UiT THE ARCTIC UNIVERSITY OF NORWAY

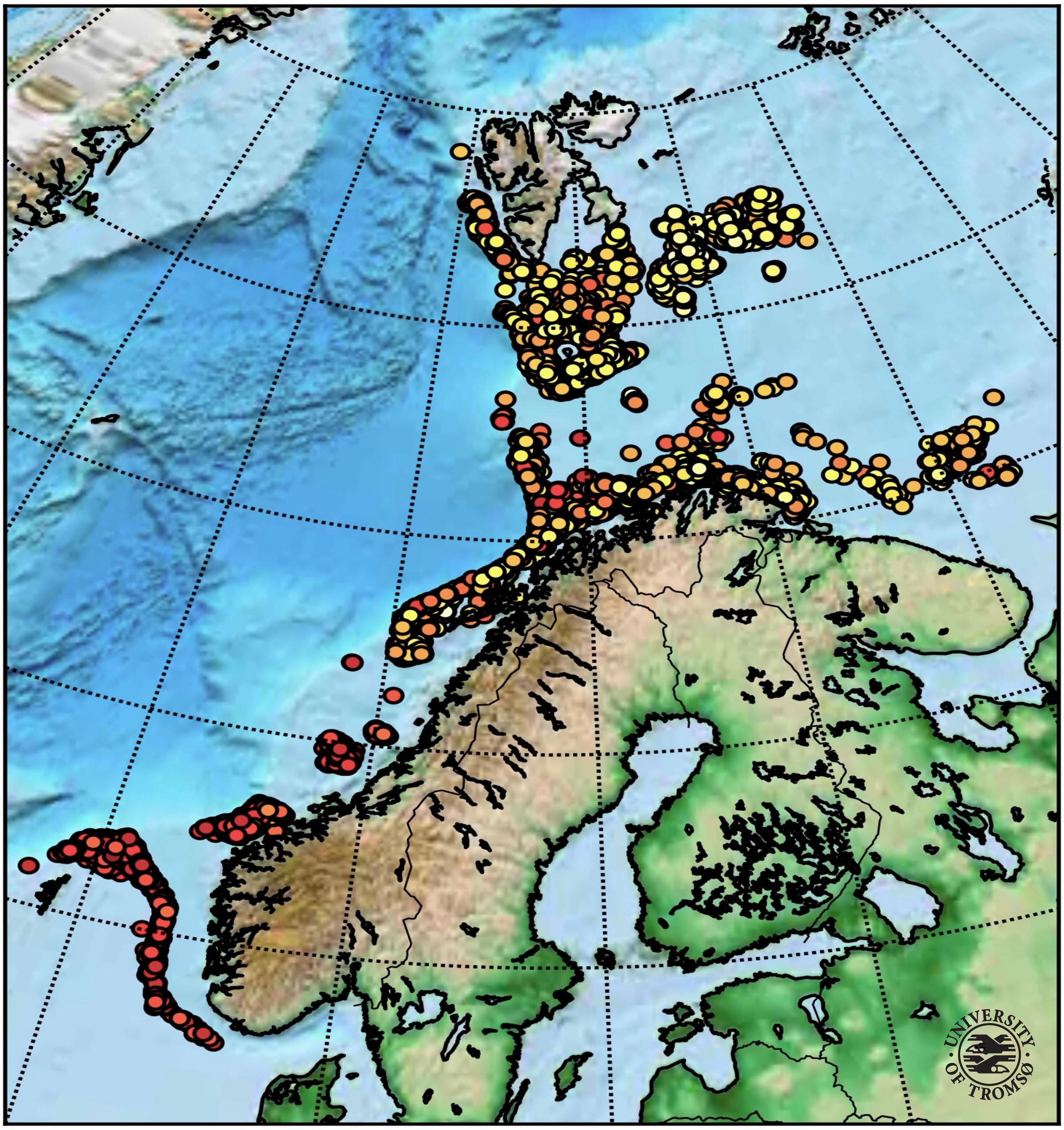
Faculty of Science and Technology
Department of Physics and Technology

An initial assessment of the possibilities of fish catch prediction using Gaussian processes

—

Sara Björk

FYS-3941 Master's thesis in applied physics and mathematics, December 2016



The cover image visualizes the 18,000 catch reports of North-east Atlantic Cod (Skrei), that are considered in this thesis. A deep red colour indicates a smaller amount of catch, while a brighter colour indicates a greater amount of catch.

Sara Björk

Abstract

The fishing and aquaculture industry is one of the largest industries of Norway. Enhanced knowledge of the distribution of fish in the ocean is important for an economical and sustainable fishing industry. This study investigates the possibilities of using Gaussian processes for regression within fish catch prediction. A dataset that combines catch reports from the Norwegian shipping company Havfisk ASA with a multitude of ocean-related data is created, and analysed in this thesis. Stochastic variational inference for Gaussian process models is used for the regression, as the method allows the use of Gaussian processes for regression on large datasets. The aim of this study is to assess the suitability of the dataset for fish catch predictions, in addition to evaluating the predictions from the Gaussian process model. Different investigations were performed within; time dependency, clustering analysis, data transformation and feature selection. The investigations indicates that Gaussian processes for regression do reveal a structure in the dataset and that the collected dataset is suitable for fish catch prediction.

Acknowledgements

First and foremost, I would like to thank my supervisor Associate Professor Robert Jenssen for your guidance, enthusiasm and for the constructive discussions. I would also thank you for introducing me to the framework of Gaussian processes and making it possible for me to attend to the Gaussian Process summer school, arranged at the Sheffield university in the UK.

Thanks to Senior Scientist Ståle Walderhaug for being my co-supervisor, and for letting me be a part of the interesting project eSushi at SINTEF Nord AS. I also want to thank my colleagues at SINTEF Nord AS for helping me extracting variables from the ocean models that I was using.

To the Norwegian shipping company Havfisk ASA for letting me get access to their logged data, and Dualog AS for providing me the data through their electronic logbook, `eFangst`.

To Annette Samuelsen at the Nansen Environmental and Remote Sensing Center for giving me access to the HYCOM-NORWECOM model, from which the biogeochemical were extracted.

To Arne Melsom and Magnar Reistad at the Norwegian Meteorological Institute for giving me access to the TOPAZ4 reanalysis product and the Nora10 hindcast model, from which the physical, wind and wave data were extracted.

To Radovan Bast and the other staff members at the high-performance computing group at UiT. Thank you for your fast and excellent service regarding the use of the supercomputer Stallo.

To my office mates and members of the most fashionable cheese club in Tromsø, Thomas A. Haugland Johansen, Rolf Ole R. Jenssen and Torgeir Brenn; thank you for your support and jokes.

To my dear friends and family, I am so grateful for your love, support, laugh and patience. I do really look forward to spend more time with you all! Special thanks to Torgeir Blæsterdalen and Theresa Rexer for proofreading this thesis.

Contents

List of Figures

List of Tables

Abbreviations

ARD, Automatic relevance determination

BFS, Backward feature selection

BIC, Bayesian information criterion

CLT, Central limit theorem

DSS, Decision support system

FFS, Forward feature selection

GP, Gaussian process

GPR, Gaussian process for regression

KL, Kullback-Leibler

LOOCV, Leave-one-out cross-validation

MAP, Maximum a posterior

ML, Machine learning

MSE, Mean square error

NN, Neural networks

PCA, Principal component analysis

RBF, Radial-basis function

RMSE, Root mean squared error

SE,	Squared exponential
STDE,	Standard Deviation Error
SVI,	Stochastic variational inference
SVM,	Support vector machine
SVR,	Support vector machine for regression
Bias_k,	Bias for the kth k-fold
bSize,	Batch size
k-fold CV,	k-fold cross validation
kg,	Kilogram
km,	Kilometre
iid,	Independent identically distributed
pdf,	Probability density function
psd,	Positive semidefinite
std,	Standard deviation
t-SNE,	t-Distributed Stochastic Neighbor Embedding

Nomenclature

$\lvert\boldsymbol{A}\rvert$	Determinant of the matrix **A**
$\bar{\boldsymbol{x}}$	Sample mean of a vector $\boldsymbol{x}$, a scalar
$\boldsymbol{f}$	Random vector of the functions values $f(\boldsymbol{x}_i)$ from training set, $i = 1, ..., n$
$\boldsymbol{f}_*$	Random vector of the functions $f(\boldsymbol{x}_{i,*})$ from test set, $i = 1, ..., n$
$\boldsymbol{I}$	Identity matrix, of size $n \times n$
$\boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_n)$	The $n \times n$ covariance matrix between the matrices $\boldsymbol{X}_n$ and $\boldsymbol{X}_n$
$\boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_*)$	The $n \times n_*$ covariance matrix between the matrices $\boldsymbol{X}_n$ and $\boldsymbol{X}_*$
$\boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*)$	The $n_* \times n_*$ covariance matrix between matrices $\boldsymbol{X}_*$ and $\boldsymbol{X}_*$
$\boldsymbol{K}(\boldsymbol{Z}_m, \boldsymbol{X}_*)$	The $m \times n_*$ covariance matrix between $\boldsymbol{Z}_m$ and $\boldsymbol{X}_*$
$\boldsymbol{K}(\boldsymbol{Z}_m, \boldsymbol{Z}_m)$	The $m \times m$ covariance matrix between the matrices $\boldsymbol{Z}_m$ and $\boldsymbol{Z}_m$ of inducing inputs
$\boldsymbol{K}_{**}$	The $n_* \times n_*$ covariance matrix between the matrices $\boldsymbol{X}_*$ and $\boldsymbol{X}_*$
$\boldsymbol{k}_*$	The covariance vector between the test point $\boldsymbol{x}_*$ and the points of $\boldsymbol{X}$
$\boldsymbol{K}_{m*}$	The $m \times n_*$ covariance matrix between the matrices $\boldsymbol{Z}_m$ and $\boldsymbol{X}_*$
$\boldsymbol{K}_{mm}$	The $m \times m$ covariance matrix between the matrices $\boldsymbol{Z}_m$ and $\boldsymbol{Z}_m$
$\boldsymbol{K}_{n*}$	The $n \times n_*$ covariance matrix between the matrices $\boldsymbol{X}$ and $\boldsymbol{X}_*$
$\boldsymbol{K}_{nn,y}$	The $n \times n$ covariance matrix of noisy observations
$\boldsymbol{K}_{nn}$	The $n \times n$ covariance matrix between the matrices $\boldsymbol{X}$ and $\boldsymbol{X}$

$\boldsymbol{m}$	Mean function of training data, vector
$\boldsymbol{m}_*$	Mean function of test data, vector
$\boldsymbol{u}$	Random vector of the latent functions values $f(z_j)$ from $k = 1, ..., m$
$\boldsymbol{X}$	Input, training matrix of size $n \times p$
$\boldsymbol{X}^T$	Transpose of the matrix $\boldsymbol{X}$
$\boldsymbol{x}^{'}$	Vector of normalized observations
$\boldsymbol{X}_i$	Multivariate random variable
$\boldsymbol{X}_*$	Input, test matrix of size $n \times p$
$\boldsymbol{x}_*$	Test input, vector of a single observation, size $1 \times p$
$\boldsymbol{y}$	Output vector of training input, using noisy observations
Z	Matrix of inducing points of size $m \times p$
$\boldsymbol{\Lambda}$	Matrix of characteristic length scales
$\boldsymbol{\mu}$	Vector of true mean values
Σ	General definition of a covariance matrix
ϵ	Additive iid. Gaussian noise
η^2	Nose variance
λ	Characteristic length scale
λ_d	Characteristic length scale for feature number d
$\mathcal{D}$	Training set of n observations
$\mathcal{N}$	Gaussian process/distribution
$\|\|\cdot\|\|$	The Euclidean distance of $\cdot$
$\overline{C}_i$	Average chlorophyll content for observation i

$\overline{D}_i$	Average depth feature for observation i
ρ	Number of optimized kernel parameters
σ^2	Signal variance
$\sqrt{s}$	Sample standard deviation, a scalar
c	Notation for number of clusters
j	Number of observations in a test set
m	Number of inducing inputs
N	Number of samples in a k-fold training data set
n	The total number of observations considered in this thesis
n_*	Number of samples in test data
p	Number of features/dimension of input data
S	Sample space
s^2	Sample variance, a scalar
t	Time index
Tr	The trace of a matrix
X	Random variable
x	The value of a random variable
y	Output point of training input

1 Introduction

The interest of analysing and finding patterns in large, historical datasets are far from new within the field of statistics. Predicting the future weather or the final score in a game are coveted skills, which have been of interest for a long time. The amount of data that are stored over the years increases, and so does the need for processing, understanding and analysing the data. Computer power and capacity has increased over the last decades, which can explain the increasing attention that the field of machine learning, abbreviated ML, has received in recent years. ML combines statistical knowledge with computer science and is based on algorithms that *learn* from datasets without explicitly being be told how to interpret the data [52].

One of the largest industries of Norway is the Norwegian fishing and aquaculture industry [16], where the profit of today's traditional fishing industry is highly dependent upon the knowledge of individual skippers and fishermen. Better knowledge of the distribution of the fish in the ocean, both quantity and location, is important for an economical and a sustainable fishing industry.

The use of ML in the development of decision support systems, abbreviated DSSs, in fisheries and aquaculture could be a useful tool for the fishing management and the fishing industry to make better decisions [50]. A DDS can be helpful for correctly determine fishing quotas, for decreasing fuel consumption and for fish catch optimization through good and accurate fish catch predictions. This thesis will focus on the possibilities of fish catch prediction through Gaussian processes for regression.

1.1 Former research

In the review by Mathisen et al. [50] stated that there today is little research on DDSs within the fisheries and the aquaculture. The two papers, [1, 37] found in the review, have developed techniques for predicting the presence of fishing banks through data obtained by various remote sensors. Both papers focused on the use of Neural Networks, abbreviated NN, [8, 81], to locate fish banks.

Neither of the data sources selected for this thesis uses remote sensors to gather data which could be helpful in fish catch prediction. This is, to the author's knowledge, the first time that the selected data sources are used for fish catch prediction which motivates the application of "new" prediction methods. Gaussian processes for regression (GPR), see Refs. [53, 65], is a probabilistic inference method that in contrast to both NN and support vector machines, abbreviated SVM, return both a prediction and an uncertainty of the predictions. The properties of Gaussian processes (GP) and the fact that many studies [10, 14, 89] have shown that GPs can provide an excellent accuracy estimation in addition to uncertainty estimates to the predictions, motivates the use of GPR for fish catch prediction.

1.2 Aim of the study

This thesis will contribute to an initial study on the possibilities of using GP for fish catch prediction.

The main contributions of this thesis include:

- A rederivation and formulation of one of the most recent state-of-the-art methods for achieving sparse GP models, i.e. stochastic variational inference (SVI) for GP models. The method, developed by Hensman et al. [32] in 2013 makes it possible to perform inference through GP on a real-time dataset that at least is of size 800,000 data points. Knowledge of the principles of SVI for GP models is necessary as it makes inference through GP possible on the large datasets that will be considered in this thesis.

- A description of the preparatory work needed in the process of identifying, collecting, assimilating and preprocessing real world data, before it initially can be used for regression analysis.

- Initial analysis and investigations in the assimilated dataset through Gaussian processes for regression. This work involves investigations on how the dataset should be modified to achieve more accurate predictions. Two different methods for feature selection, Automatic relevance determination (ARD) and Forward feature selection (FFS) have also been considered in the initial analysis and investigations.

1.3 Structure of the thesis

This thesis is structured in four parts.

Part I describes the relevant background theory needed for the investigations performed in the thesis. Ch. 2 briefly introduce different concepts of machine learning and the fundamentals of Bayes' statistic. The latter is needed for the understanding of the fundamentals of Gaussian processes, GP. Ch. 3 introduces the framework for GP and GP for regression. The principles introduced in this chapter are only suitable for small datasets, with no more than 10,000 observations. A major part of the novelty of the presented work is that this thesis will consider a dataset that is larger than what the framework of Ch. 3 is intended for. For this purpose will Ch. 4 introduce the framework of stochastic variational inference (SVI) for GP. This "new" framework allows the use of GPR for datasets that are much larger than what has been possible before.

In Part II, the assimilated dataset is described and important performance metrics are defined.

Part III contains the investigations of this thesis, where SVI for GP models are applied to the dataset described in Part II. Ch. 7 presents an initial analysis of the input data that investigates different configurations for the input data to increase the prediction accuracy by decreasing the overall Standard Deviation Error (STDE). Ch. 8 presents and compares two different methods for feature selection, the ARD method and the FFS algorithm. Ch. 9 concludes the investigations by performing a performance comparison between the best input data in Ch. 7 and Ch. 8, that yielded the lowest overall STDE with an alternative regression strategy.

Part IV concludes the thesis with a summary of the most important results, a notation over the challenges encountered during the work and finally some recommendations to future work.

1.4 Notation and definitions

In this thesis, scalars will consistently be denoted by normal italic type and matrices in bold, italic, capital letters.
The words *data points* and *observations* will be used interchangeably in the same

context, as they represent observations or measurements of real world data. Data points and observations will consistently be denoted by bold, italic types as each observation/data point will be a multidimensional vector. *Features* and *dimensions* will also be used interchangeably in this thesis. Both definitions indicate the number of parameters in a vector $\boldsymbol{x}$.

Part I

Background theory

2 Fundamentals of Machine learning and Bayesian statistics

2.1 Machine learning, an overview

This section will give an overview of the different disciplines within machine learning, that will be described more deeply in in the forthcoming sections. ML today used in a multitude of different research areas for data analysis, feature selection and data transformation [81]. Figure 2.1 gives an overview of the different categories within ML.

Supervised machine learning is a category where the computer is presented with both the training data and the desired output. The objective of supervised machine learning is for the computer to learn some general "rule" that maps the data from the input space to the output space, following the rule. Supervised learning can be divided into two different kinds of problems, *regression* or *classification* problems [81]. Regression problems consider the prediction of continuous quantities and will be the focus of Gaussian Processes for regression. Classification problems are on the other hand try to assign input data into classes, where the outputs of the classification are discrete class labels [65].

In *unsupervised machine learning*, the computer does not have any information regarding the structure of the data or the desired output. The main goal for the computer, when considering data of this ML category, is to find ways to distinguish between dissimilarities and similarities, i.e. patterns, in the data set. This is performed by different clustering techniques, that cluster similar data into groups [81].

The *data transformation and feature selection* methods are often used during preprocessing to get rid of redundant and uninformative data [39, 81].

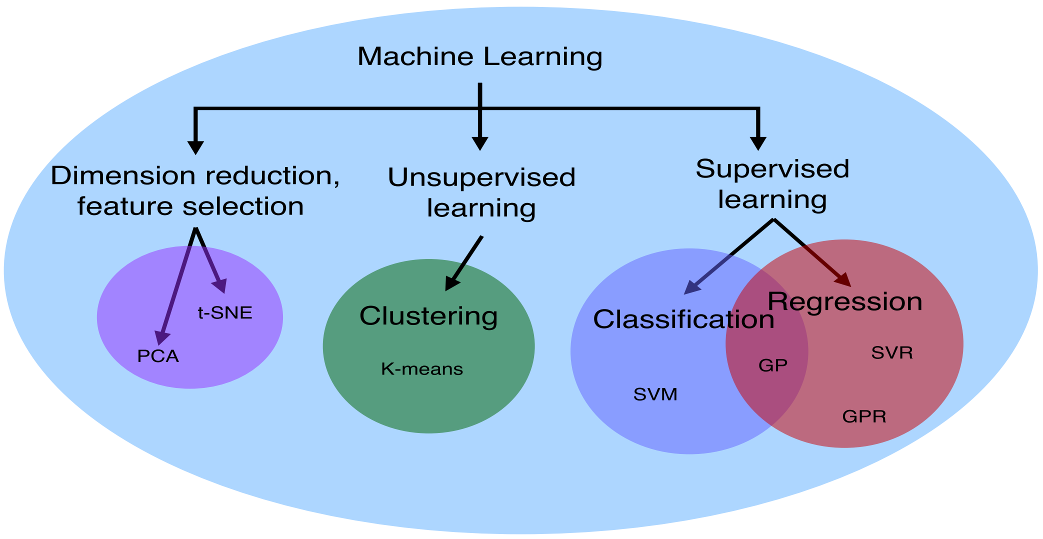


Figure 2.1: An overview of the main categories within ML.

The specific methods that are listed within the different categories in Fig. 2.1 will be introduced in later chapters and sections, and are only shown here to help the reader get an overview of the topics.

Figure 2.2 illustrates the learning procedure of an arbitrary ML model, used for prediction. During training is input data fed to the ML model. The ML model predicts the output, based on the input data. The predicted output is then compared to the actual target/validation data available during training, and a prediction error can be computed. The prediction error will be used as input in the next round of learning/training and will help the ML model to adjust model parameters in order to achieve more accurate predictions by minimizing the error. The optimization/training of the ML model proceeds until the error is constant. The optimal model setup can then be used for prediction on new data, as shown in the figure.

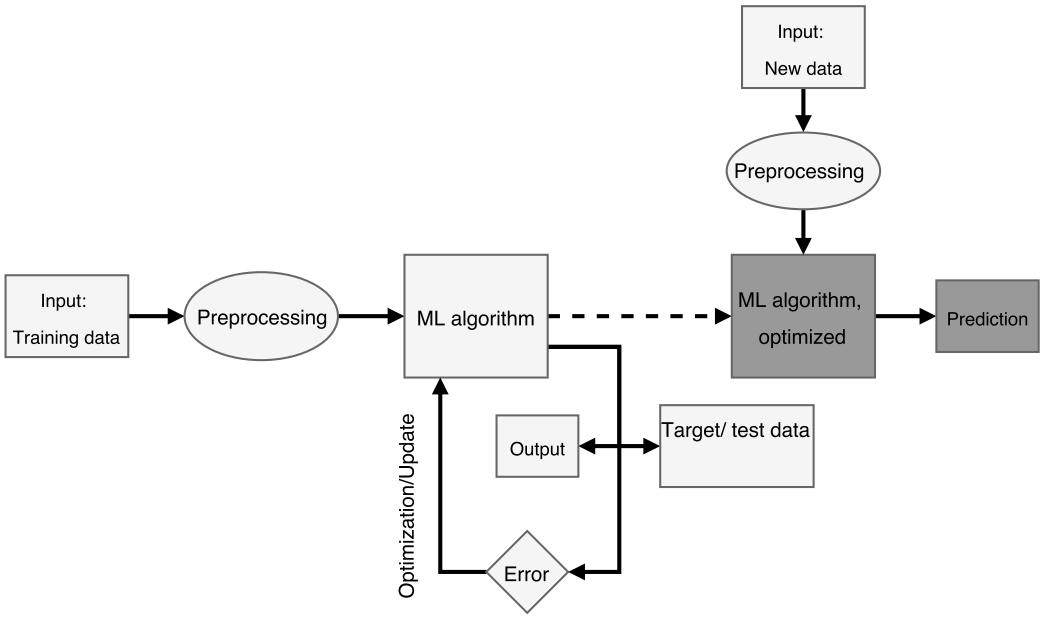


Figure 2.2: Machine learning flowchart. The dashed line indicates the transition to the optimized ML algorithm that can be used for predictions.

2.2 Bayesian statistics

Bayesian statistics introduces another approach to statistical inference than the classical or frequentist approach. The core concepts of Bayesian statistics are that parameters are viewed as random variables, and that Bayes' theorem is the backbone to the methodology [90]. This section will briefly establish and introduce the main concepts to Bayesian statistics, that in Ch. 3 and Ch. 4 are assumed to be background knowledge.

2.2.1 Fundamental Bayesian statistics

The *conditional probability* of event B given event A is given by:

$$P(B|A) = \frac{P(B,A)}{P(A)}, \tag{2.2.1}$$

where $P(B,A)$ denote the *joint probability* of the two events A and B. Applying the conditional probability of Eq. B.0.15 allows for the mathematical formulation of *Bayes' theorem* [81],

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.2.2}$$

The *posterior* distribution of the parameters $\boldsymbol{\theta}$ given some data, $\boldsymbol{x}$, is denoted $p(\boldsymbol{\theta}|\boldsymbol{x})$ and given by Bayes' rule [90],

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}} = \frac{p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{x})}, \tag{2.2.3}$$

where the denominator, i.e. the *marginal likelihood* distribution is independent of $\boldsymbol{\theta}$, because the parameter have been marginalized out. The marginal likelihood can therefore be interpreted as a normalizing constant.
The marginal likelihood in Eq. 2.2.3 for continuous data, $\boldsymbol{x}$ is defined in the following way [65],

$$p(\boldsymbol{x}) = \int_{\theta} p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}. \tag{2.2.4}$$

$p(\boldsymbol{\theta})$ in Eq. 2.2.3 is the prior distribution of the parameters in $\boldsymbol{\theta}$ and reveals prior beliefs of the parameters, before the data is presented [53]. The *likelihood function* is defined as the probability density of the observed data $\boldsymbol{x}$ given the parameters to the observations [65].

3 Gaussian Processes

This chapter will introduce the concepts of Gaussian processes (abbreviated GP), which in later chapters will be used for regression analysis of the potential catch of fish in the oceans. A GP can be interpreted as a generalization of the Gaussian distribution, where the probability distribution governs the properties of functions, instead of variables or vectors [65]. The field of Gaussian processes is divided into two major disciplines; regression and classification, where GP for classification considers the problem of assigning input data into different categories or groups. The interested reader can consult [65] for an introduction to Gaussian processes for classification, as this is beyond the scope of this work.

3.1 Background and introduction to Gaussian processes

The concept of Gaussian processes for regression is far from new and have been studied for a long time. The Wiener process is an early example of a GPR applied within time series [35]. Some background material for Wiener processes and GPR within time series can be found in [92, 93]. Gaussian processes for regression have been used in a more recent setting, where it was introduced to geostatistics during the 1970's under the name "Kriging" [78]. The process within geostatistics, named after the South African mining engineer D.G. Krige, was introduced as a method for predicting the distribution of ore grades and ore reserves through samples from drill holes. Refs. [23, 48, 49] can be consulted for an introduction to the principles of geostatistics and Kriging. The framework of Kriging is identical to the concepts of GPR [46], but is mainly concerned with two or three-dimensional cases. GPR are on the other hand considering a more general, i.e. multidimensional, input space [96].

The use of Gaussian processes within the machine learning community is a more recent development. Neal introduced in 1995 the idea that infinite neural networks converge to a Gaussian process [78]. This discovery inspired Williams and Rasmussen [96] to describe Gaussian processes in a machine learning context [65, 95]. An advantage of using Gaussian processes over neural networks is that the distribution over the parameters in the process can be treated analytically [95]. The book of Rasmussen and Williams, [65], and Chapter 15 in the book by Murphy, [53], can be consulted for good introductions to Gaussian processes for regression.

Gaussian processes, in a ML context, are used to make inferences about the relationship between input data and the output data, without explicitly modeling the whole distribution of the input data. Inferences, that can be drawn from the regression analysis, are based the conditional distribution of the target values given the input data [65].

The GPR is divided into two parts, *regression* and *prediction*. The first, regression part is considered with *revealing* the relationship between the input data and the output target values. This part has similarities with *supervised learning* within machine learning where both training and target data are available.
The second part of GPR considers prediction, where the trained process can be applied to new, unseen input data, to perform predictions of future, unseen target values.

An introduction to the theory of Gaussian processes for regression will be given in the forthcoming section, where the statistical background for Gaussian processes will be established. This will in later sections be used to formulate the function that connects the input variables with the output variables, which can be used for prediction.

3.2 Gaussian processes for regression, a theoretical review

This section will give an introduction to the framework of Gaussian processes for regression. Only real-valued data will be considered as input and output of the GP because this is the focus of this thesis. The theory presented in this section is closely related to Chapter 15, Murphy [53], and the book; *Gaussian Processes for*

Machine Learning, Rasmussen [65]. The same notations will therefore be used, when possible.

3.2.1 GP for regression

It will from now on be assumed that a training set is available, consisting of both input training data $\boldsymbol{x}$ and target (output) data y. The training set will be denoted $\mathcal{D}$, where $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1, ..., n\}$. It will initially be assumed that the input training data, $\boldsymbol{x}$, is noise-free. Let $\boldsymbol{x}_i = \{x_{i,1}, x_{i,2}, ..., x_{i,p}\}$ denote the ith multidimensional input point i.e. an observation/realization from the p-dimensional input space. The index $i = 1, ..., n$ can either be interpreted as a time variable, denoting when $\boldsymbol{x}_i$ is observed, an index defining the number of an observation, or a combination of an unique time and observation. All input training data can be combined into a matrix of multidimensional observations, it will be denoted X.

Similarly will y_i denote the target value in the output space, and $\boldsymbol{y} = \{y_1, ..., y_n\}$ will from now on denote a vector of target (output) values in the output space.

The GPR assume that the target values y_i are related the input data $\boldsymbol{x}_i$ through a function such that $y_i = f(\boldsymbol{x}_i)$. Thus, y_i represents a realization of the Gaussian process, $f(\boldsymbol{x}_i)$, at location $\boldsymbol{x}_i$. Figure 3.1 illustrates the relation ship between the multidimensional input observations and the output targets through the Gaussian process, $f(\boldsymbol{x}_i)$.

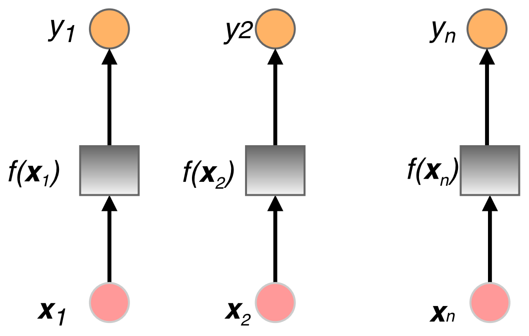


Figure 3.1: Visualization of the relationship between the input and output data through the GP.

From now on, let $\boldsymbol{f} = [f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_n)]^T$ denote the vector of all $f(\boldsymbol{x}_i)$. To make predictions of new, unseen observations, a distribution over the functions $f(\boldsymbol{x}_i)$, given the data $\mathcal{D}$ need to be defined. The approach presented here will be based on

Gaussian processes, which defines a prior over the functions, that can be converted to a posterior given $\mathcal{D}$ [53]. Using the approach of GP, the distribution over the function values, $f(\boldsymbol{x}_i)$, does not have to be explicitly defied. It will instead be enough to define the distribution over the function's values at a finite, random sets of points $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$. Defining the the prior of the functions $f(\boldsymbol{x}_i)$ to be a GP will imply that probability distribution, $p\big(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_n)\big)$ will be jointly Gaussian distribution [53, 65].

As a reminder,the general, multivariate Gaussian distribution of a p-dimensional random vector $\boldsymbol{x}$ is defined by,

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}, \tag{3.2.1}$$

where Σ is the $p \times p$ variance-covariance matrix of $\boldsymbol{x}$, $|.|$ denotes the determinant and $\boldsymbol{\mu}$ denote the $p \times 1$ vector of expectations for the random vector $\boldsymbol{x}$ [39]. Appendix A gives an example of how Eq. 3.2.1 can be used to compute a bivariate Gaussian distribution. The example in App. A can then be extended to the case of the multivariate Gauassian distribution.

An advantage of using Gaussian processes is that they are completely specified by its mean function, $\boldsymbol{m}$ and covariance function, $\boldsymbol{K}_{nn}$, in the same manner as multivariate Gaussian distributions are specified by its mean vector and covariance matrix [46, 58]. Using the approach of GP allows the prior of the functions $\boldsymbol{f} = [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), ..., f(\boldsymbol{x}_n)]^T$ to be defined in the following way,

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_n) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}_{nn}),$$

which can be simplified to

$$\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}_{nn}), \tag{3.2.2}$$

where $\boldsymbol{m}$ is a vector of expectations,

$$\boldsymbol{m} = \begin{bmatrix} E[f(\boldsymbol{x}_1)] \\ E[f(\boldsymbol{x}_2)] \\ \vdots \\ E[f(\boldsymbol{x}_n] \end{bmatrix},$$

and $\boldsymbol{K}$ is the matrix of covariance functions between vectors of the dataset in $\mathcal{D}$, formulated as,

$$\mathbf{K}_{nn} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_2, \boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}.$$

The covariance function is assumed to be a function of the input data,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \text{cov}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) = k(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{3.2.3}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ denotes two different vectors of size $1 \times p$.

