

---

# Contributions to Scalable Gaussian Processes

Doctoral Dissertation submitted to the  
Faculty of Informatics of the Università della Svizzera Italiana  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy

presented by  
Manuel Pascal Schürch

under the supervision of  
Prof. Luca M. Gambardella,  
Prof. Marco Zaffalon and Prof. Alessio Benavoli

July 2022



---

Dissertation Committee

<b>Prof. David Ginsbourger</b>	University of Bern, Switzerland
<b>Prof. Gianluigi Pillonetto</b>	University of Padova, Italy
<b>Prof. Cesare Alippi</b>	Università della Svizzera Italiana, Switzerland
<b>Prof. Rolf Krause</b>	Università della Svizzera Italiana, Switzerland
<b>Prof. Alessio Benavoli</b>	Trinity College Dublin, Ireland
<b>Prof. Marco Zaffalon</b>	Institute for Artificial Intelligence, Switzerland
<b>Prof. Luca M. Gambardella</b>	Università della Svizzera Italiana, Switzerland

Dissertation accepted on 31. July 2022



---

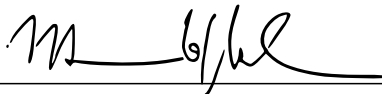
Research Advisor

**Prof. Luca M. Gambardella**

---

PhD Program Director


**Prof. Walter Binder *pro tempore***



---

Co-Advisor

**Prof. Marco Zaffalon**



---

Co-Advisor

**Prof. Alessio Benavoli**

---

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.



---

Manuel Pascal Schürch  
Lugano, 31. July 2022

*To my mother,  
who gave me the gift  
of life once again.*



Nobody ever figures out what life  
is all about, and it doesn't matter.  
Explore the world. Nearly  
everything is really interesting if  
you go into it deeply enough.

Richard P. Feynman





# Abstract

This thesis provides novel contributions to scalable Gaussian processes (GPs), which constitute an important tool in machine learning and statistics with applications ranging from social and natural science through engineering. GPs are powerful probabilistic methods with many benefits, such as their modeling flexibility, the robustness to overfitting and the availability of well-calibrated predictive uncertainty estimates. However, off-the-shelf GP inference procedures are limited to datasets with several thousand training data points because of their cubic computational complexity. This thesis presents new methodologies and novel algorithms for scaling GP regression to larger datasets by employing sequential and local methods. In particular, the first contribution of this work is a unifying GP approximation method based on a recursive formulation, which enables to train analytically a range of existing GP models in an online and distributed way. In this formulation, the so-called hyperparameters, which refer to a few parameters determining the GP, are assumed to be known. On the other hand, those can be learned by the second major contribution consisting of two novel algorithms for sequential hyperparameters estimation. These allow to scale the training of GPs up to millions of training samples. The last contribution involves a novel unifying GP approximation model exploiting sparsity and locality. Specifically, a method based on local GPs, which can share common information with a flexible correlation structure, is proposed. Thereby, this new model unifies several existing local and global GP approximation approaches. All the proposed methods in this thesis are theoretically supported and empirically tested on synthetic as well as real-world datasets with up to millions of training samples. Thereby, these new methods outperform the state-of-the-art in several tasks. This demonstrates the effectiveness of the novel GP approximations proposed in this thesis, which can achieve high-scalability without sacrificing the performance of original GPs. Therefore, this work substantially contributes to overcome the computational complexity barrier for the large-scale adoption of GPs.



# Acknowledgements

I would like to thank my supervisors Prof. Luca M. Gambardella, Prof. Marco Zaffalon, and Prof. Alessio Benavoli for the invaluable guidance, great support, helpful feedback and patience during my doctoral studies. In particular, my direct adviser Alessio provided me with constant encouragement, endless support, valuable advice, and I could count on his brilliant ideas as well as the common beneficial discussions whenever I felt the need for them. Similarly, I also wish to thank Dr. Dario Azzimonti, a senior researcher in our team and co-author for several papers, who accompanied me during the whole period of my doctoral studies on matters related to this thesis and beyond. Dario was always open for discussions ranging from high level ideas to small mathematical details, and he provided me with continuous motivation, meaningful feedback and never-ending suggestions for improvements. Moreover, I would like to thank my co-author Lucas Kania for a very interesting and fruitful research collaboration.

In general, I would like to express my gratefulness for the valuable research and teaching opportunity at *Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)* and *Università della Svizzera Italiana (USI)*. Thereby, I had the pleasure to meet many interesting people, from whom I could benefit through plenty of enriching and enjoyable discussions and stimulating activities beyond research. I am particularly thankful to my fellow PhD student Xavier Coiteux-Roy, who spent with me many unforgettable adventures in the mountains of *Ticino*.

The journey towards finishing my PhD was also influenced by several other people. In particular, a big thanks goes to my family and friends for their love and unlimited support. In particular, this thesis is dedicated to my mother, who gave me the gift of life once again in form of her kidney two years ago. I am eternally grateful for this generous donation and it reminds me every day to persistently follow my dreams and enjoy life to its best.



# Contents

<b>Contents</b>	<b>xi</b>
<b>List of Figures</b>	<b>xv</b>
<b>List of Tables</b>	<b>xvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Scientific Contributions . . . . .	3
1.3 Outline of the Thesis . . . . .	4
1.4 Associated Software . . . . .	5
<b>2 Preliminaries</b>	<b>7</b>
2.1 Basic Definitions and Properties . . . . .	7
2.1.1 Multivariate Gaussian Distributions . . . . .	8
2.1.2 Kullback-Leibler Divergence . . . . .	11
2.1.3 Numerical Optimization . . . . .	13
2.1.4 Useful Properties from Linear Algebra . . . . .	14
2.2 Linear Basis Function Model . . . . .	15
2.2.1 Additive Gaussian Noise . . . . .	16
2.2.2 Point Estimation for Regression . . . . .	17
2.3 Bayesian Probabilistic Inference . . . . .	23
2.3.1 Bayes' Rule . . . . .	23
2.3.2 Bayesian Hierarchical Model . . . . .	24
2.3.3 Bayesian Inference in Practice . . . . .	26
2.3.4 Bayesian Linear Model . . . . .	28
2.3.5 Advantages and Limitations . . . . .	38
2.4 Kernels . . . . .	42
2.4.1 Definitions and Properties . . . . .	42
2.4.2 Examples . . . . .	44

<b>3</b>	<b>Gaussian Processes</b>	<b>47</b>
3.1	From Bayesian Linear Models to GPs . . . . .	48
3.2	Function Space View . . . . .	50
3.2.1	Prior Distribution . . . . .	51
3.2.2	Inference . . . . .	52
3.3	Inference of Hyperparameters . . . . .	57
3.4	Linear Basis Function Models Revisited . . . . .	57
3.5	Advantages and Limitations of GPs . . . . .	64
<b>4</b>	<b>Approximations for Gaussian Processes</b>	<b>67</b>
4.1	Overview of GP Approximations . . . . .	68
4.2	Sparse Global GPs . . . . .	70
4.2.1	Augmented Model via Inducing Points . . . . .	70
4.2.2	Exact Inference with Approximate Prior . . . . .	73
4.2.3	Approximate Inference with Exact Prior . . . . .	82
4.2.4	Choosing Inducing Inputs and Hyperparameters . . . . .	90
4.2.5	Advantages and Disadvantages of SGPs . . . . .	94
4.3	Local Approaches . . . . .	95
4.3.1	Local Models with Single Prediction . . . . .	95
4.3.2	Product of Experts . . . . .	97
<b>5</b>	<b>Recursive Estimation for Sparse GPs</b>	<b>107</b>
5.1	Extended Bayesian Linear Model . . . . .	107
5.1.1	Likelihood . . . . .	108
5.1.2	Predictive Posterior . . . . .	109
5.1.3	Sparse GPs Revisited . . . . .	109
5.2	Recursive Sparse GP Model . . . . .	113
5.2.1	Bayesian Recursive Updating . . . . .	115
5.2.2	Recursive Joint Prior Distribution . . . . .	116
5.2.3	Recursive Updated Posterior . . . . .	117
5.2.4	Recursive Predictive Posterior . . . . .	117
5.2.5	Recursive Marginal Likelihood . . . . .	118
5.2.6	Kalman Filter Like Updating . . . . .	118
5.2.7	Information Filter Like Updating . . . . .	120
5.2.8	Recursive Variational Inference . . . . .	121
5.2.9	Examples . . . . .	123
5.2.10	Transformed Basis Functions . . . . .	125
5.2.11	Connection to Dynamical Systems . . . . .	126
5.3	Summary of Contributions . . . . .	127

<b>6</b>	<b>Sequential Hyperparameter Learning for Sparse GPs</b>	<b>129</b>
6.1	Background . . . . .	129
6.1.1	State of the Art . . . . .	131
6.2	Recursive Gradient Propagation . . . . .	134
6.2.1	Generalized Recursive Collapsed Bound . . . . .	134
6.2.2	Derivatives of Recursive Collapsed Bound . . . . .	135
6.2.3	Computational Complexity . . . . .	137
6.2.4	Mini-batch size . . . . .	138
6.2.5	Experiments . . . . .	139
6.2.6	Conclusion . . . . .	144
6.3	Sparse Information Filter for Fast Sparse GPs . . . . .	144
6.3.1	Recursive Estimation for Sparse GPs . . . . .	146
6.3.2	Information Filter for Sparse GPs . . . . .	147
6.3.3	Generalized Lower Bound based on IF . . . . .	148
6.3.4	Stochastic Hyperparameter Optimization . . . . .	149
6.3.5	Generalized Independent Lower Bound . . . . .	149
6.3.6	Recovery of the Full Posterior . . . . .	151
6.3.7	Distributed Hyperparameter Optimization . . . . .	152
6.3.8	Experiments . . . . .	153
6.4	Summary of Contributions . . . . .	158
<b>7</b>	<b>Correlated Product of Experts for Sparse GPs</b>	<b>161</b>
7.1	Background . . . . .	161
7.1.1	GP Regression . . . . .	164
7.2	Correlated Product of Experts . . . . .	165
7.2.1	Graphical Model . . . . .	166
7.2.2	Sparse and Local Prior . . . . .	169
7.2.3	Inference . . . . .	170
7.2.4	Prediction . . . . .	172
7.2.5	Properties . . . . .	174
7.2.6	Generalizations of Bayesian Linear Models . . . . .	176
7.2.7	Computational Details . . . . .	176
7.2.8	Generalized CPoE . . . . .	182
7.2.9	Additional Results . . . . .	184
7.3	Comparison . . . . .	187
7.3.1	Application . . . . .	189
7.4	CPoE via Gaussian Belief Propagation . . . . .	191
7.4.1	Construction of Junction Tree . . . . .	193
7.4.2	Factorization of Joint Distribution . . . . .	195

7.4.3	Belief Propagation . . . . .	197
7.4.4	Summary . . . . .	199
7.5	Summary of Contributions . . . . .	200
7.5.1	Conclusion . . . . .	200
7.5.2	Contributions . . . . .	200
<b>8</b>	<b>Conclusion and Future Work</b>	<b>203</b>
8.1	Summary . . . . .	203
8.2	Future Work . . . . .	204
8.2.1	Online Algorithm for Sparse GPs . . . . .	204
8.2.2	Variational Approach for CPoE . . . . .	205
8.2.3	Efficient Drawing of Posterior Samples . . . . .	205
8.2.4	Large-Scale GP Implementations for Time-Series . . . . .	206
8.2.5	Unifying GP Model for Regression and Classification . . . . .	206
8.2.6	Potential Application Areas for Large-Scale GPs . . . . .	207
	<b>Bibliography</b>	<b>209</b>
<b>A</b>	<b>Orthogonally Decoupled Variational Fourier Features</b>	<b>219</b>
<b>B</b>	<b>Appendix of Chapter 5</b>	<b>221</b>
B.1	Proofs . . . . .	221
B.2	Details for Recursive Gradient Propagation . . . . .	222
<b>C</b>	<b>Appendix of Chapter 7</b>	<b>225</b>
C.1	Proofs . . . . .	225
C.2	Additional Material . . . . .	236
C.3	Tables . . . . .	237



# Figures

1.1	Different settings to train GP models. . . . .	2
1.2	Preview of novel GP approximation model. . . . .	4
2.1	Inference in multivariate Gaussian distributions. . . . .	10
2.2	Forward and reverse KL minimization. . . . .	12
2.3	Examples of basis functions. . . . .	16
2.4	Polynomial regression. . . . .	19
2.5	Generalization capacity of non-Bayesian approaches. . . . .	22
2.6	Bayesian inference for linear regression. . . . .	33
2.7	Inference of hyperparameters. . . . .	36
2.8	Bayesian model selection for polynomial regression. . . . .	38
2.9	Bayesian linear regression with different basis functions. . . . .	40
3.1	From finite Bayesian linear models to GPs. . . . .	48
3.2	Illustration of kernel trick. . . . .	49
3.3	Inference in GPs with different kernels. . . . .	55
3.4	Inference of hyperparameters in GPs. . . . .	58
3.5	Basis function decomposition of full GPs. . . . .	61
3.6	Difference between Bayesian linear models and GPs. . . . .	62
3.7	Comparison of full GP, degenerated and augmented GP. . . . .	64
4.1	Illustration of global inducing points. . . . .	71
4.2	Graphical models of full GP and sparse GP. . . . .	72
4.3	Four different versions of sparse GPs. . . . .	81
4.4	Inference of inducing inputs. . . . .	91
4.5	Different local GP approaches. . . . .	100
5.1	Extended Bayesian linear basis function model. . . . .	112
5.2	Recursive updating for sparse GP models. . . . .	119
5.3	Full GP and batch sparse GP regression. . . . .	124

5.4	Online learning for sparse GPs. . . . .	125
5.5	Distributed learning for sparse GPs. . . . .	126
6.1	Different ways to train global sparse GP models. . . . .	130
6.2	Convergence of SRGP to batch version. . . . .	138
6.3	Effect of mini-batch size. . . . .	139
6.4	Results of simulation study. . . . .	140
6.5	Convergence for real world data. . . . .	141
6.6	Training and prediction for non-linear plant. . . . .	142
6.7	Results for non-linear plant. . . . .	143
6.8	Preview of convergence of proposed method. . . . .	145
6.9	Convergence for synthetic data I. . . . .	152
6.10	Convergence for synthetic data II. . . . .	154
7.1	Unifying view of our proposed method. . . . .	162
7.2	Graphical models of different GP approaches. . . . .	163
7.3	Different GP approximations methods. . . . .	165
7.4	Flexible correlation structure. . . . .	168
7.5	Graphical model of CPoE. . . . .	168
7.6	Sparse transition and projection matrices. . . . .	170
7.7	Sparse posterior precision approximation. . . . .	171
7.8	Prediction Aggregation in CPoE. . . . .	173
7.9	Convergence of CPoE to full GP . . . . .	175
7.10	Extensions of Bayesian linear models. . . . .	177
7.11	Illustration of graph. . . . .	178
7.12	Partial inversion. . . . .	179
7.13	Structure of covariance and precision matrices I. . . . .	184
7.14	Structure of covariance and precision matrices II. . . . .	185
7.15	Structure of precision matrices. . . . .	185
7.16	Synthetic data. . . . .	188
7.17	Convergence for CPoE. . . . .	189
7.18	Time series example. . . . .	191
7.19	General correlation structure. . . . .	192
7.20	Connection between sparse matrices and graphs. . . . .	193
7.21	Construction of junction tree. . . . .	194
7.22	Varying correlation. . . . .	196
8.1	Online approach for sparse GPs. . . . .	204
8.2	Correlated experts and junction trees. . . . .	207

# Tables

2.1	Bayesian inference in an hierarchical model. . . . .	25
2.2	Inference for hyperparameters. . . . .	35
3.1	Overview of inference for GPs. . . . .	52
3.2	Basis function decomposition of full GPs. . . . .	60
4.1	Unifying view of sparse GPs. . . . .	75
4.2	Overview of inference in sparse GPs. . . . .	76
4.3	Different approximate kernel functions. . . . .	82
4.4	Different aggregation methods for local methods. . . . .	98
5.1	Basis function decompositions for sparse GPs. . . . .	110
5.2	Unifying view of recursive sparse GPs. . . . .	115
6.1	Overview of inference approaches. . . . .	131
6.2	Different methods for estimating hyperparameters. . . . .	133
6.3	Results for synthetic data. . . . .	155
6.4	Overview of datasets. . . . .	155
6.5	Log-Likelihood for real-world data. . . . .	157
6.6	RMSE for real-world data. . . . .	157
6.7	Running times for real-world data. . . . .	157
7.1	Summary of extensions of Bayesian linear models. . . . .	177
7.2	Complexity of CPoE. . . . .	182
7.3	Results with deterministic optimization. . . . .	188
7.4	Results with stochastic optimization. . . . .	189
7.5	Summary of datasets. . . . .	190
C.1	Results for dataset <i>concrete</i> . . . . .	239
C.2	Results for dataset <i>mg</i> . . . . .	239
C.3	Results for dataset <i>space</i> . . . . .	239

---

C.4	Results for dataset <i>abalone</i> . . . . .	240
C.5	Results for dataset <i>kin</i> . . . . .	240
C.6	Results I for dataset <i>kin2</i> . . . . .	240
C.7	Results II for dataset <i>kin2</i> . . . . .	241
C.8	Results for dataset <i>cadata</i> . . . . .	241
C.9	Results for dataset <i>sarcos</i> . . . . .	241
C.10	Results for dataset <i>casp</i> . . . . .	242

# Chapter 1

## Introduction

### 1.1 Motivation

The amount of collected digital data is growing at a rapid pace and concerns basically all areas from our society. Machine learning algorithms and statistical models are employed on this data to automatically uncover hidden patterns and extract useful information that can be used for predictions and optimal decision making, which might have a great impact to the reality. Therefore, a research challenge is to design scalable and robust algorithms that reliably exploit this large amount of data, as for instance outlined in "*Steps toward robust artificial intelligence*" by Dietterich [2017]. For example, in a medical application or in a self-driving car, it is crucial that the underlying assumptions in the algorithms are comprehensible and transparent to the user, since the resulting predictions might have severe consequences. Furthermore, it is important that algorithms provide predictions, which also include some information indicating how certain they are. Moreover, the incoming data from the different sensors of a self-driving car or a medical device arrive in a continuous stream and prediction-based actions have to be done immediately. Thus, it is very beneficial to update the current knowledge with the incoming information incrementally. Therefore, the aim of this thesis consists on the development of algorithms, (i) which can be scaled to huge datasets, (ii) in which the underlying assumptions are transparent to the user, (iii) can quantify the reliability of their predictions, and (iv) can adapt their knowledge incrementally.

