour of the temperature beyond the extent of the training data. This is not the case for the $k_{basic}$ kernel which seems to be able to reproduce the testing data quite well.

The reason for the inability of $k_{tseries}$ to extrapolate into the future most likely comes down to the optimal value of the parameter $\theta_4$. As can be seen in Eqs. 20 and 17, $\theta_4$ is the time scale which controls the decay of the periodicity. It is included to allow for deviations from exact periodicity when fitting the data. Here, this approach has backfired and once beyond the range training data, our $k_{tseries}$ regression model dampens all oscillations down to almost nothing in a matter of months. If we look at Table 2, we see that the optimal value for $\theta_4$ is 2.43 years whereas we expected it to be more like 100 years. This short optimal time scale might be the result of the model taking into account variations in the seasonal cycle between different years. That is, one seasonal cycle can be quite different from the next and the model finds that the best way to accommodate for this is to

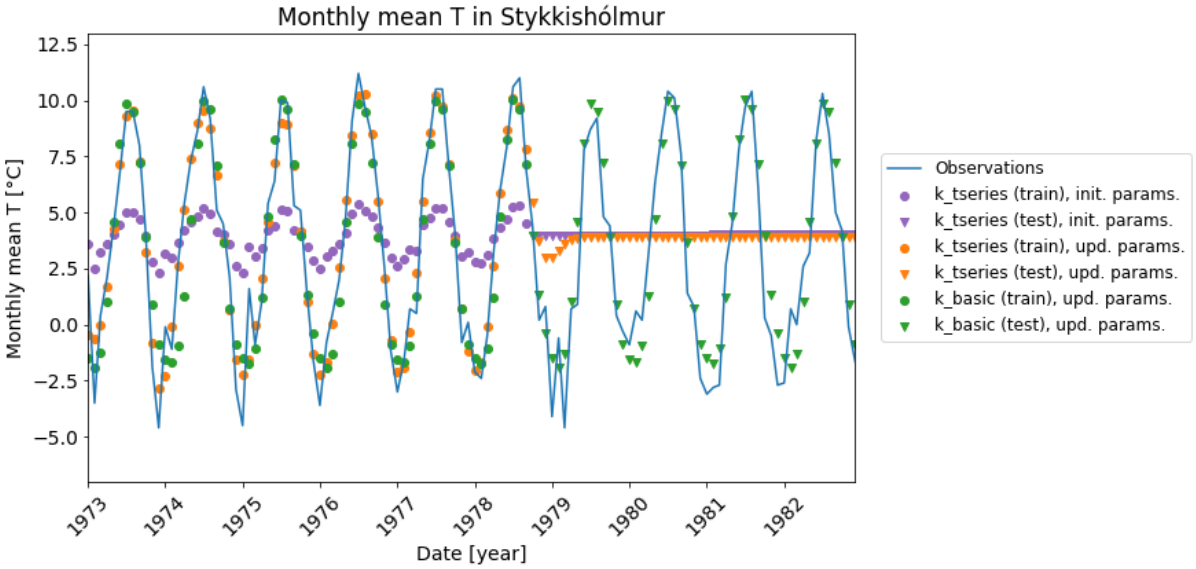


**Figure 6:** *A snippet from the Stykkishólmur temperature timeseries. The blue line represents observational data and the various dots (training) and triangles (testing) results from the Gaussian processes regression. The purple markers represent $k_{tseries}$ with initial parameters (see Table 2), the orange markers $k_{tseries}$ with updated parameters, and the green markers $k_{basic}$ with updated parameters.*

**Table 1:** *Correlation coefficients and root mean square errors (RMSE) for the results displayed in Fig. 6. The cases $c_1$, $c_2$, and $c_3$ represent the training data for $k_{tseries}$ w/ initial parameters, $k_{tseries}$ w/ updated parameters, and $k_{basic}$ w/ updated parameters respectively. In the same way cases $c_4$, $c_5$, and $c_6$ represent the testing data.*

| | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ |
|---|---|---|---|---|---|---|
| Corr. coeff. | 0.957 | 0.972 | 0.926 | 0.148 | 0.155 | 0.946 |
| RMSE [°C] | 3.75 | 1.10 | 1.76 | 4.25 | 4.24 | 1.56 |

assign a small value to $\theta_4$. We think that due to the strong annual cycle, our model should be able to extrapolate beyond the range of the training data. Not long into the future but some years or decades at least. This turned out to be the case for $k_{basic}$ but not $k_{tseries}$.