Both the covariance function and the mean function need to be chosen in some manner depending on the data. It is commonly to chose $\boldsymbol{m} = \mathbf{0}$, because the Gaussian process should be flexible enough to model the mean in a reasonable way [53]. Thus, Eq. 3.2.2 can then be redefined to:

$$\boldsymbol{f} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_{nn}) \tag{3.2.4}$$

Further information about properties of covariance functions etc. can be found in Sec. 3.2.3.

In the following will it be assumed that an arbitrary covariance function, denoted $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is available in addition to a zero mean function be assumed. It will also be assumed that *new* observations are available, in addition to the training set. These will be gathered into a matrix of test observations, denoted $\boldsymbol{X}_*$. Gaussian processes will be used to predict the outputs, $\boldsymbol{f}_*$, of the variables in $\boldsymbol{X}_*$ with help of both $\boldsymbol{f}$ and the training set, where $\boldsymbol{f}_* = [f(\boldsymbol{x}_{1,*}), ..., f(\boldsymbol{x}_{n,*})]$. By the definition of GP, the joint distribution of $\boldsymbol{f}$ and $\boldsymbol{f}_*$ will be Gaussian distributed, and will have the following form,

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_n) & \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_*) \\ \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_n) & \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right). \tag{3.2.5}$$

Henceforth, the notation will be simplified in the following way,

- $\boldsymbol{K}_{nn} = \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_n)$, the $n \times n$ covariance matrix between $\boldsymbol{X}_n$ and $\boldsymbol{X}_n$
- $\boldsymbol{K}_{n*} = \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_*)$, the $n \times n_*$ covariance matrix between $\boldsymbol{X}_n$ and $\boldsymbol{X}_*$
- $\boldsymbol{K}_{**} = \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*)$, the $n_* \times n_*$ covariance matrix between $\boldsymbol{X}_*$ and $\boldsymbol{X}_*$,

where n denote the number of observations in $\boldsymbol{X}$, while n_* denote the number of observations in $\boldsymbol{X}_*$. The covariance matrix is symmetric,[1] which implies that $\boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_*) = \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_*)^T$, where T denotes the transpose of a matrix.

The primary goal for GPR is to predict the function values of $\boldsymbol{f}_*$ for the new test set input $\boldsymbol{X}_*$. It turns out that the conditional distribution of $\boldsymbol{f}_*$ given $\boldsymbol{f}$ can be computed since both the function values of $\boldsymbol{f}$ and $\boldsymbol{X}_*$ are known. This can be expressed

1. See Sec. 3.2.3

as [64, 65],

$$\boldsymbol{f}_*|\boldsymbol{X}_*, \boldsymbol{X}_n, \boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{K}_{n*}^T \boldsymbol{K}_{nn}^{-1} \boldsymbol{f}, \boldsymbol{K}_{**} - \boldsymbol{K}_{n*}^T \boldsymbol{K}_{nn}^{-1} \boldsymbol{K}_{n*}\right), \tag{3.2.6}$$

where it is assumed that $\boldsymbol{m} = \boldsymbol{m}_* = \boldsymbol{0}$. $\boldsymbol{m}$ is the mean function of $\boldsymbol{f}$ while $\boldsymbol{m}_*$ is the mean function of $\boldsymbol{f}_*$. In the case when the mean functions are non-zero, Eq. 3.2.6 would look like:

$$\boldsymbol{f}_*|\boldsymbol{X}_*, \boldsymbol{X}_n, \boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{m}_* + \boldsymbol{K}_{n*}^T \boldsymbol{K}_{nn}^{-1} (\boldsymbol{f} - \boldsymbol{m}), \boldsymbol{K}_{**} - \boldsymbol{K}_{n*}^T \boldsymbol{K}_{nn}^{-1} \boldsymbol{K}_{n*}\right). \tag{3.2.7}$$

The expressions in Eq. 3.2.6 and Eq. 3.2.7 are well known results for jointly Gaussian distributions. The full result, and a proof for the bivariate case can be found in the Appendix B, as it is easier to compute and show than the multivariate case.

3.2.2 Noisy data

So far, it has been assumed that the observations in $\boldsymbol{X}_n$ and $\boldsymbol{X}_*$ are noise-free. In general, data collected from measurements are often corrupted by noise due to different kinds of measurements errors, or errors that occur during storage and/or processing of the data [28]. This section will consider situations that are more realistic, for work with real world datasets. It will from now on be assumed that the observed target values are given by the function values at some point, in addition to some additive noise. The most common assumption for the noise is the additive independent identically distributed (abbreviated iid) Gaussian noise, denoted ϵ [64]. The interested reader can consult Neal [57] for an example of a Student's t-distributed noise model.

Adopting the assumption of iid Gaussian noise, the observed target values will from now on be expressed in a slightly different way according to [65],

$$y_i = f(\boldsymbol{x}_i) + \epsilon,$$

where $\epsilon \sim \mathcal{N}(0, \eta^2)$ and η^2 denotes the noise variance that is by definition constant for all vectors of $\boldsymbol{X}$. Reformulating the general notation of the covariance function in Eq. 3.2.3 for noisy observations gives

$$\operatorname{cov}(y_i, y_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \eta^2 \delta_{ij}, \tag{3.2.8}$$

where $\delta_{i,j}$ denotes the Kronecker delta function,

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise .} \end{cases} \tag{3.2.9}$$

Using matrix notation for Eq. 3.2.8 result in,

$$\operatorname{cov}(\boldsymbol{y}) = \boldsymbol{K}_{nn} + \eta^2 \boldsymbol{I} = \boldsymbol{K}_{nn,y}, \tag{3.2.10}$$

where the additive noise is added to each observation of $\boldsymbol{X}$, via the $n \times n$ identity matrix $\boldsymbol{I}$, [53]. $\boldsymbol{K}_{nn,y}$ is a simplified notation for the covariance matrix with additive noise.

The joint density in Eq. 3.2.5 can now be reformulated with help of Eq. 3.2.10 under the assumption of additive iid Gaussian noise,

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K}_{nn,y} & \boldsymbol{K}_{n*} \\ \boldsymbol{K}_{n*}^T & \boldsymbol{K}_{**} \end{bmatrix}\right), \tag{3.2.11}$$

still assuming a that $\boldsymbol{m} = \boldsymbol{0}$. The conditional distribution for the noisy observations, corresponding to Eq. 3.2.6, can eventually be formulated, following [65] and [53],

$$\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}_n, \boldsymbol{y} \sim \mathcal{N}\Big(E[\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}_n, \boldsymbol{y}], \operatorname{cov}(\boldsymbol{f}_*)\Big), \tag{3.2.12}$$

where

$$\bar{\boldsymbol{f}}_* = E[\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y}] = \boldsymbol{K}_{n*}^T \boldsymbol{K}_{nn,y}^{-1} \boldsymbol{y}, \tag{3.2.13}$$

and

$$\operatorname{cov}(\boldsymbol{f}_*) = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}_{nn,y}^{-1} \boldsymbol{K}_*. \tag{3.2.14}$$

The two equations, Eq. 3.2.13 and Eq. 3.2.14 are essential for Gaussian process regression, and will be referred to as *the predictive equations for Gaussian process regression* [65].

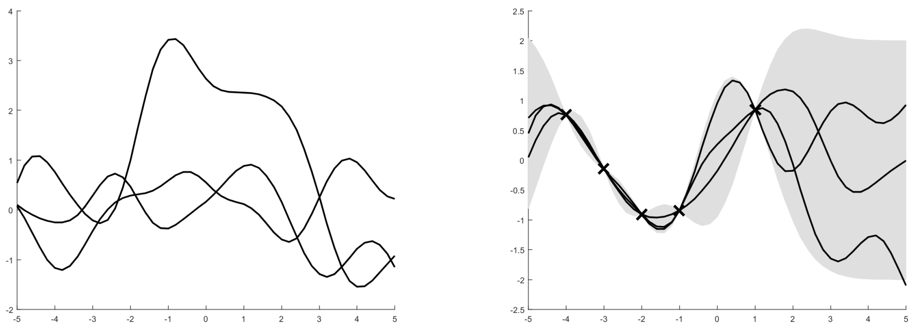
The prediction process is illustrated in Fig. 3.2, where the left panel shows three arbitrary samples from a GP prior, with a SE covariance function. The right panel of Fig. 3.2 shows three samples from a GP posterior, after conditioning on five noise-free observations. The shaded region in the right panel corresponds to the 95% confidence region [53, 65].

The predictive equations in Eq. 3.2.13 and Eq. 3.2.14 can be simplified when the test set only contains a single input, $\boldsymbol{x}_*$. In this case will $\boldsymbol{k}_*$ be a vector of the covariances between the test input and the training points of $\boldsymbol{X}$, $\boldsymbol{k}_* = [k(\boldsymbol{x}_*, \boldsymbol{x}_1), ..., k(\boldsymbol{x}_*, \boldsymbol{x}_n)]$. The covariance between of the test point, and it self, will be defined: $\boldsymbol{k}_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$. Introducing this compact notation for the single input $\boldsymbol{x}_*$ gives the *simplified predictive equations* [53, 65],

$$\bar{\boldsymbol{f}}_* = E[\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}_n, \boldsymbol{y}] = \boldsymbol{k}_*^T \boldsymbol{K}_{nn,y}^{-1} \boldsymbol{y}, \tag{3.2.15}$$

$$\operatorname{cov}(\boldsymbol{f}_*) = \boldsymbol{k}(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \boldsymbol{K}_{nn,y}^{-1} \boldsymbol{k}_*, \tag{3.2.16}$$

with $\boldsymbol{K}_{nn,y} = \boldsymbol{K}_{nn} + \eta^2 \boldsymbol{I}$.



(a) **(b)**

Figure 3.2: (a) Visualization of three functions sampled from a GP prior with a SE covariance function (see Sec. 3.2.3). (b) Visualization of three samples from a GP posterior, after conditioning on five noise-free observations. The figures are generated by `gprDemoNoiseFree` written by [65]

It can in some cases be useful to consider GP with non-zero mean functions. In these cases can Murphy [53] and Rasmussen [65] be consulted for some examples of non-zero mean functions.

3.2.3 Covariance functions

When no other references are mentioned, this section is based on [65]. As stated in Sec. 3.2.1, the Gaussian process is completely specified by its mean and covariance functions. The choice of the covariance function will play an essential part of the Gaussian process for regression and prediction, as the both predictive equations Eq. 3.2.13 and Eq. 3.2.13 (and the simplified equations Eq. 3.2.15 and Eq. 3.2.16) depend upon the covariance function.

A basic assumption for predictions is that similar inputs $\boldsymbol{x}$, should have similar output values, $\boldsymbol{y}$. Hence, training input points that are located close to a test input point should be more informative in the prediction, than training input points located further away. Assuming a GP, the covariance function should reflect this closeness.

Furthermore, valid covariance functions need to generate a positive semidefinite (abbreviated psd) covariance matrix for any points in space [46]. A symmetric and real covariance matrix $\boldsymbol{K}$ is psd if it satisfies

$$0 \leq \boldsymbol{x}^T \boldsymbol{K}_{nn} \boldsymbol{x}, \tag{3.2.17}$$

for all vectors $\boldsymbol{x}^T = [x_1, x_2, ..., x_n]$. Furthermore, a symmetric matrix $\boldsymbol{K}_{nn}$ is positive semidefinite if and only if its eigenvalues are greater than or equal to zero [39]. It can be difficult to come up with covariance functions that fullfills the requirement of generating a psd matrix. Fortunately, there are several families of covariance functions that fullfills this requirement and from which a suitable covariance function can be chosen. A well known class of positive semidefinite covariance functions are the stationary and isotropic covariance functions [95]. A stationary covariance is a function of $\boldsymbol{x}_i - \boldsymbol{x}_j$, whilst an isotropic covariance function is independent of the direction and only a function of $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$, where $\|.\|$ denotes the Euclidean distance.

It should be noted that, in the ML context, the covariance function is often referred to as a *kernel function* [26]. This thesis will use the two expressions interchangeably.

The squared exponential covariance function

One of the most popular choices of the covariance function is the squared exponential (SE), also denoted the radial-basis function (RBF) [26].

The SE covariance function is symmetric, isotropic, stationary and positive semidefinite, in addition to be infinitely differentiable and thus very smooth. The squared exponential covariance function is defined here with Gaussian noise added to the model:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \cdot \exp\left[-\frac{1}{2} \sum_{d=1}^{p} \frac{(x_{di} - x_{dj})^2}{\lambda_d^2} \right] + \eta^2 \delta_{i,j}, \tag{3.2.18}$$

where i, j denotes two different vectors of observations, and $d = 1, \ldots, p$ denotes the dimension of the input data. σ, λ and η are the hyperparameters of the squared exponential covariance function.

- λ_d is the *characteristic length scale* and it is is a horizontal scale, over which the function changes.
- σ^2 is the *signal variance*, and controls the vertical scale of the
- η^2 is the noise variance.

The characteristic length scale can either be different for each dimension of $\boldsymbol{x}$, this is indicated by its subscript d, or constant along all dimensions of $\boldsymbol{x}$. The characteristic length scale will in the latter case be denoted by λ. The SE covariance function can in the latter case be expressed in the following, simplified way,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \cdot \exp\left[-\frac{1}{2\lambda^2} \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2 \right] + \eta^2 \delta_{i,j},$$

where $\left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|^2$ denote the squared Euclidean distance between the two vectors $\boldsymbol{x}_i$

and $\boldsymbol{x}_j$. The SE function in Eq. 3.2.18 can be rewritten in a more compact form using matrix notation for two arbitrary vectors $\boldsymbol{x}_i, \boldsymbol{x}_j$,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \cdot \exp\left[-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_i - \boldsymbol{x}_j)\right] + \eta^2 \delta_{i,j}, \tag{3.2.19}$$

where the matrix $\boldsymbol{\Lambda}$ can be defined in several ways. Using this matrix: $\boldsymbol{\Lambda}_1 = \lambda^{-2}\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix gives an isotropic matrix, while $\boldsymbol{\Lambda}_2 = \mathrm{diag}(\boldsymbol{\lambda})^{-2}\boldsymbol{I}$ with $\boldsymbol{\lambda} = \lambda_1, ..., \lambda_p$ generates an matrix with different length scale along all dimensions [53]. Refs. [3, 46, 53, 65].

3.2.4 The importance of choosing the hyperparameters correctly

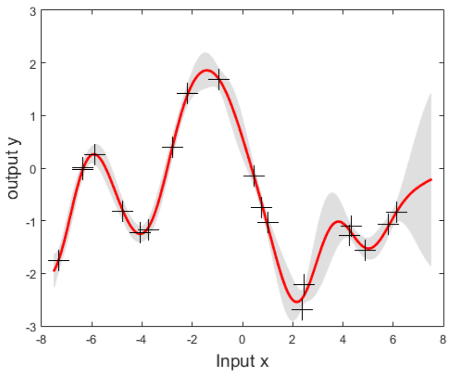
Sec. 3.2.3 introduced the hyperparameters of the SE covariance function, which have to be determined in some way to perform GP prediction. Fig. 3.3 and Fig. 3.4 will be used to examine the importance of choosing the hyperparameters correctly. The upper left panel in the both figures, Fig. 3.3 and Fig. 3.4 represents a reference figure. The reference figures were made by first sampling 20 noisy observations, marked with + signs, from a SE kernel, with the following hyperparameters; (λ, σ, η) = (1,1,0.1). Predictions were then made, conditioned on the data and recycling the same hyperparameters, i.e. (λ, σ, η) = (1,1,0.1). This results in a narrow confidence region which indicates a good fit, as can be seen in the both reference figures [53, 65].

The red curve indicates the underlying function while the shaded area in the both figures represents the 95%-confidence region, which for a Gaussian distribution mathematically can be expressed by,

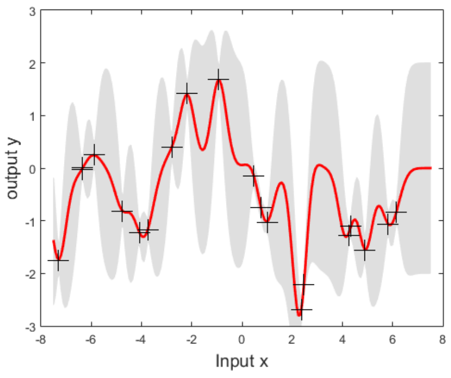
$$\bar{\boldsymbol{f}}_* \pm 2 \cdot \sqrt{\mathrm{cov}(\boldsymbol{f}_*)}. \tag{3.2.20}$$

The reference figures in Fig. 3.3a and Fig. 3.4a shows that predictions in the areas where the observations are close will yield better predictions, than in the areas where the observations are far away, This can be interpreted from the confidence region, which is narrow in areas with many observations. The opposite is true in areas with few observations, which can be seen close to $x = 2$. The importance of choosing the hyperparameters correctly will be shown in Fig. 3.3 and Fig. 3.4, by keeping two of the hyper parameters constant while the third will be changed.

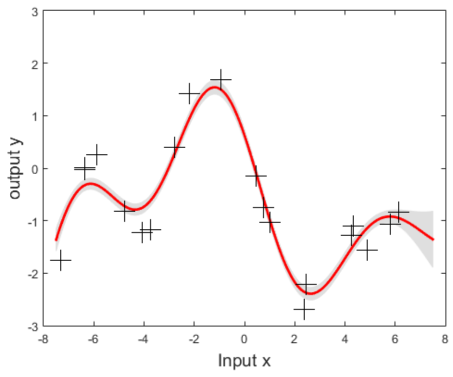
Fig. 3.3b shows the result for $\lambda = 0.3$ while Fig. 3.3c shows the result for $\lambda = 3.0$. Both σ and η are held constant at the same parameter values as in Fig. 3.3a. Decreasing the characteristic length scale results in a more wiggly function, with a confidence region that has increased rapidly, compared to Fig. 3.3a. This indicates that defining λ too small could increase the uncertainty in the predictions. On the other hand, increasing the length scale to $\lambda = 3.0$ results in a slowly varying function with a narrow confidence region, as in Fig. 3.3c. It could be noted that the the function in Fig. 3.3c does not fit all observations very well, which could result in misleading predictions.



(a)



(b)



(c)

Figure 3.3: The three figures in (a), (b) and (c) shows 20 noisy observations, marked with + sign, generated from a 1D GP defined through a SE covariance function. The shaded grey regions indicate the 95% confidence region that was achieved from the GPR. The values of the hyperparameters (λ, σ, η) in panel (a) are (1,1,0.1). The hyperparameters of (b) are set to (0.3,1,0.1) while the hyperparameters of panel (c) are set to (3.0, 1, 0.1). The figures are based on Figure 2.5 in [65], and generated by the functiongprDemoChangeHparams, written by Carl Rasmussen [65].

Turning the focus to Fig. 3.4; Fig. 3.4b shows the result of keeping the length scale and the noise variance constant at the same parameter values, as in , while σ is set to 0.1. Fig. 3.4c shows the case when the signal variance was set to $\sigma = 1.16$, while the other parameters are kept constant. The latter result, Fig. 3.4c, is similar to the reference figure (Fig. 3.4a), but there are some places where the accuracy of the predictions decreases. This can be seen around $x = 2$ and $x = 4$, which resulted in larger peaks for the confidence region. A decrease in σ, as in Fig. 3.4b, shows an almost non-existing confidence region, compared to Fig. 3.4a, concurrently as the function shows a poor fit to the observations.

Fig. 3.4d shows the result for $(\lambda, \sigma, \eta) = (1{,}1{,}0.00005)$, where the predictions function overfit the observations. The last figure, Fig. 3.4e shows the result for $(\lambda, \sigma, \eta) = (1{,}1{,}0.89)$ which result in a slowly varying function with large error bars for all observations.

The figures in Fig. 3.3 and Fig. 3.4 reveals the problem of determining the hyperparameters properly. The decision of the hyperparameters of the squared exponential covariance function will be even more complicated when the characteristic length scale is allowed to change with each input dimension. This will result in a huge variety of possible hyperparameters and distance measures, and it would be to time consuming to change the hyperparameters manually [65]. The following section will consider the the optimization of the hyperparameters through the marginal likelihood of the GP model, which is a non-manual way of optimizing the hyperparameters.

3.2.5 Hyperparameter estimation and optimization

For this section, let $\boldsymbol{\theta} = \{\lambda, \sigma, \eta\}$ denote the vector of the hyperparameters of the SE kernel function. The characteristic length scale, λ, in $\boldsymbol{\theta}$ will either be constant for all dimensions, or a separate parameter for each of the different dimensions of $\boldsymbol{x}$ (see Sec. 3.2.3).

One of the major advantages with GPs is that the hyperparameters of the covariance function can be chosen directly from the training data, instead of using more complicated methods [78]. Ideally, the optimal hyperparameters of $\boldsymbol{\theta} = \{\lambda, \sigma, \eta\}$, given a set of observations, would in a Bayesian setting be computed by the posterior distribution over the hyperparameters, see Sec. 2.2.3. The posterior distribution over the hyperparameters given the data, denoted $p(\boldsymbol{\theta}|\boldsymbol{y}, X_n)$, can be expressed in the following way [29],

$$p(\boldsymbol{\theta}|\boldsymbol{y}, X_n) = \frac{p(\boldsymbol{y}|X_n, \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{y}|X_n)}, \tag{3.2.21}$$

where $\boldsymbol{X}_n$ denote the training set and $\boldsymbol{y}$ denote the noisy observed target values. The denominator, in Eq. 3.2.21 will be independent of $\boldsymbol{\theta}$, and can therefore be treated as a normalization constant. The normalizing constant will often be ignored , see Sec. 2.2,

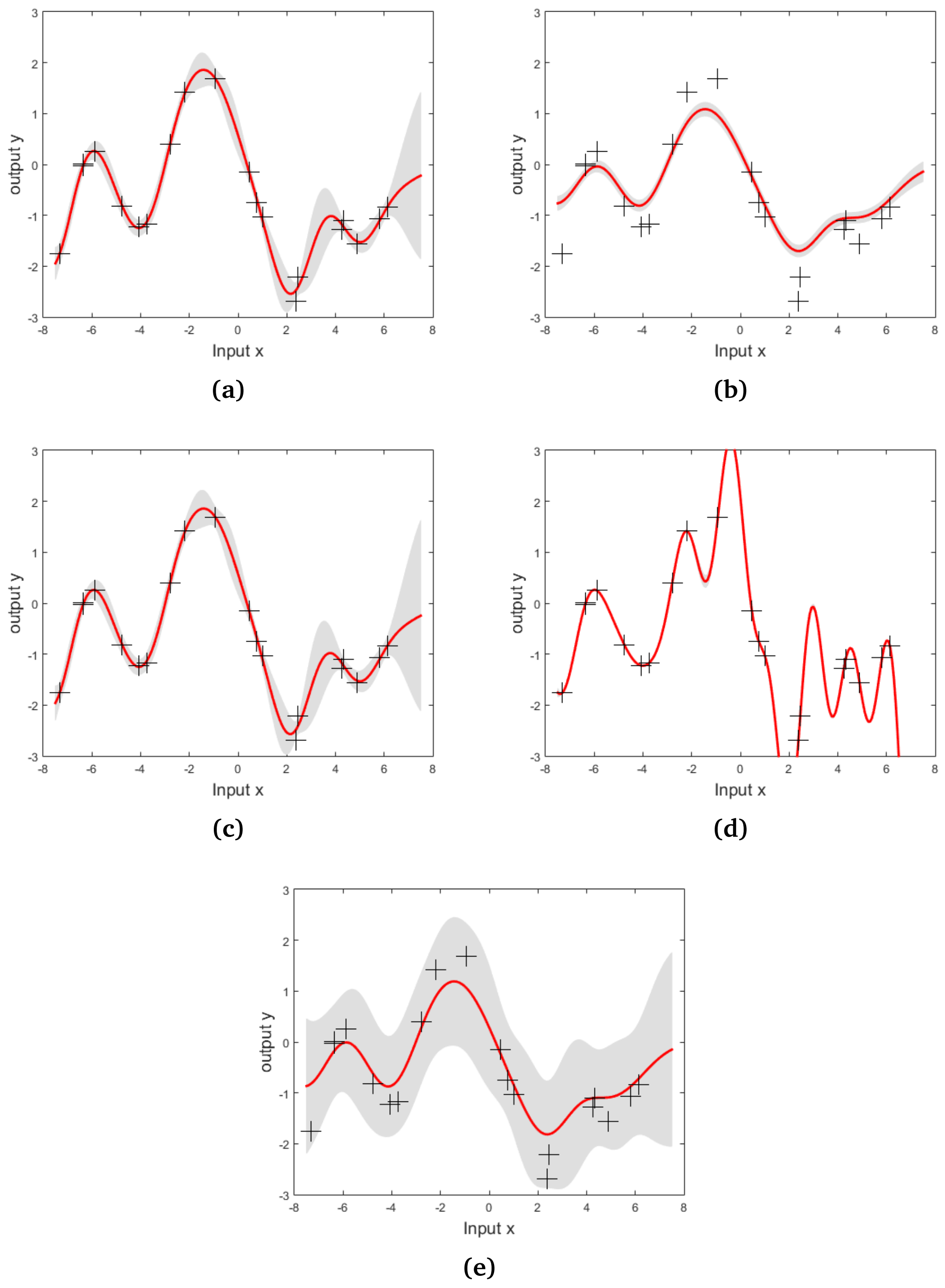


Figure 3.4: The five figures in (a)-(e) shows 20 noisy observations, marked with + sign, generated from a 1D GP with a SE covariance function. The shaded grey regions indicate the 95% confidence region that was achieved from the GPR. The values of the hyperparameters (λ, σ, η) in panel (a) are (1,1,0.1). The hyperparameters of (b) are set to (1,0.1,0.1), the hyperparameters of panel (c) are (1, 1.16, 0.1), the hyperparameters of panel (d) are (1, 1, 0.00005) while the hyperparameters of panel (e) are (1, 1, 0.89) . The figures are based on Figure 2.5 in [65], and generated by the functiongprDemoChangeHparams, written by Carl Rasmussen [65].

which gives the following simplified expression for $p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}_n)$,

$$p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}_n) \propto p(\boldsymbol{y}|\boldsymbol{X}_n, \boldsymbol{\theta})p(\boldsymbol{\theta}). \tag{3.2.22}$$

Thus, the posterior of the hyperparameters given the data are proportional to the *marginal likelihood function*, $p(\boldsymbol{y}|\boldsymbol{X}_n, \boldsymbol{\theta})$, times the prior distribution for $\boldsymbol{\theta}$ [53]. The prior distribution $p(\boldsymbol{\theta})$ should encode prior beliefs about the hyperparameters, before the data is presented [90]. Ref. [12] points out that the optimal hyperparameters could be found from Eq. 3.2.22 given knowledge about the prior density. The optimal hyper parameters will be those that maximizes the posterior density. This will be called the *maximum a posterior* (MAP) estimates, denoted $\boldsymbol{\theta}_{MAP}$, which can be fed into Eq. 3.2.13 and Eq. 3.2.14 to make predictions and calculate the uncertainty of the predictions.

A common practise to ignore the prior term in Eq. 3.2.22, when for example $p(\boldsymbol{\theta})$ is unknown [29]. The posterior distribution can then be defined in the following, simplified form,

$$p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}_n) \propto p(\boldsymbol{y}|\boldsymbol{X}_n, \boldsymbol{\theta}), \tag{3.2.23}$$

which implies that the maximum a posterior estimates of $\boldsymbol{\theta} = \{\lambda, \sigma, \eta\}$ can be computed by maximizing the marginal likelihood function of the hyperparameters. The marginal likelihood is a GP, and will by definition follow the distribution of a multivariate Gaussian distribution:

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{nn,y}), \tag{3.2.24}$$

where $\boldsymbol{K}_{nn,y} = \boldsymbol{K}_{nn} + \eta^2 \boldsymbol{I}$ is the covariance matrix for the noisy targets $\boldsymbol{y}$. For numerical reasons, the *log marginal likelihood* will be considered [12],

$$\begin{aligned} l = \log\big(p(\boldsymbol{y}|\boldsymbol{X}_n, \boldsymbol{\theta})\big) &= \frac{1}{(2\pi)^{n/2}|\boldsymbol{K}_{nn,y}|} \exp\left(-\frac{1}{2}\big(\boldsymbol{y}^T \boldsymbol{K}_{nn,y}^{-1} \boldsymbol{y}\big)\right) \\ &= -\frac{1}{2}\log\big(|\boldsymbol{K}_{nn,y}|\big) - \frac{1}{2}\boldsymbol{y}^T \boldsymbol{K}_{nn,y}^{-1}\boldsymbol{y} - \frac{n}{2}\log(2\pi). \end{aligned} \tag{3.2.25}$$

The first term of Eq. 3.2.25 is a complexity and penalty term, which measures and penalizes the complexity of the model [64]. The second term, $-\frac{1}{2}\boldsymbol{y}^T \boldsymbol{K}_{nn,y}^{-1}\boldsymbol{y}$, is the only term that involves the observed target values and thus measures the fit of the data. The last term of Eq. 3.2.25 is independent of both $\boldsymbol{X}$ and $\boldsymbol{y}$ and thus a normalization term. It can be shown, [95], that the partial derivatives of the log marginal likelihood can be expressed with respect to the hyperparameters in the following way,

$$\frac{\partial l}{\partial \theta_i} = -\frac{1}{2}\mathrm{Tr}\left(\boldsymbol{K}_{nn,y}^{-1}\frac{\partial \boldsymbol{K}_{nn,y}}{\partial \theta_i}\right) + \frac{1}{2}\boldsymbol{y}^T \boldsymbol{K}_{nn,y}^{-1}\frac{\partial \boldsymbol{K}_{nn,y}}{\partial \theta}\boldsymbol{K}_{nn,y}^{-1}\boldsymbol{y} = 0 \tag{3.2.26}$$

Thus, the hyperparameters of the squared exponential covariance function in Eq. 3.2.18 can be found by solving Eq. 3.2.26 with respect to the hyperparameters. It should be noted that the likelihood generally have multiple local optima, which could result in some *bad* solutions for the optimal hyperparameters. This problem could be avoided by using sensible priors of the hyperparameters, and calculating the MAP-estimates instead of the maximum likelihood estimates [29].

The interested reader can consult Alg. 2.1 in [65] and Alg. 15.1 in [53] for a pseudo-code of the implementation of GPR.

4

Applying GP for inference on large datasets

Once the kernel parameters are optimized, the framework for GP for inference is both elegant and simple, as the prediction of new unseen points is completely defined trough Eq. 3.2.13 and Eq. 3.2.14. The latter equation gives the uncertainty of these predictions. Predictions achieved through GP are based on a well-established foundation through the multivariate Gaussian distribution. This ensures that new predictions can be completely defined given new input data, previous input and output data. Despite this, the framework GP for regression suffers from some crucial limitations. The complexity for applying inference by GP on a dataset of size n is of $\mathcal{O}(n^3)$, which occurs during the computations of the inverse covariance matrix $\boldsymbol{K}_{nn,y}^{-1}$ [53]. Furthermore, the storage demands are of $\mathcal{O}(n^2)$ [83].
These complexities could, unfortunately, restrict the use of GPR to datasets where $n < 10,000$ observations. In order to apply GP for inference on larger datasets many different sparse and approximation methods for Gaussian processes have been suggested, where Refs. [34, 65, 72, 75, 77] can be consulted for some examples. Many of these approximation and sparse methods are based on a small set of m *inducing variables* instead of using the whole dataset of n observations. This will allow a reduction of the computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(nm^2)$, and a reduction of storage demands to $\mathcal{O}(nm)$ [32]. The many different sparse and approximate methods mentioned mainly differ in how the inducing inputs are selected [82]. Common for many of these approaches are that they are still not suitable for datasets where the

number of observations can be many millions or billions [32].