In order to achieve these goals, probabilistic methods based on so-called *Bayesian inference* constitute an attractive approach, since they model uncertainty in a principled manner, can incorporate prior assumptions transparently and can incrementally update information. Bayesian inference methods rely on *Bayes' theo-*

rem, allowing to combine prior assumptions with information about the observed data, resulting into a posterior belief from which predictions can be computed. These concepts will be more thoroughly discussed in subsequent chapters. A particular class of probabilistic methods are *Gaussian processes* (GPs), which enable to perform Bayesian inference directly in the space of functions. GPs allow to express our prior beliefs about a class of functions before observing the data, and to analytically compute our updated posterior beliefs about the functions in a way that is consistent with the data and prior beliefs. Thereby, the posterior can be used to quantify the predictive uncertainty and, therefore, to evaluate the reliability of the model. Due to the incremental updating mechanism, GPs are also promising for online settings. However, besides these advantages, GPs have a main drawback: they are not scalable to large datasets due to their cubic time and quadratic space complexity in the number of training samples. Therefore, the specific aim of this thesis is to propose novel GP approximation methods, which allow to scale GPs to large datasets while retaining the benefits of GPs. Our main approach is to divide the overall problem into smaller local subproblems. Instead operating on the whole—possibly huge—dataset  $\mathcal{D}$ , we consider in this thesis different settings, where we split the data into  $J$  smaller blocks of data  $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_J\}$  and only perform computations involving one block  $\mathcal{D}_j$  at a time as illustrated in Figure 1.1. In particular, we propose novel algorithms for the online, sequential and distributed setting, yielding highly scalable and accurate GP approximation methods, so that a range of new applications are opened up for GPs.

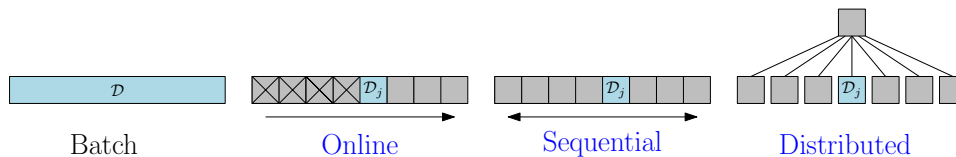


Figure 1.1. Different settings to train GP models by exploiting smaller blocks of data. Beside full batch training involving the whole dataset  $\mathcal{D}$ , we distinguish between *online*, *sequential* and *distributed* settings. In the former two cases, we assume that the data arrives as a stream of mini-batches  $\mathcal{D}_j$ . However, in the online setting, each mini-batch can be only considered once, whereas in the sequential setting, it is allowed to consider each block of data several times. In the distributed setting, we assume a central node and each mini-batch is associated with a computational node working in parallel.

## 1.2 Scientific Contributions

This thesis provides novel contributions to scalable Gaussian processes. In particular, we first review the properties of GPs with focus on regression methods, analyze the state-of-the-art methods for their approximations in the literature, and discuss their strengths and limitations. To make GPs scalable to large datasets, we then propose new methodologies and novel algorithms, theoretically supported and empirically evaluated by several experiments, demonstrating state-of-the-art performance in several tasks. The main scientific contributions are based on the following publications.

- **Recursive Estimation for Sparse Gaussian Process Regression**  
Schürch, Azzimonti, Benavoli, and Zaffalon [2020], *Automatica* 2020.
- **Orthogonally Decoupled Variational Fourier Features**  
Azzimonti, Schürch, Benavoli, and Zaffalon [2020], arXiv [2007.06363].
- **Sparse Information Filter for Fast Gaussian Process Regression**  
Kania, Schürch, Azzimonti, and Benavoli [2021], ECML 2021.
- **Correlated Product of Experts for Sparse Gaussian Process Regression**  
Schürch, Azzimonti, Benavoli, and Zaffalon [2022], *Machine Learning* journal track ECML PKDD 2022.

In this thesis, we present the content from these papers in a slightly different structure. In particular, the first part of [Schürch et al., 2020] is discussed in Chapter 5, whereas the second part is provided together with [Kania et al., 2021] in Chapter 6. Further, the work in [Schürch et al., 2022] is shown together with additional material in Chapter 7, and finally [Azzimonti et al., 2020] is summarized in Appendix A. In the following, we briefly provide a summary of the main novel content in this thesis.

- **Online and Distributed Training of Sparse GPs (Chapter 5):** We provide a unifying recursive model for the analytic training of a range of existing GP models in an online and distributed way for fixed hyperparameters (which refer to a few parameters particularly determining the GP).
- **Sequential Hyperparameters Estimation for Sparse GPs (Chapter 6):** We propose two novel algorithms for sequential hyperparameters estimation for sparse GPs, which enables to scale the training of GPs up to millions of training samples.

- **Unifying GP Approximation Method (Chapter 7):** We present a novel GP approximation method with flexible dependency structure based on locally correlated experts as illustrated in Figure 1.2. Thereby, our new model unifies several existing local and global GP approximation approaches. Moreover, we establish connections between GPs and local belief propagation methods.

An explicit description about the detailed contributions with respect to the state-of-the-art can be found at the end of each of these Chapters 5, 6 and 7.

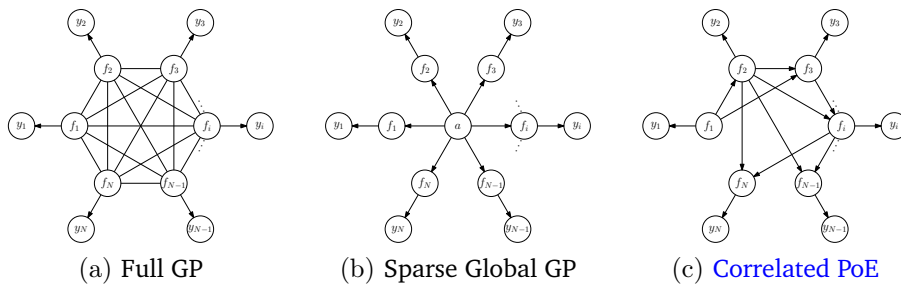


Figure 1.2. As a preview, the graphical models of full GP, an existing sparse global GP approximation, and one of our novel methods with flexible dependency structure.

### 1.3 Outline of the Thesis

The thesis is structured into three kind of chapters: the introductory and conclusive Chapters 1, 2, and 8, the reviewing Chapters 3 and 4, and the Chapters 5, 6, and 7, which contain explicit novel scientific content. In the following, we provide a brief summary about the chapters in this thesis.

- **Chapter 1:** This chapter provides an overview of the thesis. In particular, it contains the motivation, the scientific contributions, the outline, and the associated software of this thesis.
- **Chapter 2:** In this chapter, we discuss preliminary tools and concepts, which are needed in the subsequent chapters about GPs and their scalable approximations. Among other basic topics, we discuss Bayesian inference techniques and provide an example about the Bayesian linear model with basis functions.



- **Chapter 3:** In this chapter, we introduce GPs tailored to the need for machine learning and in particular for this thesis. Specifically, we focus on GP regression, for which we provide two different views, discuss their inference techniques and compare GPs with Bayesian linear models.
- **Chapter 4:** In this chapter, we present existing approximations for GPs, whereby we will focus on two classes of approximations. On the one hand, we discuss global approaches based on so-called inducing points, and on the other hand, we provide an overview over local approaches combined with prediction averaging methods.
- **Chapter 5:** In this chapter, we propose a novel unifying recursive model, which allows to train analytically a range of existing sparse GP models in an online and distributed way for fixed hyperparameters.
- **Chapter 6:** In this chapter, we present two new methods for sequential hyperparameter learning for sparse GPs, which allows to train those models with up to several millions of training data samples and achieves state-of-the-art performance, as demonstrated in several experiments.
- **Chapter 7:** In this chapter, we introduce a novel unifying GP approximation model based on local and correlated experts. Our proposed model generalizes several existing global and local approaches. Moreover, we establish connections between GPs and local belief propagation methods.
- **Chapter 8:** In this chapter, we conclude the thesis by summarizing the findings in this manuscript and outline some future work.

## 1.4 Associated Software

For each algorithm in this thesis, the corresponding software is available on Github. In particular, the source code in Python with explanations how to use it, can be found at the corresponding Github repository.

- **Recursive Estimation for Sparse Gaussian Process Regression**  
<https://github.com/manuelIDSIA/SRGP>.
- **Sparse Information Filter for Fast Gaussian Process Regression**  
<https://github.com/lkania/Sparse-IF-for-Fast-GP>.

- **Correlated Product of Experts for Sparse Gaussian Process Regression**  
<https://github.com/manuelIDSIA/CPoE> ,

# Chapter 2

## Preliminaries

In this chapter, we introduce preliminary probabilistic tools and concepts, which are needed in the subsequent chapters about Gaussian processes and their scalable approximations. In particular, we provide some basic definitions and properties in Section 2.1, which are used throughout this thesis, as for instance the multivariate Gaussian distribution, the Kullback-Leibler-divergence and some approaches for numerical optimization. Moreover, frequentistic and Bayesian approaches for inference in statistical models are discussed by the aid of a linear regression model in Sections 2.2 and 2.3, respectively. Finally, the concept of kernels is introduced in Section 2.4.

The content of this chapter is well known in the scientific community and it is mainly provided here for the sake of self-completeness of this thesis. However, these general concepts are presented in a way that is tailored to GP models and the subsequent chapters build on these tools. In particular, Section 2.3 about Bayesian inference is presented so as to introduce the formalism, which will then be used for GPs in the next chapters. For instance, the inference for the Bayesian linear model is discussed in detail, since it constitutes a simple model ideal for introducing several abstract concepts about probabilistic inference. A reader who is familiar with topics in probabilistic inference might skip this chapter except Section 2.3, where relevant concepts for the rest of the thesis are introduced.

### 2.1 Basic Definitions and Properties

In this section, we provide some standard definitions and properties used in subsequent chapters of this thesis. These properties can be found in the most standard books about multivariate statistics or machine learning, in particular, we refer the reader to Bishop [2006], Murphy [2012], and Koller and Fried-

man [2009]. First, we discuss multivariate Gaussian distributions and their main properties in Section 2.1.1, since they constitute the fundamentals of Gaussian processes. Moreover, we briefly introduce Kullback-Leibler-(KL)-divergences and their need for probabilistic inference in Section 2.1.2, the basics about numerical optimization in Section 2.1.3, and some useful properties from linear algebra in Section 2.1.4.

### 2.1.1 Multivariate Gaussian Distributions

**Definition 2.1 (Multivariate Gaussian Distribution)** *A random vector  $\mathbf{z} \in \mathbb{R}^M$ , which follows a  $M$ -variate Gaussian distribution*

$$\mathbf{z} \sim \mathcal{N}(\mathbf{z} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = p(\mathbf{z})$$

with mean vector  $\mathbb{E}[\mathbf{z}] = \boldsymbol{\mu} \in \mathbb{R}^M$  and positive-definite covariance matrix  $\text{Cov}(\mathbf{z}) = \boldsymbol{\Sigma} \in \mathbb{R}^{M \times M}$ , has a probability density function

$$p(\mathbf{z}) = (2\pi)^{-\frac{M}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{z} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{z} - \boldsymbol{\mu})\right).$$

#### 2.1.1.1 Marginal and Conditional Distributions

For two random vectors  $\mathbf{y} \in \mathbb{R}^N$  and  $\mathbf{z} \in \mathbb{R}^M$ , which are jointly Gaussian distributed, the joint  $(N + M)$ -variate Gaussian distribution can be formulated as

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_y \\ \boldsymbol{\mu}_z \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{yy} & \boldsymbol{\Sigma}_{yz} \\ \boldsymbol{\Sigma}_{zy} & \boldsymbol{\Sigma}_{zz} \end{bmatrix}\right) = p(\mathbf{y}, \mathbf{z}) \quad (2.1)$$

with the corresponding subentries, where it holds  $\boldsymbol{\Sigma}_{yz} = \boldsymbol{\Sigma}_{zy}^T$ . From this joint distribution in (2.1), the marginal Gaussian distributions are

$$\begin{aligned} \mathbf{y} &\sim \mathcal{N}(\mathbf{y} | \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_{yy}) = \int p(\mathbf{y}, \mathbf{z}) d\mathbf{z} = p(\mathbf{y}), \\ \mathbf{z} &\sim \mathcal{N}(\mathbf{z} | \boldsymbol{\mu}_z, \boldsymbol{\Sigma}_{zz}) = \int p(\mathbf{y}, \mathbf{z}) d\mathbf{y} = p(\mathbf{z}). \end{aligned} \quad (2.2)$$

Moreover, the conditional Gaussian distribution  $\mathbf{z} | \mathbf{y}$  can be derived as

$$p(\mathbf{z} | \mathbf{y}) = \frac{p(\mathbf{y}, \mathbf{z})}{p(\mathbf{y})} = \mathcal{N}\left(\mathbf{z} | \boldsymbol{\mu}_z + \boldsymbol{\Sigma}_{zy} \boldsymbol{\Sigma}_{yy}^{-1}(\mathbf{y} - \boldsymbol{\mu}_y), \boldsymbol{\Sigma}_{zz} - \boldsymbol{\Sigma}_{zy} \boldsymbol{\Sigma}_{yy}^{-1} \boldsymbol{\Sigma}_{yz}\right), \quad (2.3)$$

which is a density in  $\mathbf{z}$ . Note that  $p(\mathbf{y} | \mathbf{z})$  can be computed analogously.

Given the Gaussian marginal distribution  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$  and the Gaussian conditional distribution  $p(\mathbf{y}|\mathbf{z}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{z} + \mathbf{b}, \mathbf{Q})$  with affine mean transformation, the joint Gaussian distribution  $p(\mathbf{y}, \mathbf{z})$  is given by

$$p(\mathbf{y}, \mathbf{z}) = p(\mathbf{y}|\mathbf{z})p(\mathbf{z}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{z} \\ \mathbf{y} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu} \\ \mathbf{A}\boldsymbol{\mu} + \mathbf{b} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}\mathbf{A}^T \\ \mathbf{A}\boldsymbol{\Sigma} & \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T + \mathbf{Q} \end{bmatrix}\right). \quad (2.4)$$

The marginal distribution of  $\mathbf{y}$  is then

$$p(\mathbf{y}) = \int p(\mathbf{y}, \mathbf{z}) d\mathbf{z} = \int p(\mathbf{y}|\mathbf{z})p(\mathbf{z}) d\mathbf{z} = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T + \mathbf{Q}). \quad (2.5)$$

The opposite conditional distribution  $\mathbf{z}|\mathbf{y}$  can be computed by

$$p(\mathbf{z}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{y})} = \frac{p(\mathbf{y}, \mathbf{z})}{p(\mathbf{y})} = \mathcal{N}(\mathbf{z}|\mathbf{P}(\mathbf{A}^T\mathbf{Q}^{-1}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Sigma}^{-1})\boldsymbol{\mu}, \mathbf{P}), \quad (2.6)$$

where  $\mathbf{P} = (\boldsymbol{\Sigma}^{-1} + \mathbf{A}^T\mathbf{Q}^{-1}\mathbf{A})^{-1}$ .