Another instance where our guess for initial parameter value is way of is in the case of $\theta_7$. This parameter represents correlated noise, that is noise which carries from one month to the next. We assumed this time scale to be on the order of weeks to months but the model finds an optimal value on the order of tens of thousands of years. These discrepancies between our estimates and the optimal values, which turn out not to be overly realistic, indicate that the carefully structured $k_{tseries}$ kernels is perhaps not so carefully structured after all. This inference is supported by our previous results which showed that $k_{tseries}$ did not outperform $k_{basic}$ in a meaningful way. In fact, the main advantage of $k_{tseries}$ is that it is less sensitive to changes in external parameters such as the train:test split and the amount of noise added to the training data. On the other hand, its main weakness is that it is much slower than $k_{basic}$. When all things are considered,

it might be better to use a simpler and more general kernel when doing a Gaussian processes regression. This conclusion is further supported when we take into account that unlike when dealing with a monthly mean temperature series, often we do not know or understand the underlying behaviour of our data.

**Table 2:** *The coefficients $\theta$ used for $k_{tseries}$ in Fig. 6. In the top row we have the manually chosen initial values (Init.) and in the second the updated (Upd.) values after the model was trained on the first 80% of the datapoints from the Stykkishólmur temperature timeseries.*

| | $\theta_0$ [°C] | $\theta_1$ [years] | $\theta_2$ [°C] | $\theta_3$ [°C] | $\theta_4$ [years] | $\theta_5$ [-] | $\theta_6$ [°C] | $\theta_7$ [years] | $\theta_8$ [°C] |
|---|---|---|---|---|---|---|---|---|---|
| Init. | 0.5 | 185 | 0 | 12 | 100 | 10 | 0.5 | 0.1 | 0.5 |
| Upd. | 0.00316 | $1 \cdot 10^5$ | $6 \cdot 10^{-5}$ | 4.9 | 2.43 | 20.5 | 3.85 | $4.63 \cdot 10^4$ | 0.0626 |

# 5 Conclusions

In this report we performed regression by Gaussian processes on a 185 year long observational timeseries of surface air temperature. This timeseries includes monthly mean values from Stykkishólmur in western Iceland.

So-called kernels are an important part of Gaussian processes since they define the prior in the function space. The choice of kernel is therefore the basis for the Gaussian process posterior and hence the output from the regression. In this report we investigated three different kernels. We constructed one of them in a way we thought it might be able to capture the main features of the temperature timeseries. This includes the periodic behaviour as cold winters and warm summer oscillate on an annual basis, a possible long-term trend, and noise or internal variability in the climate system and the measurements. We call this kernel $k_{tseries}$. The second kernel, termed $k_{period}$, is the periodic part of the $k_{tseries}$ kernel. By using it, we attempted to capture the main feature of the timeseries, namely the periodic behaviour, but with less computational cost than required by $k_{tseries}$. This kernel takes into account that the annual cycle can change considerably from one year to the next, therefore providing flexibility in the fit. Finally we used a very basic kernel, termed $k_{basic}$. This kernel is also supposed to capture the periodicity but unlike $k_{period}$, $k_{basic}$ is an un-modified out-of-the-box kernel from the machine learning library Scikit-Learn [4]. The other kernels are also based on functions from Scikit-Learn but they have been modified to fit our problem better.

We found that the best kernel for the job was either the most complex one, $k_{tseries}$, or the most basic, $k_{basic}$. After varying external parameters related to the optimization of the regression, the amount of noise added to the observations, and the ratio between testing and training data, we saw that although $k_{tseries}$ gives more consistent results over a range of external parameters, in some cases $k_{basic}$ is able to match or even outperform the best results achieved by $k_{tseries}$. We also made an informal experiment where we compared the running times of the regression model using each of the kernels and found that $k_{tseries}$ needs almost sixfold the time $k_{basic}$ needs for the same task. As a results we cannot recommend one kernel over the other when it comes to the task reported upon here. However, if we were to perform a Gaussian processes regression on data we have much less prior knowledge about than a meteorological temperature timeseries, we strongly recommend using a simple kernel.

# References


[1] Christopher K Williams and Carl E Rasmussen. *Gaussian processes for machine learning*. MIT press, 2006. ISBN: 026218253X.

[2] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006. ISBN: 978-0387-31073-2.

[3] Icelandic Met Office. *Tímaraðir fyrir valdar veðurstöðvar (Timeseries for chosen weather stations)*. 2022. URL: `https://vedur.is/vedur/vedurfar/medaltalstoflur/` (visited on June 7, 2022).

[4] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[5] Richard H Byrd et al. "A limited memory algorithm for bound constrained optimization". In: *SIAM Journal on scientific computing* 16.5 (1995), pp. 1190–1208.

[6] Dong C Liu and Jorge Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1 (1989), pp. 503–528.

[7] David Duvenaud. "Automatic model construction with Gaussian processes". PhD thesis. University of Cambridge, 2014. DOI: `https://doi.org/10.17863/CAM.14087`.