This chapter focus on a newly developed algorithm that is called stochastic variational inference, SVI, for Gaussian process models. The algorithm combines inducing variables with SVI to fit the GP to the data [32]. The method presented in [32] was published in 2013 and is one of the most recent methods for achieving sparse GP models. Hensman et al. [32] proved that their method works well for inference on a real-time dataset of size 800,000 data points, and state that SVI for GP also could be applied to datasets containing millions of data points. This method has been chosen among others both because it is a very recent method, and because it has been demonstrated to work well on large datasets.

An understanding of how the method of SVI for Gaussian processes works will be established in the following sections. The theory presented will follow the structure in [32] closely, and will therefore start by introducing the main ideas behind sparse GPs, and how the inducing variables can be derived using variational inference and learning. This introduction will closely follow [82] and [83], where the technical report in [83] has more details than [82]. This chapter will then proceed by briefly introducing the main parts of SVI for GPs.

4.1 Sparse GP and variational learning

4.1.1 Aim of sparse GP methods

Sec. 3.2.5 stated that the estimation and optimization of the hyperparameters, $\boldsymbol{\theta}$, of the GP model is crucial for achieving accurate and good predictions. It was also stated that the optimal hyperparameters could be estimated by maximizing the log marginal likelihood:

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{0}, \eta^2 \boldsymbol{I} + \boldsymbol{K}_{nn}), \tag{4.1.1}$$

which involves the computation of the $\boldsymbol{K}_{nn,y}^{-1}$ matrix, where $\boldsymbol{K}_{nn,y} = \eta^2 \boldsymbol{I} + \boldsymbol{K}_{nn}$.

The aim of sparse GP methods is to perform matrix operations, such as inversion and diagonalisations, in the most time efficient way [21]. Instead of using all n observations, approximate or sparse methods for GP will only consider a small set of m inducing inputs, denoted $\boldsymbol{Z} = \{z_i\}_{i=1}^m$. It is assumed that $m \ll n$, which yields a time complexity that scales as $\mathcal{O}(nm^2)$ [82]. In order to apply a sparse GP method for inference, both the input points in $\boldsymbol{Z} = \{z_i\}_{i=1}^m$ and the hyperparameters of $\boldsymbol{\theta} = \{\lambda, \sigma, \eta\}$ must be learned and optimized. The input points in $\boldsymbol{Z} = \{z_i\}_{i=1}^m$ and $\boldsymbol{\theta}$ can be inferred by an approximation of the true log marginal likelihood in Eq. 4.1.1, [82].

Introducing the notation

This subsection will introduce some new notation, and restating some of the notations used for GP, in the previous chapter to make it easier to follow the computations. For consistency with [32], the same notation will be adopted when possible.

Let $\boldsymbol{y}$ denote an observed output/data vector where each y_i is a noisy observation of the function $f(\boldsymbol{x}_i)$ for the training points in $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^{n}$. Since $f(\cdot)$ is not observed directly, it will be referred to as an unobserved or latent function [82]. The latent function, $f(\cdot)$, is assumed to be a Gaussian process prior with zero mean and a covariance function that is specified through a set of hyperparameters. The noisy observations y_i will be corrupted by independent Gaussian noise such that

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \eta^2), \quad i = 1, ..., n. \tag{4.1.2}$$

Thus, $\boldsymbol{y}$ is Gaussian distributed and $\boldsymbol{f}$ contains the latent function values at the points $\boldsymbol{X}_n$ (see Sec. 3.2.1).

A set of *inducing variables* will be introduced here, that is a significant part of most sparse methods for Gaussian process regression. The inducing variables are denoted $Z = \{z_i\}_{i=1}^{m}$, and are points that live in the same space as $\boldsymbol{X}_n$. It could be assumed that Z_m is a subset of $\boldsymbol{X}_n$, where m denotes the number of inducing variables. The vector $\boldsymbol{u}$ is defined to contain the values of the latent function $f(\cdot)$ at the points $Z = \{z_i\}_{i=1}^{m}$, i.e.

$$\boldsymbol{f} = \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_n) \end{bmatrix}, \quad \boldsymbol{u} = \begin{bmatrix} f(\mathbf{z}_1) \\ f(\mathbf{z}_2) \\ \vdots \\ f(\mathbf{z}_m) \end{bmatrix}.$$

Since $f(\cdot)$ follows a Gaussian process, will $\mathbf{y}$, $\mathbf{f}$ and $\mathbf{u}$ be distributed in the following way

$$\begin{aligned} \boldsymbol{y} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_{nn} + \frac{1}{\eta^2}\boldsymbol{I}) \\ \boldsymbol{f} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_{nn}) \\ \boldsymbol{u} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_{mm}), \end{aligned} \tag{4.1.3}$$

where $\boldsymbol{K}_{nn} = \boldsymbol{K}(\boldsymbol{X}_n, \boldsymbol{X}_n)$ is of size $n \times n$ and $\boldsymbol{K}_{mm} = \boldsymbol{K}(Z_m, Z_m)$ is of size $m \times m$. $\boldsymbol{K}_{nm} = \boldsymbol{K}_{mn}^T$ will in what follows be the covariance function between all inducing points Z_m and the training points $\boldsymbol{X}_n$.

The notation introduced here can now be used to formulate an approximation to the true log marginal likelihood function. The following section will focus on the differences between the sparse GP method introduced in [82] (and used in [32]) and the current, common form for approximation methods of GP, such that the novelty of [82] can be emphasized.

4.1.2 Difference between current sparse methods and the variational approximation

Two different state-of-the-art approximation methods for GP are the sparse pseudo-inputs GP method (SPGP), which is proposed in [77], and the likelihood approximation proposed by [72] which is called the Projected Latent Variables (PLV) method [63]. These two methods approximate the log marginal likelihood function in the following way

$$\widetilde{p(\boldsymbol{y})} \sim \mathcal{N}(\mathbf{0}, \eta^2 \boldsymbol{I} + \boldsymbol{Q}_{nn}), \tag{4.1.4}$$

where

$$\boldsymbol{Q}_{nn} = \mathrm{diag}[\boldsymbol{K}_{nn} - \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^T] + \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^T \quad \text{in SPSG}, \tag{4.1.5}$$

and where

$$\boldsymbol{Q}_{nn} = \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^T \quad \text{in PLV}. \tag{4.1.6}$$

$\boldsymbol{Q}_{nn}$ is an approximation of the true covariance function $\boldsymbol{K}_{nn}$ in both Eq. 4.1.5 and Eq. 4.1.6. The PLV form of $\boldsymbol{Q}_{nn}$, $\boldsymbol{Q}_{nn} = \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^T$, is also called the Nyström approximation [82]. The interested reader can consult [94] for a derivation of the method.

Comparing Eq. 4.1.1 with Eq. 4.1.4, it is apparent that $\widetilde{p(\boldsymbol{y})}$ is obtained by modifying the GP prior. This and similar approximation methods focus on maximizing the marginal likelihood of the modified GP model with respect to the inducing inputs $\mathbf{Z}_m$. The approximation in Eq. 4.1.4 will not ensure a reliable approximation of the exact GP model because there is no assurance that the difference between the modified GP and the exact GP model is minimized [82].

The work presented by Titsias [82] instead focuses on minimizing the Kullback-Leibler (KL)-divergence between the exact posterior GP, given by the predictive equations in Eq. 3.2.13 and Eq. 3.2.14, and a variational approximation to the GP posterior. The inducing inputs $\mathbf{Z}_m$ will in this setting be defined to be variational parameters that, together with the model hyperparameters, are jointly selected such that the KL-divergence between the approximate posterior and the true posterior is minimized [82].
The KL-divergence is a common measure of differences between probability distributions [8]. For two distributions f and g is is defined as

$$KL(f||g) = \int f(\boldsymbol{x}) \log\left(\frac{f(\boldsymbol{x})}{g(\boldsymbol{x})}\right), \tag{4.1.7}$$

for some data $\mathbf{x}$. Minimizing the KL-divergence between f and g can be viewed as minimizing the grey area in Fig. 4.1. Obviously, this will occur when the two distributions f and g are as similar as possible.

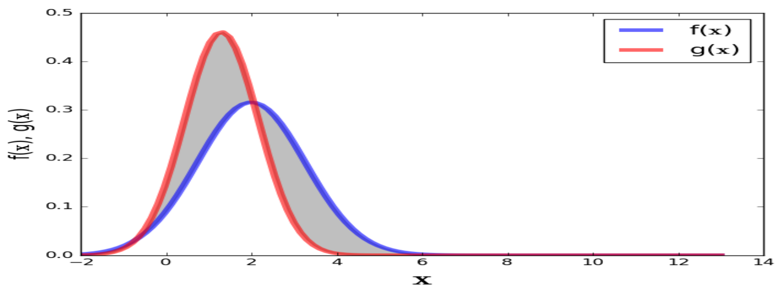


Figure 4.1: Visualization of two arbitrary distributions f and g. Minimizing the KL-divergence corresponds to minimize the grey shaded area in this figure.

4.2 Introduction to variational inference for approximating posterior GP

The aim of this section is to define a sparse method that directly approximates the posterior GP without modifying the GP prior. Variational methods for inference and learning are often used for this kind of problems since they can be used to approximate the exact GP posterior distribution [6]. This section will briefly introduce the concept of how variational inference can be used to both approximate the exact posterior GP and specify the inducing points $\mathbf{Z}_m$. The introduction will follow the work presented in [82] closely.

The exact posterior GP, given by Eq. 3.2.13 and Eq. 3.2.14, can more formally be described through the predictive Gaussian distribution [65]. Using the predictive Gaussian distribution to formulate the prediction of the output $\boldsymbol{f}_*$ given new observations $\boldsymbol{X}_*$ gives

$$\begin{aligned} p(\boldsymbol{f}_*|\boldsymbol{y}) &= \int p(\boldsymbol{f}_*, \boldsymbol{f}|\boldsymbol{y}, X_n, X_*) d\boldsymbol{f} \\ \text{Using:} \quad p(\boldsymbol{f}_*, \boldsymbol{f}) &= p(\boldsymbol{f}_*|\boldsymbol{f}) p(\boldsymbol{f}) \\ &= \int p(\boldsymbol{f}_*|\boldsymbol{f}, \boldsymbol{y}, X_n, X_*) p(\boldsymbol{f}|\boldsymbol{y}, X_n) d\boldsymbol{f}. \end{aligned} \tag{4.2.1}$$

The dependency of $\boldsymbol{X}_n$ and $\boldsymbol{X}_*$ will for simplicity be neglected from now on, which allows the following formulation

$$p(\boldsymbol{f}_*|\boldsymbol{y}) = \int p(\boldsymbol{f}_*|\boldsymbol{f},\boldsymbol{y})p(\boldsymbol{f}|\boldsymbol{y})d\boldsymbol{f}. \tag{4.2.2}$$

Here, $p(\boldsymbol{f}_*|\boldsymbol{f},\boldsymbol{y})$ denote the conditional GP prior, while $p(\boldsymbol{f}|\boldsymbol{y})$ denote the posterior distribution over the *training latent function values*. Eq. 4.2.2 shows that the predictive Gaussian distribution is computed by marginalization over the function values $\boldsymbol{f}$, see Ch. 2.2. The posterior GP, given by the integral in Eq. 4.2.2, can be approximated using a small set of m inducing variables $\boldsymbol{u}$, that is evaluated at the points $Z = \{z_i\}_{i=1}^m$.

Assuming that $\boldsymbol{f}_*$ are independent from the training inputs, $\boldsymbol{X}_n$ [82], allows the following reformulation of Eq. 4.2.2

$$\begin{aligned} p(\boldsymbol{f}_*|\boldsymbol{y}) &= \int p(\boldsymbol{f}_*,\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})d\boldsymbol{f}d\boldsymbol{u} \\ &= \int p(\boldsymbol{f}_*|\boldsymbol{f},\boldsymbol{u},\boldsymbol{y})p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})d\boldsymbol{f}d\boldsymbol{u} \\ &= \int p(\boldsymbol{f}_*|\boldsymbol{f},\boldsymbol{u},\boldsymbol{y})p(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{y})p(\boldsymbol{u}|\boldsymbol{y})d\boldsymbol{f}d\boldsymbol{u}. \end{aligned} \tag{4.2.3}$$

Furthermore, assuming that $\boldsymbol{u}$ is a *sufficient statistic* for the parameter $\boldsymbol{f}$ in the sense that $\boldsymbol{f}_*$ and $\boldsymbol{f}$ are independent given $\boldsymbol{u}$ implies that the conditional prior over any finite set of the function points in $\boldsymbol{f}_*$ can be computed through $\boldsymbol{u}$ only [82]. Thus, it holds that $p(\boldsymbol{f}_*|\boldsymbol{u},\boldsymbol{f}) = p(\boldsymbol{f}_*|\boldsymbol{u})$ for any $\boldsymbol{f}_*$. Using this, Eq. 4.2.3 can be rewritten in the following way:

$$\begin{aligned} p(\boldsymbol{f}_*|\boldsymbol{y}) &= \int p(\boldsymbol{f}_*|\boldsymbol{u},\boldsymbol{y})p(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{y})p(\boldsymbol{u}|\boldsymbol{y})d\boldsymbol{f}d\boldsymbol{u} \\ & \text{Define:} \quad \phi(\boldsymbol{u}) = p(\boldsymbol{u}|\boldsymbol{y}) \\ &= \int p(\boldsymbol{f}_*|\boldsymbol{u},\boldsymbol{y})\Big[\int p(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{y})d\boldsymbol{f}\Big]\phi(\boldsymbol{u})d\boldsymbol{u} \\ & \text{where} \quad \int p(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{y})d\boldsymbol{f} = 1 \\ &= \int p(\boldsymbol{f}_*|\boldsymbol{u},\boldsymbol{y})\phi(\boldsymbol{u})d\boldsymbol{u}. \end{aligned} \tag{4.2.4}$$

The last line of Eq. 4.2.4 can be reformulated to $q(\boldsymbol{f}_*, \boldsymbol{u}) = p(\boldsymbol{f}_*, \boldsymbol{u}|\boldsymbol{y})p(\boldsymbol{u}|\boldsymbol{y}) = p(\boldsymbol{f}_*|\boldsymbol{u}, \boldsymbol{y})\phi(\boldsymbol{u})$, using conditional probability once and defining $q(\boldsymbol{f}_*) = p(\boldsymbol{f}_*|\boldsymbol{y})$. Eq. 4.2.4 can then be written in the following, simplified form

$$q(\boldsymbol{f}_*) = \int q(\boldsymbol{f}_*, \boldsymbol{u})d\boldsymbol{u}. \tag{4.2.5}$$

The distribution in Eq. 4.2.5 is a *new* predictive distribution, similar to the one defined in Eq. 4.2.1, but the predictive distribution in Eq. 4.2.5 is only dependent on m function points through the inducing variables $\boldsymbol{u}$. This is an advantage over Eq. 4.2.1, which depends on n inducing variables through $\boldsymbol{f}$. As long as the assumption of $\boldsymbol{u}$ being a sufficient statistic for $\boldsymbol{f}$ holds will Eq. 4.2.5 be an exact approximation of Eq. 4.2.1. Consequently, Eq. 4.2.5 can be computed in $\mathcal{O}(nm^2)$ time instead of $\mathcal{O}(n^3)$ [83].

It will, unfortunately, be hard to find the inducing variables $\boldsymbol{u}$ such that they are sufficient statistics. It is more likely that $q(\boldsymbol{f}_*)$ only will be an approximation of the exact predictive distribution $p(\boldsymbol{f}_*|\boldsymbol{y})$ in Eq. 4.2.1. In order to optimize the quality of this approximation, [83] defines $\phi(\boldsymbol{u})$ to be a “free” variational distribution where $\phi(\boldsymbol{u}) \neq p(\boldsymbol{u}|\boldsymbol{y})$ with $q(\boldsymbol{f}_*) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{A})$.

The predictive posterior distribution in Eq. 4.2.4 (and Eq. 4.2.5) will be Gaussian, and it is possible to show that the mean and covariance functions of the approximate posterior GP can be formulated in the following way [82],

$$\begin{aligned} \bar{\mathrm{f}}'_* &= \boldsymbol{K}_{*m}\boldsymbol{K}_{mm}^{-1}\tilde{\boldsymbol{\mu}} \\ \mathrm{cov}(\mathrm{f}'_*) &= \boldsymbol{K}_{**} - \boldsymbol{K}_{m*}^T\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{m*} + \boldsymbol{K}_{m*}^T\boldsymbol{K}_{mm}^{-1}\boldsymbol{A}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{m*}. \end{aligned} \tag{4.2.6}$$

The approximate predictive equations in Eq. 4.2.6 are the general form of a sparse posterior GP. The quality of how well Eq. 4.2.5 approximates Eq. 4.2.1 will depend on where the inducing points $\boldsymbol{Z}_m$ are located, thus both the ϕ-distribution, i.e. $(\boldsymbol{\mu}, \boldsymbol{A})$ and $\boldsymbol{Z}_m$ needs to be selected in an appropriate way. Titsias [82] solves this issue by applying a variational inference method that jointly specifies $\boldsymbol{Z}_m$ and the ϕ-distribution, and treats the inducing inputs as variational parameters. The inducing points $\boldsymbol{Z}_m$ and the ϕ-distribution will be selected such that the KL divergence between the augmented true posterior, $p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y}) = p(\boldsymbol{f}|\boldsymbol{u}, \boldsymbol{y})p(\boldsymbol{u}|\boldsymbol{y})$, and the augmented variational posterior distribution $q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})$ is minimized, [83]. Using the definition in [40] allow the following formulation of the KL-divergence

$$\mathrm{KL}(q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})||p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})) = \int q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y}) \log\left[\frac{q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})}{p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})}\right] d\boldsymbol{f} d\boldsymbol{u}. \tag{4.2.7}$$

Clearly, this KL-divergence will be minimized when there are $m = n$ inducing variables that are located at exact the same positions as the original training points, i.e. $\boldsymbol{u} = \boldsymbol{f}$ and $\boldsymbol{K}_{mm} = \boldsymbol{K}_{nm} = \boldsymbol{K}_{nn}$ [32]. This will however not yield any computational benefits, and this is why m is restricted to $m \ll n$. Minimization of the KL-divergence in Eq. 4.2.7 is equivalent to maximizing the variational lower bound on the true augmented log marginal likelihood [40], i.e.

$$\max\left(log\left[p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{u})\right]\right). \tag{4.2.8}$$

Perfroming the maximization in Eq. 4.2.8 will determine the variational quantities $\boldsymbol{Z}_m$ and the ϕ-distribution [83], thus:

$$\begin{aligned}\log\left[p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{u})\right] &= \log\left[\int p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})d\boldsymbol{f}d\boldsymbol{u}\right] \\ &= \log\left[\int p(\boldsymbol{f}|\boldsymbol{u})\phi(\boldsymbol{u})\left(\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{f}|\boldsymbol{u})\phi(\boldsymbol{u})}\right)d\boldsymbol{f}d\boldsymbol{u}\right].\end{aligned} \tag{4.2.9}$$

Let $U = p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})/\big(p(\boldsymbol{f}|\boldsymbol{u})\phi(\boldsymbol{u})\big)$ and $V = p(\boldsymbol{f}|\boldsymbol{u})\phi(\boldsymbol{u})$, the expression within the integral can then be formulated as $E[U]$ with respect to V, i.e. the expectation of U with respect to V. The computation of the integral in the second line of Eq. 4.2.9 can be simplified by applying Jensen's inequality. The concavity of the logarithm function, and Jensen's inequality will imply that $\log E[f(\boldsymbol{y})] \geq E[\log f(\boldsymbol{y})]$ is true for any random vector $\boldsymbol{y}$ [36]. Thus, applying Jensen's inequality to Eq. 4.2.9 yields

$$\begin{aligned}\log\left[p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{u})\right] &\geq \int p(\boldsymbol{f}|\boldsymbol{u})\phi(\boldsymbol{u})\log\left(\frac{p(\boldsymbol{y}|\boldsymbol{f})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}p(\boldsymbol{u})}{\cancel{p(\boldsymbol{f}|\boldsymbol{u})}\phi(\boldsymbol{u})}\right)d\boldsymbol{f}d\boldsymbol{u} \\ &= \int \phi(\boldsymbol{u})\left(\left[\int p(\boldsymbol{f}|\boldsymbol{u})\log p(\boldsymbol{y}|\boldsymbol{f})d\boldsymbol{f}\right] + \log\frac{p(\boldsymbol{u})}{\phi(\boldsymbol{u})}\right)d\boldsymbol{u}.\end{aligned} \tag{4.2.10}$$

It is possible to show[1], that the integral within the square brackets will be

$$\int p(\boldsymbol{f}|\boldsymbol{u})\log p(\boldsymbol{y}|\boldsymbol{f})d\boldsymbol{f} = \log\left[\mathcal{N}(\boldsymbol{\alpha},\eta^2\boldsymbol{I})\right] - \frac{1}{2\eta^2}Tr(\widetilde{\boldsymbol{K}}), \tag{4.2.11}$$

where $\boldsymbol{\alpha} = \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{u}$, $\widetilde{\boldsymbol{K}} = \boldsymbol{K}_{nn} - \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn} = \boldsymbol{K}_{nn} - \boldsymbol{Q}_{nn}$ and Tr denotes the trace. Insert Eq. 4.2.11 into Eq. 4.2.10 and reversing the Jensen's inequality, i.e.

1. The interested reader can consult [83] for the derivation of this expression

moving the logarithm outside of the integral in Eq. 4.2.10 yields the following objective function, i.e. lower bound, after maximization

$$L_2 = \log \mathcal{N}(0, \eta^2 \boldsymbol{I} + \boldsymbol{Q}_{nn}) - \frac{1}{2\eta^2} Tr(\widetilde{\boldsymbol{K}}), \tag{4.2.12}$$

where $\boldsymbol{Q}_{nn}\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}$. The interested reader can consult [83] for the complete derivation of L_2. This lower bound matches the lower bound, also denoted L_2, in [32], with $\beta = 1/\eta^2$. The *new* marginal likelihood function, Eq. 4.2.12, derived by [82] differentiates from the frequently used marginal likelihood function given by Eq. 4.1.4 through the regularization trace term: $-\frac{1}{2\eta^2}Tr(\widetilde{\boldsymbol{K}})$. This term will be zero when $\boldsymbol{K}_{nn} = \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}$, which occurs when the variational distribution $q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})$ approximates the true posterior distribution exactly. $\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}$ is still referred to as the Nyström approximation, [82]. The trace term corresponds to the squared error of predicting the training latent values $\boldsymbol{f}$ from $\boldsymbol{u}$, i.e. only using m inducing inputs instead of all n training inputs. Thus, L_2 attempts to both maximize the log likelihood and simultaneous minimizing the regularization term [82, 83].

The lower bound given by Eq. 4.2.12 can be computed in $\mathcal{N}(nm^2)$ time, which clearly is an enhancement compared to $\mathcal{N}(n^3)$. A maximization of the lower bound can be achieved by optimizing over $\boldsymbol{Z}_m$, and also the number of inducing inputs, m. The inducing inputs will determine the flexibility of the variational distribution $q(\boldsymbol{f}, \boldsymbol{u}|y) = p(f|\boldsymbol{u}, \boldsymbol{y})\phi(\boldsymbol{u})^*$ in Eq. 4.2.7, where $\phi(\boldsymbol{u})^*$ is defined as the optimal distribution [82].

$\phi(\boldsymbol{u})^*$ can be computed by inserting Eq. 4.2.11 into Eq. 4.2.10 and differentiate it with respect to $\phi(\boldsymbol{u})$. After some computations, see [83], can the optimal variational distribution $\phi(\boldsymbol{u})^*$ be defined in the following way

$$\phi(\boldsymbol{u})^* \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}, \boldsymbol{A}), \tag{4.2.13}$$

where

$$\tilde{\boldsymbol{\mu}} = \eta^{-2}\boldsymbol{K}_{mm}(\boldsymbol{K}_{mm} + \eta^{-2}\boldsymbol{K}_{mn}\boldsymbol{K}_{nm})^{-1}\boldsymbol{K}_{mn}, \tag{4.2.14}$$

and

$$\boldsymbol{A} = \boldsymbol{K}_{mm}(\boldsymbol{K}_{mm} + \eta^{-2}\boldsymbol{K}_{mn}\boldsymbol{K}_{nm})^{-1}\boldsymbol{K}_{mm}. \tag{4.2.15}$$

The variational GP is fully specified by the optimal variational distribution $\phi(\boldsymbol{u})^*$. Substituting for $(\tilde{\boldsymbol{\mu}}, \boldsymbol{A})$ into Eq. 4.2.6 will allow for the possibility of making predictions in unseen points using only m inducing inputs [82].

This section has focused on the ideas from [82]and describes how the GP posterior can be approximated through $\phi(\boldsymbol{u})^*$. Stochastic variational inference for Gaussian process models, introduced in [32], are based on another formulation of these results. In order to embrace the framework of SVI for GP models, the following section will be devoted to a derivation, confirming that the distribution in Eq. 4.2.13 corresponds to the implicit approximating distribution $q(\boldsymbol{u})$ presented in [32]. When these results have been confirmed, Sec. 4.3 will focus on a brief introduction to the main ideas of SVI for GP models.

4.2.1 Confirming the results

The aim of this section is to work through the necessary calculations that shows that the optimal variational distribution $\phi(\boldsymbol{u})^*$ defined by Eq. 4.2.13, corresponds to the implicit approximating distribution $q(\boldsymbol{u})$ in [32]. The implicit approximating distribution in [32] is defined in the following way

$$q(\boldsymbol{u}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{A}}), \tag{4.2.16}$$

where

$$\begin{aligned}\hat{\boldsymbol{\mu}} &= \beta \hat{A}^{-1} \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \boldsymbol{y} \\ \hat{\boldsymbol{A}} &= \beta \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} + \boldsymbol{K}_{mm}^{-1}.\end{aligned} \tag{4.2.17}$$

Hensman et al. [32] actually use $\boldsymbol{\Lambda}$ instead of $\hat{\boldsymbol{A}}$, but this symbol is already used for the characteristic length scale matrix, so $\hat{\boldsymbol{A}}$ will represent the covariance matrix in the approximation distribution $q(\boldsymbol{u})$.

Let $\widehat{\boldsymbol{K}_{mm}}$ define a *new* matrix, defined as $\widehat{\boldsymbol{K}_{mm}} = \boldsymbol{K}_{mn}\boldsymbol{K}_{nm}$. Note that $\beta = 1/\eta^2$ is called the precision, and will be used from now on in the expressions of Eq. 4.2.14 and Eq. 4.2.15. The following known matrix manipulations [51] will be used, here shown for three arbitrary matrices $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$ that all are invertible

$$\begin{aligned}\boldsymbol{U}(\boldsymbol{V} + \boldsymbol{W}) &= \boldsymbol{U}\boldsymbol{V} + \boldsymbol{U}\boldsymbol{W} \quad (1) \\ (\boldsymbol{U}\boldsymbol{V})^{-1} &= \boldsymbol{V}^{-1}\boldsymbol{U}^{-1} \quad (2) \\ (\boldsymbol{U}^{-1})^{-1} &= \boldsymbol{U} \quad (3) \\ (\boldsymbol{V} + \boldsymbol{W})\boldsymbol{U} &= \boldsymbol{V}\boldsymbol{U} + \boldsymbol{W}\boldsymbol{U} \quad (4).\end{aligned} \tag{4.2.18}$$

It is important to note that both $\boldsymbol{K}_{nn}$ and $\boldsymbol{K}_{mm}$ are assumed to be invertible. If this was not the case, the predictions in Eq. 3.2.13, Eq. 3.2.14 and Eq. 4.2.6 would not be possible to compute, since the inverse of $\boldsymbol{K}_{nn}$ and $\boldsymbol{K}_{mm}$ need to be computed in order to achieve predictions. This section starts by confirming that $\tilde{\boldsymbol{\mu}} = \hat{\boldsymbol{\mu}}$

$$\begin{aligned}
\hat{\boldsymbol{\mu}} &= \beta\left(\beta \boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1} + \boldsymbol{K}_{mm}^{-1}\right)^{-1}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{y}, \\
&\quad \text{using Eq. 4.2.18 (3) with } \boldsymbol{V}^{-1} = (\cdot)^{-1} \text{ and } \boldsymbol{U}^{-1} = \boldsymbol{K}_{mm}^{-1} \\
&= \beta\left(\boldsymbol{K}_{mm}\left[\beta \boldsymbol{K}_{mm}^{-1}\widehat{\boldsymbol{K}_{mm}}\boldsymbol{K}_{mm}^{-1} + \boldsymbol{K}_{mm}\right]\right)^{-1}\boldsymbol{K}_{mn},
\end{aligned} \tag{4.2.19}$$

using Eq. 4.2.18 (1) within the square brackets

$$\begin{aligned}
\hat{\boldsymbol{\mu}} &= \beta\left[\beta\widehat{\boldsymbol{K}_{mm}}\boldsymbol{K}_{mm}^{-1} + \boldsymbol{I}\right]^{-1}\boldsymbol{K}_{mn}\boldsymbol{y} \\
&= \underline{\beta\left[\boldsymbol{I} + \beta\widehat{\boldsymbol{K}_{mm}}\boldsymbol{K}_{mm}^{-1}\right]^{-1}\boldsymbol{K}_{mn}\boldsymbol{y}}.
\end{aligned} \tag{4.2.20}$$

Now focusing on $\tilde{\boldsymbol{\mu}}$,

$$\begin{aligned}
\tilde{\boldsymbol{\mu}} &= \beta \boldsymbol{K}_{mm}\left(\boldsymbol{K}_{mm} + \beta\widehat{\boldsymbol{K}_{mm}}\right)^{-1}\boldsymbol{K}_{mn}\boldsymbol{y} \\
&\quad \text{Using Eq. 4.2.18 (3)} \\
&= \beta\left(\boldsymbol{K}_{mm}^{-1}\right)^{-1}\left(\boldsymbol{K}_{mm} + \beta\widehat{\boldsymbol{K}_{mm}}\right)^{-1}\boldsymbol{K}_{mn}\boldsymbol{y} \\
&\quad \text{Using Eq. 4.2.18 (2)} \\
&= \beta\left[\left(\boldsymbol{K}_{mm} + \beta\widehat{\boldsymbol{K}_{mm}}\right)\boldsymbol{K}_{mm}^{-1}\right]^{-1}\boldsymbol{K}_{mn}\boldsymbol{y} \\
&\quad \text{Using Eq. 4.2.18 (4) within the square brackets:} \\
&= \underline{\beta\left[\boldsymbol{I} + \beta\widehat{\boldsymbol{K}_{mm}}\boldsymbol{K}_{mm}^{-1}\right]^{-1}\boldsymbol{K}_{mn}\boldsymbol{y}}.
\end{aligned} \tag{4.2.21}$$

This shows that Eq. 4.2.20 is equal to Eq. 4.2.22, which implies that $\tilde{\boldsymbol{\mu}} = \hat{\boldsymbol{\mu}}$.

Now, turning the attention towards the covariance and showing that $\boldsymbol{A} = \hat{\boldsymbol{A}}$,

$$\boldsymbol{A} = \boldsymbol{K}_{mm}(\boldsymbol{K}_{mm} + \beta\widehat{\boldsymbol{K}_{mm}})^{-1}\boldsymbol{K}_{mm}$$

Using the trick of Eq. 4.2.18 (3) on both sides of the parenthesis:

$$= \left[\boldsymbol{K}_{mm}^{-1}\left(\boldsymbol{K}_{mm} + \beta\widehat{\boldsymbol{K}_{mm}}\right)\boldsymbol{K}_{mm}^{-1}\right]^{-1} \tag{4.2.22}$$

$$= \left(\boldsymbol{K}_{mm}^{-1} + \beta\widehat{\boldsymbol{K}_{mm}}^{-1}\right)^{-1}$$

which is the same expression as in Eq. 4.2.17 with $\widehat{\boldsymbol{K}_{mm}} = \boldsymbol{K}_{mn}\boldsymbol{K}_{nm}$.

This confirms that the interpretation of Titsias's work in the previous sections corresponds to the same rederivation presented in [32].