In Figure 2.1, we illustrate marginalization and conditioning for the joint Gaussian distribution  $p(\mathbf{z}, \mathbf{y})$ . As a preview for subsequent chapters, if we interpret  $\mathbf{z}$  as an unobserved variable and  $\mathbf{y}$  as an observed variable, then  $p(\mathbf{z})$  is called prior,  $p(\mathbf{y}|\mathbf{z})$  likelihood,  $p(\mathbf{z}|\mathbf{y})$  posterior and  $p(\mathbf{y})$  marginal likelihood. Note that these simple operations correspond to inference with Gaussian processes, as we will thoroughly discuss in Chapter 3.

### 2.1.1.2 Entropy of Multivariate Gaussian

The *Entropy*  $H$  of a  $M$ -variate Gaussian random vector  $\mathbf{z}$  with density  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$  is defined as

$$H[\mathbf{z}] = H[p(\mathbf{z})] = \frac{1}{2}(\log |\boldsymbol{\Sigma}| + M(1 + \log 2\pi)), \quad (2.7)$$

where we use  $\log$  as the natural logarithm and thus the entropy is measured in nats (natural units).

### 2.1.1.3 Gaussians in Canonical Form

Alternatively to Definition 2.1 of a multivariate Gaussian distribution  $\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$  parametrized by the mean  $\boldsymbol{\mu}$  and covariance  $\boldsymbol{\Sigma}$ , it can also be parameterized by the *canonical* parameters  $\boldsymbol{\eta} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$  and  $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ .

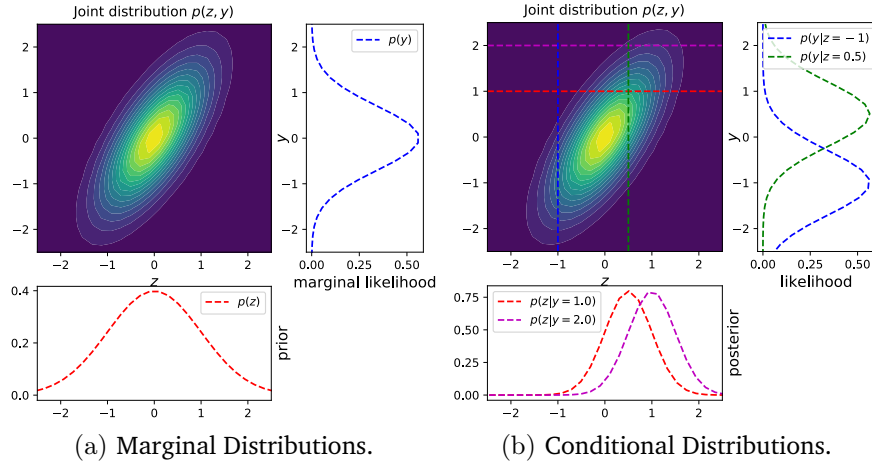


Figure 2.1. Illustration for marginalization and conditioning for the joint Gaussian distribution  $p(\mathbf{z}, \mathbf{y})$  as described in Equations (2.2) and (2.3), respectively.

**Definition 2.2 (Multivariate Gaussian Distribution in Canonical Form)** A random vector  $\mathbf{z} \in \mathbb{R}^M$ , which follows a  $M$ -variate Gaussian distribution, can be written as

$$\mathbf{z} \sim \mathcal{N}^{-1}(\mathbf{z} | \boldsymbol{\eta}, \boldsymbol{\Lambda}) = p(\mathbf{z})$$

with natural mean vector  $\boldsymbol{\eta} \in \mathbb{R}^M$  and precision matrix  $\text{Cov}(\mathbf{z})^{-1} = \boldsymbol{\Lambda} \in \mathbb{R}^{M \times M}$ , has a probability density function

$$p(\mathbf{z}) = (2\pi)^{-\frac{M}{2}} |\boldsymbol{\Lambda}|^{\frac{1}{2}} \exp\left(-\frac{1}{2} \mathbf{z}^T \boldsymbol{\Lambda} \mathbf{z} + \boldsymbol{\eta}^T \mathbf{z} - \frac{1}{2} \boldsymbol{\eta}^T \boldsymbol{\Lambda}^{-1} \boldsymbol{\eta}\right).$$

It holds  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}^{-1}(\mathbf{z} | \boldsymbol{\eta}, \boldsymbol{\Lambda})$  with  $\boldsymbol{\mu} = \boldsymbol{\Lambda}^{-1} \boldsymbol{\eta}$  and  $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1}$ .

The joint distribution in (2.1) can equivalently be formulated in canonical form

$$p(\mathbf{y}, \mathbf{z}) = \mathcal{N}^{-1}\left(\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\eta}_y \\ \boldsymbol{\eta}_z \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Lambda}_{yy} & \boldsymbol{\Lambda}_{yz} \\ \boldsymbol{\Lambda}_{zy} & \boldsymbol{\Lambda}_{zz} \end{bmatrix}\right) \quad (2.8)$$

with the corresponding subentries, where  $\boldsymbol{\Lambda}_{yz} = \boldsymbol{\Lambda}_{zy}^T$ . From this joint distribution in (2.8), the marginal Gaussian distribution in canonical form can be computed

$$p(\mathbf{z}) = \int p(\mathbf{y}, \mathbf{z}) d\mathbf{y} = \mathcal{N}^{-1}\left(\mathbf{z} | \boldsymbol{\eta}_z - \boldsymbol{\Lambda}_{zy} \boldsymbol{\Lambda}_{yy}^{-1} \boldsymbol{\eta}_y, \boldsymbol{\Lambda}_{zz} - \boldsymbol{\Lambda}_{zy} \boldsymbol{\Lambda}_{yy}^{-1} \boldsymbol{\Lambda}_{yz}\right). \quad (2.9)$$

Moreover, the conditional Gaussian distribution  $\mathbf{z} | \mathbf{y}$  can be derived as

$$p(\mathbf{z} | \mathbf{y}) = \frac{p(\mathbf{y}, \mathbf{z})}{p(\mathbf{y})} = \mathcal{N}^{-1}\left(\mathbf{z} | \boldsymbol{\eta}_z - \boldsymbol{\Lambda}_{zy} \mathbf{y}, \boldsymbol{\Lambda}_{zz}\right). \quad (2.10)$$

Note that, the marginal distribution  $p(\mathbf{y})$  and the conditional distribution  $p(\mathbf{y}|\mathbf{z})$  can be computed analogously.

#### 2.1.1.4 Product of Gaussians

Assume some weights  $a_j \in \mathbb{R}$  and multivariate Gaussians  $p_j(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \mathcal{N}^{-1}(\mathbf{x}|\boldsymbol{\eta}_j, \boldsymbol{\Lambda}_j)$  for  $j = 1, \dots, J$ . The weighted product over these Gaussians densities can be explicitly computed as

$$\prod_{j=1}^J p_j(\mathbf{x})^{a_j} = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}^{-1}(\mathbf{x}|\boldsymbol{\eta}, \boldsymbol{\Lambda}), \quad (2.11)$$

with

$$\boldsymbol{\Lambda} = \sum_{j=1}^J a_j \boldsymbol{\Lambda}_j \quad \text{and} \quad \boldsymbol{\eta} = \sum_{j=1}^J a_j \boldsymbol{\eta}_j,$$

as well as

$$\boldsymbol{\Sigma} = \left( \sum_{j=1}^J a_j \boldsymbol{\Sigma}_j^{-1} \right)^{-1} \quad \text{and} \quad \boldsymbol{\mu} = \boldsymbol{\Sigma} \left( \sum_{j=1}^J a_j \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j \right),$$

respectively. Note that, the Gaussians are only properly defined as long as  $\boldsymbol{\Sigma}$  positive-definite (for instance if  $a_j \equiv 1$  or  $\sum_{j=1}^J a_j = 1$ ) [Koller and Friedman, 2009, 14.2.1].

### 2.1.2 Kullback-Leibler Divergence

The *Kullback-Leibler-(KL)-divergence* is a statistical distance which measures the difference from a base probability distribution  $p$  to a second distribution  $q$  [Bishop, 2006, Chapter 1]. This divergence is asymmetric in the two distributions  $p$  and  $q$ , so that we distinguish between the *forward KL* and *reverse KL* denoted as  $\text{KL}(q||p)$  and  $\text{KL}(p||q)$ , respectively.

**Definition 2.3 (Reverse Kullback-Leibler-(KL)-Divergence)** For continuous probability densities  $p$  and  $q$ , the (reverse) Kullback-Leibler-(KL)-divergence between  $p(\mathbf{z})$  and  $q(\mathbf{z})$  is defined as

$$\text{KL}[p(\mathbf{z}) || q(\mathbf{z})] = \int p(\mathbf{z}) \log \frac{p(\mathbf{z})}{q(\mathbf{z})} d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})} \left[ \log \frac{p(\mathbf{z})}{q(\mathbf{z})} \right]. \quad (2.12)$$

Similarly, the definition of the forward KL is obtained by replacing  $p$  and  $q$  in Definition 2.3.

## 2.1.2.1 KL between Two Multivariate Gaussians

The (reverse) KL between two  $M$ -variate Gaussians  $p(\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$  and  $q(\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$  satisfies

$$2 \text{KL}[p(\mathbf{z})||q(\mathbf{z})] = \text{tr}(\boldsymbol{\Sigma}_q^{-1}\boldsymbol{\Sigma}_p) - M + (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^T \boldsymbol{\Sigma}_q^{-1}(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p) + \log \frac{|\boldsymbol{\Sigma}_q|}{|\boldsymbol{\Sigma}_p|}. \quad (2.13)$$

## 2.1.2.2 Minimization of KL

In statistical inference, the KL can be used to approximate an intractable distribution  $p$  with a simpler distribution  $q$ . That is, the task is to find a distribution  $q$  with a particular structure which is close to the true distribution  $p$  in the KL-divergence sense. In particular, minimizing the reverse KL can be formulated as

$$q^*(\mathbf{z}) = \arg \min_{q(\mathbf{z})} \text{KL}[p(\mathbf{z}) || q(\mathbf{z})]$$

and similarly the minimization of the forward KL

$$q^*(\mathbf{z}) = \arg \min_{q(\mathbf{z})} \text{KL}[q(\mathbf{z}) || p(\mathbf{z})].$$

Note that the minimization of the forward KL often constitutes a simpler optimization task, since the expectation in (2.12) is taken with respect to the approximate distribution  $q$  instead of the intractable distribution  $p$ . In Figure 2.2, we provide an example for the two directions of the KL minimization between a mixture of Gaussians  $p$  and approximate Gaussian distribution  $q$ .

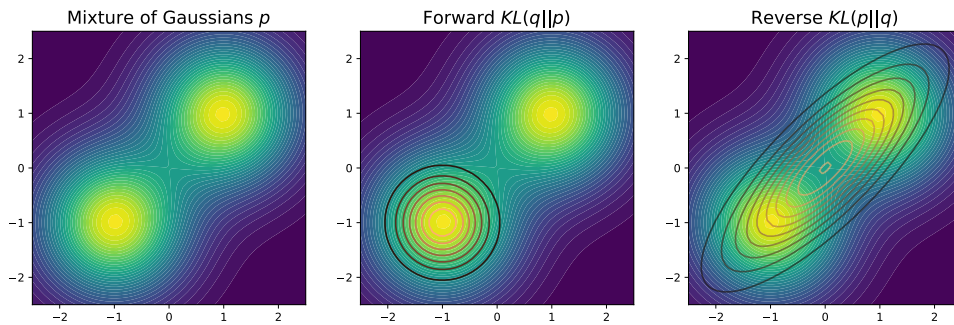


Figure 2.2. Illustration of forward and reverse KL minimization between a mixture of Gaussians  $p$  and approximate distribution  $q$ .



### 2.1.3 Numerical Optimization

In this section, we briefly introduce some basic concepts from numerical optimization which are extensively needed later for our proposed methods. In particular, we present two algorithms from deterministic and stochastic optimization. We refer to [Wright et al., 1999] for more details about these topics.

For a scalar-valued function  $\mathcal{L}(\boldsymbol{\theta}) : \mathbb{R}^p \rightarrow \mathbb{R}$  with multivariate parameter  $\boldsymbol{\theta} = [\theta_1, \dots, \theta_p]^T \in \mathbb{R}^p$ , we formulate a minimization or maximization problem as

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \mathbb{R}^p} \mathcal{L}(\boldsymbol{\theta}) = \arg \max_{\boldsymbol{\theta} \in \mathbb{R}^p} -\mathcal{L}(\boldsymbol{\theta}),$$

where we see that each maximization task can be transformed into a minimization task by considering the negative objective function. Ideally, we would like to compute the optimal solution  $\boldsymbol{\theta}^*$  analytically, however, for many interesting problems, this is not possible, so that we have to resort to numerical approaches. We briefly introduce two numerical optimization categories, namely *deterministic* and *stochastic* approaches. In this work, we interpret the objective function  $\mathcal{L}$  as a loss function depending on observed data  $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_J\}$  with  $J$  mini-batches  $\mathcal{D}_j$ . In the deterministic case, we consider optimization tasks of the form

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \mathbb{R}^p} \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}),$$

where the whole data  $\mathcal{D}$  is involved, whereas in the stochastic case, we assume that the loss function can be decomposed into a sum of objective functions  $\mathcal{L}_j$  only depending on the mini-batch of data  $\mathcal{D}_j$ , that is,

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \mathbb{R}^p} \frac{1}{J} \sum_{j=1}^J \mathcal{L}_j(\boldsymbol{\theta}, \mathcal{D}_j).$$

#### 2.1.3.1 Deterministic Optimization

Deterministic numerical optimization algorithm can be further categorized into derivative-free (for instance [Nelder and Mead, 1965]), first-order (e.g. gradient descent [Cauchy et al., 1847]) or second-order algorithms (for instance Newton-method e.g. [Wright et al., 1999] or BFGS [Fletcher, 1987]). In the following of this work, we assume that the function  $\mathcal{L}$  is differentiable, so that the corresponding gradient vector can be computed  $\Delta \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) = [\frac{\partial \mathcal{L}(\boldsymbol{\theta}, \mathcal{D})}{\partial \theta_1}, \dots, \frac{\partial \mathcal{L}(\boldsymbol{\theta}, \mathcal{D})}{\partial \theta_p}]^T \in \mathbb{R}^p$ , involving the partial derivatives  $\frac{\partial \mathcal{L}(\boldsymbol{\theta}, \mathcal{D})}{\partial \theta_p}$ . We briefly discuss *gradient descent (GD)*

algorithm [Cauchy et al., 1847], which is an iterative procedure, where in each step the parameter  $\boldsymbol{\theta}$  is updated in direction of the negative gradient direction,

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \gamma_t \Delta \mathcal{L}(\boldsymbol{\theta}^{(t)}, \mathcal{D}), \quad (2.14)$$

with a small learning rate  $\gamma_t \in \mathbb{R}_+$  and initial guess  $\boldsymbol{\theta}^{(0)}$ . Under some conditions [Fletcher, 2005], gradient descent is guaranteed to converge to a local minimum of the objective function and we refer to Wright et al. [1999] for more details about this topic.

### 2.1.3.2 Stochastic Optimization

*Stochastic gradient descent (SGD)* [Robbins and Monro, 1951] is a stochastic approximation of gradient descent, since for large data  $\mathcal{D}$ , the gradient computation  $\Delta \mathcal{L}(\boldsymbol{\theta}, \mathcal{D})$  is expensive. We assume that the loss function decomposes into a sum of  $J$  terms  $\mathcal{L}_j$ , each involving only mini-batch  $\mathcal{D}_j$ , that is,

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) = \frac{1}{J} \sum_{j=1}^J \mathcal{L}_j(\boldsymbol{\theta}, \mathcal{D}_j).$$

In this case, the gradient update in (2.14) is replaced by

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \gamma_t \Delta \mathcal{L}_j(\boldsymbol{\theta}^{(t)}, \mathcal{D}_j), \quad (2.15)$$

where mini-batch  $\mathcal{D}_j$  is selected randomly at each step  $t$ . This classic stochastic gradient descent update scheme is generally sensitive to appropriate learning rates  $\gamma_t$ . In order to achieve fast convergence, one is tempted to increase the learning rates, however, this might induce numerical instability. There are several approaches for improved and adaptive learning rates, consider for instance [Rumelhart et al., 1986; Duchi et al., 2011; Kingma and Ba, 2014]. However, all these methods still depend on a parameter, which has to be carefully chosen manually. If not otherwise stated in the following of this work by referring to SGD, we mean the algorithm of [Kingma and Ba, 2014].