4.3 Stochastic variational inference for GP

In the previous sections, an introduction to sparse GP models was presented, through the concept of variational inference. The introduction was based on an additional interpretation of the work presented in [82], and does not follow the rederivation of the variational approach to inducing variables that is presented in [32]. A brief summation of the main results from the previous sections can be found in the following subsection, which highlights the starting point of stochastic variational inference.

Summarizing the main results

Sec. 4.2 introduced the approximate predictive equations in Eq. 4.2.6, which provide the general form of the sparse posterior GP. The quality of how well the new predictive distribution, $q(\boldsymbol{f}_*)$ approximates the exact predictive distribution, denoted $p(\boldsymbol{f}_*|\boldsymbol{y})$, will depend on the number of inducing points $\boldsymbol{Z}_m$, where they are located and how the variational distribution ϕ was defined. The KL-divergence was introduced such that the distance between the augmented true posterior $p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y}) = p(\boldsymbol{f}|\boldsymbol{u}, \boldsymbol{y})p(\boldsymbol{u}|\boldsymbol{y})$, and the augmented variational posterior distribution $q(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y})$ was minimized. Instead of minimizing the KL-divergence directly it is a common practice, see [32, 36, 40, 82] to maximize what is called the variational lower bound of the true marginal likelihood.

This will result in the objective function, denoted L_2 in Eq. 4.2.12. Maximization, i.e. optimization, of this lower bound can, as described in Sec. 4.2, be done by optimizing over Z_m, which also will modify the optimal variational distribution $q(\boldsymbol{uu})^*$. The previous sections did not mention how the optimization should be accomplished, and this will be the starting point of the stochastic variational inference method proposed in [36].

The rest of this section will introduce the main concepts behind stochastic variational inference, that allow the use of variational inference for very large datasets, and will follow the work presented in [32]. The concept of stochastic variational inference is initially introduced in [36], which can be consulted for the main ideas behind SVI. The authors of [36] do not consider how the ideas of SVI can be combined with Gaussian processes, and this is where the novelty of the work presented in [32] lies. This section will start with a brief introduction to the main concepts of SVI, and then focus on how [32] combines these ideas with the concepts of Gaussian processes.

The main idea proposed in [36] is to use stochastic optimization, [79], for the optimization of the objective function that was found through the method of variational inference. Combing stochastic optimization with variational inference result in the method SVI [36]. Stochastic optimization focuses on finding the maximum of an objective function, by following noisy, but unbiased estimates of its gradient (i.e. the slope [18]) with decreasing step size. The requirement for applying SVI to a model is that it contains a set of *global* variables [36], that can be factorized into observations and inducing variables [32]. SVI is a powerful tool for applying inference to very large datasets when this requirement is fulfilled [32]. The introduction of the inducing variables $\boldsymbol{u}$ in the previous section will guarantee for the GP model to be appropriate for SVI. Unfortunately will the dependencies between the observations be re-introduced when the marginalization over $\boldsymbol{u}$ (Eq. 4.2.4) is performed, which will eliminate the global parameters.

Hensman et al. [32] presents a way to work around this problem by introducing an additional variational distribution $q(\boldsymbol{u})$, that unlike the variational distribution in Eq. 4.2.16, is explicit. This additional variational distribution will allow for SVI, which can be used to optimize the kernel hyperparameters, and the noise precision ($\beta = 1/\eta^2$) by performing standard stochastic gradient descent alongside the variational parameters. The work proposed by [32] allows for a gradient approximation to the natural gradient, where the data can be considered either separately or in mini batches [32]. The interested reader can consult [32] and [33] for a deeper insight to how the global variables are defined and how natural gradients are computed.

Fig. 4.2 is borrowed from Hensman et al. [32] and describes visually how the approximate GP posterior converges towards the exact GP posterior when SVI are applied for GPR. Ten mini batches were used in total and only three inducing points are considered during each step.

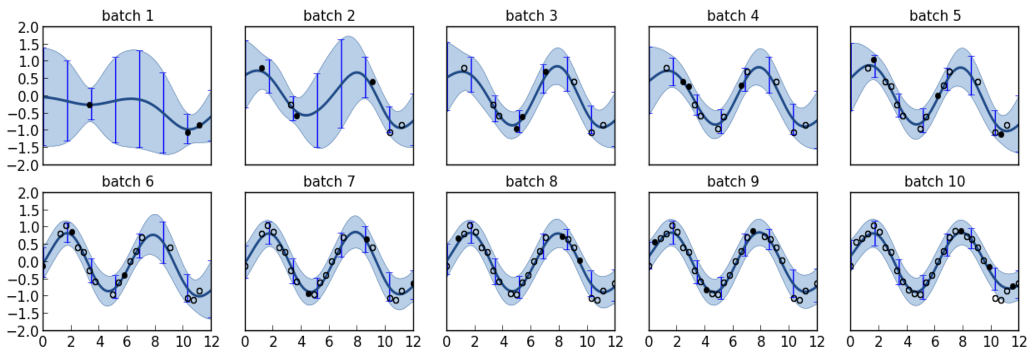


Figure 4.2: Example of how a GPR problem is optimized thought stochastic variational inference. The optimization is performed in 10 mini batches, and each panel of the figure shows the posterior GP after optimization of the points considered in each mini batch. The black, dark points indicate which of the points that were considered for each mini batch. The hollow points show points that have been considered during optimization of previous mini batches. The vertical error bars in this figure represents the explicit variational distribution $q(\boldsymbol{u})$. The black curve indicates the true posterior GP. This figure is borrowed from [32] because it explains the method of applying SVI for GP in such a good way.

The authors of [32] have implemented the method of SVI for GP as a part of the GPy Gaussian process toolkit, `http://github.com/SheffieldML/GPy`. The implementation requires that the user specifies the batch size and the number of inducing variables that should be considered.

Part II

Description of the data, preprocessing and validation

5 Data sources

The identification and collection of the data that will be used trough this thesis was conducted in collaboration with the other team members of the eSushi project team at SINTEF Nord. The process of identifying, collecting, assimilating and preprocessing the data has been a major part of the preparatory work for this thesis. This chapter will, therefore, be devoted to a description of the preparatory work, the identified data sources and how different features were assimilated into a dataset. The last part of this chapter will describe how a baseline for the forthcoming analysis was defined through the preprocessing of the assimilated dataset.

A selection of a data source, or a feature, was motivated by the overall question of this thesis; is it possible to achieve accurate predictions of the quantity of fish catch through the interpretation of different oceanic features? The group of different data sources, from which the features were extracted, was selected based on their ability to describe the state of the ocean, and/or the nutrient content in the ocean.

It is important to notice that this is the first time, to the author's knowledge, that the selected different data sources have been assimilated with catch reports from the Norwegian shipping company Havfisk ASA for fish catch prediction. This work is a major part of the novelty of the work presented in this thesis. Thus, the definition of the baseline in the preprocessing part of this thesis was mostly defined through what seems to work and the wish to start these initial analysis through a smaller case instead of using all the available data which could confuse the understanding of the results.

5.1 Data from the Havfisk ASA data base

The first selected data source was provided by Dualog AS's electronic logbook, `eFangst`, and contains fish catch reports from the Norwegian shipping company Havfisk ASA. The vessels, belonging to Havfisk ASA, are logging each activity that they perform during one *trip*. This database contains, among other things, information about the activities that the vessels from Havfisk ASA performs at the ocean. The following will briefly introduce some of the main parts from the Havfisk ASA database, henceforth denoted the Havfisk database,

- The logged data in the database ranges from November 2006 to June 2015.
- A *trip* is defined as the time between a vessel leaves a port, and lasts until the vessel arrives to port again.
- *To launch a haul* will denote the activity that is performed when a vessel lowers its fishing gear into the water.
- The *haul* ends when the fishing gear is brought aboard on the ship.
- For each haul a vessel launch, the *date*, *time*, and *location* is logged for both the *launch* and the retraction of the *haul*.
- It is possible for a vessel to launch several hauls during the same trip.
- All catch of fish that are brought on board is logged in numerous of ways. This includes the different species of fish, the quantity of catch for each individual species in units of kg (kilogram) and the gear specification used.
- Only start- and finish positions are registered during trawling in the database. This implies that there is no knowledge of the accurate position(s) where the fish actually was caught.

Due to the last point, it will be assumed that the position where the fish is caught will be equivalent to the vessels reported haul position. This assumption will be independent of the actual gear specification. The haul position will from now on be denoted the *catch position(s)*.

Because the purpose of this thesis is to find out if it is possible to predict the quantity of catch of fish from the collected data sources, the actually logged quantity of catch will be defined to be the correct output from the prediction model. In the GP context, the actual quantity of catch will, therefore, be denoted with $\mathbf{y}$ or f_* depending on if the quantity of catch is used for training or prediction. The reported catch positions, consisting of longitude and latitude coordinates will together with the reported day and time be used as input to the ML regression model. Figure 5.1 shows how the data from the Havfisk is partitioned into input and output data.

5.2 Additional data sources and their different features

The following three data sources have been selected since they can provide the catch position in the database from Havfisk ASA with additional knowledge of the state of the ocean. The selected variables from each data source, i.e. model, will from now on be referred to as features or variables. The intuition is that if these features are chosen appropriately they will be able to help the ML method to recover some pattern in the dataset, that may be informative in for predictions.

5.2.1 The TOPAZ4 model

TOPAZ4 is an Arctic Ocean physics reanalysis product provided by the Copernicus Marine environment monitoring service, CMEMS[1]. Documentation over the physical variables, achieved from the TOPAZ4 reanalysis product, can be found in the quality information document [97] and [68]. These two references are also the main information sources for this reanalysis product.

TOPAZ4 have for example assimilated Sea surface temperature from NOAA (National Oceanic and Atmospheric Administration)[2] and the OSTIA system (i.e. Operational Sea Surface Temperature and Sea Ice Analysis) with in situ temperature and salinity from hydrographic cruises. The following two references can be consulted for additional information about the OSTIA system; [24, 80]. Some model specifications for TOPAZ4,

- TOPAZ4 uses version 2.2.18 of HYCOM, see Ref. [9] , (the HYbrid Coordinate Ocean Model- NORWegian ECOlogical Model).

- The TOPAZ4 model covers the North Atlantic and the Arctic basins, which also is shown in Figure 2 in [97].

- The current model has a temporal coverage from 1991-01-15, time 00:00:00 to 2014-12-15, time 00:00:00.

1. See `http://marine.copernicus.eu/`.
2. See `http://www.noaa.gov/`

More about the model's specification can be found in Ref. [19], where the entire dataset can be downloaded after registration. The code for the model can be accessed from: `https://svn.nersc.no/repos/hycom` or browsed at `https://svn.nersc.no/hycom/browser`.

Table 5.1 list the features that are achieved from the TOPAZ4 reanalysis product, which all are real physical variables [97]. The reanalysis product contains data of both monthly means and daily means. The feature *bottom temperature* was only available in monthly means, which explains why daily means was not chosen for this feature. Figure 5.1 shows how the features from the TOPAZ4 model are assimilated

Table 5.1: List over the physical variables achieved from the TOPAZ4 reanalysis product, where [·] indicate the unit.

Physical variables
Bottom temperature [°C], monthly means
Sea surface temperature [°C], daily means
Sea surface salinity [g/kg], daily means
Depth, [m]

into the input training and test datasets.

5.2.2 The HYCOM-NORWECOM model

This section will give a brief review of the HYCOM-NORWECOM model (HYbrid Coordinate Ocean Model-NORWegian ECOlogical Model), as well as the biochemical parameters achieved from the model. Reference [69] can be consulted for an in-depth assessment of the revised HYCOM-NORWECOM v2.1 model. The HYCOM-NORWECOM model is a physical-biological modelling system that combines the ocean model HYCOM [9] with the physical-chemical-biological NORWECOM [73] model system. A user's manual of different versions of the HYCOM model can be found at: `http://hycom.org/hycom/documentation`.

- The HYCOM-NORWECOM v2.1 model is tested and evaluated on both local, in situ observations, as well as satellite data.
- The HYCOM-NORWECOM v2.1 model could provide this thesis with features from year 1997 up to, and including 2014.
- The model covers the North Atlantic and the Arctic Ocean.

The available features from the model are listed in Tab. 5.2, where the feature DEPT is

Table 5.2: List over the biogeochemical variables that are retrieved from the revised HYCOM-NORWECOM v2.1 model. DEPT are in the model defined to be the phosphate part of Detritus.

Biogeochemical variables		
Nitrogen	Phosphate	Silicate
Flagellates	Diatoms	Primary production
Microzooplankton	Mesozooplankton	Detritus
Biogenic silica	DEPT	Chlorophyll
	Model Depth	

the phosphate part of Detritus. Both features close to the surface, and near the bottom of the model are extracted from this model as well as the depth of the model. The units of the different features are omitted in Tab. 5.2 because they where not provided by the model. Figure 5.1 shows how the features from the HYCOM-NORWECOM v2.1 model are assimilated into the input training and test datasets.

5.2.3 The Nora10 model

NORA10 is a regional hindcast model, that is a downscaling of the ERA40 reanalysis to a spatial resolution of 10-11 km through the High Resolution Limited Area Model (HIRLAM, see Ref. [84]) of version 6.4.2, [27]. Data from the NORA10 model are according to Reistad et al. [67] accurate enough to be used in the designing and planning of offshore installations and operations. The methods of the downscaling, as well as assessments of the accuracy of the model, are well described in [67]. The wave simulations in NORA10 are achieved from a modified version of the WAM [42] cycle 4 model [2]. Some additional specifications regarding the NORA10 hindcast model,

- The ERA40 dataset consists of a reanalysis of meteorological observations, which spans the period from September 1957 to August 2002 [85] and have a resolution of approximately 125 km [67].
- HIRLAM is an atmospheric model with a 10 km horizontal resolution [67].
- The geographical scope of NORA10 is the North-East Atlantic Ocean, which includes the North Sea, the Norwegian Sea and the Barents Sea, [2].

The variables, provided by the NORA10 model, are listed in Table 5.3. Only wind and wave features near the sea surface was extracted since the effects of the wind and waves are strongest close to the surface.

Table 5.3: List over the wind and wave variables that can be retrieved from the NORA10 model. The letter(s) within [·] indicate the unit of the variable. A variable without [·] indicates that there were no knowledge about the unit of that variable.

Variables from NORA10	
Significant wave height [m]	Peak wave period [s]
Significant wave period [s]	Peak wave direction [°C]
Wave direction	Significant wind wave height [m]
Peak wave period wind [s]	Peak wave direction wind [°C]
Mean wave direction wind [°C]	Significant swell wave height [m]
Peak wave period swell [s]	Mean wave period swell [s]
Mean wave direction swell [°C]	Stokes drift x-velocity [ms^{-1}]
Stokes drift y-velocity [ms^{-1}]	Wind speed [ms^{-1}]
Wind direction	Peak wave direction
Peak Wave Period Swim	

5.3 Additional features

Two additional features were chosen to be included in the forthcoming analysis; the *lunar phase* and the *distance to land*.

5.3.1 Lunar phase

There are some studies, e.g. Poisson et al. [62], that claims that the lunar cycle could affect the catch rate of some fish species. For this reason will the phase of the moon, i.e. the lunar phase, be considered as an additional feature. The Python function `moon.py`[3] was used to calculate the "actual" lunar phase at the same time, date and position as the catch of fish was brought aboard on the vessel. The function returns a value between 0.0-1.0, which indicates the percentage of illumination of a full circle. Thus the "actual" lunar phase is defined to be the percentage of illumination of a full circle , where 0.0 indicates *new moon*, while 1.0 indicates *full moon*.

5.3.2 Distance to land

The shortest *distance to land* was computed from each of the catch reports, for each catch position. The unit of this feature will be in kilometers (km). The *distance to*

3. The code `moon.py` can be retrieved from:
https://github.com/KitWallace/routefinder/blob/master/moon.py
The Python-function is based on a code written by John Walker, where the algorithms are achieved from the book: *Practical astronomy with your calculator, P. Duffett-Smith, 2nd Edition*.

land feature was selected because it is possible that that different fish species are caught at different distances from land. This implies that this feature can be used to distinguish between different fish species.

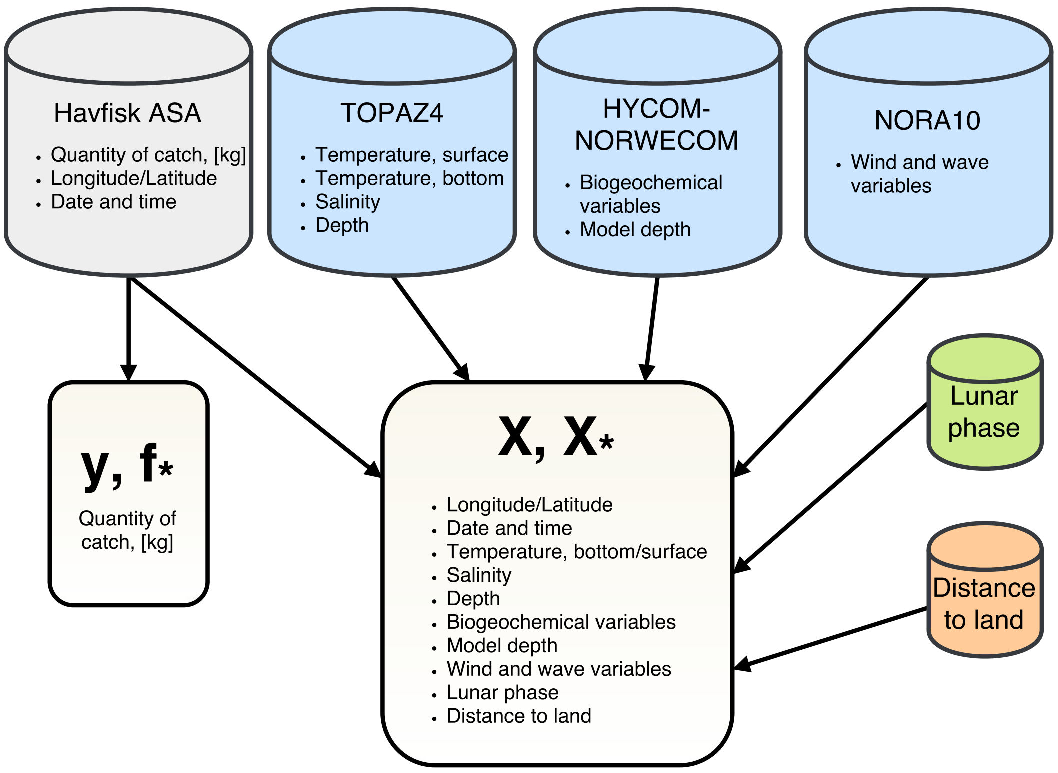


Figure 5.1: Visualization of how the different data sources are assimilated to an input training set, $\mathbf{X}$, and an input test/test set, X_*. The output training data, $\boldsymbol{y}$, and the output test/test data, f_*, are achieved from the Havfisk ASA dataset. The three blue boxes denote different models, i.e. Nora10, HYCOM-NORWECOM v2.1 and TOPAZ4, the grey box denote a database, the green and the peach coloured box denote additional features, while the beige boxes denote assimilated input and output data.

5.4 Combining the features to create datasets

All features, for each catch report in the database, were be aggregated, see Fig. 5.1, into the data matrices X, X_*. The data matrix is similar to the one shown in Fig. 5.2. Each row in Fig. 5.2 represents an observation, i.e. a catch report combined with the corresponding features from the NORA10, HYCOM-NORWECOM v2.1 and TOPAZ4 models and the two additional features (lunar phase and distance to land). Each column represents a specific feature. Thus, x_{12} denote feature 2 of catch report number

1, e.g. the daily mean of the salinity content in the ocean at date YYYY-MM-DD and time HH:MM:SS.sss at the position for which the first catch report was reported. In the same manner will x_{14} denote feature 4 of catch report number 1, e.g. the nitrogen content in the ocean at date YYYY-MM-DD and time HH:MM:SS.sss at the position for which the first catch report was reported etc.

A combination of all the features listed in Sec. 5.1, Sec. 5.2.1, Sec. 5.2.2, Sec. 5.2.3 and Sec. 5.3 result in 53 different features since the HYCOM-NORWECOM model provides the dataset with both a bottom and surface variable for each feature.

$$X, X_* = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix}$$

Figure 5.2: Visualization of how the data in the input data matrices X and X_* are structured.

5.5 Defining a baseline for data analysis

The following section will lead the reader through the different choices and decisions that were made for the first analysis of the collected variables that this thesis will perform. The different choices and decisions will in the end of this section define the baseline for the forthcoming analysis of the large available data material. The curious reader can consult Sec. 5.5.5 on page 53 for the full summary over the baseline, presented here.

5.5.1 The primary production and the chlorophyll features

Some initial tests were performed on the whole dataset of 51 features to find out if some of the features could cause any problem in the forthcoming analysing work. It turns out that the features "primary production bottom" and "primary production surface" will give rise to a lot of "linear algebra"-errors in the programming language Python. These both parameters will, therefore, be neglected from now on in the forthcoming analysis. The minimum, maximum and sample mean values of "primary production bottom" and "primary production surface" are all very small and close to zero, see Tab. 5.4, which possibly could cause the "linear algebra"-errors in Python.

Table 5.4: Table over the minimum, maximum and sample mean values for the variable Primary production surface/bottom.

Variable	Minimum value	Maximum value	Sample mean value
Primary production, surface	0	0.0005239400	0.0000192306
Primary production, bottom	0	0.0001016857	0.0000006195

Using both the feature “Chlorophyll bottom” and “Chlorophyll surface” will in the same manner as the “primary production” feature cause “linear algebra”-errors in Python. After some trial and error, it turns out that the use of an average value for the chlorophyll content solved the problem. The chlorophyll content in the ocean at each catch report and position i will from now on be defined as,

$$\overline{C}_i = \frac{1}{2}\Big(\mathrm{C}_{\text{bottom}, i} + \mathrm{C}_{\text{surface}, i}\Big). \tag{5.5.1}$$

5.5.2 The depth feature

Two different depth features was selected, see Fig. 5.1, see Tab. 5.1, and Tab. 5.2. The idea of using a depth feature is that it could give some additional information to the model regarding why fish species are located at specific places. Furthermore, it is possible that the model could distinguish between different fish species based on the depth of the ocean.

Some pre-inspection of the two depth features from the TOPAZ4 model and the HYCOM-NORWECOM model showed that the two features are relatively similar. Thus, the depth feature, that will be used from now on will be defined to be the average depth from the two models at each catch position. The average depth feature will be defined $\bar{D}_i$ for catch report i in the data base. it will be computed by,

$$\overline{D}_i = \frac{1}{2}\Big(\mathrm{D}_{\text{TOPAZ4}, i} + \mathrm{D}_{\text{HYCOM-NORWECOM}, i}\Big). \tag{5.5.2}$$

5.5.3 Defining the gear specification and a species of fish

There is a variety of different possible ways to investigate and analyse the whole collected dataset, but it will not be possible to do it all during the time limit of this thesis. Since the work in this thesis presents a first initial analysis of the collected dataset in a machine learning context, some limitations will, therefore, be defined here and used in the forthcoming analysis of this thesis. This implies choosing a species of fish and a specific gear specification.

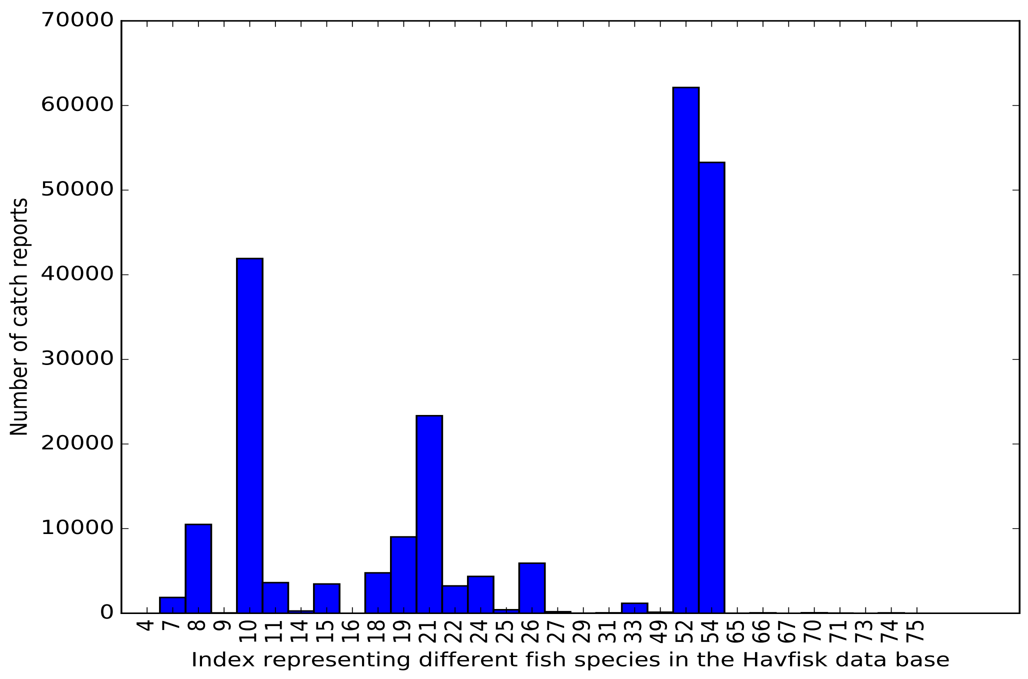


Figure 5.3: Histogram over the number of catch reports per species of fish represented in the database. The indices on the x-axis represents the same index that is used in the Havfisk database.

The collected dataset is large and consists of catch reports over many years and over 31 different fish species. The work presented in this thesis will only focus on one species of fish, mainly to narrow the scope but also because it is not known in advance if GP or other ML methods for regression performs best on a single species or a multitude of species. The histogram in Fig. 5.3 will be used to select which species of fish that will be the focus of the forthcoming investigations. Figure 5.3 shows a histogram over the different fish species in the database, where the x-axis represents the identification number of the different fish species, used in the Havfisk database. The y-axis represents the frequency over the number of catch reports for a specific species of fish. The quantity of catch per fish species is not taken into consideration in Fig. 5.3. The species of fish with identification number 52, i.e. North-East Atlantic Cod, will be considered in the forthcoming analysis since Fig. 5.3 shows that this fish species has the highest number of catch reports. This implies that there also will be more information about this species of fish compared to other species.

The North-East Atlantic cod are caught with single trawl or double trawl, where the double trawl potentially could yield a larger amount of catch compared with the single trawl. Single trawl was chosen arbitrarily since most of the fish catch of North-East Atlantic cod was caught with this gear specification. Figure 5.4 shows an example a single trawl.

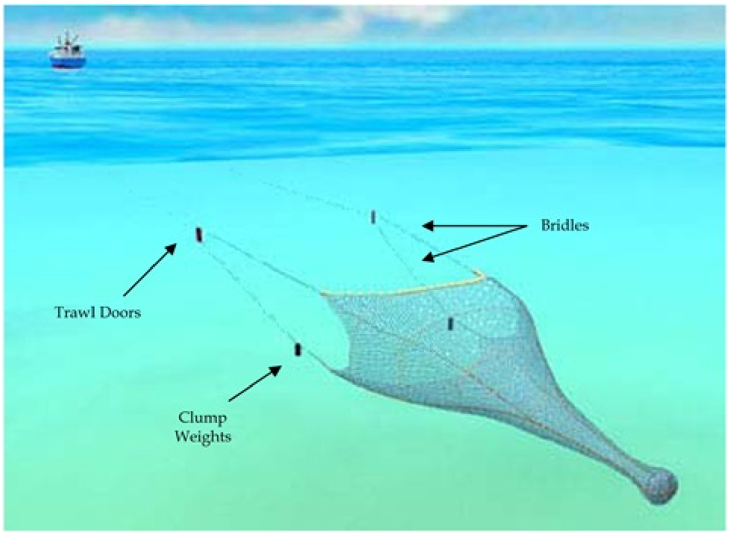


Figure 5.4: Example of a single trawl [4]

5.5.4 Defining a time scope

Some of the additional ML algorithms that are considered in this thesis, see for example Sec. 7.5 at page 76, had some numerical problems with datasets consisting of more than 18,000 data points. Therefore, the time scope of this thesis is defined to be the 18,000 first catch reports/observations of the years 2007-2011. The selected 18,000 catch reports are visualized on the map in Fig. 5.5 For visualization purposes was the quantity of catch transformed by taking its logarithm to the base of 10. The colourbar indicates the actual quantity of catch in kg. The figure indicates a higher quantity of catch in the upper half of the Norwegian Sea, and especially east of Bjørnøya.

5.5.5 The defined baseline

The following list summarizes the baseline for the forthcoming analysis,

- The both primary production features will not be considered in the forth coming analysis.
- The average chlorophyll content at each catch position for each catch report will be used from now on.
- The depth feature is defined in Eq. 5.5.2 and is the average of the depth in the TOPAZ4 and the HYCOM-NORWECOM model.

4. The image is retrieved from http://www.fao.org/fishery/topic/4080/en

- The three points above have decreased the number if features to 49 different features. The features that will be used in this thesis is listed in App. C.

- Only the 18,000 first catch reports over North-East Atlantic Cod (Skrei) caught with a single trawl during the years 2007-2011 will be considered.

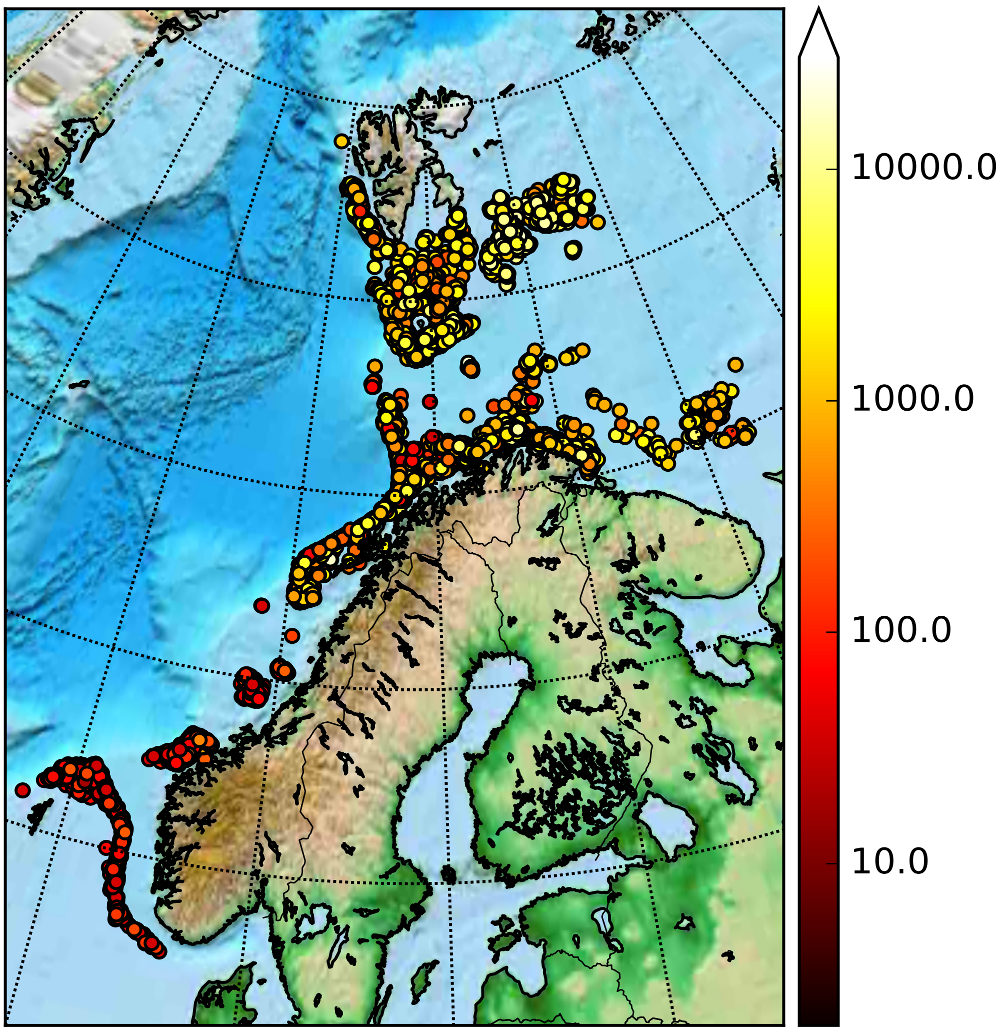


Figure 5.5: Map of all reported catch for the 18,000 observations of North East Atlantic Cod. The red/yellow marks denote one quantity of catch per position. Red indicates a lower quantity of catch while yellow indicates a higher quantity of catch

6
Preprocessing and assessment of the results

6.1 Data preprocessing and transformation

Machine learning algorithms, like the Gaussian processes, are used to automatically extract information from machine-readable datasets. The success of the ML algorithms usually depends on the quality of the data that is fed to the algorithm, where data of high quality will lead to results of high quality. Data preprocessing is an umbrella term that covers many different methods and techniques that all aims to increase the quality of the data before it is fed to the ML algorithm [43].

The measurement unit of the features in a dataset can affect the forthcoming data analysis. For example; a feature measured in kilograms or grams, or measured in meters or centimeteres, may lead to different results. *Data normalization* or *data transformation* are techniques that can be applied to a dataset to avoid dependence on the choice of measurement units [31]. The words transformation or normalization will in this thesis be used interchangeably in the same context.
Data normalization is performed on the dataset to scale the maximum and minimum values of the features to lower values such as [-1,1] or [0.0, 1.1]. This will, in general, give the feature a larger range [31]. The two most common methods for data transformation will be presented in what follows.

For the two transformations, let A denote a feature, which can be represented by numeric values and let $\boldsymbol{x} = [x_1, ..., x_n]$ be the vector of n observations of A. Furthermore, let $\boldsymbol{x}^{'}$ denote the vector of n normalized observations of A.

The **Min-max normalization** performs a linear transformation of the original observed values in $\boldsymbol{x}$. Let $\min_A$ denote the minimum value of A, and let $\max_A$ denote the maximum value of A. The min-max normalization maps a value $\boldsymbol{x}$ of A to $\boldsymbol{x}^{'}$, where all $x_i^{'}$ are in the "new" range $[\min_A^{'}, \max_A^{'}]$. The mapping is achieved by computing

$$\boldsymbol{x}^{'} = \frac{\boldsymbol{x} - \min_A}{\max_A - \min_A} \cdot (\max_A^{'} - \min_A^{'}) + \min_A^{'}. \tag{6.1.1}$$

The **zero-mean normalization** are a normalization that are based on the (sample) mean value of A, denoted $\bar{A}$ and the (sample) standard deviation of A, denoted s_A. The zero-mean normalization is defined as,

$$\boldsymbol{x}^{'} = \frac{\boldsymbol{x} - \bar{A}}{s_A}, \tag{6.1.2}$$

where the sample mean of the observations of feature A can be computed by

$$\bar{A} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{for } i = 1, ..., n. \tag{6.1.3}$$

Furthermore, the sample standard deviation, s_A, is the positive square root of the sample variance, that in this setting is given by

$$s_A^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{A})^2, \quad \text{for } i = 1, ..., n. \tag{6.1.4}$$

The both normalization techniques are achieved from, [43] and [31], which can be consulted for further preprocessing techniques.

6.2 Cross validation

Cross validation, CV, will be used to compare different data configurations or experiments with each other. This is a statistical method for comparing and evaluating the outcome of different machine learning models against each other. Cross validation partitions the dataset into parts, i.e. groups, where one part will be used to train the GP or ML regression model, and the other to validate i.e. test the model and its performance [66]. The three most common methods for CV are called the *Hold-out Validation*, *Leave-one-out Cross-validation* and *K-fold cross validation* [65, 66, 81].

The so-called *Hold-out Validation* it is both simple and fast [13]. In the hold-out validation method, the whole dataset is divided into two disjoint sets, a training and a testing set. The test will be completely independent of the training set, and will only be considered after the model has been trained. This method will only iterate through all points in the dataset once, which imply that it will be less time-consuming than the other two CV methods. The downside of this method is that it is highly dependent on the choice of the training/test splits [66]. Furthermore, a small test set can give rise to a performance estimate that may have large variance [65].

In the *k-fold cross validation*, abbreviated k-fold CV, the dataset consisting of n observations is partitioned into k equally sized, disjoint sets. k sequential iterations are performed, one for each of the k-folds. In each of the k iterations, the test set will consist of the kth k-fold, while the training set is the union of the other $k-1$ sets [66]. The procedure will be repeated k times, such that all k subsets will be used in both a training and test set [65]. Figure 6.1 shows an illustration of k-fold CV, where the data has been portioned into $k = 8$ k-folds. The grey box indicates which part of the data that is considered for testing, while the green boxes are combined into the training set.

The overall performance of the model is computed by, for example, the k-fold Bias, that is presented in the following section.

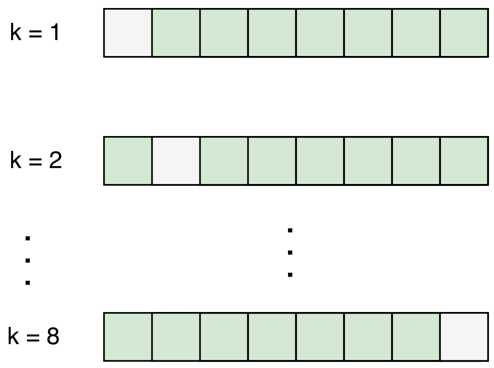


Figure 6.1: Illustration of k-fold CV, with $k = 8$ folds. The whole available data set consist of all the data in the eight boxes. The grey box indicate the part of the data that is considered for testing in one round, while the green boxes indicates the training data.

The *Leave-one-out Cross-validation* method, also denoted LOOCV, is a special case of the k-fold CV with $k = n$. This implies that n iterations are performed, one for each observation in the dataset. For each of the n iterations; one single observation at the time is considered to be the test set while the rest $n-1$ observations are considered to be the training set [66]. When n is a large number, this model selection method can come with a large computational cost [65]. Similar to k-fold CV, the overall performance of the model can be computed by for example the k-fold Bias.

LOOCV is a popular method when the size of the whole dataset is small [66], but performing k-fold CV is less time consuming than the LOOCV, as only k iterations are performed instead of N. It is assumed that $k \ll n$. This thesis will only consider *k-fold cross validation* since it would be to time consuming to perform LOOCV on the dataset that this thesis considers.

6.3 Performance validation

This section will define different validation methodologies that in Ch. 7-Ch. 9 can be used to evaluate the outcome from different outcomes from the k-fold CV. The following three statistical parameters,

- the root mean squared error, henceforth RMSE
- the Bias
- the standard deviation error, henceforth STDE

have been used in several different settings to evaluate the model performance, see for example [15, 47, 91], and will therefore also be used in this thesis.

For this section, let $\boldsymbol{f}_{*,k}$ denote the *true* output in the test set of the kth k-fold, thus $\boldsymbol{f}_{*,k} = [f(\boldsymbol{x}_{1*,k}), \ldots, f(\boldsymbol{x}_{j*,k})]$. Furthermore, let $\hat{\boldsymbol{f}}_{*,k}$ denote the estimated/predicted output of the GP or ML regression model, from the kth k-fold, i.e $\hat{\boldsymbol{f}}_{*,k} = [\hat{f}(\boldsymbol{x}_{1*,k}), \ldots, \hat{f}(\boldsymbol{x}_{j*,k})]$. The index j denote the size of a k-fold in both cases.
In addition, let $\boldsymbol{f}_*$ denote the *true* output from *all* the 600 k-folds, i.e. $\boldsymbol{f}_* = [f(\boldsymbol{x}_{1*}), \ldots, f(\boldsymbol{x}_{n*})]$, where $n = 18,000$ in this thesis. Furthermore, let $\hat{\boldsymbol{f}}_*$ denote the estimated/predicted output of the GP or ML regression model from *all* the the 600 k-folds, i.e. $\hat{\boldsymbol{f}}_* = [\hat{f}(\boldsymbol{x}_{1*}), \ldots, \hat{f}(\boldsymbol{x}_{n*})]$.

The overall RMSE can then be computed by

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{f}(\boldsymbol{x}_{i*}) - f(\boldsymbol{x}_{i*})\right)^2}. \tag{6.3.1}$$

A small overall RMSE, i.e. close to zero, indicates that the estimated output is close to the true output, while a larger overall RMSE indicates that estimated output deviates from the true output. The square root in Eq. 6.3.1 guarantees that the overall RMSE will have the same unit as $\hat{f}(\boldsymbol{x}_{i*})$ and $f(\boldsymbol{x}_{i*})$ [91].

An interpretation of the overall Bias or the k-fold Bias, abbreviated Bias$_k$ will be used to evaluate the data tendency. A positive overall Bias or Bias_k indicates that the predicted outputs tend to be an overestimate of the true output, while a negative overall Bias or Bias_k indicates that the predicted outputs tend to be an underestimate of the true output [15]. The overall Bias is defined as,

$$\text{Bias} = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{f}(\boldsymbol{x}_{i*}) - f(\boldsymbol{x}_{i*})\right), \tag{6.3.2}$$

while Bias_k is defined as,

$$\text{Bias}_k = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{f}(\boldsymbol{x}_{i*,k}) - f(\boldsymbol{x}_{i*,k})\right), \tag{6.3.3}$$

where N indicates the size, i.e. the number of observations/data points of a k-fold. Plotting a histogram over Bias_k for $i = 1, \ldots, k$ can give an indication of the distribution of the Bias for the k-fold CV of an experiment. Both the Bias and the Bias_k will have the same unit as $\hat{f}(\boldsymbol{x}_{i*})$ and $f(\boldsymbol{x}_{i*})$.

The overall STDE can be computed through the overall RMSE and the overall Bias in the following way,

$$\text{STDE} = \sqrt{(\text{RMSE})^2 - (\text{Bias})^2}. \tag{6.3.4}$$

The square root in the expression of the overall STDE ensures that it will have the same units as the RMSE and the Bias.

The overall RMSE, Bias and STDE will be used to compare the overall performance of different experiments and investigations. The overall STDE will be prioritised when different experiments are be compared to each other [15], since it both considers the error through the RMSE and the tendency through the Bias.

6.4 BIC

The Bayesian information criterion, abbreviated BIC, will be used when different GP models are compared with each other, see Sec. 8.3. The choice of using BIC is inspired by the work presented by Duvenaud [26] who chose to use this criterion because of its simplicity for model comparison, since it can be used to approximate the integral of the marginal likelihood over all free parameters in the GP model [71].

$$BIC_\rho = \log\left[\text{likelihood}(\text{optimized GP model})\right] - \frac{1}{2}\cdot\rho\cdot\log\left[n\right], \tag{6.4.1}$$

where ρ denote number of kernel parameters in the GP model, n denote the number of observations/data points in the dataset. BIC should penalize the marginal likelihood of a GP model in proportion to how many parameters the model has. The model with the highest BIC will be considered to be the optimal [26].

6.5 Comparing the quantity of catch

Two additional measurements will be considered in the forthcoming experiments an analysis in order to set the Bias_k introduced in Sec. 6.3 in context to the expected average quantity of catch in the k-folds. The actual average quantity of catch per k-fold, denoted $\text{Quant}_{\text{k-fold}}$ can be computed by the following,

$$\text{Quant}_{\text{k-fold}} = \frac{1}{k} \sum_{j=1}^{k} \sum_{i=1}^{N} f(\boldsymbol{x}_{i*,k}, \quad \text{for} j = 1, \ldots, k \text{ and } i = 1, \ldots, N, \tag{6.5.1}$$

where j denote the index of the kth k-fold, k is the number of k-folds and N is the number of observations/data points in a k-fold. Similarly can the average predicted quantity of catch per k-fold, $\widehat{\text{Quant}}_{\text{k-fold}}$ be computed by,

$$\widehat{\text{Quant}}_{\text{k-fold}} = \frac{1}{k} \sum_{j=1}^{k} \sum_{i=1}^{N} \hat{f}(\boldsymbol{x}_{i*,k}) \quad \text{for} j = 1, \ldots, k \text{ and } i = 1, \ldots, N, \tag{6.5.2}$$

recalling that $\hat{f}(\boldsymbol{x}_{i*,k})$ denote the predicted quantity of catch for observation i in k-fold k while $f(\boldsymbol{x}_{i*,k})$ denote the actual quantity of catch for observation i in k-fold k. Both $\text{Quant}_{\text{k-fold}}$ and $\widehat{\text{Quant}}_{\text{k-fold}}$ will in this thesis be given in kg.

Part III

Data analysis, feature selection and kernel design

7 Initial data analysis

In order to achieve god predictions using different ML methods and GP for regression, the dataset must possess a meaningful and informative structure for the method to be successful. A dataset without an underlying structure could result in poor predictions, as the data looks too similar to the regression method. This chapter will investigate different methods for understanding and extracting information from the collected data. Regression through SVI for GP models will be the considered regression method of this chapter, with a batch size of 1,000 observations and 200 inducing variables. The number of inducing variables and the batch size was arbitrary defined to be a fifth of what Hensman et al. [32] used in their experiment of airline delays. The investigations in this chapter will only consider the SE/RBF-kernel with a characteristic length scale in each feature dimension.

The term data configuration will be used as an umbrella term describing how the dataset can be modified in different ways. This chapter will focus on the following three topics *time dependency*, *clustering analysis* and *data transformation*.

The same first 18,000 catch reports of North East Atlantic Cod, with their corresponding 49 features, will be used through the whole chapter. For simplicity, the term *observation* will be used as a collective term that combines a catch report, i.e. the quantity of catch, longitude and latitude, with its corresponding physical, biogeochemical, wind and wave features.

The focus of this chapter is to establish an understanding of how the data should be modified and presented to the GP model in order to achieve as good predictions as possible. The aim with the different dataset configurations is to increase the accuracy of the predicted quantity of catch for an observation. This will be done by modifying the input data, i.e. the remaining features of the observation, in different ways. Common for all the experiments and analysis presented in this chapter is that they are all based on k-fold CV.

The number of k-folds will be set to $k = 600$, which implies that there will be 600 test sets, each consisting of $N = 30$ observations. All the 600 test sets are independent of each other, which implies that there will be 600 independent performance metrics. The value/size of k was chosen such that the distribution of Bias_k should, according to the central limit theorem (CLT), converge towards a normal distribution [90]. A histogram over the Bias_k, for $k = 600$, would under this assumption be bell shaped. A skewed histogram over the Bias_k will, on the other hand, violate the assumption that the Bias_k follows a normal distribution. By definition, this can indicate that there are some over or underestimation tendency in the predictions for a specific experiment, as discussed in Sec. 6.3.

During the 600 iterations of the k-fold CV, the GP model will be trained on 17,970 observations. The model is then used to predict quantity of catch for the remaining 30 observations. For consistency, the k-fold CV method with $k = 600$, will be used in the rest of this thesis so that the result from the predictions can be compared and evaluated with each other.

7.1 Investigating the time dependency between observations

This section will focus on the annual, monthly and daily pattern in the dataset. The aim of this section is to investigate if more accurate predictions could be achieved by preserving the time dependency between the observations. Investigations in this section will be performed in two parts; one where the time dependency between the observations is preserved and one where the order of the observations is randomized.
In the first part, the 18,000 observations were ordered in ascending time order from the 1st of January 2007 at time 00:00:000 until the most recent observation. The data was then partitioned into the 600 k-folds.
In the second part, the order of the 18,000 observations were randomized before they were divided into the 600 k-folds.
It should be mentioned that the time dependency is not perfectly preserved in the first part of the experiments, since the construction of the k-folds will introduce time gaps in the k-fold training datasets, as shown in Fig. 6.1.

The actual average quantity of catch per k-fold was computed separately for both cases to examine if there were any difference between the ordered or the randomized data. This was not the case, and therefore $\text{Quant}_{\text{k-fold}} = 70,334$ kg per k-fold will be used for both cases. This will be a reference value that the Bias_k can be compared to in both this section and the forthcoming sections of this chapter.

7.1.1 Result, investigating the time dependency

Figure 7.1 shows a visualization of the Bias_k, using both randomized and ordered training data in kg. The upper panel of Fig. 7.1 shows two Bias histograms, where the bins on the x-axis give an interval that represents the computed Bias for each of the k folds, the y-axis represents the occurrence of a specific Bias_k. The lower panel of Fig. 7.1 shows a scatter plot of the Bias_k for all the k-folds, for the both the randomized and ordered data.

The overall RMSE, Bias and STDE was computed by Eq. 6.3.1, Eq. 6.3.3 and Eq. 6.3.4,for the both parts of this investigation and are aggregated into Tab. 7.1. In the same manner was the average predicted quantity of catch per k-fold computed for the two cases by Eq. 6.5.2 and summarized in Tab. 7.1.

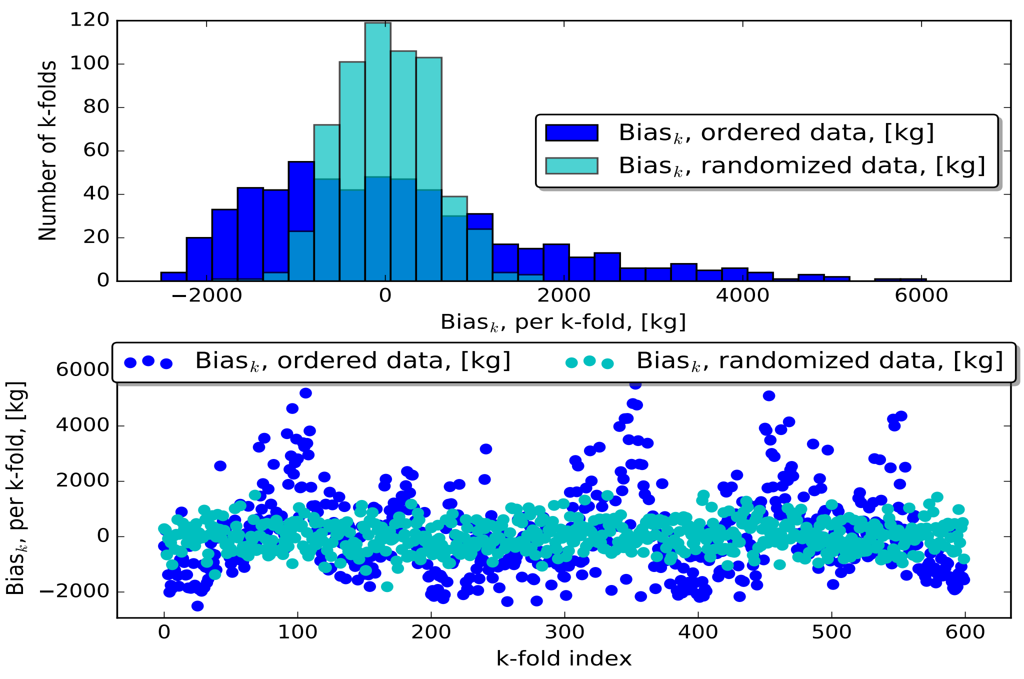


Figure 7.1: Upper panel: Histogram over the Bias_k from the two cases of the investigations of Sec. 7.1. Lower panel: Scatter plot over the Bias_k from the two cases. All values in the table are in the unit kg.

Table 7.1: The overall RMSE, Bias and STDE for the two cases in Sec. 7.1 are shown in row 1-3. The last row represents the computed average predicted quantity of catch per k-fold for the both cases in Sec. 7.1. All values in the table are in the unit, kg.

	Ordered data	Randomized data
Overall RMSE	2,917	2,342
Overall Bias	94	23
Overall STDE	2,916	2,342
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 67,527 kg/k-fold	Average 69,636 kg/k-fold

7.1.2 Discussion, investigating the time dependency

Although only 18,000 observations were used, the results in Tab. 7.1 indicate that it is better to present the data to the model in a randomized order, as the overall RMSE, Bias and STDE decreased when the order of the observations were randomized. Furthermore, the average predicted quantity of catch per k-fold, $\widehat{\text{Quant}}_{\text{k-fold}}$ for the randomized data only differs with 698 kg of fish from the $\text{Quant}_{\text{k-fold}}$-value of 70,334 kg/k-fold.

The both panels of Fig. 7.1 indicates that the variation in the Bias_k is larger when the data presented to the GP model is ordered in time. A randomization of the order resulted in a Bias_k distribution with both shorter tails and with an approximately bell-shaped structure. On the other hand, ordered input data results in a Bias_k distribution that is more skewed and with a larger variance. The computed Bias in Tab. 7.1 indicates that there is an overestimation tendency in the predictions for the both cases, and that is approximately four times larger for the ordered data which can explain the skewness shown in the upper panel of Fig. 7.1. The scatter plot in the lower panel of Fig. 7.1 indicates that there could be an underlying systematic error in the predictions, using ordered data, as the scatter plot almost indicate a periodic pattern.

The results of the investigations in this section indicate that it is better to present the data to the GP model in a randomized way. This could indicate that an observation from another period of the year, or another year, is more important to the GP model than the observations from the days before. In other words, preserving the annual, monthly and daily patterns will not increase the prediction accuracy. Even though the evidence of this section suggests that randomizing the input data leads to smaller Bias, STDE and RMSE, the forthcoming experiments of this chapter will include both ordered and randomized data to ensure that this is the case for other data set configurations.

7.2 Clustering and data transformation

In the previous section, all training data from each of the k-folds were used sequentially to train the GP model. There were no considerations taken regarding if some of the observations in the k-fold training set would be more or less important in the prediction.

This section and the following sections of this chapter will, therefore, focus on the following questions,

1. Is there a pattern or structure in the data?
2. Is it possible to achieve better predictions, using only the observations of the k-fold training data that are most similar to the k-fold test data?
3. Is it possible to perform a linear or non-linear mapping of the data from the high dimensional space to a lower dimensional space, and achieve better predictions in this low-dimensional space?

7.3 Revealing the structure of the data

This section will focus on the first question by visually interpret the whole dataset of 18,000 observations. As the data is of 49-dimensions, it will not be possible to visualize the high-dimensional data. It will, therefore, be natural to introduce a mapping, or transformation method here that will allow the data to be transformed to a 2-dimensional space for visualization.

7.3.1 Principal component analysis

One of the most used data transformation methods in the literature is the *principal component analysis*, PCA, that has been introduced in a multitude of textbooks, see for example [8, 11, 20, 39, 59, 81]. Principal component analysis is a popular method for feature generation, dimensionality reduction and data interpretation. PCA can also provide information regarding the variance-covariance structure of the original data [39, 81].

The basic idea of PCA is that the original p-dimensional data can be explained by a smaller set of p-components, i.e. new variables, that is a *linear* combination of the original variables [11], thus PCA is a *linear transformation method* [81]. The linear combination is chosen such that the resulting p-components are uncorrelated, and so that the variance of the mapping is maximized. In other words, PCA aims to keep low-dimensional, dissimilar transformed observations far away from each other [45]. This is achieved by considering the eigenvectors and eigenvalues of the covariance matrix of the input data, see [8, 81]. The reference literature listed above can be consulted for further insight into the PCA method.

7.3.2 Result and discussion of the PCA mapping

The 18,000 observations were transformed to a 2-dimensional space through PCA, that is visualized in Fig. 7.2. Figure 7.2 shows one compact cluster of transformed observations with some outliers. An evaluation of the outcome in Fig. 7.2 suggests,

- The data is non-linearly separable in the high-dimensional space.
- There is no underlying structure in the data.
- It is not optimal to transform the data through PCA to a 2-dimensional space.

A linear data transformation method is not expected to perform well on non-linearly separable data, and this can explain the unsatisfactory transformation result in Fig. 7.2. A reasonable procedure would be to apply a non-linear transformation to the original data, and check for structures in the non-linearly transformed data, before it is possible to suggest that there is no underlying structure in the data. If this also results in a transformed dataset with a non-visual structure it could suggest that it would be worth trying to transform the data to a 3-dimensional space with the PCA mapping to find out if this is more suitable for the data.

7.3.3 t-Distributed Stochastic Neighbor Embedding

The t-Distributed Stochastic Neighbor Embedding algorithm, also denoted t-SNSE, is a non-linear dimensionality reduction technique. The method models each high-dimensional point by a two or three-dimensional representation of the point, and can therefore only be considered for a transformation to a two or three-dimensional space. It ensures that similar objects in the high-dimensional space are modelled by nearby points in the two or three-dimensional transformation, while dissimilar points in the high-dimensional space are modelled far away in the transformation [45]. The algorithm is proposed in [45] and developed by Laurens van der Maaten and Geoffrey Hinton. An implementation of the software in multiple programming languages can be retrieved from `https://lvdmaaten.github.io/tsne/`. An additional implementation for the programming language Python is also provided by Scikit-learn [60]. The following references can be consulted for additional reading of the t-SNE algorithm [86, 87, 88].

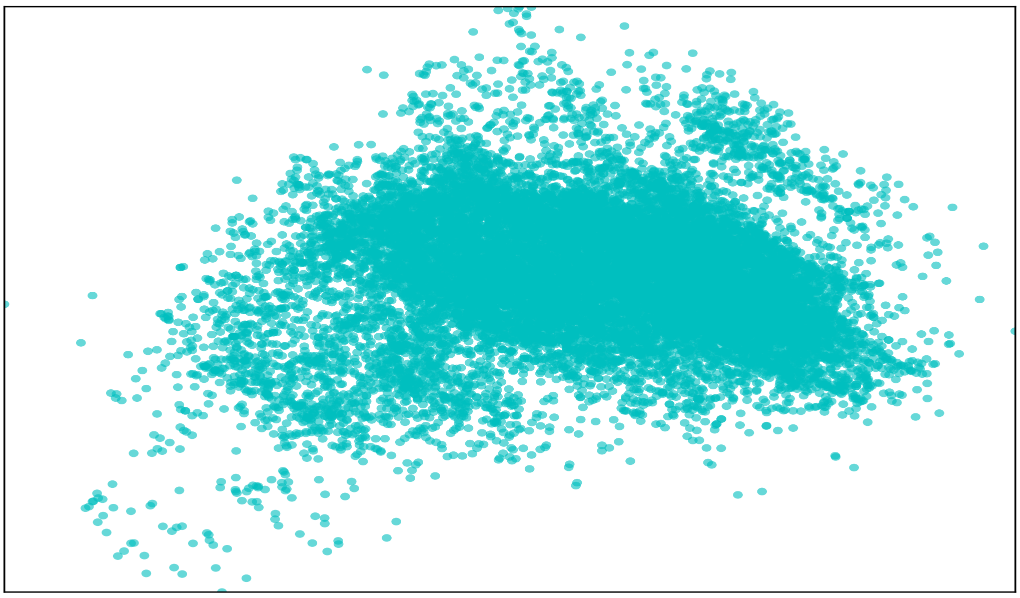


Figure 7.2: Visualization of the 18,000 observations after a transformation to the 2-dimensional space with PCA. Each of the points in the figure corresponds to an unique observation.

7.3.4 Result and discussion of the t-SNE mapping

The t-SNE algorithm was applied to the same 18,000 observations described above, for dimensionality reduction to the 2-dimensional space. It turned out that the t-SNE implementation, provided by the scikit-learn package for Python [60], was the best alternative for the dataset considered in this thesis. The resulting low-dimensional interpretation of the observations, using the scikit-learn package implementation for the t-SNE mapping, is visualized in Fig. 7.3.

Figure 7.3 indicates that there is an underlying structure in the high-dimensional data that have been preserved during the transformation. The results from the t-SNE transformation also indicate that the high-dimensional data is non-linearly separable. Because Fig. 7.3 is more informative than Fig. 7.2 will the forthcoming analysis only consider the t-SNE transformation of the data.

The interested reader should note that only the default parameters were used in the t-SNE mapping that is presented in this section and in Sec. 7.5. It is possible that the tuning of the parameters of the t-SNE algorithm could have resulted in a better mapping, see [87], but this was not considered as it is beyond the scope of this thesis.

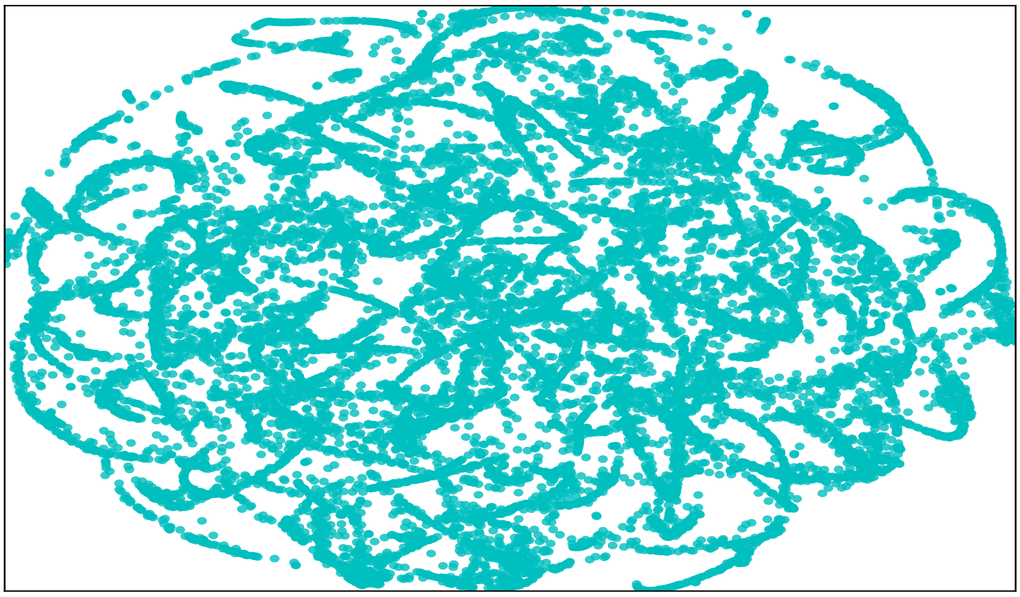


Figure 7.3: Visualization of the 18,000 observations after a transformation to the 2-dimensional space with t-SNE. Each of the points in the figure corresponds to an unique observation.

7.4 Clustering analysis

The second question listed in Sec. 7.2 will be investigated by clustering analysis of the training data. Clustering analysis arises from *unsupervised machine learning* and considers the task of separating a dataset into groups depending on similarities or dissimilarities between observations or points in the dataset [8]. The following references [25, 38, 81] can be consulted for an introduction to the majority of today's different clustering algorithms. Common for almost all clustering techniques is that they require that the number of clusters, c, is known in advance and/or a predefined similarity/dissimilarity measure [81].

The idea to tackle the second question of Sec. 7.2 is to sequentially cluster each of the 600 k-fold training sets into c different clusters. The cluster that is most similar to each of the 600 k-fold test sets will be defined to be the new training set. This procedure implies that the training sets in each of the 600 k-fold iterations will be smaller, i.e. consist of less than 17,970 observations, but hopefully more similar to the test sets. The forthcoming investigation will try to find out if the use of a training set similar to the test set results in better predictions, than using the whole, initial training sets. The following procedure explains how the *new* 600 k-folds training sets, referred to as the *k-fold cluster training sets*, was selected through k-means clustering.

Algorithm 1 Code

```
procedure K-MEANS CLUSTERING FOR K-FOLD CV
    for $i$ = 0 to 600 do
        % Cluster all observations of k-fold Training set i in c clusters,
        % Store the centroid coordinate to each cluster:

        cluster, centroid = kmeansClustering(kfoldTrainingSet[i])
        cent_val = mean(kfoldTrainingSet[i])   % Centre of kfoldTrainingSet[i]
        Dist = []   % Empty list
        Ind = 0   % Index
        for j = 0 to $c$ do
            % Find closest cluster to kfoldTestSet[i]:
            Dist[j] = EuclideanDist(centroid[j],cent_value)
            Ind = index(min(Dist))
        end
        New kfoldTrainingset[i] is closest cluster to kfoldTestSet[i],
        clust_kfoldTrainingSet[i] = cluster[:,Ind]
    end
```

The 600 clustered training datasets were then sequentially used as inputs to the GP model during training, before the GP model was used to predict the potential catch at the 600 different k-fold test sets. The investigation will, in conformity with the investigations in Sec. 7.1 be performed in two parts.