## 2.1.4 Useful Properties from Linear Algebra

### 2.1.4.1 Inversion and Determinant Lemma

Consider invertible matrices  $\mathbf{A} \in \mathbb{R}^{B \times B}$ ,  $\mathbf{C} \in \mathbb{R}^{M \times M}$  and matrices  $\mathbf{U} \in \mathbb{R}^{B \times M}$ ,  $\mathbf{V} \in \mathbb{R}^{M \times B}$ . In this case, we can compute the inverse

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1} \mathbf{U} (\mathbf{C}^{-1} + \mathbf{VA}^{-1} \mathbf{U})^{-1} \mathbf{VA}^{-1} \quad (2.16)$$

and the determinant

$$|\mathbf{A} + \mathbf{UCV}| = |\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U}||\mathbf{C}||\mathbf{A}|. \quad (2.17)$$

#### 2.1.4.2 Inversion and Determinant of Block Matrices

Given an invertible and symmetric block matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix}.$$

The inverse  $\mathbf{M}^{-1}$  can be computed as

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{Z}^{-1}\mathbf{B}^T\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{Z}^{-1} \\ -\mathbf{Z}^{-1}\mathbf{B}^T\mathbf{A}^{-1} & \mathbf{Z}^{-1} \end{bmatrix} \quad (2.18)$$

with  $\mathbf{Z} = \mathbf{D} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}$ . Moreover, the determinant  $|\mathbf{M}|$  can be computed as

$$|\mathbf{M}| = |\mathbf{A}| |\mathbf{D} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}| = |\mathbf{D}| |\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^T|. \quad (2.19)$$

## 2.2 Linear Basis Function Model

In statistics and machine learning, often mappings  $f : \mathcal{X} \rightarrow \mathcal{Y}$  from some *input* domain  $\mathcal{X}$  to some *output* domain  $\mathcal{Y}$  are modelled. In the supervised setting, that is, given  $N$  observed data samples  $\mathcal{D} = \{y_i, \mathbf{x}_i\}_{i=1}^N$  with  $\mathbf{x}_i \in \mathcal{X}$  and  $y_i \in \mathcal{Y}$ , the task is to learn the unknown function  $f(\mathbf{x})$  for each  $\mathbf{x} \in \mathcal{X}$  so that the observed output values are close to the function values of the inputs, that is  $y_i \approx f(\mathbf{x}_i)$ . If  $\mathcal{Y} = \mathbb{R}$ , the task is also known as *regression*, if  $\mathcal{Y}$  is categorical, it is called *classification*. In this thesis, we mainly focus on the regression case, therefore, we assume  $\mathcal{Y} = \mathbb{R}$  if not otherwise stated. Moreover, we assume  $\mathcal{X} = \mathbb{R}^D$  for the sake of simplicity, however, the most concepts can be adapted to a more general space.

In this section, we introduce a *parametric* linear model with basis functions, in particular, we consider the following model  $f : \mathbb{R}^D \rightarrow \mathbb{R}$ ,

$$f(\mathbf{x}) = \sum_{j=1}^M w_j \phi_j(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}, \quad (2.20)$$

involving the input vector  $\mathbf{x} = [x_1, \dots, x_D]^T \in \mathbb{R}^D$ , the weight vector  $\mathbf{w} = [w_1, \dots, w_M]^T \in \mathbb{R}^M$  and the basis function vector  $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \dots, \phi_M(\mathbf{x})]^T \in \mathbb{R}^M$

$\mathbb{R}^M$  with the  $j$ th basis function  $\phi_j(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}$ . Importantly, this model is linear in the *weights*  $\mathbf{w}$  and not in the *input*  $\mathbf{x}$ , which is illustrated in Figure 2.3 for different basis functions  $\phi(\mathbf{x})$ .

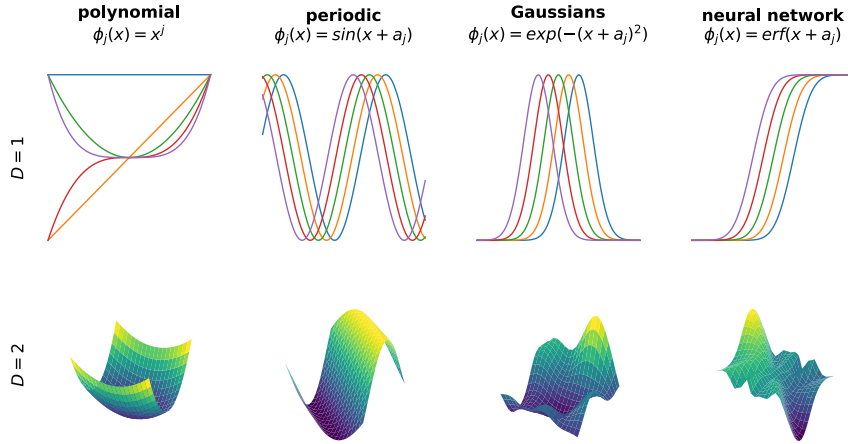


Figure 2.3. Examples of basis functions  $\phi(\mathbf{x})$  in one and two dimension which illustrates that the linear basis function model in (2.20) represent a very broad class of complicated functions (for instance also neural networks).

### 2.2.1 Additive Gaussian Noise

We further assume that the observed values  $y_1, \dots, y_N$  are perturbed observations of the underlying function values  $f(\mathbf{x}_1), \dots, f(\mathbf{x}_N)$ . In particular, an additive noise model is assumed, that is,

$$y_i = y(\mathbf{x}_i) = f(\mathbf{x}_i) + \varepsilon_i, \quad (2.21)$$

where  $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$  are independent Gaussian variables with zero mean and noise variance  $\sigma_n^2$ . These assumptions give rise to the *likelihood* function

$$p(y_i | \mathbf{w}) = \mathcal{N}(y_i | \phi(\mathbf{x}_i)^T \mathbf{w}, \sigma_n^2),$$

which describes the probability of the output observation  $y_i \in \mathbb{R}$  given the parameters  $\mathbf{w} \in \mathbb{R}^M$ . Due to the independence assumption of the noise, the joint likelihood factorizes over the  $N$  output samples  $\mathbf{y} = [y_1, \dots, y_N] \in \mathbb{R}^N$ . Therefore, the joint likelihood can be written as

$$p(\mathbf{y} | \mathbf{w}) = \prod_{i=1}^N p(y_i | \mathbf{w}) = \prod_{i=1}^N \mathcal{N}(y_i | \phi(\mathbf{x}_i)^T \mathbf{w}, \sigma_n^2) = \mathcal{N}(\mathbf{y} | \Phi_{\mathbf{x}} \mathbf{w}, \sigma_n^2 \mathbb{I}), \quad (2.22)$$

where the input matrix is denoted as  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^T \in \mathbb{R}^{N \times D}$  and the basis function matrix  $\Phi_{\mathbf{X}} = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_N)]^T$ , which is particularly defined as

$$\Phi_{\mathbf{X}} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{N \times M}.$$

Note that, there are many other choices for the noise model resulting in different likelihood distributions. For instance, if we assume a Bernoulli likelihood leads to logistic regression, which corresponds to linear classification [Murphy, 2012, Section 1.4.6].

## 2.2.2 Point Estimation for Regression

### 2.2.2.1 Least-Squares Approach

The *least-squares* (LS) approach tries to find a solution for  $\mathbf{y} \approx f(\mathbf{X})$  by minimizing the difference between the observed outputs  $\mathbf{y}$  with the function evaluations  $f(\mathbf{X})$  of the input  $\mathbf{X}$  in the  $L^2$ -norm sense. This can be formulated as

$$\begin{aligned} \mathbf{w}_{LS} &= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - f(\mathbf{X})\|_2^2 = \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}\|_2^2 \\ &= (\Phi_{\mathbf{X}}^T \Phi_{\mathbf{X}})^{-1} \Phi_{\mathbf{X}}^T \mathbf{y}, \end{aligned} \quad (2.23)$$

where the solution can be obtained by setting the derivative w.r.t.  $\mathbf{w}$  to zero. Since the least-squares approach is prone to overfitting or the matrix  $\Phi_{\mathbf{X}}^T \Phi_{\mathbf{X}}$  could even be singular (for instance if there are many correlated features or even  $M > N$ ), a common approach is to introduce a regularization term in the objective function to balance between fit and regularity. This leads to *regularized least-squares* or *ridge regression*

$$\begin{aligned} \mathbf{w}_{ridge} &= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ &= (\Phi_{\mathbf{X}}^T \Phi_{\mathbf{X}} + \lambda \mathbb{I})^{-1} \Phi_{\mathbf{X}}^T \mathbf{y}, \end{aligned} \quad (2.24)$$

where  $\lambda$  controls the influence of the regularization and is added in the solution to the diagonal of  $\Phi_{\mathbf{X}}^T \Phi_{\mathbf{X}}$ , which has the effect that all eigenvalues of the matrix  $\Phi_{\mathbf{X}}^T \Phi_{\mathbf{X}}$  are shifted away from 0. Regularization allows training of complex models without severe over-fitting, essentially by limiting the effective model complexity. However, finding the optimal  $\lambda$  in ridge regression is crucial and not an easy task. Ideally, we would like to find the  $\lambda$  which gives a minimal error on an

additional validation dataset  $\mathcal{D}_{\text{val}} = \{y_j, \mathbf{x}_j\}_{j=1}^V$ , which is different to the training data  $\mathcal{D} = \{y_i, \mathbf{x}_i\}_{i=1}^N$ . Thereby, the size  $V$  should be as large as possible, however, large validation sets are often not realistic in practice, so that the most common approach is to select  $\lambda$  via cross-validation based on different splits of the training dataset. We refer to [Bishop, 2006, Ch. 3] for more details.

**Example 2.1 (Polynomial Regression)** For input space  $D = 1$  and thus  $x \in \mathbb{R}$ , we consider the polynomial regression model

$$f_p(x) = \sum_{j=0}^P w_j x^j = \phi(x)^T \mathbf{w},$$

which is an instance of the linear basis function model (2.20) with the polynomial basis function

$$\phi(x) = [1, x, x^2, \dots, x^P]^T \in \mathbb{R}^M$$

for some degree  $P = M - 1$ . The basis function matrix can be computed by

$$\Phi_X = \begin{bmatrix} 1 & x_1 & \cdots & x_1^P \\ 1 & x_2 & \cdots & x_2^P \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & \cdots & x_N^P \end{bmatrix} \in \mathbb{R}^{N \times M}.$$

We want to compare the solution  $\mathbf{w}_{LS}$  for least-squares in (2.23) and the regularized version  $\mathbf{w}_{ridge}$  in (2.24) for the models  $f_p(x)$  for  $P = 1, \dots, 10$ . We generated  $N = 14$  training data samples from a sine function with additive Gaussian noise. The results are illustrated in Figure 2.4 for  $\mathbf{w}_{LS}$  in blue and  $\mathbf{w}_{ridge}$  in orange. We can observe, that for increasing  $P$  both estimates can represent more complex functions, however, for the regularized version, there is a smoothing effect compared to the unregularized. The numbers in the bottom left corner indicate the root-mean-squared-error (RMSE), which is defined as  $\sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2}$  for the training data  $\mathcal{D} = \{y_i, \mathbf{x}_i\}_{i=1}^N$  and  $\sqrt{\frac{1}{M} \sum_{j=1}^M (y_j - f(\mathbf{x}_j))^2}$  for an additional validation data set  $\mathcal{D}_{\text{val}} = \{y_j, \mathbf{x}_j\}_{j=1}^M$ . In this example, we generated  $M = 100$  validation data samples, so that the validation error is a very good estimate for the true generalization error. Note that, such a big validation set is not realistic for real data, we use it here just for illustration purposes. For each order  $P$ , the optimal  $\lambda_*$  is chosen such that the RMSE error is minimized on the validation set. We notice, that the training error for ordinary least-squares is minimal for  $P = 10$ , however, the validation error is best for  $P = 5$ . On the other hand, for regularized least-squares, the training error is minimal for  $P = 7$  and validation error for  $P = 5$ . Note that for

ridge regression, the training error is a much better estimate for the validation error, which means that a better trade-off between fitting the data and generalization is achieved. Consider also Example 2.2 and Figure 2.5.

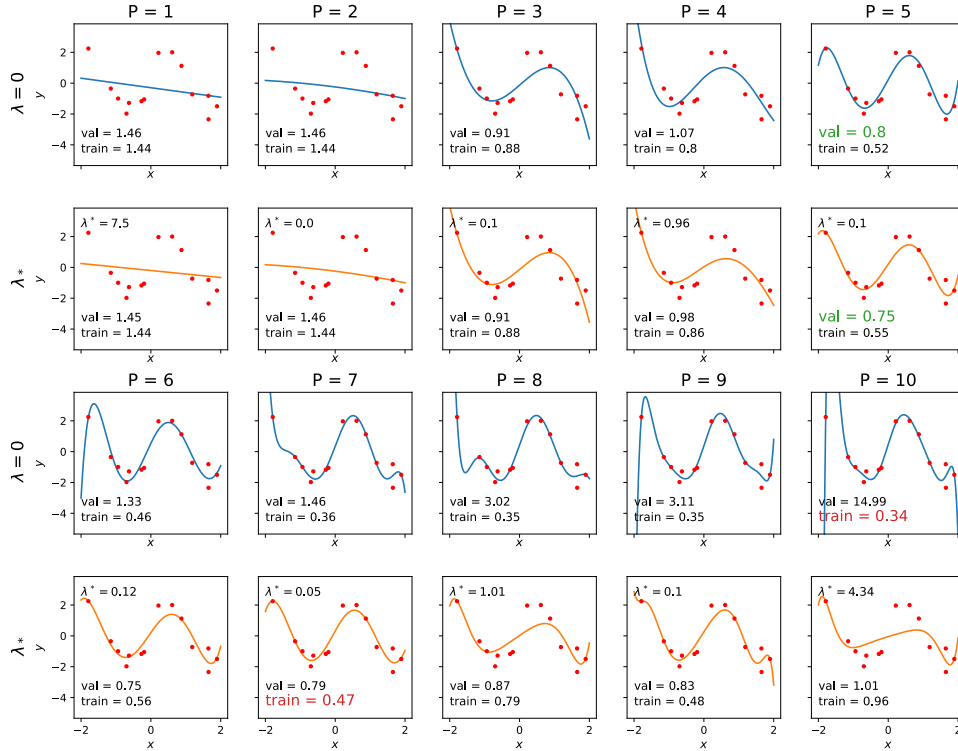


Figure 2.4. Polynomial regression with least-squares (blue) and ridge regression (orange) for Example 2.1.

### 2.2.2.2 Probabilistic Interpretation of Least-Squares

The point estimates for the weights in the previous section can also be derived with a probabilistic interpretation. In particular, we can maximize the likelihood function  $p(\mathbf{y}|\mathbf{w})$  (2.22), leading to

$$\begin{aligned}
 \mathbf{w}_{ML} &= \arg \max_{\mathbf{w} \in \mathbb{R}^M} p(\mathbf{y}|\mathbf{w}) = \arg \max_{\mathbf{w} \in \mathbb{R}^M} \log p(\mathbf{y}|\mathbf{w}) \\
 &= \arg \max_{\mathbf{w} \in \mathbb{R}^M} \log \mathcal{N}(\mathbf{y} | \Phi_{\mathbf{X}} \mathbf{w}, \sigma_n^2 \mathbf{I}) \\
 &= \arg \max_{\mathbf{w} \in \mathbb{R}^M} -\frac{1}{2\sigma_n^2} (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w})^T (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}) \\
 &= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}\|_2^2 = \mathbf{w}_{LS}.
 \end{aligned}$$

Therefore, we can understand the least-squares approach as *maximum likelihood* (ML) estimation.

**Definition 2.4 (Maximum Likelihood I (ML-I))** *Maximizing the likelihood distribution  $p(\mathbf{y}|\mathbf{w})$  as a function of the parameter  $\mathbf{w}$ , i.e.*

$$\mathbf{w}^* = \arg \max_{\mathbf{w} \in \mathbb{R}^M} p(\mathbf{y}|\mathbf{w})$$

*is called the maximum likelihood approach. For reasons which will be clear later, we denote this type of approach maximum likelihood inference of level I, abbreviated as ML-I.*

The probabilistic analog of regularization is to incorporate some additional assumptions of the parameters by specifying a distribution prior to seeing the data. For instance, we could assume a Gaussian distribution for the weights  $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma_0^2 \mathbb{I})$  with mean zero and isotropic covariance with variance  $\sigma_0^2$ , which corresponds to the assumption that all parameter are pairwise independent and they are most likely around zero with some standard deviation  $\sigma_0$ , however, large values are unlikely. Instead maximizing only the likelihood as above, we can instead maximize the likelihood times the distribution of the weights, consider for instance [Bishop, 2006, Ch. 3].