The *k-means algorithm* has been chosen for the investigation of the second question, due to its computational and conceptual simplicity [38]. The k-means algorithm is initialized by a predefined number of clusters, denoted c, and by an arbitrary definition of the mean estimates for the clusters, i.e. centroids. Each data point in the dataset is then defined to belong to the cluster for which the distance to its corresponding centroid is minimized. The squared Euclidean distance is used to measure the distance between centroids and the points that should be clustered [81]. Since the k-means considers the minimizing of a distance, the algorithm will in general work best for compact clusters [54].

The main drawbacks to the algorithm are that the number of clusters are needed as an input, and that the algorithm is sensitive to noise and outliers. A poor estimate of c could result in a poor clustering result, where the underlying structure of the dataset is not revealed properly. Outliers could, for example, influence the algorithm in such a way that it could form new clusters from the outliers [81].

7.4.1 Defining c

The so-called Elbow-criterion is a common rule of thumb method that can be used to find the optimal number of clusters within a dataset. The core of this method is a graphical pinpointing of the smallest number of clusters that exposes the variance of the data satisfactory [7]. The Elbow-method is conducted by plotting the percentage of variance explained by the clusters as a function of the number of clusters. The first clusters will explain a lot of the variance in the dataset, but at some point will the gain of adding another cluster yield an increase of explained variance. This point will result in an angle, i.e. an *elbow*, on the graph which indicates that the number of clusters should be chosen at this point [4, 7]. The drawback with this method is that the elbow cannot always be unambiguously identified [4].

7.4.2 Finding the number of clusters within the dataset

Since it would be too time-consuming to apply the Elbow-criterion to all the 600 different k-fold training sets, the Elbow- criterion was only applied to the whole training set of 18,000 observations once. The resulting number of clusters found in the whole dataset was then used for each of the 600 k-fold training sets in the both experiments. This procedure was considered to be appropriate, since the k-fold training sets are defined through the 18,000 observations.

The elbow-method was executed by sequentially increasing the number of clusters in the k-means algorithm from one cluster until 18 clusters. The maximum number of clusters, 18, was arbitrary set to a large number. The first elbow-graph that got a distinct elbow was kept, and this elbow was set to define the number of clusters for all of the 600 k-fold training datasets. Figure 7.4 shows the first elbow-graph with a distinct elbow, located at 6 clusters, while Fig. 7.5 shows the resulting six clusters plotted on the same t-SNE plot shown in Fig. 7.3. It should be pointed out that the Elbow-method had to be restarted several times, since the elbow was not always visible in the graph.

The number of clusters found was only evaluated on the whole dataset, when the number of observations was ordered by time in ascending order. For the sake of simplicity will also 6 clusters be adopted for the same data, when the order between the samples have been randomized, see Sec. 7.1. This decision seems appropriate as this section investigates in whether clustering analysis can be used to define the training data and not on finding the optimal number of clusters.

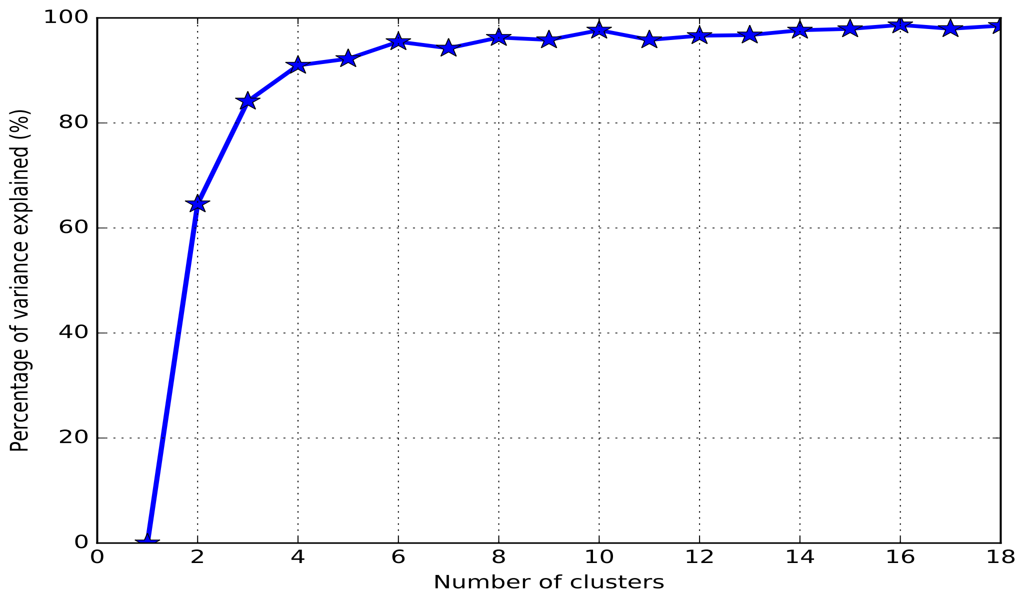


Figure 7.4: Using Elbow method to find out an initial number of clusters. The graph indicates that the optimal number of clusters for the 18,000 observations are 6.

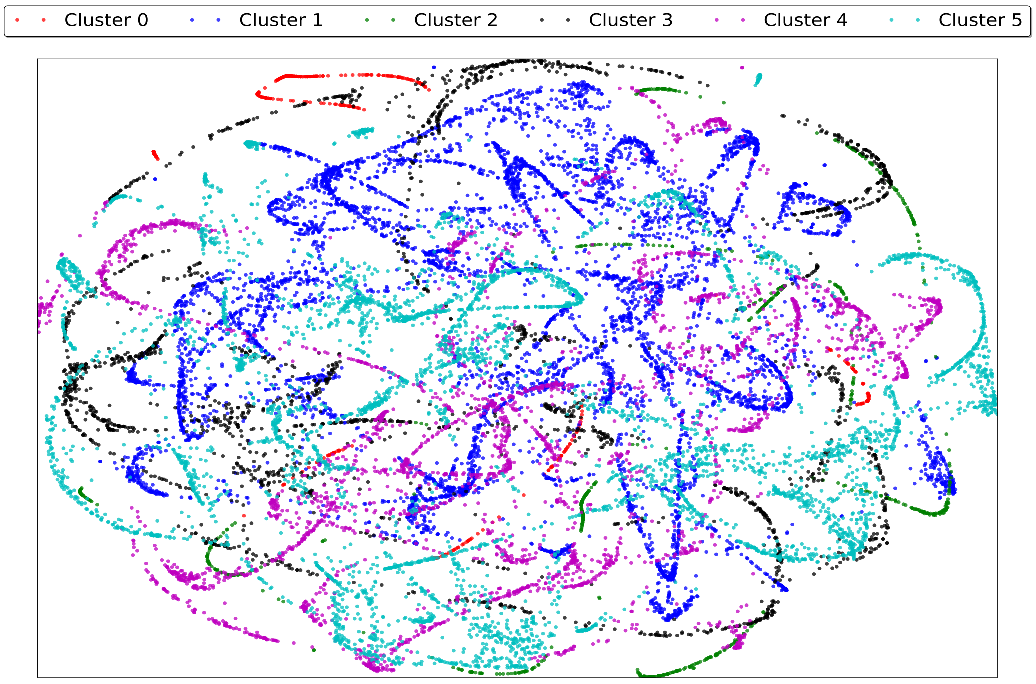


Figure 7.5: Visualization of the *optimal* 6 clusters of the 18,000 observations. The cluster affiliation is shown on the t-SNE transformed data shown in Fig. 7.3.

Table 7.2: Result of investigations in Sec. 7.4.3, where k-means was introduced to find the training data that was most similar to the test data. The second column shows the results when the time dependency was preserved. The third column shows the results when the time dependency was not preserved. All values in the table are given in kg.

	Ordered data	Randomized data
Overall RMSE	3,048	2,576
Overall Bias	-58	289
Overall STDE	3,048	2,560
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 72,072 kg	Average 61,661 kg

7.4.3 Results, using k-means to initialize the training data

The upper panel of Fig. 7.6 shows the histograms over the Bias_k per k-fold found in the investigations when clustered data was used as input to the GP regression model. The lower panel of Fig. 7.6 shows a scatter plot over the Bias_k for all the k-folds, for the both cases of the investigation.

Table 7.2 summarizes the overall RMSE, Bias and the STDE from the investigations of this section in combination with the $\widehat{\text{Quant}}_{\text{k-fold}}$.

7.4.4 Discussion the results of k-means clustering

The results in Tab. 7.2 indicates once again that the overall RMSE decreases in magnitude when the clustered training data is defined through randomized input data instead. Furthermore, the overall Bias indicates that the GP model tends to underestimate the predictions when the input data is ordered, while it tends to overestimate the predictions when the input data is randomized. It should be noted that the Bias is higher for the case with randomized data, though that the overall STDE prefers the randomized data.

Table 7.2 shows that the $\widehat{\text{Quant}}_{\text{k-fold}}$ value is closer to the actual average quantity of catch per k-fold, $\text{Quant}_{\text{k-fold}}$, for the case with ordered data. An explanation to this result could be that the clusters were defined through the ordered data, and not for the two datasets separately.

Figure 7.6 indicates once again that the distribution over Bias_k have a smaller variation and shorter tails for the case when randomized input data is used. Comparing the distribution of Bias_k for the ordered data in Fig. 7.6 with the corresponding distribution in Fig. 7.1 it should be noticed that the distribution in Fig. 7.6 seems more symmetric

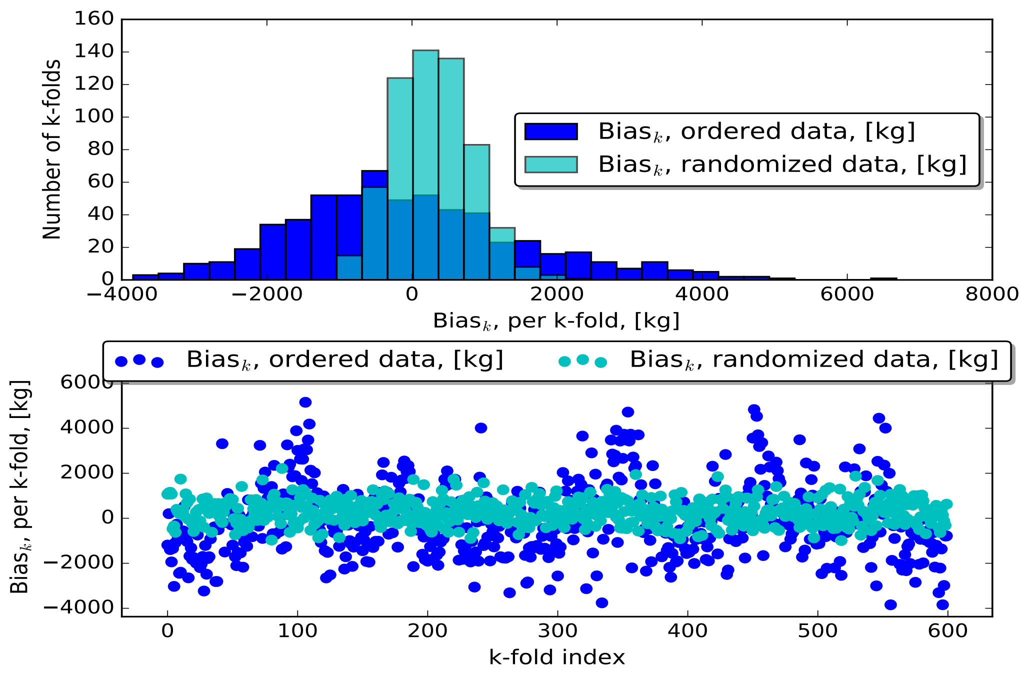


Figure 7.6: Upper panel: Histogram over the Bias$_k$ per k-fold from the two cases of the investigations in Sec. 7.4.3. Values on the x-axis represent the Bias$_k$ per k-fold in kg, per k-fold. Values on the y-axis sum the number of different k-folds that falls within a specific range of the Bias$_k$ per k-fold.
Lower panel: Scatter plot over the Bias$_k$ per k-fold for the two cases. Values on the x-axis denote which k-fold that is plotted, while the y-axis gives the Bias$_k$ per k-fold per k-fold. All values in the table are given in kg.

thought the longer tail on the right-hand side. This visual interpretation is also confirmed by the overall Bias that in magnitude has been halved in Tab. 7.2, using clustered input data, compared with the investigations in Tab. 7.1.

The scatter plot in the lower panel of Fig. 7.6 indicates once again that the distribution over Bias_k is more evenly and tighter distributed around zero, for the randomized data. The scatter plot of the Bias_k for the ordered data indicates once again some pattern or periodicity, that could indicate a systematic error.

Despite this, it is important to notice that the predictions performed in this section have been computed on a GP model that have been trained on a training dataset that is smaller than the 17,970 observations used in Sec. 7.1. Using fewer observations during the training of the GP model will reduce the time it takes to optimize the model, which can be preferable if the dataset is much larger than today's 18,000 observations.

7.5 Dimension reduction

This section will investigate the third and last question of Sec. 7.2 and will focus on the t-SNE algorithm to achieve the dimension reduction, since the investigations in Sec. 7.5.1 indicates that the data could be non-linearly separable. Data transformation through t-SNE is time-demanding, and therefore will the dimension reduction only be performed on the whole dataset of 18,000 observations and 49 different features. The transformation was performed to a 2-dimensional space to simplify the visualization of the data. A transformation to a 3-dimensional space was not considered in this section as the focus is on the concept of using transformed as input data to the GP model for achieving better predictions.

The transformed data was then partitioned into the 600 k-folds in the same manner that is described in the previous sections, i.e. using both ordered and randomized order of the transformed data.

7.5.1 Results, using t-SNE transformed data for dimension reduction

Table 7.3 summarizes the computed overall RMSE, Bias and the STDE in addition to the $\widehat{\text{Quant}}_{\text{k-fold}}$ values for the experiments in Sec. 7.5, considering both randomizes and ordered data.
Figure 7.7 visualizes the resulting histograms and scatter plots over the Bias_k.

Table 7.3: Result of performing predictions on transformed data, using t-SNE. All values in the table are given in kg.

	Ordered data	Randomized data
Overall RMSE	3,010	3,010
Overall Bias	0.0135	0.0235
Overall STDE	3,011	3,010
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 70,334 kg/k-fold	Average 70,333 kg/k-fold

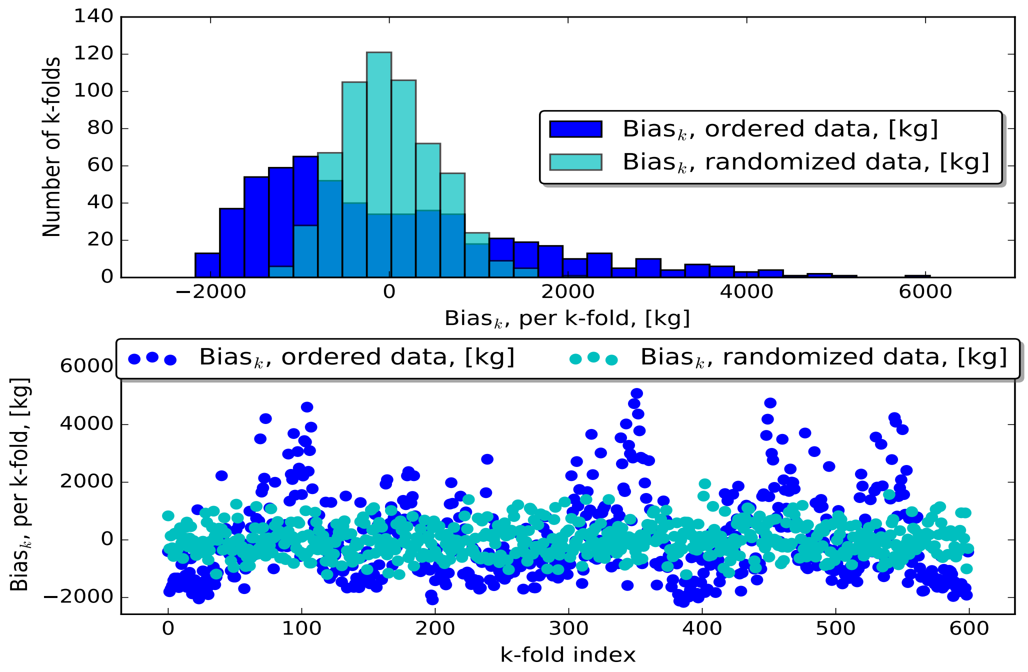


Figure 7.7: Upper panel: Histogram over the Bias$_k$ for the two cases of the investigations in Sec. 7.5.1. Values on the x-axis represent the Bias$_k$ in kg, per k-fold. Values on the y-axis sum the number of different k-folds that falls within a specific range of the Bias$_k$.
Lower panel: Scatter plot over the Bias$_k$ from the two cases. The values on the x-axis denote which k-fold that is plotted, while the y-axis gives the Bias$_k$ per k-fold. All values in the table are given in kg.

7.5.2 Discussion, using t-SNE transformed data for dimension reduction

An interpretation of the overall RMSE, Bias and STDE in Tab. 7.3 indicates that there are (almost) nothing to gain using ordered or randomized transformed input data. The exception is the Bias that is slightly lower in the case with ordered data. The overall STDEs are high for the both cases and indicates that the t-SNE transformation of the data to a 2-dimensional space is not the most proper choice for the given dataset. The high overall STDE could indicate that a lot of information in the original dataset is lost during the transformation to the 2-dimensional space through the t-SNE algorithm.

The two panels in Fig. 7.7 indicates that the distribution of the Bias_k is tighter and more bell-shaped for the randomized data than for the ordered data. It could be of interest to note that the resulting panels in Fig. 7.7 indicates that the computed Bias within each k-fold has a much higher variance than the overall Bias in Tab. 7.3. These conflicting results could indicate that the transformed input data is not optimal for the GP model, and/or that there is some systematic prediction error.

Furthermore, the interested reader may have observed that the predicted average quantity of catch within a k-fold $\widehat{\text{Quant}}_{\text{k-fold}}$ is identical to the true average quantity of catch within a k-fold, i.e. $\text{Quant}_{\text{k-fold}} = 70,334$ kg per k-fold. This is the best results so far and indicates the transformed data is very informative for the GP model under training and optimization.

The inconsistent and contradictory results of the investigations, using a 2-dimensional transformation of the input data motivates for some additional investigation in the results. The additional investigations will be presented in the forthcoming section.

7.5.3 Additional investigations

The first row of Tab. 7.4 summarizes the minimum and maximum values of the predictions, using the ordered or randomized input data from the t-SNE transformation. The second row in the same table summarizes the minimum and maximum values of the actual reported catch, the for intervals are based on all 18,000 observations. The two intervals from the predictions in Tab. 7.4 indicates that the GP model actually does not manage to capture the structure in the transformed input data as the predictions are far away from the actual quantity of catch. These problems are also illustrated in Fig. 7.8 where the actual reported quantity of catch in k-fold number zero is plotted against the predictions from the GP model using t-SNE transformed data. Figure 7.8 clearly shows that the prediction from the GP model are constant and that the GP model does not manage to capture any structure in the input data. The resulting predictions in Fig. 7.8 are based on the ordered data, but the from the randomized,

Table 7.4: First row: intervals for the minimum and maximum values of the predictions in Sec. 7.5. Second row: intervals for the minimum and maximum values of the actual reported quantity of catch. All values in the table are given in kg and are computed for all 18,000 observations.

Overall [min; max] quantity of catch	Ordered data	Randomized data
in predictions	[2,297; 2,376] kg	[2,155; 2,455] kg
in actual reported catch	[2; 30,539] kg	[2; 30,539] kg

transformed data show the same tendencies. K-fold number 0 was arbitrary chosen for visualization, but the same tendencies visualized in Fig. 7.8, can be found in the other 599 k-folds. The outcome of the additional testing in this section highlights the importance of thorough testing, not rushing to a premature conclusion based on a single test result. The predicted average quantity of catch in a k-fold, $\widehat{\text{Quant}}_{\text{k-fold}}$, is apparently not the best measurement to base the conclusions on, whereas the overall RMSE and the STDE seem to be better validation measurements.

This additional investigation indicates that no further focus should be put into the use of a t-SNE transformation of the data to a 2-dimensional space.

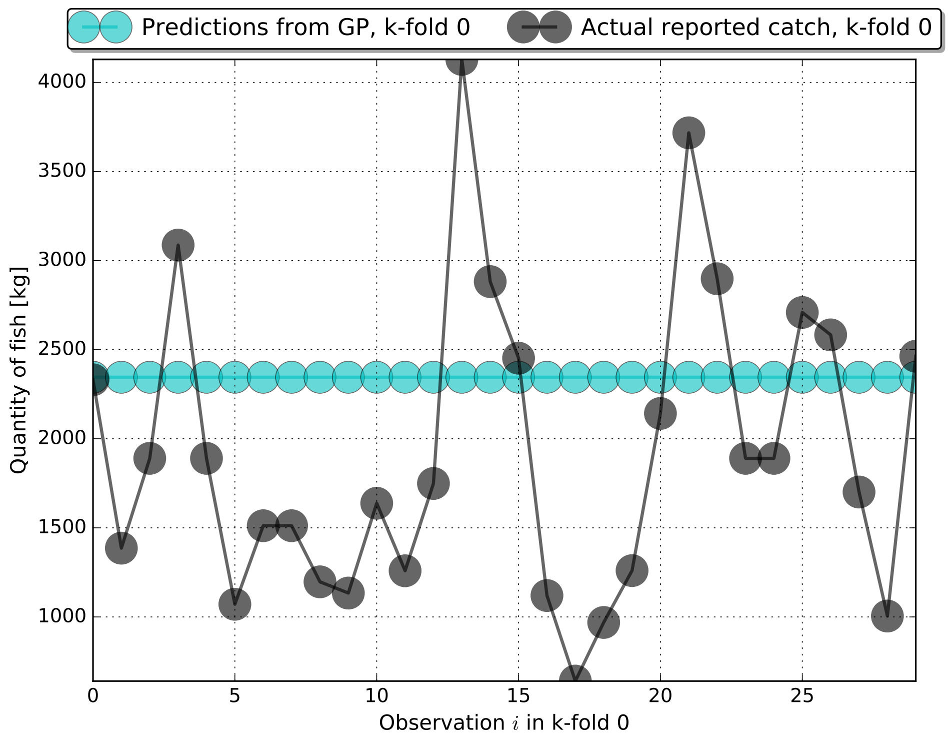


Figure 7.8: Visualization of the predictions from Sec. 7.5 for k-fold number 0. The black points indicate the true, reported catch wile the blue points indicates the predictions from the GP model for the ordered, transformed input data. The lines between the predicted/actual catch reports are only used to simplify the interpretation of the figure.

8 Feature selection

So far have all 49 features been represented in previous experiments, without considering if some of the features could be more or less important for the GP model. This chapter will, therefore, focus on investigating the features that have been assembled for this thesis through *feature selection*.

The sections considering feature selection will both try to find out if it is possible to find a selection of features that are more important for the model than the whole collection of the 49 features. These sections will also investigate how well the GPR performs, using only a selection of the most important features.

In order to compare the prediction results of this chapter with the experiments performed in the previous chapter, will this chapter also consider the same 18,000 observations, in addition to applying k-fold CV with 600 k-folds. The experiments of this chapter will also be performed in two parts, one where the time dependency between the observations are preserved and the other where the order of the observations are randomized before the k-folds are chosen. The same randomization will be used in this chapter so that the results from the randomized data in this chapter can be compared to the corresponding results from the previous chapter.

As a reminder to the reader, the actual average quantity of catch per k-fold is $\text{Quant}_{\text{k-fold}} = 70,334$ kg of fish, and will be referred to when comparing to the Bias_k.

Regression through SVI for GP is the only ML method for regression that will be considered in this chapter, with a batch size of 1,000 observations and 200 inducing variables. Only the SE-kernel with a characteristic length scale in each feature dimension will be considered, such that the results in this chapter can be compared with the previous chapter of data analysis.

Different features will mostly be referred to with a number in the range 0-48, as this is the indexing the programming language Python uses. The feature indices are preferred over the whole feature names to not confuse the reader with long feature names. A full list of the features, with their actual names, can be found in the appendix, see App. C.

8.1 Feature selection

Feature selection is an important part in many data analysis and ML tasks since feature redundancy and uninformative features can result in a ML model with poor performance. Thus, the work presented in this section will be an important part for understanding the assembled data. The following sections aims to investigate the following two questions,

1. which features are more important than the other?

2. will the predictions from the GPR model be more, less or equally accurate, using a fewer number of features for prediction?

Two different approaches to feature selection will be considered in the following two sections, i.e. the method of *Automatic relevance determination*, henceforth ARD, and the *Forward feature selection* algorithm, henceforth FFS algorithm. ARD was selected since it is a natural way of arranging the features after their importance when considering GP and kernels with a characteristic length scale per each feature dimension, see Sec. 3.2.3. The FFS algorithm for feature selection was selected both because it is traditionally more used, and known outside of the framework of GP [22].

The two following sections, Sec. 8.2 and Sec. 8.3 will both start with an introduction of the main concepts to the respective feature selection methods. After the introduction will the two sections continue with the actual feature selection, where the ten most important features will be selected among the whole set of 49 features.

The threshold was set to ten features since it corresponds to approximately 20% of all available features. It was important to not select too few or too many features, since selecting too few features could in worst case give the GP model to little information and result in bad predictions. Selecting too many features could on the other hand not ensure that the redundant and uninformative features have been sorted out.

The feature selection will in the both sections be performed on the whole set of 18,000 observations, since it would be to time-consuming to apply the feature selections on each of the 600 k-folds. The ten most important features will then be combined into a new optimal set of size $18,000 \times 10$. This optimal set can then be partitioned into 600 k-folds before they eventually can be presented to the GP model for regression. The two sections will individually discuss the benefits/disadvantages of using ordered or randomized data. Section 8.3.2 will conclude the feature selection part of this chapter by discussing and comparing the results from the two different feature selection methods with each other. The results of this chapter will be evaluated with the same concepts that were introduced in Sec. 6.3.

8.2 Automatic relevance determination

In the GP setting, the inverse characteristic length scales λ_d^{-2} in the squared exponential covariance function, Eq. 3.2.18, could be used for interpreting the relevance of each feature in the dataset, [56, 65]. This method is called automatic relevance determination, ARD, and evaluate the magnitude of the inverse λ_d^{-2}. A large length scale in specific feature dimension results in a covariance function that is almost independent of this feature dimension. These features can then, in the ARD setting, be interpreted to be almost unimportant. Similarly, a small length scale in specific feature dimension indicates that this feature plays a significant role for the covariance function and by that also a significant role in the prediction procedure.

This section will investigate if the method of ARD for selecting important features from the original dataset can result in better and more accurate predictions, than considering all features. A GP model will in the following section be trained and optimized on the whole dataset of 18,000 observations. The inverse characteristic length scales will then be interpreted, and the ten most important features, i.e. those with the smallest characteristic length scales will then be combined to a new optimal set.

It is important to notice that the input data to the GP model needs to be normalized so that the optimized length scales over each feature dimension can be interpreted in the same way.

8.2.1 Results, investigating if ARD gives better predictions

Figure 8.1 shows a histogram over the relative importances between the 49 different inverse characteristic length scales after optimization. Higher bars indicates that the corresponding feature is more important for the GP model, than the other features. The upper panel shows the histogram of the inverse characteristic length scales for the case with the ordered input data, while the lower panel of Fig. 8.1 shows the relative importance for the GP model trained on randomized input data. The purple bars indicates the ten most important features for each of the two cases.

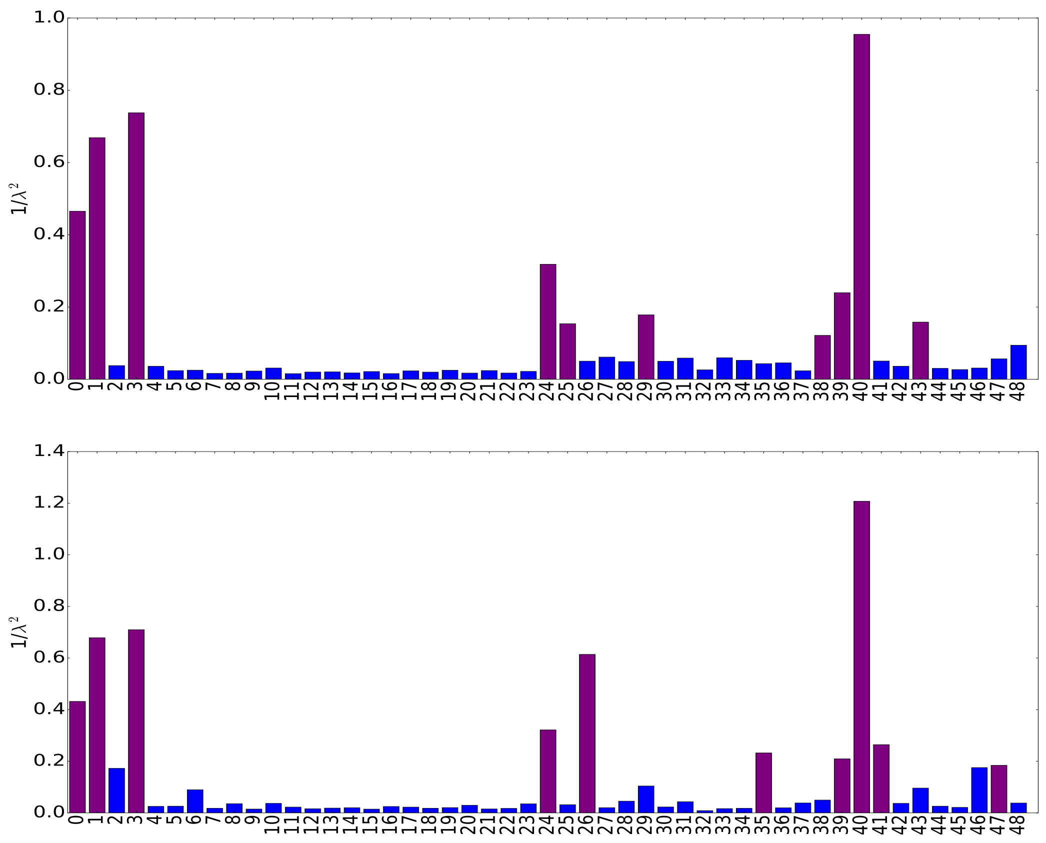


Figure 8.1: Visualization of the inverse length scales found by the ARD method for all features. The red bars indicates the ten most important features, according to the method of ARD. Upper panel visualizes the result for the ordered data while the lower panel visualizes the result for the randomized input data.

Table 8.1 list the ten most important features with their corresponding feature name in descending order, for both the case with ordered input data and randomized input data. The ten most important features from the ordered (randomized) data were then combined into a new optimal dataset of size $18,000 \times 10$, which then was then partitioned into the 600 different k-folds. K-fold CV was then applied to the

Table 8.1: Table representing the ten most important features, found by ARD. The most important feature is on the top and the 10th most important feature on the bottom.

Ordered data	Randomized data
40, MesozoonplanctonBottom	40 MesozoonplanctonBottom
3, Temperature	3, Temperature
1, Longitude	1, Longitude
0, Latitude	26, NitrogenBottom
24, DistanceToLand	0, Latitude
39, MesozoonplanctonSurface	24, DistanceToLand
29, SilicateSurface	41, DetriusSurface
43, DEPTSurface	35, OxygenSurface
25, NitrogenSurface	39, MesozoonplanctonSurface
38, MicroplanctonBottom	47, AvgDepth

Table 8.2: The overall RMSE, Bias and STDE in addition to the average predicted quantity of catch per k-fold, $\widehat{\text{Quant}}_{\text{k-fold}}$. The results are computed, using only the ten most important features during training and prediction. The ten features were achieved through the ARD method. All values in the table are given in kg.