**Definition 2.5 (Maximum a Posteriori I (MAP-I))** *Maximizing the product of the likelihood  $p(\mathbf{y}|\mathbf{w})$  times a prior distribution  $p(\mathbf{w})$  as a function of the parameter  $\mathbf{w}$ , that is,*

$$\mathbf{w}^* = \arg \max_{\mathbf{w} \in \mathbb{R}^M} p(\mathbf{y}|\mathbf{w}) p(\mathbf{w})$$

*is called the maximum-a-posteriori approach, here abbreviated as MAP-I.*

Note that we will thoroughly discuss prior distributions in the next sections. We continue here to show that the solution of MAP for the weights correspond ex-



actly to the solution obtained by ridge regression.

$$\begin{aligned}
\mathbf{w}_{MAP} &= \arg \max_{\mathbf{w} \in \mathbb{R}^M} p(\mathbf{y}|\mathbf{w})p(\mathbf{w}) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^M} \log p(\mathbf{y}|\mathbf{w}) + \log p(\mathbf{w}) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^M} \log \mathcal{N}(\mathbf{y}|\Phi_X \mathbf{w}, \sigma_n^2 \mathbb{I}) + \log \mathcal{N}(\mathbf{0}, \sigma_0^2 \mathbb{I}) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^M} -\frac{1}{2\sigma_n^2}(\mathbf{y} - \Phi_X \mathbf{w})^T(\mathbf{y} - \Phi_X \mathbf{w}) - \frac{1}{2\sigma_0^2} \mathbf{w}^T \mathbf{w} \\
&= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \frac{1}{\sigma_n^2} \|\mathbf{y} - \Phi_X \mathbf{w}\|_2^2 + \frac{1}{\sigma_0^2} \|\mathbf{w}\|_2^2 \\
&= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - \Phi_X \mathbf{w}\|_2^2 + \frac{\sigma_n^2}{\sigma_0^2} \|\mathbf{w}\|_2^2 \\
&= \mathbf{w}_{ridge}.
\end{aligned}$$

We note that MAP-I for a Gaussian likelihood with noise variance  $\sigma_n^2$  and a Gaussian prior with variance  $\sigma_0^2$  correspond to ridge regression with  $\lambda = \frac{\sigma_n^2}{\sigma_0^2}$ . In the case  $\sigma_0 = 1$ , which is reasonable when the output data is standardized, finding the regularization parameter  $\lambda$  in ridge regression correspond to estimating the noise in the data. For  $\sigma_0^2 \rightarrow \infty$ , the regularization term  $\lambda \rightarrow 0$  vanish, which means no constraints for the weights. In this case, the prior converges to an uniform distribution and we can note that ML/LS estimation correspond to MAP with uniform prior.

**Example 2.2 (Generalization Capacity and Model Selection)** *For the Example 2.1 with Figure 2.4, the generalization capacity of the two models ordinary least squares and ridges regression are compared for choosing the optimal model complexity. In Figure 2.5, the RMSE of the training and validation data is depicted for least-squares and ridge regression for increasing polynomial order  $P$ . We observe that the training error for least-squares decreases for increasing order  $P$ , however, the validation error becomes very bad, which means that this model has severe difficulties with overfitting the data. On the other hand, the training error for ridge regression is a good estimate for the validation error, which means that this model can generalize its learned pattern well to new data points. For both models, the optimal model according to the validation error is for  $P = 5$ , but according the training error it is  $P = 10$  and  $P = 7$ , respectively. In particular, the training error for least-squares is always minimal for the highest considered  $P$ . Note that the performance of ridge regression depends strongly on the size and quality of the validation set. In this toy example, the validation set represents basically the truth, so that it is clear,*

that the training error correlates strongly with the validation error when choosing the optimal  $\lambda_*$  with the validation set.

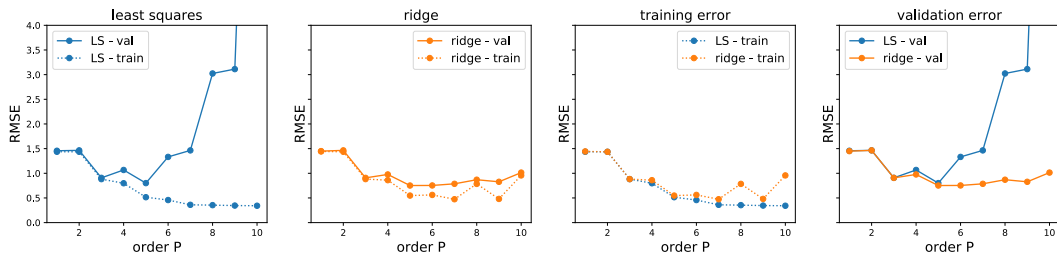


Figure 2.5. Generalization capacity for least-squares (blue) and ridge regression (orange) for increasing polynomial order  $P$  in Example 2.2. The training error for least-squares is not a good estimate for the validation error, whereas the training error correlates strongly with the validation of ridge-regression.

### 2.2.2.3 Limitations of Non-Bayesian Approaches

Many classical machine learning approaches for finding the optimal parameters of a model are based on optimizing a loss function, as for instance illustrated by the  $L^2$  norm in the least-squares case. This approach constitutes a well-founded and flexible tool for inference of many models, however, the generalization capacity is intrinsically limited. In particular, when only a small number of training data samples are available compared to the complexity of the model (e.g. number of parameters), they might suffer from severe overfitting issues. Adding a regularization term to the loss function as illustrated for ridge regression partially solves the generalization problem. However, finding the regularization parameter via validation-based approaches (e.g. cross-validation) is computationally expensive and wasteful of valuable data. In Section 2.2.2.2, we have seen that regularization can be understood as incorporating probabilistically additional knowledge about the solution before seeing the data. Bayesian probabilistic inference generalize this approach, which will be discussed in the next Section 2.3.

## 2.3 Bayesian Probabilistic Inference

In this section, we briefly discuss the main concepts of Bayesian inference and Bayesian model selection. The description is general but with a focus on aspects which will be important for GPs. In particular, we discuss Bayes' rule in Section 2.3.1, Bayesian hierarchical models in Section 2.3.2, practical consideration in Section 2.3.3, and the Bayesian treatment of the linear basis function model in Section 2.3.4. This section is solely based on [Murphy, 2012, 5.4-5.6], [Rasmussen and Williams, 2006, Ch. 5], and [Bishop, 2006, Ch. 3].

### 2.3.1 Bayes' Rule

In a nutshell, *Bayes' rule* can be seen as transforming the belief about a parameter  $\mathbf{z}$  *before* and *after* seeing the data  $\mathcal{D}$ . In particular, it allows to combine prior information about the parameter  $\mathbf{z}$  with the observed data  $\mathcal{D}$  to a distribution representing the information of the parameter incorporating the data. In mathematical terms, this can be formulated as

$$p(\mathbf{z}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{z}) p(\mathbf{z})}{p(\mathcal{D})}, \quad (2.25)$$

where we assume  $p(\mathcal{D}) > 0$  and  $p(\mathbf{z})$  is the *prior* distribution encoding our prior information or assumptions about the parameter  $\mathbf{z}$  before seeing the data. On the other hand, the term  $p(\mathbf{z}|\mathcal{D})$  on the left in (2.25) is the *posterior* distribution including the information from the data, which encodes useful knowledge about the data summarized by the parameter. The prior and the posterior distributions are connected by the *likelihood* distribution  $p(\mathcal{D}|\mathbf{z})$ , which describes how likely the data is for a given parameter and is used to probabilistically describe the data generating process. Note that the posterior distribution is proportional to the product of the likelihood and the prior

$$p(\mathbf{z}|\mathcal{D}) \propto p(\mathcal{D}|\mathbf{z}) p(\mathbf{z}),$$

since the denominator  $p(\mathcal{D})$  in (2.25) is independent of the parameter  $\mathbf{z}$ . The denominator is acting as a normalization constant and can thus be computed by integrating over the nominator in (2.25)

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z}. \quad (2.26)$$

This quantity is called *marginal likelihood* (or also *evidence*) and describes the probability distribution of the data.

### 2.3.2 Bayesian Hierarchical Model

Usually, a Bayesian model is specified as a hierarchical model of parameters. In our case, we assume that the main *parameter*  $\mathbf{w}$  are at the first level. For instance in the linear basis function model in Section 2.2, all weights for the basis functions correspond to the main parameters. At the second level are the *hyperparameters*  $\boldsymbol{\theta}$ , which control the distribution of the parameters at the first level. For instance, the noise variance  $\sigma_n^2$  or the prior variance  $\sigma_0^2$  discussed in Section 2.2.2.2 in the prior distribution for  $\mathbf{w}$  belongs to this category. At the bottom level, there is usually a (discrete) set of possible models  $\mathcal{M}$  under consideration. In the linear basis function model this might correspond to the models with different number of basis functions.

Bayesian inference corresponds to repetitively applying *Bayes' rule* (2.25) to each parameter at each level of the hierarchical model. This is shown in the following for the model with three levels with parameters  $\mathbf{w}$ ,  $\boldsymbol{\theta}$  and  $\mathcal{M}$  and data  $\mathcal{D} = \{\mathbf{y}, \mathbf{X}\}$ , where we assume that the input data  $\mathbf{X}$  is deterministic and fixed. First, we have to specify all prior distributions, that is, we have to encode our prior assumptions of the model  $p(\mathcal{M})$ , the hyperparameters  $p(\boldsymbol{\theta}|\mathcal{M})$  in a given model and the parameters  $p(\mathbf{w}|\boldsymbol{\theta}, \mathcal{M})$  for a given model and fixed hyperparameters, respectively. Note that those choices strongly depend on the particular application, however, in this section they are assumed to be general and will be specified in the subsequent sections. Additionally to the priors, we have to specify the likelihood on the first level

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M}) \tag{2.27}$$

determining the data generating process for a fixed model, fixed input data and given parameter and hyperparameter. For instance in the linear basis function model, we have discussed the likelihood  $p(\mathbf{y}|\mathbf{w})$  with Gaussian additive noise in Section 2.2.1. In a Bayesian setting, the likelihood can be always implicitly understood as the likelihood in (2.27), that is, conditioned on the input data, the hyperparameters in a fixed model; and similarly for the prior distributions.

**Remark 2.1** *In the rest of this thesis - except this section - when discussing Bayesian inference on the **first** level in a Bayesian model, we usually omit the explicit dependencies on the parameters on the lower levels for the sake of simplicity. Concretely, if it is clear from the context and not otherwise stated, we assume that the hyperparameters  $\boldsymbol{\theta}$ , the input data  $\mathbf{X}$  and the model  $\mathcal{M}$  are already known or fixed.*

	variable		posterior	likelihood	prior	marginal lik.	equation
I	parameter	$\mathbf{w}$	$p(\mathbf{w} \mathbf{y}, \mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$	$p(\mathbf{y} \mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})$	$p(\mathbf{w} \boldsymbol{\theta}, \mathcal{M})$	$p(\mathbf{y} \mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$	(2.28), (2.29)
II	hyperpar.	$\boldsymbol{\theta}$	$p(\boldsymbol{\theta} \mathbf{y}, \mathbf{X}, \mathcal{M})$	$p(\mathbf{y} \mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$	$p(\boldsymbol{\theta} \mathcal{M})$	$p(\mathbf{y} \mathbf{X}, \mathcal{M})$	(2.30), (2.31)
III	model	$\mathcal{M}$	$p(\mathcal{M} \mathbf{y}, \mathbf{X})$	$p(\mathbf{y} \mathbf{X}, \mathcal{M})$	$p(\mathcal{M})$	$p(\mathbf{y} \mathbf{X})$	(2.32), (2.33)

Table 2.1. Summary of Bayesian inference in an hierarchical model depicted for three variables  $\mathbf{w}$ ,  $\boldsymbol{\theta}$  and  $\mathcal{M}$ . Every prior on each level and the likelihood on the first level has to be specified, then the posterior for each parameter can be computed (if it is tractable) by repetitively applying Bayes' rule (2.25). Note that the marginal likelihood of the previous level acts as likelihood in the next level.

Generally, the posterior in a Bayesian model on the first level can be computed by applying Bayes' rule (2.25) the first time yielding

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) = \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M}) p(\mathbf{w}|\boldsymbol{\theta}, \mathcal{M})}{p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M})}. \quad (2.28)$$

Thereby, the prior over the parameter  $p(\mathbf{w}|\boldsymbol{\theta}, \mathcal{M})$  is combined with the specified likelihood (2.27). The denominator, which is called marginal likelihood of the first level, can be obtained by computing the integral over the numerator in (2.28)

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M}) p(\mathbf{w}|\boldsymbol{\theta}, \mathcal{M}) d\mathbf{w}. \quad (2.29)$$

Note that this is usually meant by "the" marginal likelihood in a Bayesian model, however, there is a marginal likelihood at each level. At the second level, we analogously express the posterior over the hyperparameters by combining the prior distribution  $p(\boldsymbol{\theta}|\mathcal{M})$  and the previous marginal likelihood (2.29) leading to the posterior

$$p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{X}, \mathcal{M}) = \frac{p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta}|\mathcal{M})}{p(\mathbf{y}|\mathbf{X}, \mathcal{M})}. \quad (2.30)$$

Note that the prior over the hyperparameters is also called *hyper-prior* distribution and we want to emphasize that the previous marginal likelihood on the level of the weights plays the role of the likelihood for the hyperparameters.

**Remark 2.2** *In a Bayesian hierarchical model, the **marginal likelihood** of the previous level acts as the likelihood for the current level.*

Again, the marginal likelihood (2.26) for the second level can be obtained by

$$p(\mathbf{y}|\mathbf{X}, \mathcal{M}) = \int p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta}|\mathcal{M}) d\boldsymbol{\theta}. \quad (2.31)$$

At the third level of the model, the posterior for the model can be formulated as

$$p(\mathcal{M}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{X}, \mathcal{M}) p(\mathcal{M})}{p(\mathbf{y}|\mathbf{X})}, \quad (2.32)$$

where again the previous marginal likelihood (2.31) is combined with the prior. Assuming that the models  $\mathcal{M}$  is a discrete random variable with range  $\mathcal{M} \in \{M_1, \dots, M_J\}$ , the normalization in the denominator can be computed as

$$p(\mathbf{y}|\mathbf{X}) = \sum_{j=1}^J p(\mathbf{y}|\mathbf{X}, \mathcal{M} = M_j) p(\mathcal{M} = M_j). \quad (2.33)$$

This is known as Bayesian model selection, as described for instance in [Bishop, 2006, Chapter 3].

### 2.3.3 Bayesian Inference in Practice

Computing the marginal likelihood requires, in general, the computation of multivariate integrals. Depending on the particular model, these integrals might not be analytically tractable and we have to resort either to analytical approximations (e.g. variational approach) [Bishop, 2006, Ch. 10] or Markov chain Monte Carlo (MCMC) methods [Bishop, 2006, Ch. 11]. We focus in this work on an analytic approach based on the marginal likelihood from the previous level, which is called *empirical Bayes* [Murphy, 2012, Section 5.6].

#### 2.3.3.1 Level II Inference: Hyperparameters

In particular, the computation of the integral in (2.31) might be difficult, since the marginal likelihood  $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$  of the hyperparameters are often a highly non-linear function in  $\boldsymbol{\theta}$ . Therefore, instead of computing the full posterior (2.30) of the second level, a common approach is to maximize the marginal likelihood in (2.29) with respect to the hyperparameters.

**Definition 2.6 (Maximum Likelihood II (ML-II))** *Maximizing the marginal likelihood  $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$  of the hyperparameters  $\boldsymbol{\theta}$  in the second level of a Bayesian model, that is,*

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$$

is called *maximum likelihood approach at level II* or also just *maximum marginal likelihood* [Murphy, 2012, Ch. 5].

Maximizing the marginal likelihood constitutes a cheap and often effective alternative for computing the intractable full posterior  $p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{X}, \mathcal{M})$ . However, of course, one should be careful with such an optimization step, since it opens up the possibility of overfitting, in particular if there are many hyperparameters. Note that in principle similar arguments can be made as in the discussion around the usual maximum likelihood approach of type I, as highlighted in Section 2.2.2. However, compared to ML-I or MAP-I treatment of the hyperparameters, an additional level of uncertainty is taken into account and the risk of overfitting is less severe for the second level, since the marginal likelihood has an interesting regularization property.

**Remark 2.3 (Regularization Effect of ML-II)** *The maximum likelihood type II approach, obtained by maximizing the marginal likelihood for the hyperparameters, has the property that it automatically incorporates a trade-off between model fit and model complexity. This can be explained by Eq. (2.29), which shows that the ML-II objective  $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$  is the expectation of the ML-I objective  $p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})$  under the prior distribution  $p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})$ , that is,*

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) = \mathbb{E}_{p(\mathbf{w}|\boldsymbol{\theta}, \mathcal{M})} [p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})].$$

*Thus, due to the expectation over  $\mathbf{w}$ , the ML-II is less prone to overfitting than the ML-I and incorporates implicitly the model complexity.*

Therefore, using the maximum marginal likelihood for finding the hyperparameters automatically prevents overfitting up to some degree and constitutes a valuable approach in many situations. This property is also illustrated in Example 2.4 and 2.5.

Alternatively, a further improvement to the ML-II approach is to still use priors on level II, but to only compute the *maximizer* of the posterior distribution instead the whole intractable posterior distribution in (2.30). This is based on the fact that it is easier in the most cases to only find the maximizer of a distribution than computing the whole distribution. In particular, the maximizer is independent of the normalization, which corresponds to compute an hard integral (2.31).

**Definition 2.7 (Maximum a Posteriori II (MAP-II))** *Using the hyperparameters, which correspond to the maximum of the posterior distribution, is called the maximum-*

*a-posterior type II (MAP-II) approach [Murphy, 2012, Ch. 5.6]. In particular, maximizing the posterior can be written as*

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{X}, \mathcal{M}) \\ &= \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta}|\mathcal{M}),\end{aligned}$$

where we exploited the fact, that for computing the maximizer, the integral in (2.31) is not needed.

The MAP-II approach constitutes often a cheap and effective alternative for computing the intractable full posterior  $p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{X}, \mathcal{M})$  but using some prior knowledge about the hyperparameters, as we will see in Example 2.4.

### 2.3.3.2 Level III Inference: Model Selection

Approximate inference at level II by a point estimate, and in particular not computing the normalization (2.31), has also consequences for inference at level III corresponding to model selection. We have seen that we can circumvent the computation of the normalization or marginal likelihood at level II (2.31) by a point estimate either by the ML-II or MAP-II approach and get still valuable information for the hyperparameters. However, this marginal likelihood  $p(\mathbf{y}|\mathbf{X}, \mathcal{M})$  at level II acts as likelihood at level III, which is needed for inference. A solution for approximate inference for the model  $\mathcal{M}$  in the third level can be obtained by plugging the maximizer  $\boldsymbol{\theta}^*$  into the marginal likelihood  $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}^*, \mathcal{M})$  and using a local expansion (e.g. Laplace approximation [Bishop, 2006, Chapter 10]) around  $\boldsymbol{\theta}^*$  for computing the integral  $p(\mathbf{y}|\mathbf{X}, \mathcal{M})$  (2.31). The prior over the model  $p(\mathcal{M})$  in (2.32) is often taken to be flat, so that no model is preferred over another a-priori. In this case, the probability for the model is proportional  $p(\mathcal{M}|\mathbf{y}, \mathbf{X}) \propto p(\mathbf{y}|\mathbf{X}, \mathcal{M})$  to the approximated marginal likelihood in (2.31) and can be used for model selection [Bishop, 2006, Chapter 3] as illustrated in Example 2.5.

### 2.3.4 Bayesian Linear Model

In this section, we discuss Bayesian inference for the linear basis function model introduced in Section 2.2, which is also known as *Bayesian linear regression* [Bishop, 2006, Ch. 3]. Note that, in this section we focus on Bayesian inference



of the weights, so that by likelihood, prior or posterior we refer to the corresponding distribution on the first level in a Bayesian hierarchical model. For the sake of simplicity, we omit the explicit conditioning on the input data and the chosen model as noted in Remark 2.1. The discussion about approaches for inference on the second level is provided in Section 2.3.4.6.

#### 2.3.4.1 Likelihood

We briefly recall the assumptions of the linear regression model in Section 2.2 with basis functions and Gaussian additive noise as introduced in Section 2.2.1. The model can be summarized by

$$\begin{aligned} y(x) &= f(\mathbf{x}) + \varepsilon, \\ f(\mathbf{x}) &= \phi(\mathbf{x})^T \mathbf{w}, \\ \varepsilon &\sim \mathcal{N}(\mathbf{0}, \sigma^2), \end{aligned}$$

which give rise to the Gaussian regression likelihood

$$p(\mathbf{y}|\mathbf{w}) = \mathcal{N}(\mathbf{y}|\Phi_{\mathbf{x}}\mathbf{w}, \sigma_n^2\mathbb{I}). \quad (2.34)$$

#### 2.3.4.2 Prior

In a Bayesian setting, a *prior* distribution is specified over the parameters  $\mathbf{w}$ , expressing our beliefs about the parameters before considering the data. We put a zero mean Gaussian prior with prior covariance  $\Sigma_0 \in \mathbb{R}^{M \times M}$  on the weights

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \Sigma_0), \quad (2.35)$$

so that Bayesian inference can be applied.

#### 2.3.4.3 Posterior

Bayes' theorem (2.25) for computing the *posterior* of the weights  $\mathbf{w}$  given the data  $\mathbf{y}$  can be formulated as

$$p(\mathbf{w}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{y})} \propto p(\mathbf{y}|\mathbf{w})p(\mathbf{w}), \quad (2.36)$$

where the likelihood (2.34) and the prior (2.35) are combined to compute the posterior knowledge. Since both distributions are Gaussian, the product is again Gaussian

$$\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{y}|\Phi_{\mathbf{x}}\mathbf{w}, \sigma_n^2\mathbb{I})\mathcal{N}(\mathbf{w}|\mathbf{0}, \Sigma_0), \quad (2.37)$$

where the posterior mean  $\boldsymbol{\mu}$  and posterior covariance  $\boldsymbol{\Sigma}$  of the weights  $\boldsymbol{w}$  can be analytically computed by (2.6), yielding

$$\boldsymbol{\mu} = \frac{1}{\sigma_n^2} \boldsymbol{\Sigma} \boldsymbol{\Phi}_X^T \boldsymbol{y} \quad \text{and} \quad \boldsymbol{\Sigma} = \left( \boldsymbol{\Sigma}_0^{-1} + \frac{1}{\sigma_n^2} \boldsymbol{\Phi}_X^T \boldsymbol{\Phi}_X \right)^{-1}. \quad (2.38)$$

Note that the mean  $\boldsymbol{\mu}$  has the same structure as the point estimate of ridge regression in (2.24), however, here we have additional information due to the posterior covariance. By applying the inversion lemma (2.16) to  $\boldsymbol{\Sigma}$ , we can alternatively write for the posterior moments

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}_0 \boldsymbol{\Phi}_X^T \boldsymbol{P}^{-1} \boldsymbol{y} \quad \text{and} \quad \boldsymbol{\Sigma} = \boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0 \boldsymbol{\Phi}_X^T \boldsymbol{P}^{-1} \boldsymbol{\Phi}_X \boldsymbol{\Sigma}_0, \quad (2.39)$$

where we introduce the covariance matrix

$$\boldsymbol{P} = \boldsymbol{\Phi}_X \boldsymbol{\Sigma}_0 \boldsymbol{\Phi}_X^T + \sigma_n^2 \mathbb{I}. \quad (2.40)$$

**Remark 2.4 (Efficient Inversion)** *Note that computing the posterior moments in (2.39) requires the inversion  $\boldsymbol{P}^{-1} \in \mathbb{R}^{N \times N}$  which is in  $\mathcal{O}(N^3)$ , whereas computing the posterior covariance  $\boldsymbol{\Sigma} \in \mathbb{R}^{M \times M}$  in (2.38) is  $\mathcal{O}(M^3)$ . Which one is preferable from a computational point of view depends whether the number of basis functions  $M$  or the number of data samples  $N$  is larger.*

#### 2.3.4.4 Prediction

For a new query point  $\boldsymbol{x}_* \in \mathbb{R}^D$ , we can find the predictive posterior distribution  $p(y(\boldsymbol{x}_*)|\boldsymbol{y})$  by averaging over all possible parameter values, weighted by their posterior probability, that is,

$$\begin{aligned} p(y(\boldsymbol{x}_*)|\boldsymbol{y}) &= \int p(y(\boldsymbol{x}_*)|\boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{y}) d\boldsymbol{w} \\ &= \int \mathcal{N}(y(\boldsymbol{x}_*) | \boldsymbol{\phi}(\boldsymbol{x}_*)^T \boldsymbol{w}, \sigma_n^2) \mathcal{N}(\boldsymbol{w} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\boldsymbol{w} \\ &= \mathcal{N}(y(\boldsymbol{x}_*) | \boldsymbol{\phi}(\boldsymbol{x}_*)^T \boldsymbol{\mu}, \boldsymbol{\phi}(\boldsymbol{x}_*)^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\boldsymbol{x}_*) + \sigma_n^2), \end{aligned} \quad (2.41)$$

where the integral of two Gaussians can be computed by (2.5). Using  $\boldsymbol{\mu}$  in Eq. (2.39) and introducing  $\boldsymbol{\phi}_* = \boldsymbol{\phi}(\boldsymbol{x}_*)^T \in \mathbb{R}^{1 \times M}$  to ease notation, the mean of the posterior predictive distribution (2.41) can be written as

$$\boldsymbol{\mu}(\boldsymbol{x}_*) = \boldsymbol{\phi}_* \boldsymbol{\mu} = \boldsymbol{\phi}_* \boldsymbol{\Sigma}_0 \boldsymbol{\Phi}_X^T \boldsymbol{P}^{-1} \boldsymbol{y} \quad (2.42)$$

and similarly for the variance with  $\Sigma$  in Eq. (2.39)

$$\sigma^2(\mathbf{x}_*) = \phi_* \Sigma \phi_*^T + \sigma_n^2 = \phi_* \Sigma_0 \phi_*^T - \phi_* \Sigma_0 \Phi_X^T \mathbf{P}^{-1} \Phi_X \Sigma_0 \phi_*^T + \sigma_n^2. \quad (2.43)$$

In order to summarize the predictive posterior distribution, we can compute the  $100(1 - \alpha)\%$ -credible interval, denoted as  $CI_{1-\alpha}$ , with limits  $[c_0, c_1]$  satisfying

$$\int_{c_0}^{c_1} p(y(\mathbf{x}_*) | \mathbf{y}) dy(\mathbf{x}_*) = 1 - \alpha.$$

Since the predictive posterior distribution is Gaussian, the 95%-credible interval for instance can be computed as

$$CI_{0.95}(\mathbf{x}_*) = \mu(\mathbf{x}_*) \pm 1.96 \sqrt{\sigma^2(\mathbf{x}_*)} \quad (2.44)$$

involving the mean and the variance of the predictive posterior distribution.

#### 2.3.4.5 Marginal Likelihood

Note that the denominator  $p(\mathbf{y})$  in (2.36) is independent of the weights  $\mathbf{w}$  and thus acting as a normalization constant. It can be computed for the linear Gaussian model by marginalizing or integrating out the weights from the product of the likelihood  $p(\mathbf{y}|\mathbf{w})$  in (2.22) times the prior  $p(\mathbf{w})$  in (2.35), that is,

$$\begin{aligned} p(\mathbf{y}) &= \int p(\mathbf{y}|\mathbf{w})p(\mathbf{w}) d\mathbf{w} \\ &= \int \mathcal{N}(\mathbf{y}|\Phi_X \mathbf{w}, \sigma_n^2 \mathbb{I}) \mathcal{N}(\mathbf{w}|\mathbf{0}, \Sigma_0) d\mathbf{w} \\ &= \mathcal{N}(\mathbf{y}|\mathbf{0}, \Phi_X \Sigma_0 \Phi_X^T + \sigma_n^2 \mathbb{I}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{P}), \end{aligned} \quad (2.45)$$

where we used (2.5) to compute the integral and we can recognize again the covariance matrix  $\mathbf{P}$  in (2.40).

**Example 2.3 (Bayesian Linear Regression, Level I)** Assume the polynomial mapping  $\phi(\mathbf{x}) = [1, x^{(1)}, \dots, x^{(P)}]$  for the model in (2.20) with  $P = 1$  and thus

$$f(x) = w_0 + w_1 x = [1, x][w_0, w_1]^T = \phi(x)^T \mathbf{w}. \quad (2.46)$$

We assume a zero-mean isotropic Gaussian prior over the parameters  $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbb{I})$  together with the Gaussian likelihood  $p(y_i|\mathbf{w}) = \mathcal{N}(y_i|\phi(\mathbf{x}_i)^T \mathbf{w}, \sigma_n^2)$  of  $N$  training data samples with inputs  $\mathbf{X} = [x_1, \dots, x_N]^T \in \mathbb{R}^{N \times 1}$  and the corresponding outputs  $\mathbf{y} = [y_1, \dots, y_N]^T \in \mathbb{R}^N$ . We generated  $N = 40$  data samples

from the true model in (2.46) with  $w_0 = -0.2$ ,  $w_1 = 0.5$  and fixed variances  $\sigma_0^2 = 0.3$ ,  $\sigma_n^2 = 0.03$ . In the top row of Figure 2.6, the likelihoods  $p(y_i|\mathbf{w}) = \mathcal{N}(y_i|\phi(\mathbf{x}_i)^T\mathbf{w}, \sigma_n^2)$  are shown for some samples  $i$  as a function in the weights  $\mathbf{w}$ , where the yellow areas indicate the most likely parameters  $\mathbf{w}$  which are consistent with the data sample  $(y_i, \mathbf{x}_i)$ . Defining the augmented input matrix  $\Phi_X = [\mathbf{1}, \mathbf{X}] \in \mathbb{R}^{N \times 2}$  and using (2.38), the posterior over the weights  $p(\mathbf{w}|\mathbf{y}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$  can be computed by

$$\boldsymbol{\mu} = \sigma_n^{-2} \boldsymbol{\Sigma} \Phi_X^T \mathbf{y} \in \mathbb{R}^2 \quad \text{and} \quad \boldsymbol{\Sigma}^{-1} = \sigma_0^{-2} \mathbb{I} + \sigma_n^{-2} \Phi_X^T \Phi_X \in \mathbb{R}^{2 \times 2}.$$

In the second row in Figure 2.6, the prior  $p(\mathbf{w})$  and several cumulative posteriors  $p(\mathbf{w}|\mathbf{y}_{1:i})$  are shown, where the latter is obtained by combining the prior and all previous likelihoods including the  $i$ th. We can observe that the yellow area for posterior  $i$ , corresponding to the most likely weights given all previous seen data  $\mathbf{y}_{1:i}$ , shrinks as the more data is included in the model, meaning that the confidence about the inferred estimate increases. For a query point  $\mathbf{x}_* \in \mathbb{R}$  and  $\phi_* = [1, \mathbf{x}_*] \in \mathbb{R}^{1 \times 2}$ , the predictive distribution  $p(y(\mathbf{x}_*)|\mathbf{w})$  with mean  $\mu(\mathbf{x}_*) = \phi_* \boldsymbol{\mu}$  and variance  $\sigma^2(\mathbf{x}_*) = \phi_* \boldsymbol{\Sigma} \phi_*^T + \sigma_n^2$  as given in Eqs. (2.42) and (2.43), respectively, can be used to compute the 95%-credible interval

$$CI_{0.95}(\mathbf{x}_*) = \mu(\mathbf{x}_*) \pm 1.96 \sqrt{\sigma^2(\mathbf{x}_*)}.$$

In the third row in Figure 2.6, the  $\mu(\mathbf{x}_*)$  (green solid line) as well as the credible interval  $CI_{0.95}(\mathbf{x}_*)$  (shaded area) are shown for a range of query points  $\mathbf{x}_*$  on the  $x$ -axis. We can observe that for increasing number of samples, the mean gets closer to the true function (dotted red line) and the size of the shaded area decreases, corresponding to the higher confidence of the predictive distribution. Consider Example 2.4 for the continuation of this example for level II inference.

#### 2.3.4.6 Level II Inference: Hyperparameters

In this section we discuss level II inference for the hyperparameters  $\boldsymbol{\theta}$ . In the previous section, the prior covariance  $\boldsymbol{\Sigma}_0$ , the noise variance  $\sigma_n^2$  and possible hyperparameters for the basis functions  $\phi$  are treated as known or fixed. However, in practice these hyperparameters are usually unknown and have to be inferred as well. As discussed in Section 2.3.3, in particular the normalization constant of the posterior at level II is in general analytically intractable due to the non-linear relationship. Here we focus on the empirical Bayesian approach as introduced in Section 2.3.3.