	Ordered data	Randomized data
Overall RMSE	2,904	2,911
Overall Bias	53	8.4
Overall STDE	2,904	2,910
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 68,758 kg/k-fold	Average 70,082 kg/k-fold

two different optimal datasets and Tab. 8.2 summarizes the overall RMSE, Bias and STDE from the results in addition to the predicted average quantity of catch per k-fold, $\widehat{\text{Quant}}_{\text{k-fold}}$. A histogram over the distribution of the Bias_k for the two cases, are visualized in the upper panel of Fig. 8.2, while the lower panel of the same figure shows a scatter plot over Bias_k for each k-fold for the both cases.

8.2.2 Discussion, investigating if ARD gives better predictions

Many of the important features listed in Tab. 8.1 and visualized in Fig. 8.1 are represented as important when both ordered data and randomized data are used. The features *MesozoonplanctonBottom, Temperature* and *Longitude* are listed to be very important for the both cases. Furthermore, it is important to notice that the ARD method did not find any of the wind and wave features, index 6-23, or the lunar phase, index 5, to be of any relevance for the GP model, see Fig. 8.1.

The histogram over the Bias_k distribution in Fig. 8.2 deviates from the previous

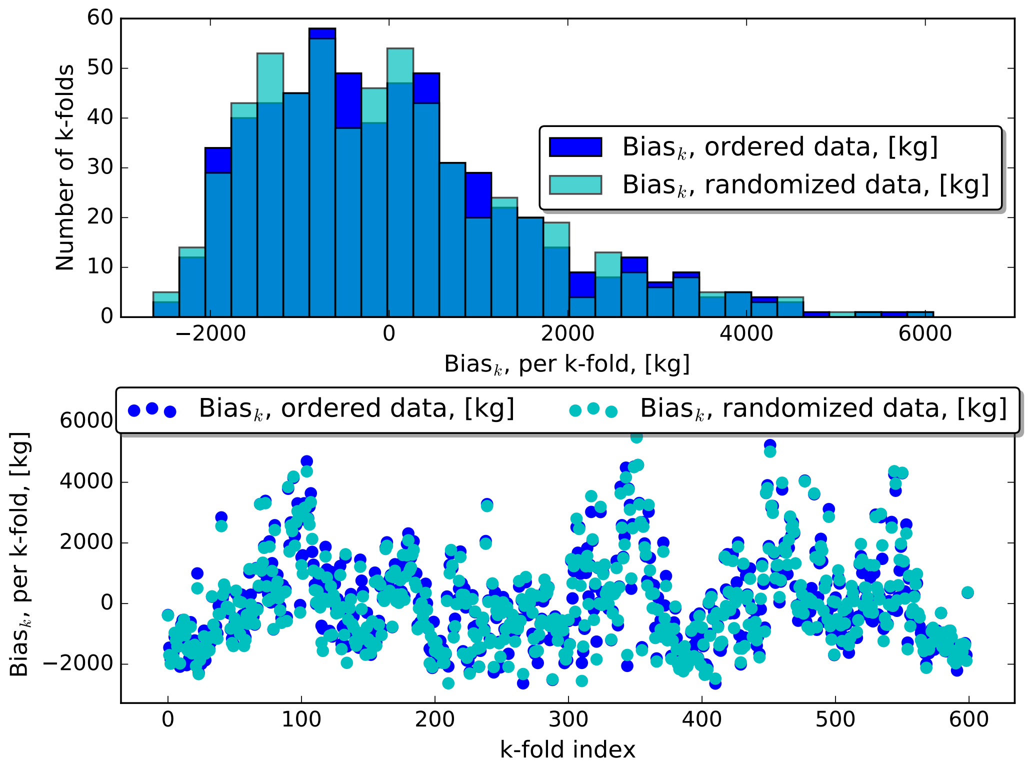


Figure 8.2: Upper panel: histogram over the distribution of Bias$_k$ for the both cases. The x-axis represents the value of Bias$_k$ in kg while the y-axis counts each k-fold that falls within a bin. Lower panel: scatter plot of Bias$_k$ per k-fold. The x-axis represents the k-fold index while the y-axis represents the average prediction error per k-fold.

histogram over the Bias_k distributions, see Ch. 7, since the Bias_k distribution over the randomized data is almost identical to the corresponding distribution of the ordered data. The both Bias_k distributions are skewed and have long tails, which could indicate that there is some systematic error in the predictions. An interpretation of the scatter plot in the lower panel of Fig. 8.2 also indicates that there could be a systematic error in the method where the ARD was used to select the ten most important features to the optimal input dataset. The scatter plot also indicates that the Bias_k for the randomized data does not follow the structure the can bee seen in the previous corresponding plots of Ch. 7.1.

Table 8.2 shows for the first time that overall RMSE is higher for the case with randomized data, than for the case with the ordered data. Despite this, it should be mentioned that the difference between the $\widehat{\text{Quant}}_{\text{k-fold}}$ and the $\text{Quant}_{\text{k-fold}}$ is only of 252 kg for the randomized data.

It is possible that the selection of the 20% most important features through the ARD method still causes redundant features, which can explain the poorer prediction results for especially the randomized data in this section. It could, therefore, be of interest to repeat the procedure explained in these sections but with a fewer number of important features, say the 10% most important features before the ARD method could be rejected in favour for another feature selection method. This will not be considered in this thesis due to the lack of time.

8.3 Forward feature selection

This section will focus on another conventional feature selection algorithm that in the literature is referred to as the *forward feature selection algorithm*, henceforth FFS algorithm [22]. The following references can, for example, be consulted for examples of the FFS algorithm, [22, 30, 41, 61]. The FFS algorithm is known as one of two basic *sequential* feature selection algorithms, the other is examined in the opposite way as the FFS algorithm and is therefore called the *backward feature selection algorithm*, BFS algorithm [61]. The FFS algorithm starts with one or zero features and selects one of the other possible features to the model according to a specific selection criteria. More features will then iteratively be added to the optimal set for the model, until a stopping criterion is met [44]. Features can only be added to the optimal set of features, and it is not possible to remove features from the optimal set during the FFS procedure. It is computational faster to optimize fewer features, especially in the case of GP with kernels that have one characteristic length scale for each dimension. This motivates the selection of the FFS algorithm over the BFS, which is initialized with all available features.

The FFS algorithm for GPR was implemented as a part of the work in this thesis since it was not possible to find any available implementation of the FFS algorithm in Python that was suitable for GPR. The algorithm assumes that the input training data is normalized to have zero mean and unity variance, while the test data is normalized to have zero mean. The selection criterion for selecting a feature or a combination of features over other features will be the Bayesian information criterion. In each round of the FFS algorithm will the BIC be computed for each possible candidate feature to the optimal set. The feature that receives the highest BIC in a round will be selected to the optimal set. A pseudo-code to the implementation of the forward search algorithm can be found in App. D.

The aim with this section is to find out if this additional feature selection method can result in other important features than the method of ARD. This section will, in addition, investigate if the important features, found by the FFS algorithm result in better, equally or worse prediction results than the method of ARD. In order to compare the FFS algorithm against the ARD method will the FFS algorithm terminate when the ten most important features are found. These ten features will then be combined into a new optimal set of $18,000 \times 10$ observations. In accordance with previous investigations will k-fold CV be used and the investigations will be performed on both randomized and ordered input data.

8.3.1 Results, investigating if FFS gives better predictions

Applying the forward search algorithm on the 49 features results in the ten most important features, shown in Tab. 8.3. The features are sorted in order so that the feature on the top of the list in Tab. 8.3 is the first one to be added to the optimal set, and the last feature on the list is the 10th and last feature to be added to the combination of the others in the optimal set. The left column shows the ten most important features for the first part of this investigation, when the order and time dependency is kept between the observations. The second column shows the result from the case where the order of the observations was randomized before they were introduced to the FFS algorithm. The important features were found by computing the BIC for each feature or each combination of the features in a round.

The two optimal sets found through the FFS were sequentially used for regression by SVI for GP, where k-fold CV once again was used on the 18,000 observations. The aim of the regression was to find out if the features found by FFS gave better, equally or worse predictions, compared with predictions achieved from the ARD method.

Table 8.4 summarizes the overall RMSE, Bias and STDE in addition to the predicted average quantity of catch per k-fold for the two cases. Figure 8.3 shows both a histogram visualization over the resulting Bias_k distribution and a scatter plot of the same distribution for the both cases.

Table 8.3: Table representing the ten most important features, found by the FFAS procedure. The most important feature is on the top and the 10th most important feature on the bottom. The features are given with both their defined number and the corresponding feature names, see App. C.

Ordered data	Randomized data
28, PhosphateBottom	35, OxygenSurface
3, Temperature	43, DEPTSurface
48, AvgChlorophyll	33, DiatomsSurface
19, StokesDriftXVelocity	32, FlagellatesBottom
7, MeanWaveDirectionWind	14, PeakWavePeriodSwim
33, DiatomsSurface	2, Salinty
6, MeanWaveDirectionSwell	0, Latitude
32, FlagellatesBottom	16, SignificantWaveHeight
4, BottomTemperature	48, AvgChlorophyll
14, PeakWavePeriodSwim	27, PhosphateSurface

Table 8.4: The overall RMSE, Bias and STDE and the $\widehat{\text{Quant}}_{\text{k-fold}}$, using only the ten most important features during training and prediction. The ten features was achieved through the FFS algorithm, and all values in the table are given in kg.

	Ordered data	Randomized data
Overall RMSE	2,890	2,537
Overall Bias	26	-49
Overall STDE	2,889	2,536
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 69,567 kg/k-fold	Average 71,817 kg/k-fold

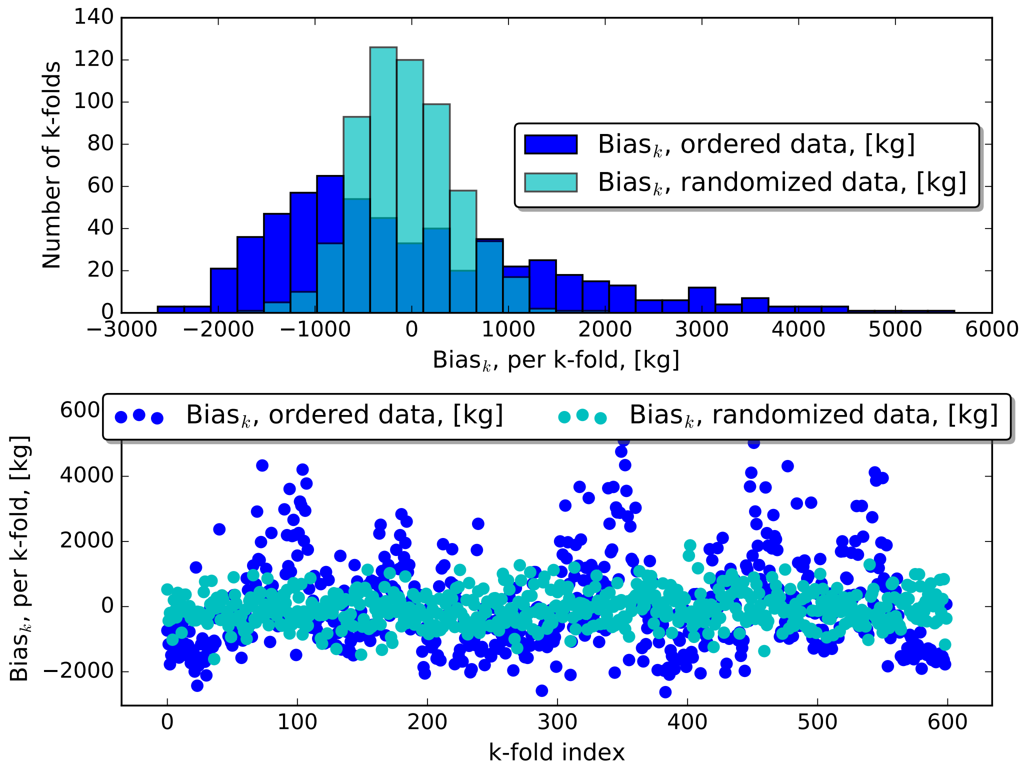


Figure 8.3: Upper panel: histogram of the distribution of Bias$_k$ for the both cases. The x-axis represents the value of Bias$_k$ in kg while the y-axis counts each k-fold that falls within a bin. Lower panel: scatter plot of Bias$_k$ per k-fold. The x-axis represents the k-fold index while the y-axis represents the average prediction error per k-fold.

8.3.2 Discussion the FFS results

The following four features *FlagelatesBottom, PeakWavePeriodSwim, DiatomsSurface* and *AvgChlorophyll* can be found in the both lists of Tab. 8.3, but are not selected to be important on the same round of the FFS algorithm. Besides these features, it can be of interest to note that neither the feature *Longitude* or the feature *Latitude* have been selected to be included in the optimal set of the ordered data. This implies that the results from the predictions presented in Tab. 8.4 and Fig. 8.3 are not based on the coordinates where the fish have been caught. In other words, the predictions for the ordered data presented in Sec. 8.3.1 are entirely based on features that describe the state of the ocean and not a position in the ocean.

The structure histogram in Fig. 8.3 reminds of the structure of the histograms in Fig. 7.1, Fig. 7.6 and Fig. 7.7 where distribution of the Bias_k for the randomized data clearly deviates from the distribution of the Bias_k for the ordered data. Similarly shows the scatter plot in the lower panel of Fig. 8.3 that the variance of the Bias_k is smaller for using the randomized data, than using the ordered data.

Table 8.4 is consistent with the results of Ch. 7.1 in the way that the overall RMSE decreases when the data is randomized. The overall RMSE in Tab. 7.1 is slightly better than the overall RMSE for the both cases in Tab. 8.4. Furthermore, the overall STDE is lower for the ordered data in Tab. 8.4, and only slightly higher for the randomized data, compared to Tab. 7.1. This is positive news since it indicates that when the ten optimal features are learned from the FFS algorithm, the accuracy in the overall RMSE will be comparable with the overall RMSE using five times as many features. It will also be computational faster to optimize a GP model with only ten features than 49 features.

The results and the resulting features from the method of ARD compared with the results and the resulting features from the FFS algorithm indicates that the FFS method for finding the ten most optimal features is a better feature selection method. The time costs of using the FFS algorithm is higher since a GP model have to be trained for each feature in each round of the FFS algorithm. This can be compared with the ARD method that only trains a GP model once and optimizes the 49 features once.

A reason to the different results, using feature selection through ARD or FFS, can be that the different models base their indication on an important feature on different criteria. The method of ARD provides information about which features that are more or less important for the GP model in the prediction. The features have been optimized such that the GP model returns the optimal predictions according to the marginal likelihood.

The features found by the FFS algorithm are, on the other hand, defined through the optimized GP model that yielded the highest BIC, among other optimized models. Thus, the BIC can not directly be used to interpret which features that are more or less important for a GP model as it only gives an overall assessment of the optimized model.

9 A performance comparison, focusing on an alternative regression strategy

So far has the focus of this thesis been on interpreting and evaluating the outcome from GPR for the collected data set. The previous chapters have demonstrated how the prediction error can increase or decrease with different data set configurations, and by considering a different number of features in the input data.

Many different studies have been performed in different research areas, see [10, 14, 17, 89], stating that GPs performs well in a regression context, compared to alternative strategies for regression. The work presented in this chapter will examine if this also is true for the data set collected and used in this thesis. This examination will compare the results of the best data set configuration, presented in this thesis, with another ML method for regression to find out how well GPs for regression performs.

The overall STDE indicates that the use of randomized data in Sec. 7.1 yields the so far best predictions, and this data set configuration will also be considered in the following investigation. Support vector machines, abbreviated SVM, will be the alternative regression strategy that will be considered in this section because it, in conformity with GPs, is a kernel based method [8]. The main concept of SVM for regression, abbreviated SVR [74], is to predict the input data through a function $f(\boldsymbol{x})$, that only allows to deviate with $\pm\epsilon$, at most, from the true targets or outputs

Table 9.1: The overall RMSE, Bias and STDE for the results of the randomized data in Tab. 7.1 and for SVR, using the same randomized data as introduced in Sec. 7.1. The last row represents the computed average predicted quantity of catch per k-fold for the both cases, all values in the table are given in kg.

	Results randomized data, Sec. 7.1	Results randomized data, using SVR
Overall RMSE	2342	3147
Overall Bias	23	1149
Overall STDE	2342	2930
$\widehat{\text{Quant}}_{\text{k-fold}}$	Average 69,636 kg/k-fold	Average 35,862 kg/k-fold

y_i. Principles of SVR are beyond the scope of this thesis, and will therefore not be introduced here. The interested reader can consult Refs. [8, 74, 76] for the principles of SVM and SVR.

The investigation in this section will also consider k-fold CV in the same setting as described in Sec. 7.1 so that the results can be comparable. Furthermore, the SVR model will also be defined with a SE/RBF-kernel such that this experiment will be in consistence with previous experiments. Tuning of the parameters of the SVR model, see Refs. [8, 74, 76], is not a part of the scope of this thesis, and therefore will only the default parameters be considered in the scikit-learn software package [60].

9.1 Evaluation of the performance comparison

Table 9.1 summarizes the results from the GP model in Tab. 7.1 for the randomized data with the corresponding results from the SVR model. Figure 9.1 visualizes the distribution of the Bias$_k$, in the upper panel, while the lower panel shows the scatter plots of the Bias$_k$ for the both cases. The results of Tab. 9.1 indicates that in overall will the predictions from the GP yield prediction estimates with an overall smaller error and Bias. Furthermore, the overall STDE is smaller for the results from the GP model, and the $\widehat{\text{Quant}}_{\text{k-fold}}$ is closer to the average quantity of catch per k-fold, i.e in average 70,334 kg of fish per k-fold.

Figure 9.2 visualizes the prediction outcome from k-fold 1, which was arbitrary chosen, for the performance comparison of this chapter. Black points in the figure indicate the actual reported catch in k-fold 1, light blue points indicate the predictions from the GPR while the blue points indicate the predictions from the SVR. The light blue region visualizes the 95% confidence region for the outcome from the GP model. The predictions are connected with line segments for the illustrative purpose. The outcome illustrated in Fig. 9.2 visualizes both that the GP model captures the structure

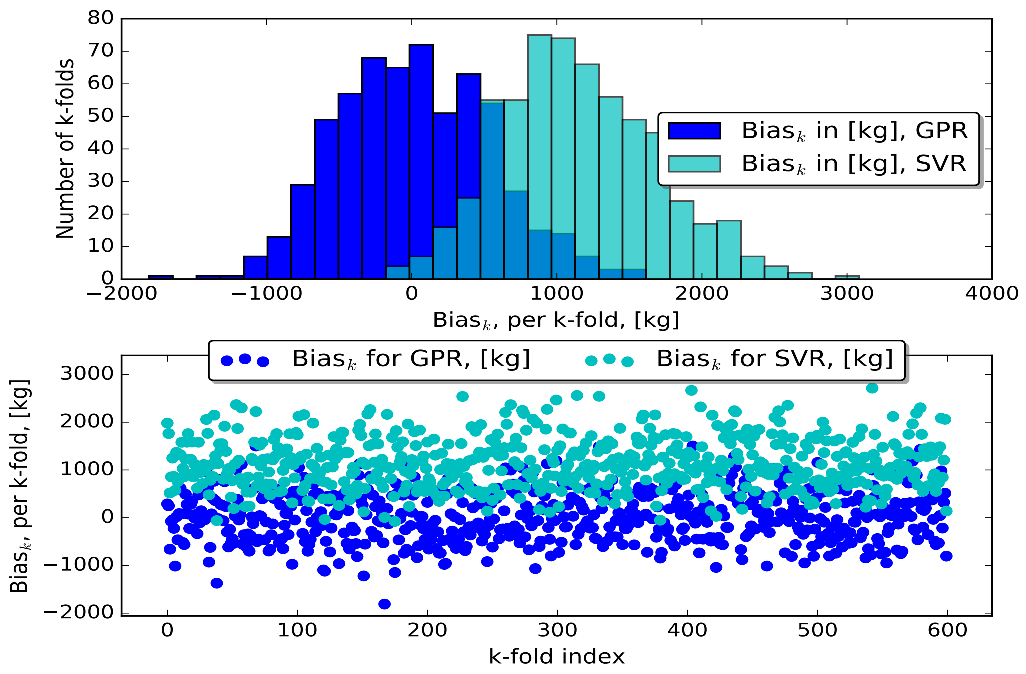


Figure 9.1: Upper panel: Distribution of Bias$_k$ per k-fold for the two experiments. Lower panel: Scatter plot of Bias$_k$ per k-fold for the two experiments.

in the input data and yield reasonable predictions, and GPR perfroms better than the SVR.

The computed overall from the SVR is the highest in all the tests that have been performed, and indicates that the SVM overestimates the predictions. These tendencies can also be seen in Fig. 9.1 which in both panels for the SVR predictions indicates an overestimation that in comparison with the Bias$_k$, computed for the GP predictions, is extra evident.

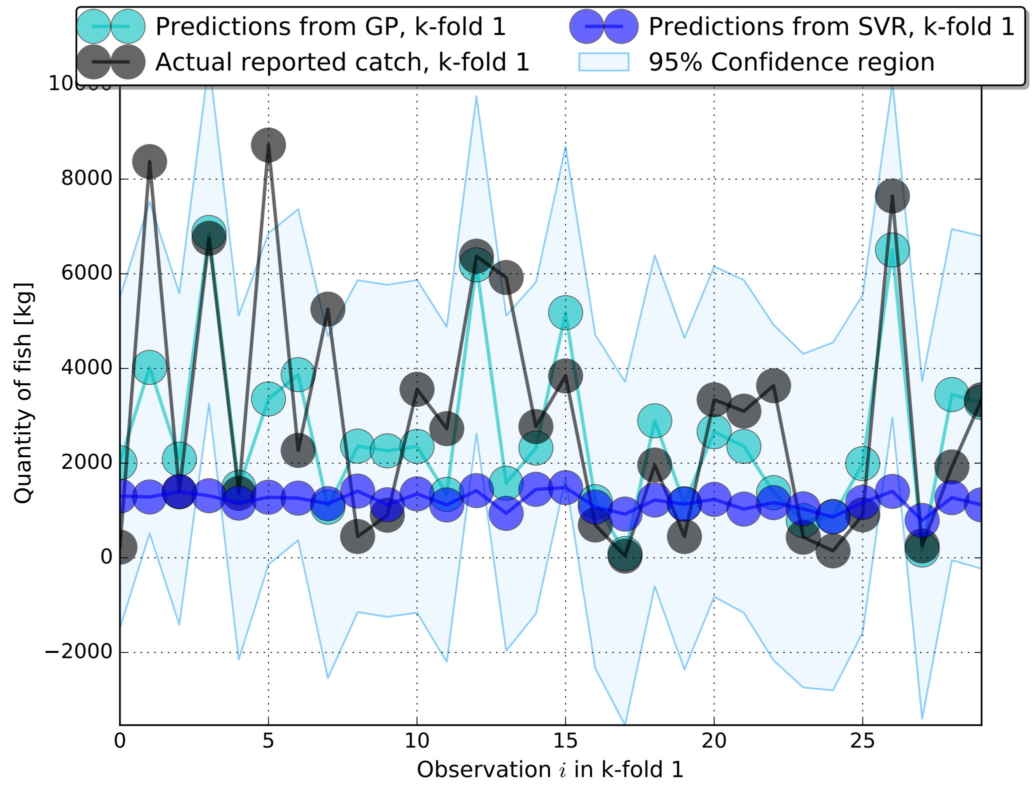


Figure 9.2: Visualization of the prediction for k-fold 1. Black points indicate the actual reported catch in k-fold 1, light blue points indicate the predictions from the GPR while the blue points indicate the predictions from the SVR. The light blue region visualizes the 95% confidence region for the outcome from the GP model.

Part IV

10 Concluding remarks

The focus of this thesis has been on investigating the possibilities of fish catch prediction using Gaussian processes. This work has for the first time combined catch reports from the Norwegian shipping company Havfisk ASA with a multitude of ocean-related data. GP for regression was considered, and this is the first time that the method of SVI for GP have been considered on the specific assimilated dataset.

In addition to initial investigations on the assimilated dataset, this thesis has also focused on the suitability of GPs for large datasets and to assess the predictions from the GP model. A summary of the main results and findings of the investigations presented in this thesis follows bellow, while Sec. 10.2 conclude this thesis with some recommendations for future work.

10.1 Summary

Different investigations have been performed to evaluate how the input dataset, used in the GP model, can be configured in the best way for achieving as accurate predictions as possible. All test except one, where ARD was applied for feature extraction, showed that the randomization of the input data resulted in a lower overall STDE compared to the use of input data that is ordered by time.

Using as large training dataset as possible (i.e. 17,970 observations and 49 features), and randomizing the order of the input data resulted in the lowest overall STDE, i.e.

STDE = 2,342. The second lowest overall STDE, i.e. STDE = 2,536 was achieved using the FFS algorithm to extract the ten most important features and perform predictions on an input dataset of size 17970×10 observations.

The investigations in this thesis indicate that SVI for GP is a suitable tool for performing regression in a GP setting on large datasets. Furthermore, only one test (using a t-SNE transformation on the input data) indicated that GP model did not catch the structure in the dataset, since it gave an almost constant estimation of the potential catch of fish independent of the structure of the input data. Thus, the investigations of this thesis indicate that GP for regression does reveal a structure in the dataset and manages to capture this to perform predictions that are non-random. A performance comparison between GPR and SVR indicated that, in accordance with the literature, that GPR does yield a lower overall STDE when the same datasets were considered.

10.2 Recommendations to future work

The scope of the performed investigations was wide, in order to investigate as many topics as possible during the time limit of this thesis. Still, there are numerous of interesting cases and things that could have been investigated further. This section will list some of the topics that could be considered in a future analysis, either for improving the GP regression model or to investigate further some of the topics discussed.

- **Kernel design**
  It has been mentioned in the thesis that the selection of an appropriate kernel (or combination of kernels) is important to achieve accurate predictions. The work presented by Duvenaud [26] for kernel design is very promising and should be considered in future investigations.

- **Additional data**
  After the analysis of the data considered in this thesis began, more recent data has been available. It is of interest to determine if the accuracy in the predictions increases if more data are used during the training and the optimization of the GP model. Furthermore, the scope of this thesis was set to one species of fish and a specific gear specification. Additional investigations could focus on investigating if the GP model is sensitive for different species of fish and/or different gear specifications.

- **Features**
  It is a well-known fact that the pressure changes with altitude, so does also the pressure below the sea surface [55]. The use of pressure as an addition feature could be useful to determine if different fish species can be found on different

altitudes etc.

Moreover, most of the features considered for this thesis were achieved from different models over areas close to the reported catch positions. It is interesting to investigate the possibility to equip the vessels with different sensors that could log (almost) the same features, but at the exact positions at which the fish is caught. This will give features that more exactly describes the state of the ocean. It is also possible that it could be both less time consuming and more economic to process data from “one” source instead of assimilating data from at least four different oceanic models with a catch database, as done for the preparatory work of this thesis.

In addition, many different reasons can explain why a vessel reports lower or higher catches for specific species of fish than usual. The use of historical and today’s Norwegian fish quotas in combination with the input data could be a good supplement to achieve good predictions.

- **Additional feature analysis and selection**
  The feature analysis and selection part of this thesis indicated that there are potential in using feature selection to find features that are more relevant than others, and to achieve predictions that are almost as accurate as using all accessible features. More effort should be put into additional feature analysis and selection. The sensitivity study of GP for oceanic chlorophyll content, presented by Blix et al. [10], could be an interesting method to consider and to compare to the ARD method and the FFS algorithm presented in the thesis.

- **Clustering and data transformation**
  The work presented did only consider PCA and t-SNE transformation of the data to a 2-dimensional space. Further investigations could for example be put into the 3-dimensional space. In addition, alternative data transformation methods could result in better results, where the *Kernel PCA* method [70] and the method of *Laplacian eigenmaps* [5] are two suggestions.

  Moreover, an inspection of the visualized t-SNE transformation, see Fig. 7.3, suggest that the selected k-means clustering algorithm may not be the best choice for the data since it does not possess compact clusters.

- **Catch positions**
  The FFS algorithm, applied on the ordered data, resulted in a lower overall STDE compared to the corresponding experiment of Sec. 7.1, although ten features were used. The predictions from the FFS algorithm for the ordered data did not consider the coordinates. This motivates an investigation in how important the coordinates in the ocean are for good predictions.

- **Additional recommendations to future work**
  - Investigate the possibilities to achieve better predictions, using for example SVR when the parameters of the model are tuned and optimized.
  - Investigate in what causes the GP model to perform very well on some predictions and worse on other.

Appendices

A

Derivation of the bivariate Gaussian distribution

The multivariate Gaussian distribution in Eq. 3.2.1 can be used to achieve an expression for the bivariate Gaussian distribution, $f(x_1, x_2)$. Let the covariance matrix between two random variables X_1 and X_2 be represented by:

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix},$$

where $\sigma_{11} = \mathrm{Var}(X_1)$, $\sigma_{22} = \mathrm{Var}(X_2)$ and $\sigma_{12} = \mathrm{Cov}(X_1, X_2)$. σ_{11}, σ_{12}, σ_{21} and σ_{22} represents scalar values. Due to symmetry will $\sigma_{12} = \sigma_{21}$. The (linear) correlation between X_1 and X_2 can be represented by,

$$\mathrm{Corr}(X_1, X_2) = \rho_{12} = \sigma_{12}/\sqrt{\sigma_{11}\sigma_{22}}. \tag{A.0.1}$$

Solving Eq. A.0.1 with respect to σ_{12} allows the following formulation of the covariance function: $\sigma_{12} = \rho_{12}\sqrt{\sigma_{11}\sigma_{22}}$. Furthermore, let the expectations of X_1 and X_2 be represented by the following vector,

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}.$$

The inverse of any 2×2 matrix **A** is given by [39],

$$\mathrm{A}^{-1} = \frac{1}{|\mathrm{A}|}\begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}, \tag{A.0.2}$$

where $|.|$ denote the determinant.

Finding the determinant of $|\Sigma|$ for the bivariate Gaussian distribution,

$$\begin{aligned} |\Sigma| &= \sigma_{11}\sigma_{12} - \sigma_{12}^2 \\ &= \sigma_{11}\sigma_{12} - \left(\rho_{12}\sqrt{\sigma_{11}\sigma_{22}}\right)^2 \\ &= \sigma_{11}\sigma_{12} - \rho_{12}^2\sigma_{11}\sigma_{22} \\ &= \sigma_{11}\sigma_{12}(1-\rho_{12}^2). \end{aligned} \tag{A.0.3}$$

Thus, the inverse of the covariance matrix can then be expressed with help of Eq. A.0.2,

$$\Sigma^{-1} = \frac{1}{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}\begin{bmatrix} \sigma_{22} & -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}} \\ -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}} & \sigma_{11} \end{bmatrix}. \tag{A.0.4}$$

Computing the argument of the expectation in Eq. 3.2.1: $(x-\boldsymbol{\mu})^T\Sigma^{-1}(x-\boldsymbol{\mu})$,

$$(\mathrm{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathrm{x}-\boldsymbol{\mu}) = \left[x_1-\mu_1, x_2-\mu_2\right]\frac{1}{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}\times$$

$$\begin{bmatrix} \sigma_{22} & -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}} \\ -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}} & \sigma_{11} \end{bmatrix}\begin{bmatrix} x_1-\mu_1 \\ x_2-\mu 2 \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_{22}(x_1-\mu_1) & -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}}(x_2-\mu_2) \\ -\rho_{12}\sqrt{\sigma_{11}\sigma_{22}}(x_1-\mu_1) & \sigma_{11}(x_2-\mu_2) \end{bmatrix}\times$$

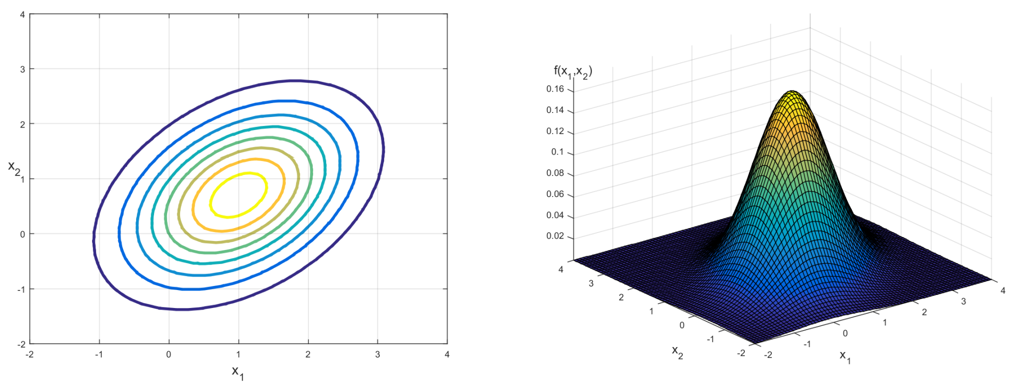
$$\begin{bmatrix} x_1-\mu_1 \\ x_2-\mu 2 \end{bmatrix}\frac{1}{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}$$

$$= \frac{\sigma_{22}(x_1-\mu_1)^2 - 2(x_1-\mu_1)(x_2-\mu_2)\rho_{12}\sqrt{\sigma_{11}}\sqrt{\sigma_{22}} + \sigma_{11}(x_2-\mu_2)^2}{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}. \tag{A.0.5}$$

The argument of the exponential term can then be expressed in the following way, after some simplifications,

$$(\mathrm{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathrm{x}-\boldsymbol{\mu}) =$$

$$\frac{1}{(1-\rho_{12}^2)}\left[\left(\frac{x_1-\mu_1}{\sqrt{\sigma_{11}}}\right)^2 - 2\rho_{12}\left(\frac{(x_1-\mu_1)(x_2-\mu_2)}{\sqrt{\sigma_{11}\sigma_{22}}}\right) + \left(\frac{x_2-\mu_2}{\sqrt{\sigma_{22}}}\right)^2\right]. \tag{A.0.6}$$



(a) Contour plot. **(b)** Surface plot.