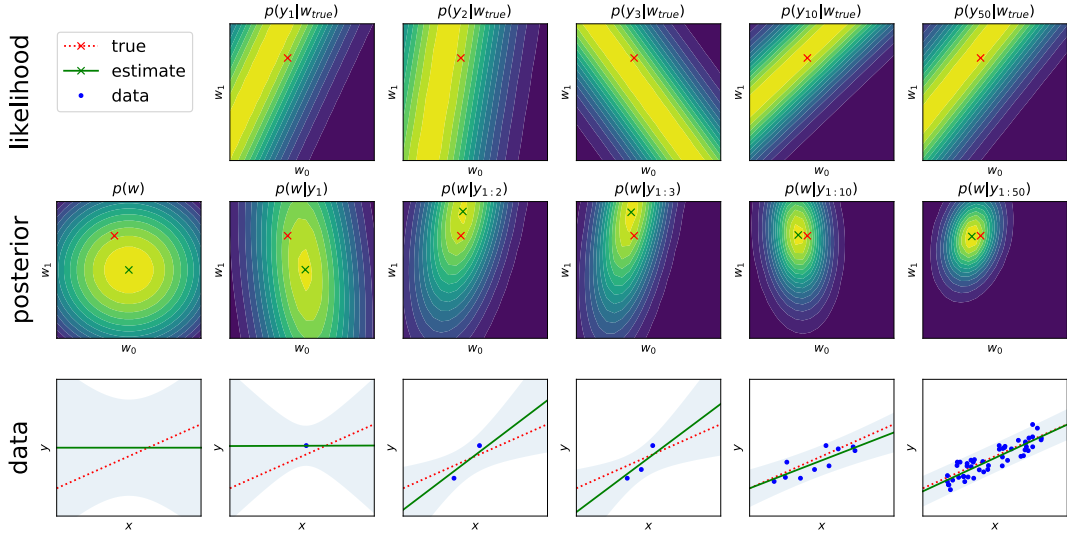


Figure 2.6. Illustration of Example 2.3 of Bayesian inference level II for linear regression.

### Maximum Likelihood II

According to Definition 2.6, we use the marginal likelihood  $p(\mathbf{y}) = p(\mathbf{y}|\boldsymbol{\theta})$  of the first level in (2.45) as the objective function, which yields

$$\begin{aligned}
 \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\boldsymbol{\theta}) \\
 &= \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) \\
 &= \arg \max_{\boldsymbol{\theta}} -\frac{1}{2} (\mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}| + N \log 2\pi) \\
 &= \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}|,
 \end{aligned} \tag{2.47}$$

where the marginal likelihood covariance in (2.40) is explicitly parametrized by  $\boldsymbol{\theta}$ , that is,  $\mathbf{P}_{\boldsymbol{\theta}} = \Phi_X \Sigma_0 \Phi_X^T + \sigma_n^2 \mathbb{I}$ . Note that, the maximizer in (2.47) might not be unique and the solution may consist of several values. However, in practice, the goal is often only to find a local maximizer via gradient-based techniques as discussed in Section 2.1.3.

### Maximum a Posterior II

Alternatively to the maximum likelihood II approach, we can use the approach in Definition 2.7. This means, we introduce some hyper-priors  $p(\boldsymbol{\theta})$  on the hyperparameters  $\boldsymbol{\theta}$  and use the (intractable) posterior  $p(\boldsymbol{\theta}|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})$  as

the objective function, that is,

$$\begin{aligned}
\boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{y}) \\
&= \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \\
&= \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \\
&= \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}| - 2 \log p(\boldsymbol{\theta}).
\end{aligned} \tag{2.48}$$

Since the most hyperparameters  $\boldsymbol{\theta}_j \in \boldsymbol{\theta}$  in probabilistic models correspond to some variances  $\sigma_j^2 \in \mathbb{R}^+$ , we use in this work *Log-Normal* hyper-priors for all hyperparameters as not otherwise stated. Thereby, we use the transformation with the natural logarithm  $\boldsymbol{\theta}_j = \log \sigma_j$ , which is often very appropriate in order to get a clear picture for small variances and leads to smooth objective functions. Using a Log-Normal prior distribution for the variable  $\sigma_j = \exp(\boldsymbol{\theta}_j)$ , that is,

$$p(\sigma_j) = \log \mathcal{N}(\sigma_j | \eta_j, \nu_j)$$

for some means  $\eta_j$  and variances  $\nu_j$ , leads to a Gaussian prior in the transformed space

$$p(\boldsymbol{\theta}_j) = \mathcal{N}(\boldsymbol{\theta}_j | \eta_j, \nu_j).$$

In the following we assume that the hyper-priors are pairwise independent, yielding the joint hyper-prior

$$p(\boldsymbol{\theta}) = \prod_{j=1}^{|\boldsymbol{\theta}|} p(\boldsymbol{\theta}_j) = \mathcal{N}(\boldsymbol{\theta} | \boldsymbol{\eta}, \boldsymbol{\nu}), \tag{2.49}$$

where  $\boldsymbol{\eta}$  is the vector containing all  $\eta_j$  and  $\boldsymbol{\nu}$  the diagonal matrix with  $\nu_j$  on the diagonal. For this kind of hyper-priors, the MAP-II in (2.48) becomes

$$\begin{aligned}
\boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \\
&= \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}| + (\boldsymbol{\theta} - \boldsymbol{\eta})^T \boldsymbol{\nu}^{-1} (\boldsymbol{\theta} - \boldsymbol{\eta})
\end{aligned} \tag{2.50}$$

which can be numerically optimized with respect to each hyperparameter  $\boldsymbol{\theta}_j \in \boldsymbol{\theta}$ .

Note that it is important to implement the minimization in (2.47) and (2.50) efficiently, therefore it is beneficial to provide also the derivatives with respect to  $\boldsymbol{\theta}$  analytically, as described in Section 2.1.3. Moreover, depending on the size of  $M$  and  $N$ , it might be more efficient to apply the inversion (2.16) and determinant lemma (2.17), respectively, to  $\mathbf{P}_{\boldsymbol{\theta}}$  and directly work with the Cholesky decomposition of  $\mathbf{P}_{\boldsymbol{\theta}}$ .

method	likelihood function $p(\mathbf{y} \boldsymbol{\theta})$	prior function $p(\boldsymbol{\theta})$	definition
ML-I	$p(\mathbf{y} \mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})$		Def. 2.4, Eq. (2.34)
MAP-I	$p(\mathbf{y} \mathbf{X}, \mathbf{w}, \boldsymbol{\theta}, \mathcal{M})$	$p(\boldsymbol{\theta} \mathbf{w}, \mathcal{M})$	Def. 2.5
ML-II	$p(\mathbf{y} \mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$		Def. 2.6, Eq. (2.47)
MAP-II	$p(\mathbf{y} \mathbf{X}, \boldsymbol{\theta}, \mathcal{M})$	$p(\boldsymbol{\theta} \mathcal{M})$	Def. 2.7, Eq. (2.50)

Table 2.2. Summary of different approaches for level II inference for hyperparameters.

**Example 2.4 (Bayesian Linear Regression, Level II)** For the Example 2.3, we have the hyperparameters  $\boldsymbol{\theta} = [\log \sigma_0, \log \sigma_n]^T = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2]^T$ . In this example, we compare the different inference approaches ML-I, MAP-I, ML-II and MAP-II for the treatment of the hyperparameters  $\boldsymbol{\theta}$ , summarized in Table 2.2. The corresponding objective functions are depicted in Figure 2.7. For ML-I, the objective function becomes

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\mathbf{w}, \boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \frac{1}{\sigma_n^2} (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w})^T (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}),$$

which is completely independent of  $\sigma_0$  and thus it cannot be estimated, which can also be validated in the second row of Figure 2.7. Note that the weights  $\mathbf{w}$  for ML-I (and also MAP-I), are set to the true weights which are used for generating the data (see Example 2.3). In practice, the weights and the hyperparameters have to be estimated jointly. For MAP-I, we also use the hyper-priors as described in (2.49), in particular, we use Log-Normal hyper-priors with zero mean and variances equal to 1 on the hyperparameters  $\boldsymbol{\theta} = [\log \sigma_0, \log \sigma_n]^T$ , that is,

$$p(\boldsymbol{\theta}) = \mathcal{N} \left( \begin{bmatrix} \log \sigma_n \\ \log \sigma_0 \end{bmatrix} \middle| \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right). \quad (2.51)$$

In the transformed space, this corresponds to an hyper-prior mean  $\mathbb{E}[\sigma_n] = \mathbb{E}[\sigma_0] = \exp\left(\frac{1}{2}\right) \approx 1.65$  and hyper-prior variance  $\text{Var}[\sigma_n] = \text{Var}[\sigma_0] = \exp(1)^2 - \exp(1) \approx 4.67$  corresponding to a standard deviation of around 2.16. Note that these hyper-priors are quite flat or uninformative compared to the true values used to generate the data  $\sigma_0^2 = 0.3$ ,  $\sigma_n^2 = 0.03$  and thus  $\sigma_0 \approx 0.55$  and  $\sigma_n \approx 0.17$ . Using these hyper-priors, the optimization problem for MAP-I becomes

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\mathbf{w}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \frac{1}{\sigma_n^2} (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w})^T (\mathbf{y} - \Phi_{\mathbf{X}} \mathbf{w}) + \boldsymbol{\theta}^T \boldsymbol{\theta}.$$

Note that the optimal estimate for  $\sigma_0$  correspond exactly to the hyper-prior mean  $\approx 1.65$  as depicted in the third row of Figure 2.7. On the other hand, for a small

number of observed samples, the prior on the noise  $\sigma_n$  has a significant effect compared to the ML-I estimate, however, for large observed samples, it converges to ML-I. In both cases, the true noise level, depicted with the red cross, cannot be recovered.

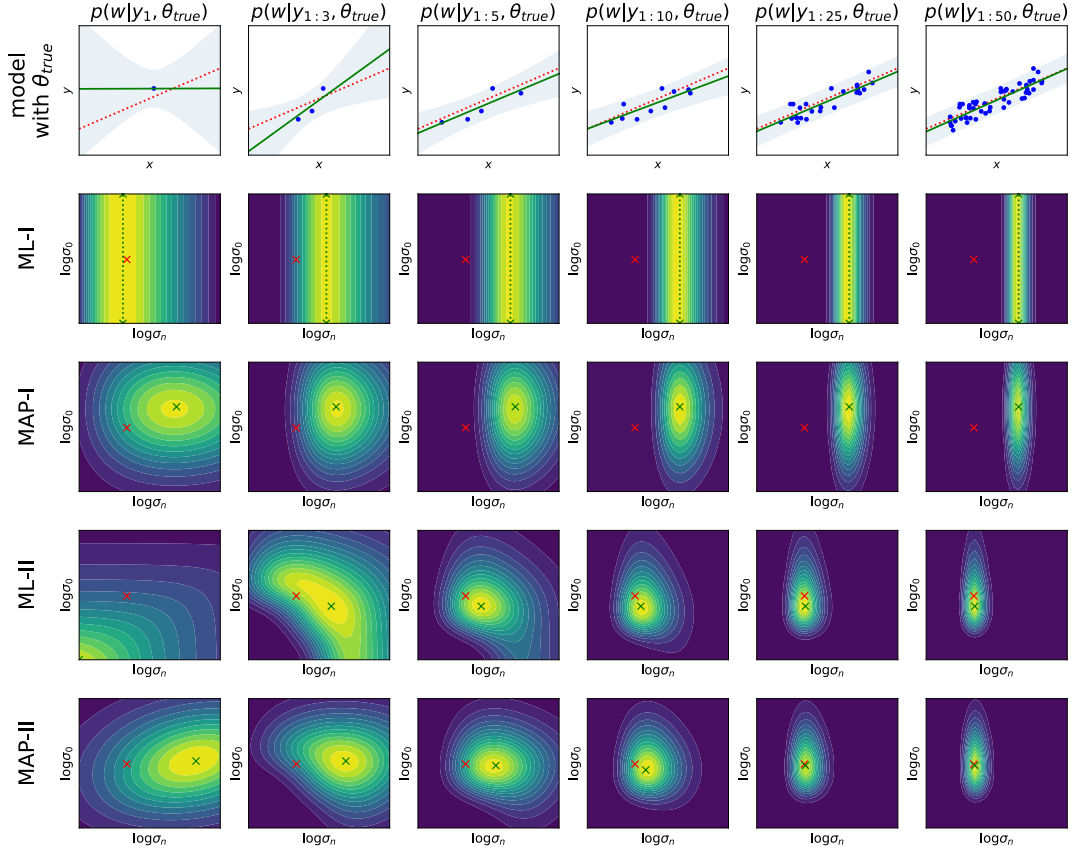


Figure 2.7. Inference at level II for hyperparameters  $\theta = [\log \sigma_0, \log \sigma_n]$  with different methods as summarized in Table 2.2 and discussed in Example 2.4. In each row from the left to the right, more observed data samples are available to train the models. Note that in the first row, the true weights and true hyperparameters are used. We can observe that the influence of the hyper-prior has only a significant (but positive) effect for small number of samples and vanishes for large number of data. Moreover, the two approaches in the bottom based on the marginal likelihood recover the true hyperparameters depicted as red crosses.

Next, we use empirical Bayes for the hyperparameters, which involves the marginal likelihood in (2.45) given as  $p(\mathbf{y}|\theta) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{P}_\theta)$  with  $\mathbf{P}_\theta = \sigma_0^2 \Phi_X \Phi_X^T + \sigma_n^2 \mathbb{I}$ . The ML-II in (2.47) becomes therefore

$$\theta^* = \arg \max_{\theta} p(\mathbf{y}|\theta) = \arg \min_{\theta} \mathbf{y}^T \mathbf{P}_\theta^{-1} \mathbf{y} + \log |\mathbf{P}_\theta|,$$



which is depicted in the fourth row in Figure 2.7. For small number of samples, it still has some identifiable issues for both hyperparameters, which can be overcome for larger number of samples and recovers basically the true hyperparameters. Similarly, for MAP-II, we use again the hyper-priors in (2.51), yielding the objective function

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}| + \boldsymbol{\theta}^T \boldsymbol{\theta},$$

which is depicted in the bottom row of Figure 2.7. We notice that for a small number of samples, the prior has a significant effect which makes the optimization problem well-defined. However, for large number of samples, the influence of the hyper-prior vanishes and it recovers the ML-II solution corresponding basically to the true hyperparameters. Note that the objective function for MAP-II corresponds to the exact Bayesian posterior, which is infeasible to compute including the normalization, however, computing (numerically) the maximizer is tractable. The mechanism for level II inference in Gaussian processes are exactly the same and an example will be given in Figure 3.4.

#### 2.3.4.7 Level III Inference: Model Selection

**Example 2.5 (Bayesian Linear Regression, Level III)** In this example we illustrate the automatic trade-off between model fit and model complexity for Bayesian model selection or level III inference based on the marginal likelihood for estimating the optimal order  $P$  for polynomial regression in Example 2.1. For the purpose of demonstration, we assume fixed and known hyperparameters  $\sigma_0^2 = 1$  and  $\sigma_n^2 = 0.1$  on the second level in order to focus on level III inference. The models  $\mathcal{M}$  under consideration in this example correspond to the linear basis function models with polynomial basis functions of degree  $P$ , that is,  $\mathcal{M} \in \{f_1, \dots, f_{10}\}$  similarly as in Example 2.2. We assume a uniform prior distribution  $p(\mathcal{M} = f_p) = \frac{1}{10}$  which means that we do not favor any of the models a priori. According to (2.32), the posterior distribution of model  $f_p$  given the data is

$$p(\mathcal{M} = f_p | \mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y} | \mathbf{X}, \mathcal{M} = f_p) p(\mathcal{M} = f_p)}{p(\mathbf{y} | \mathbf{X})}, \quad (2.52)$$

where the normalization can be computed as (2.33), however, since the prior is uniform, we already know that  $p(\mathbf{y} | \mathbf{X}) = 1$  and thus the posterior probabilities are just the likelihood divided by 10. The likelihood  $p(\mathbf{y} | \mathbf{X}, \mathcal{M} = f_p)$  corresponds to the marginal likelihood from the previous level, which is in this case just the usual marginal likelihood by integrating over the weights  $\mathbf{w}$  as already computed

in (2.45). Note that there it is denoted as  $p(\mathbf{y})$  and only implicitly conditioned on the input data and model, compare also Remark 2.1.

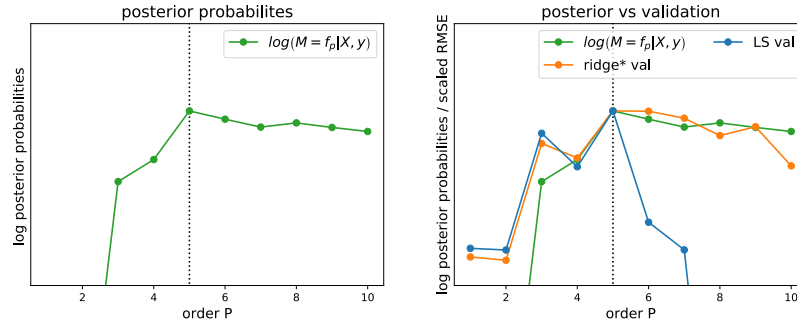


Figure 2.8. Bayesian model selection for polynomial regression.