Figure A.1: Two plots visualizing a bivariate Gaussian distribution.

Hence, the bivariate Gaussian distribution $f(x_1, x_2)$ can then be formulated, with help of Eq. A.0.3 and Eq. A.0.6 in the following way:

$$f(x_1, x_2) = \frac{1}{2\pi\sqrt{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}} \times$$

$$\exp\left(-\frac{1}{2(1-\rho_{12}^2)}\left[\left(\frac{x_1-\mu_1}{\sqrt{\sigma_{11}}}\right)^2 - 2\rho_{12}\left(\frac{(x_1-\mu_1)(x_2-\mu_2)}{\sqrt{\sigma_{11}\sigma_{22}}}\right) + \left(\frac{x_2-\mu_2}{\sqrt{\sigma_{22}}}\right)^2\right]\right). \tag{A.0.7}$$

The left panel of Fig. A.1 shows a contour plot of a bivariate Gaussian distribution while the right panel of the same figure shows a surface plot of the same distribution. The expression in Eq. A.0.7 was used for plotting. The parameters of the bivariate distribution was chosen in the following way,

$$\mu_1 = 1, \mu_2 = 0.7, \sigma_{11} = \sigma_{22} = 1 \text{ and } \sigma_{12} = 0.4.$$

B The conditional distribution of f_* given f

The conditional distributions of f_* given $\mathbf{f}$ is based on a well known result for Gaussian distributions, that can be found in [39]. The full result will be restated here:

Let $Z = \left[\begin{array}{c} Z_1 \\ \hdashline Z_2 \end{array}\right] \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma)$, where p denotes number of features of Z.

Furthermore, let the mean vector of Z be formulated by: $\boldsymbol{\mu} = \left[\begin{array}{c} \boldsymbol{\mu}_1 \\ \hdashline \boldsymbol{\mu}_2 \end{array}\right]$,
and let the variance-covariance matrix of Z be formulated by:
$\Sigma = \left[\begin{array}{c:c} \Sigma_{11} & \Sigma_{12} \\ \hdashline \Sigma_{21} & \Sigma_{22} \end{array}\right]$, with $|\Sigma_{22}| > 0$.

The conditional distribution of Z_1, given that $Z_2 = z_2$, will follow a Gaussian distribution with,

$$\begin{gathered} \text{Mean} = \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(z_2 - \boldsymbol{\mu}_2) \\ \text{and} \\ \text{Covariance} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \end{gathered} \tag{B.0.1}$$

The result given in is generalizable for any multivariate Gaussian distributed random matrix of vectors, $\boldsymbol{X}$. This section will show that the multivariate result given in the previous section will be true for a bivariate Gaussian distribution. The conditional density of X_1, given that $X_2 = x_2$, for any bivariate distribution is defined by:

$$f(x_1|x_2) = \frac{f(x_1, x_2)}{f(x_2)}, \tag{B.0.2}$$

where $f(x_1, x_2)$ denote the joint (bivariate) density of $f(x_1)$ and $f(x_2)$, $f(x_2)$ denote the marginal distribution of X_2. It will be assumed that $f(x_1, x_2)$ is a bivariate Gaussian distribution with,

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}, \text{ and } \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}.$$

According to the result, will $f(x_1|x_2)$ be distributed in the following way;

$$f(x_1|x_2) \sim \mathcal{N}\left(\mu_1 - \frac{\sigma_{12}}{\sigma_{22}}(x_2 - \mu_2), \sigma_{11} - \frac{\sigma_{12}^2}{\sigma_{22}}\right), \tag{B.0.3}$$

which in the forthcoming will be shown.

The bivariate Gaussian distribution is expressed in Eq. A.0.7, and the exponential term will be restated here for convenience:

$$\exp\{\,\cdot\,\} = \exp\Big\{-\frac{1}{2(1-\rho_{12}^2)}\Big[A + \Big(\frac{x_2-\mu_2}{\sqrt{\sigma_{22}}}\Big)^2\Big]\Big\},$$

with

$$A = \left(\frac{(x_1-\mu_1)^2}{\sigma_{11}} - \left(\frac{2\rho_{12}(x_1-\mu_1)(x_2-\mu_2)}{\sqrt{\sigma_{11}\sigma_{22}}}\right)\right). \tag{B.0.4}$$

Factorizing out $1/\sigma_{11}$ results in the following,

$$A = \frac{1}{\sigma_{11}}\Big[(x_1-\mu_1)^2 - 2\rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_1-\mu_1)(x_2-\mu_2)\Big]. \tag{B.0.5}$$

The expression in Eq. B.0.5 is similar to the right side of the algebraic identity $(a-b)^2 = a^2 - 2ba + b^2$ with,

$$\begin{aligned} a &= (x_1-\mu_1) \\ b &= \rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2). \end{aligned} \tag{B.0.6}$$

The following computations are performed such that A can be formulated on the following form: $A = (a-b)^2 - b^2$, for reasons that will clear soon.

$$\begin{aligned} A &= \frac{1}{\sigma_{11}}\Big[(x_1-\mu_1)^2 - 2\rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_1-\mu_1)(x_2-\mu_2) \\ &\quad + \Big(\rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2)\Big)^2 - (\rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2)\Big)^2\Big] \\ &= \frac{1}{\sigma_{11}}\Big[(x_1-\mu_1) - \rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2)\Big]^2 - \frac{\rho_{12}^2}{\sigma_{22}}(x_2-\mu_2)^2 \\ &= c(a-b)^2 - b^2, \end{aligned} \tag{B.0.7}$$

where $c = 1/\sigma_{11}$ is a constant. Inserting for A from Eq. B.0.7 into Eq. B yields,

$$\begin{aligned} \exp\{\,\cdot\,\} &= \exp\Big\{-\frac{1}{2(1-\rho_{12}^2)}\Big[\frac{1}{\sigma_{11}}\Big((x_1-\mu_1) - \rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2)\Big)^2 \\ &\quad - \frac{\rho_{12}^2}{\sigma_{22}}(x_2-\mu_2)^2 + \frac{(x_2-\mu_2)^2}{\sigma_{22}}\Big]\Big\} \\ &= \exp\Big\{-\frac{1}{2(1-\rho_{12}^2)}\Big[\frac{1}{\sigma_{11}}\Big(x_1 - \big(\mu_1 - \rho_{12}\frac{\sqrt{\sigma_{11}}}{\sqrt{\sigma_{22}}}(x_2-\mu_2)\big)\Big)^2\Big] \\ &\quad - \frac{1}{2(1-\rho_{12}^2)}\Big(\frac{1}{\sigma_{22}} - \frac{\rho_{12}^2}{\sigma_{22}}\Big)(x_2-\mu_2)^2\Big\}. \end{aligned} \tag{B.0.8}$$

Using the relationship between the correlation and the covariance within the square bracket, i.e. $\rho_{12} = \sigma_{12}/(\sqrt{\sigma_{11}}\sqrt{\sigma_{22}})$,

$$\exp\{\,\cdot\,\} = \exp\Bigg\{-\frac{1}{2(1-\rho_{12}^2)}\Bigg[\frac{1}{\sigma_{11}}\Big(x_1 - \big(\mu_1 - \frac{\sigma_{12}\cancel{\sqrt{\sigma_{11}}}}{\sqrt{\sigma_{22}}\cancel{\sqrt{\sigma_{11}}}\sigma_{22}}(x_2-\mu_2)\big)\Big)^2\Bigg]$$

$$-\frac{1}{\cancel{2(1-\rho_{12}^2)}}\cancel{(1-\rho_{12}^2)}(x_2-\mu_2)^2\Bigg\}$$

$$= \exp\Bigg\{-\frac{1}{2(1-\rho_{12}^2)}\Bigg[\frac{1}{\sigma_{11}}\Big(x_1 - \big(\mu_1 - \frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\big)\Big)^2\Bigg] - \frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}\Bigg\}. \tag{B.0.9}$$

Hence, the (whole) bivariate Gaussian distribution can now be expressed in the following form,

$$\frac{1}{2\pi\sqrt{\sigma_{11}\sigma_{12}(1-\rho_{12}^2)}}\exp\Bigg\{-\frac{1}{2(1-\rho_{12}^2)}\Bigg[\frac{1}{\sigma_{11}}\Big(x_1-\big(\mu_1-\frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\big)\Big)^2\Bigg]-\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}} \tag{B.0.10}$$

The marginal density of the Gaussian distribution, $f(x_2)$, is given by,

$$f(x_2) = \frac{1}{\sqrt{2\pi}\sqrt{\sigma_{22}}}\exp\Bigg\{-\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}\Bigg\}. \tag{B.0.11}$$

The conditional density of X_1, given that $X_2 = x_2$ can now be computed, when Eq. B.0.11 is inserted in the denominator of Eq. B.0.2, and Eq. B.0.11 is inserted in the numerator of the same equation

$$f(x_1|x_2) = B \times \frac{\exp\Big\{-\frac{1}{2(1-\rho_{12}^2)}\Big[\frac{1}{\sigma_{11}}\Big(x_1-\big(\mu_1-\frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\big)\Big)^2\Big]-\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}\Big\}}{\exp\Big\{-\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}\Big\}}, \tag{B.0.12}$$

where B contain all terms independent of both x_1 and x_2:

$$B = \Bigg(\frac{1}{2\pi\sqrt{\sigma_{11}}\sqrt{\sigma_{12}}\sqrt{(1-\rho_{12}^2)}}\Bigg)\Bigg/\Bigg(\frac{1}{\sqrt{2\pi}\sqrt{\sigma_{22}}}\Bigg)$$

$$= \frac{\sqrt{2\pi}\sqrt{\sigma_{22}}}{2\pi\sqrt{\sigma_{11}}\sqrt{\sigma_{12}}\sqrt{(1-\rho_{12}^2)}} \tag{B.0.13}$$

$$= \frac{1}{\sqrt{\sigma_{11}(1-\rho_{12}^2)}}.$$

Inserting for B in Eq. B.0.12, and using the the exponential identity: $\exp(a)/\exp(b) = \exp(a-b)$ yields,

$$
\begin{aligned}
f(x_1|x_2) &= B \times \exp^{\left\{-\frac{1}{2\sigma_{11}(1-\rho_{12}^2)}\left(x_1-\left(\mu_1-\frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\right)\right)^2\right] - \cancel{\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}} - \left(-\cancel{\frac{1}{2}\frac{(x_2-\mu_2)^2}{\sigma_{22}}}\right)\right\}} \\
&= \frac{1}{\sqrt{\sigma_{11}(1-\rho_{12}^2)}} \times \exp^{\left\{-\frac{1}{2\sigma_{11}(1-\rho_{12}^2)}\left(x_1-\left(\mu_1-\frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\right)\right)^2\right\}} .
\end{aligned}
\tag{B.0.14}
$$

Using the following to substitute for the correlation,

$$\sigma_{11}(1-\rho_{12}^2) = \sigma_{11}(1-\frac{\sigma_{12}^2}{\sigma_{11}\sigma_{22}}) = \sigma_{11} - \frac{\sigma_{12}^2}{\sigma_{22}}$$

The conditional density $f(x_1, x_2)$ can then be written in the following way:

$$f(x_1|x_2) = \frac{1}{\sqrt{\sigma_{11} - \frac{\sigma_{12}^2}{\sigma_{22}}}} \times \exp^{\left\{-\frac{1}{2}\cdot\frac{1}{\sigma_{11}-\frac{\sigma_{12}^2}{\sigma_{22}}}\left(x_1-\left(\mu_1-\frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2)\right)\right)^2\right\}} , \tag{B.0.15}$$

which can be recognized as Gaussian distribution:

$$f(x_1|x_2) \sim \mathcal{N}\left(\mu_1 - \frac{\sigma_{12}}{\sigma_{22}}(x_2-\mu_2), \sigma_{11} - \frac{\sigma_{12}^2}{\sigma_{22}}\right), \tag{B.0.16}$$

and corresponds to Eq. B.0.3, which should be shown.

/C Feature names

List of the feature name for each feature index used through the thesis. Python start to index values from “o”, which is why the same indexing have been adopted also here.

0	Latitude
1	Longitude
2	Salinity
3	Temperature
4	BottomTemperature
5	LunarPhase
6	MeanWaveDirectionSwell
7	MeanWaveDirectionWind
8	MeanWavePeriodSwell
9	PeakWaveDirection

10 PeakWaveDirectionSwell

11 PeakWaveDirectionWind

12 PeakWavePeriod

13 PeakWavePeriodSwell

14 PeakWavePeriodSwim

15 SignificantSwellWaveHeight

16 SignificantWaveHeight

17 SignificantWavePeriod

18 SignificantWindWaveHeight

19 StokesDriftXVelocity

20 StokesDriftYVelocity

21 WaveDirection

22 WindDirection

23 WindSpeed

24 DistanceToLand

25 NitrogenSurface

26 NitrogenBottom

27 PhosphateSurface

28 PhosphateBottom

29 SilicateSurface

30 SilicateBottom

31 FlagellatesSurface

32 FlagellatesBottom

33 DiatomsSurface

34 DiatomsBottom

35 OxygenSurface

36 OxygenBottom

37 MicroplanctonSurface

38 MicroplanctonBottom

39 MesozoonplanctonSurface

40 MesozoonplanctonBottom

41 DetriusSurface

42 DetriusBottom

43 DEPTSurface

44 DEPTBottom

45 SISSurface

46 SISBottom

47 AvgDepth

48 AvgChlorophyll

D Forward feature selection, pseudo-code

This section provides a pseudo-code for the FFS algorithm used in Sec. 8.3. The list bellow gives an overview of some of the parameters mentioned in the pseudo-code. The pseudo-code assumes SVI model for GP.

- ρ is the number of kernel parameters, which increases automatically in each round of the algorithm when more features are added.
- m denote the number of inducing variables in the GP model, while $bSize$ denote the batch size of the GP model.
- $\boldsymbol{X}$ denote the input training data while $\boldsymbol{y}$ denote the output values from the test data. $\boldsymbol{X}$ are assumed to be normalized according to z-mean normalization in Eq. 6.1.2, while $\boldsymbol{y}$ is normalized to have zero mean.
- p denote the number of features/variables in $\boldsymbol{X}$.
- n denote number of observations in $\boldsymbol{X}$.
- $kernel$ is a predefined GP kernel.

Algorithm 2 Code

```
procedure FORWARD SEARCH
    Input: $\boldsymbol{X}$, $\boldsymbol{y}$, m, bSize, Nstep, p, kernel, n
    Initialization:
    XtrSplit = split($\boldsymbol{X}, p$)   % Split $\boldsymbol{X}$ according to its features
    BIC_list = []   % Empty list
    ind = 0   % Index
    $\rho = 1$   % Only one feature
First round forward selection:
    for i = 0 to p do
        $\boldsymbol{X}$ = XtrSplit(:, $i$)   % All observations for feature i
        % Define the GP model through: $\boldsymbol{X}$, $\boldsymbol{y}$, m, bSize, kernel
        % Optimize the GP model
        % Calculate the BIC for feature i, see Eq. 6.4.1
        % Define the GP model through: $\boldsymbol{X}$, $\boldsymbol{y}$, m, bSize, kernel
        $\mathrm{BIC}_\rho[i] = \log\left[\text{likelihood(optimized GP model)}\right] - \frac{1}{2}\cdot\rho\cdot\log\left[n\right]$
    end
    % Find the feature that maximizes the BIC:
    OptFeat = XtrSplit(max(BIC))
    BIC_list(ind) = index(max(BIC))
    $\rho = 2$
    for j = 1 to Nstep do
        for i = 0 to p do
            if i not in BIC_list then
                %   Concatenate features from early rounds with new features
                $\boldsymbol{X}$ = concat(XtrSplit(:,i),OptFeat)
                % Repeat step 10-15
            else
                BIC[i] = very small number
                end
            OptFeat = concat(OptFeat,XtrSplit(max(BIC)))
            BIC_list(ind) = index(max(BIC))
            ind = ind + 1
        end
    $\rho = \rho + 1$
    end
```

Bibliography

[1] A. R. A. Iglesias, B. Arcay and M. Cotos. A support system for fisheries based on neural networks. *1st International Workshop on Artificial Neural Networks and Intelligent Information Processing*, 2005.

[2] O. J. Aarnes, Ø. Breivik, and M. Reistad. Wave extremes in the Northeast Atlantic. *Journal of Climate*, 25(5):1529–1543, 2012. ISSN 0894-8755.

[3] P. Abrahamsen. *A review of Gaussian random fields and correlation functions*. Norsk Regnesentral/Norwegian Computing Center, 1997. ISBN 8253904355.

[4] A. T. Azar, S. A. El-Said, and A. E. Hassanien. Fuzzy and hard clustering analysis for thyroid disease. *Computer methods and programs in biomedicine*, 111(1):1–16, 2013. ISSN 0169-2607.

[5] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[6] J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics*, 7:453–464, 2003.

[7] P. Bholowalia and A. Kumar. Ebk-means: A clustering technique based on elbow method and k-means in wsn. *International Journal of Computer Applications*, 105(9), 2014. ISSN 0975-8887.

[8] C. Bishop. *Pattern recognition and machine learning*. Springer, 2006. ISBN 978-0-387-31073-2.

[9] R. Bleck. An oceanic general circulation model framed in hybrid isopycnic-cartesian coordinates. *Ocean modelling*, 4(1):55–88, 2002. ISSN 1463-5003.

[10] K. Blix, G. Camps-Valls, and R. Jenssen. Sensitivity analysis of Gaussian processes for oceanic chlorophyll prediction. In *2015 IEEE International Geoscience and*

Remote Sensing Symposium (IGARSS), pages 996–999. IEEE. ISBN 2153-6996.

[11] I. Borg and P. J. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005. ISBN 0387251502.

[12] P. Boyle. *Gaussian processes for regression and optimisation*. Doctoral thesis, Victoria University of Wellington, New Zealand, 2007.

[13] J. D. Camm, J. J. Cochran, M. J. Fry, J. W. Ohlmann, D. R. Anderson, D. J. Sweeney, and W. T. A. *Essentials of Business Analytics*, pages 350–353. Cengage Learning, 2nd edition edition, 2016.

[14] G. Camps-Valls, J. Verrelst, J. Munoz-Mari, V. Laparra, F. Mateo-Jimenez, and J. Gomez-Dans. A survey on Gaussian processes for earth-observation data analysis: A comprehensive investigation. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):58–78, 2016. ISSN 2168-6831.

[15] D. Carvalho, A. Rocha, M. Gómez-Gesteira, and C. Santos. A sensitivity study of the WRF model in wind simulation for an area of high wind energy. *Environmental Modelling & Software*, 33:23–34, 2012. ISSN 1364-8152.

[16] Central Bureau of Statistics. *Statistical Yearbook 2013*. 2013.

[17] N. Chen, Z. Qian, I. T. Nabney, and X. Meng. Wind power forecasts using Gaussian processes and numerical weather prediction. *IEEE Transactions on Power Systems*, 29(2):656–665, 2014. ISSN 0885-8950.

[18] S. Colley. *Vector calculus*. Pearson, 2012. ISBN 978-0-321-81875-1.

[19] Copernicus Marine Environment Monitoring Service. Arctic Ocean physics reanalysis (1991-2014), n.d. URL `http://marine.copernicus.eu/services-portfolio/access-to-products/?option=com_csw&view=details&product_id=ARCTIC_REANALYSIS_PHYS_002_003`.

[20] T. F. Cox and M. A. Cox. *Multidimensional scaling*. CRC press, 2000. ISBN 1420036122.

[21] L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002. ISSN 0899-7667.

[22] K. Deng. *OMEGA: On-line memory-based general purpose system classifier*. Doctorial thesis, Carnegie Mellon University, Pittsburgh Pennsylvania, 1998.

[23] P. J. Diggle, J. Tawn, and R. Moyeed. Model based geostatistics. *Journal of the*

Royal Statistical Society: Series C (Applied Statistics), 47(3):299–350, 1998. ISSN 1467-9876.

[24] C. J. Donlon, M. Martin, J. Stark, J. Roberts-Jones, E. Fiedler, and W. Wimmer. The operational sea surface temperature and sea ice analysis (OSTIA) system. *Remote Sensing of Environment*, 116:140–158, 2012. ISSN 0034-4257.

[25] S. G. D. Duda O. Richard, Hart E. Peter. *Pattern classification*. Wiley, 2001. ISBN 0-471-05669-3.

[26] D. Duvenaud. *Automatic model construction with Gaussian processes*. Doctoral thesis, University of Cambridge, Cambridge, 2014.

[27] B. R. Furevik and H. Haakenstad. Near-surface marine wind profiles from rawinsonde and NORA10 hindcast. *Journal of Geophysical Research: Atmospheres*, 117(D23), 2012. ISSN 2156-2202.

[28] L. P. F. Garćia, A. C. de Carvalho, and A. C. Lorena. *Noisy data set identification*, pages 629–638. Springer, 2013. ISBN 3642408451.

[29] M. N. Gibbs. *Bayesian Gaussian processes for regression and classification*. Doctorial thesis, University of Cambridge, Cambridge, 1998.

[30] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[31] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Elsevier, 2011. ISBN 0123814804.

[32] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, page 282. Citeseer.

[33] J. Hensman, M. Rattray, and N. D. Lawrence. Fast variational inference in the conjugate exponential family. In *Advances in Neural Information Processing Systems*, pages 2888–2896, 2013.

[34] R. Herbrich, N. D. Lawrence, and M. Seeger. Fast sparse Gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems*, pages 609–616.

[35] M. Hitsuda. Representation of Gaussian processes equivalent to Wiener. *Osaka J. Math*, 5:299–312, 1968.

[36] M. D. Hoffman, D. M. Blei, C. Wang, and J. W. Paisley. Stochastic variational

inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[37] A. Iglesias, C. Dafonte, B. Arcay, and J. M. Cotos. Integration of remote sensing techniques and connectionist models for decision support in fishing catches. *Environmental Modelling & Software*, 22(6):862–870, 2007. ISSN 1364-8152.

[38] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999. ISSN 0360-0300.

[39] R. A. Johnson and D. W. Wichern. *Applied multivariate statistical analysis*. Pearson Education Limited, 2013. ISBN 9781292024943.

[40] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999. ISSN 0885-6125.

[41] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1):273–324, 1997. ISSN 0004-3702.

[42] G. J. Komen, L. Cavaleri, M. Donelan, K. Hasselmann, S. Hasselmann, and P. Janssen. *Dynamics and modelling of ocean waves*. Cambridge university press, 1994. ISBN 0521577810.

[43] S. B. Kotsiantis, D. Kanellopoulos, and P. Pintelas. Data preprocessing for supervised learning. *International Journal of Computer Science*, pages 111–117, 2006.

[44] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. *Embedded methods*, pages 137–165. Springer, 2006.

[45] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[46] D. J. MacKay. Gaussian processes-a replacement for supervised neural networks? 1997. URL `http://www-clmc.usc.edu/publications/M/mackay-LectureNotesGP.pdf`.

[47] H. Madsen, P. Pinson, G. Kariniotakis, H. A. Nielsen, and T. S. Nielsen. Standardizing the performance evaluation of short-term wind power prediction models. *Wind Engineering*, 29(6):475–489, 2005. ISSN 0309-524X.

[48] G. Matheron. Principles of geostatistics. *Economic geology*, 58(8):1246–1266, 1963. ISSN 0361-0128.

[49] G. Matheron. The intrinsic random functions and their applications. *Advances in applied probability*, pages 439–468, 1973. ISSN 0001-8678.

[50] B. M. Mathisen, P. Haro, B. Hanssen, S. Björk, , and S. Walderhaug. Decision support systems in fisheries and aquaculture: A systematic review. *ArXiv e-prints*, 2016.

[51] C. D. Meyer. *Matrix analysis and applied linear algebra*, volume 2. Siam, 2000. ISBN 0898714540.

[52] T. M. Mitchell. Machine learning and data mining. *Communications of the ACM*, 42(11):30–36, 1999. ISSN 0001-0782.

[53] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012. ISBN 0262018020.

[54] G. Nagy. *State of the art in pattern recognition*, volume 56. IEEE, 1968.

[55] National Oceanic and Atmospheric Administration. How does pressure change with ocean depth? URL `http://oceanservice.noaa.gov/facts/pressure.html`.

[56] R. M. Neal. *Bayesian learning for neural networks*. Doctorial thesis, University of Toronto, Toronto, 1995.

[57] R. M. Neal. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. *arXiv preprint physics/9701026*, 1997.

[58] C. J. Paciorek. *Nonstationary Gaussian processes for regression and spatial modelling*. Doctorial thesis, Carnegie Mellon University, Pittsburgh Pennsylvania, 2003.

[59] F. Palumbo, D. Vistocco, and A. Morineau. *Huge multidimensional data visualization: back to the virtue of principal coordinates and dendrograms in the new computer age*, pages 349–387. Springer, 2008.

[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, , B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, , R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, , D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[61] J. M. Pena, J. A. Lozano, P. Larrañaga, and I. n. Inza. Dimensionality reduction in unsupervised learning of conditional Gaussian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):590–603, 2001. ISSN 0162-8828.

[62] F. Poisson, J.-C. Gaertner, M. Taquet, J.-P. Durbec, and K. Bigelow. Effects of lunar cycle and fishing operations on longline-caught pelagic fish: fishing performance, capture time, and survival of fish. *Fishery Bulletin*, 108(3):268–281, 2010. ISSN 0090-0656.

[63] J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6 (Dec):1939–1959, 2005.

[64] C. E. Rasmussen. *Gaussian processes in machine learning*, pages 63–71. Springer, 2004. ISBN 3540231226.

[65] C. E. Rasmussen. *Gaussian processes for machine learning*. MIT Press, 2006. ISBN ISBN 026218253X.

[66] P. Refaeilzadeh, L. Tang, and H. Liu. *Cross-validation*, pages 532–538. Springer, 2009. ISBN 0387355448.

[67] M. Reistad, Ø. Breivik, H. Haakenstad, O. J. Aarnes, B. R. Furevik, and J. Bidlot. A high-resolution hindcast of wind and waves for the North Sea, the Norwegian Sea, and the Barents Sea. *Journal of Geophysical Research: Oceans*, 116(C5), 2011. ISSN 2156-2202.

[68] P. Sakov, F. Counillon, L. Bertino, N. Finck, and C. Renkl. Quality information document. 2015.

[69] A. Samuelsen, C. Hansen, and H. Wehde. Tuning and assessment of the HYCOM-NORWECOM V2. 1 biogeochemical modeling system for the North Atlantic and Arctic oceans. *Geoscientific Model Development*, 8(7):2187–2202, 2015. ISSN 1991-959X.

[70] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[71] G. Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2): 461–464, 1978. ISSN 0090-5364.

[72] M. Seeger, C. Williams, and N. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Artificial Intelligence and Statistics 9*.

[73] M. D. Skogen and H. Sõiland. A user's guide to NORVECOM V2. 0. the Norwegian ecological model system. 1998. ISSN 0071-5638.

[74] A. Smola and V. Vapnik. Support vector regression machines. *Advances in neural*

information processing systems, 9:155–161, 1997.

[75] A. J. Smola and P. L. Bartlett. Sparse greedy Gaussian process regression. In *Advances in Neural Information Processing Systems*, pages 619–625.

[76] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004. ISSN 0960-3174.

[77] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264.

[78] E. L. Snelson. *Flexible and efficient Gaussian process models for machine learning*. Doctorial thesis, University of London, London, 2007.

[79] J. C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65. John Wiley & Sons, 2005. ISBN 0471441902.

[80] J. D. Stark, C. J. Donlon, M. J. Martin, and M. E. McCulloch. OSTIA: An operational, high resolution, real time, global sea surface temperature analysis system. In *Oceans 2007-Europe*, pages 1–4. IEEE. ISBN 1424406358.

[81] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Elsevier Science, Burlington, 4th ed. edition, 2008. ISBN 1-59749-272-8.

[82] M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *AISTATS*, volume 12, pages 567–574.

[83] M. K. Titsias. Variational model selection for sparse Gaussian process regression. Report, University of Manchester, UK, 2009.

[84] P. Unden, L. Rontu, H. Järvinen, P. Lynch, J. Calvo, G. Cats, J. Cuxart, K. Eerola, C. Fortelius, and J. A. Garcia-Moya. HIRLAM-5 scientific documentation. 2002.

[85] S. M. Uppala, P. Kållberg, A. Simmons, U. Andrae, V. d. Bechtold, M. Fiorino, J. Gibson, J. Haseler, A. Hernandez, and G. Kelly. The ERA-40 re-analysis. *Quarterly Journal of the Royal Meteorological Society*, 131(612):2961–3012, 2005. ISSN 1477-870X.

[86] L. van der Maaten. Learning a parametric embedding by preserving local structure. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS-09)*, volume 500, pages 384–391.

[87] L. Van Der Maaten. Accelerating t-SNE using tree-based algorithms. *Journal of machine learning research*, 15(1):3221–3245, 2014.

[88] L. Van der Maaten and G. Hinton. Visualizing non-metric similarities in multiple maps. *Machine learning*, 87(1):33–55, 2012. ISSN 0885-6125.

[89] J. Verrelst, L. Alonso, J. P. R. Caicedo, J. Moreno, and G. Camps-Valls. Gaussian process retrieval of chlorophyll content from imaging spectroscopy data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6 (2):867–874, 2013. ISSN 1939-1404.

[90] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Yee. *Probability & Statistics for Engineers and Scientists*. Pearson, 9th ed, international edition edition, 2012. ISBN 0-32174823-9.

[91] B. A. Walther and J. L. Moore. The concepts of bias, precision and accuracy, and their use in testing the performance of species richness estimators, with a literature review of estimator performance. *Ecography*, 28(6):815–829, 2005. ISSN 1600-0587.

[92] N. Wiener. *Cybernetics: or Control and Communication in the Animal and the Machine,*. MIT Press, 1948.

[93] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, MA, 1949.

[94] C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems 13*, 2001.

[95] C. K. Williams. *Prediction with Gaussian processes: From linear regression to linear prediction and beyond*, pages 599–621. Springer, 1998. ISBN 9401061041.

[96] C. K. Williams and C. E. Rasmussen. Gaussian processes for regression. 1996. ISSN 0262201070.

[97] J. Xie, P. Sakov, F. Counillon, L. Bertino, N. Finck, and C. Renkl. Quality information document. *History*, 2, 2016.