In Figure 2.8 on the left, the log of the posterior probabilities  $\log p(\mathcal{M} = f_p | \mathbf{y}, \mathbf{X})$  against the degree  $P$  is depicted. We can observe that the log probabilities for the model of degree 1 or 2 are very low, and are clearly peaked at order  $P = 5$  corresponding to the most probable model according to the data. Interestingly, we have found similar results in Example 2.2, where we discussed validation strategies for point estimates. In Figure 2.8 on the right, the log posterior probabilities and the RMSE of a rather large validation set for ridge and least-squares regression are depicted. Note that for ridge\*, we allowed to optimize the  $\lambda^*$  on the validation set. The two curves for ridge regression with large validation set and the one obtained with Bayesian model selection via marginal likelihood are very similar. This is remarkable since this means that we can estimate the generalization error of a model only with training data without any further validation data!

### 2.3.5 Advantages and Limitations

#### 2.3.5.1 Advantages of Bayesian Inference

We briefly summarize the benefits of Bayesian inference with focus on the linear basis function model for regression.

- **Prior Knowledge:** The Bayesian paradigm constitutes a clean way to incorporate a priori knowledge in a principled way in form of priors and thus partially solves the regularization issue of non-Bayesian methods.
- **Robust against Overfitting:** Since the weights (or in general the whole set of hypothesis) are integrated out, Bayesian methods are more robust

against overfitting. This has the effect that also for a small number of data the generalization capacity shows good performance.

- **Analytic Inference:** For given basis functions, the predictive posterior distribution of a Bayesian linear model can be computed analytically in closed form for a Gaussian prior and Gaussian likelihood.
- **Credible Intervals:** Bayesian approaches provide a framework to model uncertainty. As a consequence, the predictions correspond to a whole distribution, so that well-calibrated credible intervals for the predictions can be computed.
- **Model Selection:** Since the marginal likelihood of the data given a model can be analytically computed, Bayesian approaches provide a principled way of comparing different models and to automatically trade-off between data fit and complexity of the model (e.g. number of basis functions).
- **Empirical Bayes:** From a practical point of view, empirical Bayesian methods (ML-II and MAP-II) constitute a powerful tool to efficiently infer hyperparameters without the need of any further validation set.

**Example 2.6 (Bayesian Linear Model with Basis Functions)** *In this example, we compare the Bayesian solution of the linear regression task with 4 different choices of basis functions for which the results are depicted in Figure 2.9. In particular, for the basis function model introduced in (2.20), we define the parametrized basis function*

$$\boldsymbol{\phi}_\theta(x) = [\phi_1(x), \dots, \phi_M(x)]^T \in \mathbb{R}^M$$

*involving some parameters  $\boldsymbol{\theta}$ , so that the model becomes*

$$f(x) = \sum_{j=1}^M w_j \phi_j(x) = \boldsymbol{\phi}_\theta(x)^T \mathbf{w}$$

*with  $x \in \mathbb{R}$ . For the individual basis  $\phi_j(\mathbf{x})$ , we use 4 different choices, in particular a polynomial basis*

$$\phi_j(\mathbf{x}) = (\theta_j x + \theta_{M+j})^j, \quad (2.53)$$

*a periodic basis*

$$\phi_j(\mathbf{x}) = \sin(\theta_j x + \theta_{M+j}),$$

a Gaussian basis

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{(x - \theta_j)^2}{\theta_{M+1}^2}\right), \quad (2.54)$$

and a neural network basis

$$\phi_j(\mathbf{x}) = \text{erf}(\theta_j x + \theta_{M+j}), \quad (2.55)$$

where  $\text{erf}$  is the error-function.

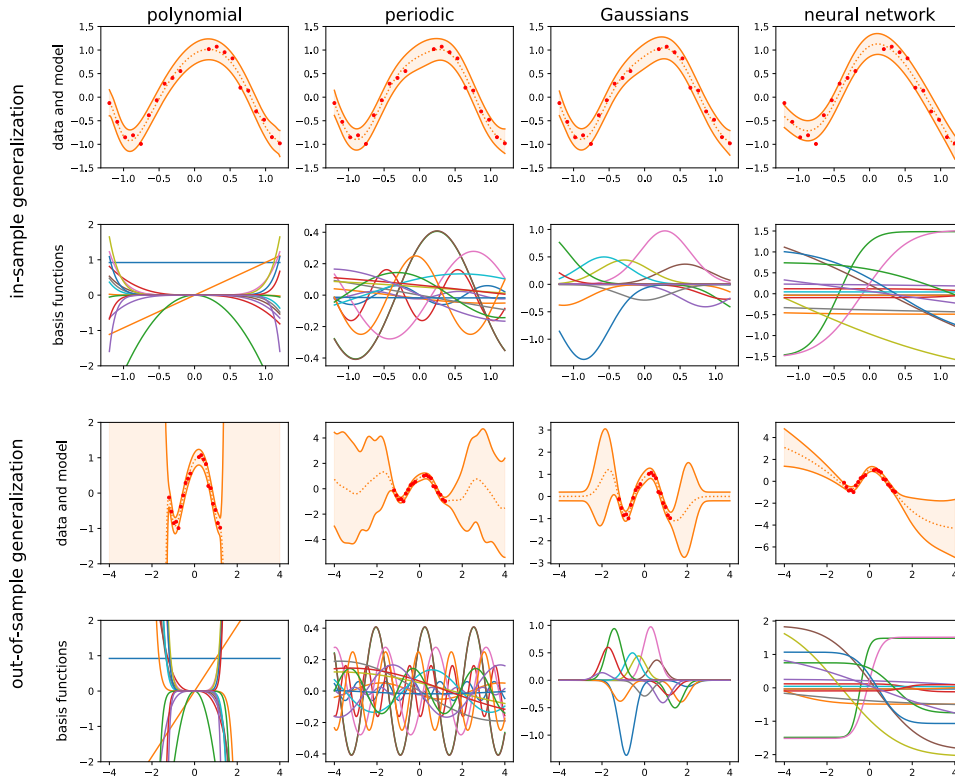


Figure 2.9. Bayesian linear regression with 4 different choices of basis functions (per columns). In the first two rows, the  $x$ -range is  $[-1, 1]$ , whereas in the third and fourth row  $[-4, 4]$ . We observe that the in-sample generalization is good for all choices of basis functions, however, the out-of-sample generalization is intrinsically limited by the **finite** number of basis functions.

We generated  $N = 20$  data samples in the  $x$ -range  $[-1, 1]$  and used  $M = 15$  number of basis functions. We used for each model the ML-II approach to find optimal hyperparameters  $\theta$ , so that the Bayesian predictive distribution with moments (2.42), (2.43) and the corresponding credible intervals (2.44) can be afterwards

analytically computed. The resulting models are depicted in the first and third row in Figure 2.9 for the  $x$ -range  $[-1, 1]$  and  $[-4, 4]$ , respectively. In the second and fourth row, the weighted basis functions  $w_j \phi_j(\mathbf{x})$  are depicted. We can observe in the first row, that each column show basically the same predictive posterior distribution independent of the choice of the particular basis functions, which are shown in the second row. It seems, that also the credible intervals are reliable. However, if we zoom out of the  $x$ -range where training samples are available, we can observe that the mean, and in particular the uncertainty estimates for the predictions, are indeed not very reliable outside regions with training data. This means, for fixed (and enough) number of basis functions, Bayesian regression can indeed achieve a good generalization capacity in regions with enough training data, however, the out-of-sample generalization is intrinsically limited by the **finite** number of basis functions.

#### 2.3.5.2 Limitations of Finite Basis Functions

Although Bayesian linear regression with arbitrary basis functions have many advantages compared to non-Bayesian approaches, they are limited regarding some aspects.

- **Choice of Basis Functions:** Finding suitable basis functions for given data is often a hard task. On the one hand, the basis should be complex enough to model the pattern in the data and on the other hand, the model should not overfit with good generalization for new data. This problem is also related to feature design or the choice of variables in statistical models. A possible easier approach constitute so-called kernel based methods as discussed in Section 2.4.
- **Number of Basis Functions:** Determining the right number of basis functions often relies on heuristic approaches. If we choose too few of them, the pattern in the data cannot be described well, if it is chosen too large, the danger of overfitting might be still there although Bayesian inference is more robust compared to non-Bayesian approaches.
- **Out-of-sample Generalization:** For a fixed number of basis functions, in particular when they are chosen depending on the training data, they always only model the function in regions where training data is available as illustrated in Example 2.6. Of course, one can not expect that it can find patterns without training data, but one can expect that at least the model

knows that it does not know, which means that the provided uncertainty information should be reliable.

The two latter mentioned limitations are both related to the fact that the number of basis functions is **finite**. Gaussian processes can intuitively be understood as the generalization of linear regression with **infinitely** many basis functions.

## 2.4 Kernels

In this section, we introduce the concept of *kernels*, for which we follow [Rasmussen and Williams, 2006, Chapter 4] and refer to [Smola and Schölkopf, 1998] for a thorough discussion. A kernel function

$$k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$$

$$\mathbf{x}, \mathbf{x}' \mapsto k(\mathbf{x}, \mathbf{x}')$$

is a real-valued function for two arguments  $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$  in the input space. It is assumed to be symmetric, i.e.  $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$ , so that it can be interpreted as a similarity measure between two points  $\mathbf{x}$  and  $\mathbf{x}'$ . In this work, we focus on the case  $\mathcal{X} = \mathbb{R}^D$ , however, the concept of kernels is more general and the input space can also include for instance graphs or strings [Smola and Schölkopf, 1998].

### 2.4.1 Definitions and Properties

**Definition 2.8 (Kernel Matrix)** For a kernel  $k$ , we define the kernel matrix

$$\mathbf{K}_{AB} = k(\mathbf{A}, \mathbf{B}) = \begin{bmatrix} k(\mathbf{a}_1, \mathbf{b}_1) & k(\mathbf{a}_1, \mathbf{b}_2) & \cdots & k(\mathbf{a}_1, \mathbf{b}_M) \\ k(\mathbf{a}_2, \mathbf{b}_1) & k(\mathbf{a}_2, \mathbf{b}_2) & \cdots & k(\mathbf{a}_2, \mathbf{b}_M) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{a}_N, \mathbf{b}_1) & k(\mathbf{a}_N, \mathbf{b}_2) & \cdots & k(\mathbf{a}_N, \mathbf{b}_M) \end{bmatrix} \in \mathbb{R}^{N \times M} \quad (2.56)$$

for input matrices  $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_N]^T \in \mathbb{R}^{N \times D}$  and  $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_M]^T \in \mathbb{R}^{M \times D}$ .

For the case  $\mathbf{A} = \mathbf{B}$ , the kernel matrix  $\mathbf{K}_{AA} = k(\mathbf{A}, \mathbf{A}) \in \mathbb{R}^{N \times N}$  is called *Gram matrix*. If the Gram matrix is positive semi-definite, i.e.  $\mathbf{a}^T \mathbf{K}_{AA} \mathbf{a} \geq 0$  for any input  $\mathbf{A} \in \mathbb{R}^{N \times D}$  and any vector  $\mathbf{a} \in \mathbb{R}^N$ , the corresponding kernel  $k$  is called *positive semi-definite kernel* (or also Mercer kernel). In the following of this thesis, we assume that the kernel is always *positive semi-definite*, as not otherwise stated.

**Definition 2.9 (Stationary and Isotropic Kernels)** *If the kernel function  $k(\mathbf{x}, \mathbf{x}')$  is a function in the difference  $\mathbf{x} - \mathbf{x}'$ , it is called a **stationary** kernel. This means that it is invariant to translations in the input space. Further, if the kernel is a function of  $|\mathbf{x} - \mathbf{x}'|$ , it is called **isotropic** kernel, and it is thus invariant to all rigid motions.*

Kernels are related to basis functions as introduced in previous sections. For instance, for any finite basis functions  $\phi(\mathbf{x}) \in \mathbb{R}^M$ , the inner product  $\phi(\mathbf{x})^T \phi(\mathbf{x}')$  is a valid kernel function  $k(\mathbf{x}, \mathbf{x}')$ .

**Example 2.7 (From Basis Functions to Kernels)** *Consider for example the 6-dimensional basis function*

$$\phi(\mathbf{x}) = [1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2]^T \in \mathbb{R}^6$$

for a two dimensional input point  $\mathbf{x} = [x_1, x_2]^T \in \mathbb{R}^D$ , where the scaling factor  $\sqrt{2}$  will become clear later. We can compute the inner product

$$\begin{aligned} \phi(\mathbf{x})^T \phi(\mathbf{x}') &= 1 + 2x_1x_1' + 2x_2x_2' + 2x_1x_1'x_2x_2' + (x_1x_1')^2 + (x_2x_2')^2 \\ &= (1 + x_1x_1' + x_2x_2')^2 \\ &= (1 + \mathbf{x}^T \mathbf{x}')^2 = k(\mathbf{x}, \mathbf{x}'), \end{aligned}$$

which is known as the quadratic polynomial kernel. This example illustrates also that it might be beneficial to work directly with kernels, which implicitly correspond to work in a possible high-dimensional basis function space.

Note, also the opposite is true under some circumstances. In particular, for any continuous kernel on a compact domain, there exist corresponding basis functions. In particular, any positive semi-definite kernel can be decomposed into (possible) infinite number of basis functions.

**Definition 2.10 (Mercer's Theorem)** *Any continuous positive definite kernel  $k$  on a compact domain can be decomposed as an infinite sum*

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{x}'), \quad (2.57)$$

where  $\lambda_i$  correspond to the  $i$ th **eigenvalue** corresponding to the **eigenfunction**  $\phi_i(\mathbf{x})$  of the kernel  $k$ .

We refer for the general case and for more details to [Rasmussen and Williams, 2006, Theorem 4.2]. From an intuitive point of view, Mercer's theorem is just the infinite-dimensional generalization of the diagonalization of a positive definite kernel matrix. For instance, consider the diagonalization of the Gram matrix  $\mathbf{K}_{AA} = \mathbf{U}\mathbf{D}\mathbf{U}^T \in \mathbb{R}^{N \times N}$  with the  $N$  eigenvectors in the columns of  $\mathbf{U}$  as the finite eigenfunctions and the  $N$  eigenvalues on the diagonal of  $\mathbf{D}$ . However, Mercer's theorem is more general as it holds also for the infinite dimensional Gram matrix for  $N \rightarrow \infty$ .

**Definition 2.11 (Degenerate Kernel)** *A kernel  $k$  with finite number of non-zero eigenvalues in the Mercer decomposition (2.57) is called a **degenerate** kernel. On the contrary, a kernel with infinite number of non-zero eigenvalues is called **non-degenerate** kernel.*

For instance, any kernel resulting from finite basis functions  $\phi(\mathbf{x}) \in \mathbb{R}^M$  like in Example 2.7 have at most  $M$  non-zeros eigenvalues and thus correspond to degenerate kernels.

Note that, symmetric positive definite functions are also called covariances in the statistics literature, hence, kernels can also be interpreted as covariance functions. For instance, the kernel specified by the two *input* data points  $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$  can be used to model the covariance

$$k(\mathbf{x}, \mathbf{x}') = \text{Cov}(f(\mathbf{x}), f(\mathbf{x}'))$$

between the *outputs*  $f(\mathbf{x}), f(\mathbf{x}') \in \mathbb{R}$  of a latent function  $f$ , which will be used for Gaussian processes in the next Chapter. Thereby, we will see that the choice of the kernel determines the properties of the function  $f$ , as it will be illustrated in Figure 3.3.

## 2.4.2 Examples

In the following, we provide some examples of kernels which will be used in subsequent chapters. For more details about kernels and examples we refer to [Smola and Schölkopf, 1998] and [Rasmussen and Williams, 2006].

### 2.4.2.1 Polynomial Kernel

For the bias  $\sigma_0 \in \mathbb{R}$  and polynomial order  $P \in \mathbb{N}^+$ , we define the polynomial kernel

$$k(\mathbf{x}, \mathbf{x}') = (\sigma_0^2 + \mathbf{x}^T \mathbf{x}')^P.$$



This kernel corresponds to the basis functions defined in (2.53) in a Bayesian linear model. In the case  $P = 1$ , the polynomial kernel simplifies to a linear kernel  $\sigma_0^2 + \mathbf{x}^T \mathbf{x}'$ .

#### 2.4.2.2 Squared-Exponential Kernel

A general *squared-exponential* (SE) kernel is defined as

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{(\mathbf{x} - \mathbf{x}')^T \mathbf{\Lambda}^{-1}(\mathbf{x} - \mathbf{x}')}{2}\right),$$

where  $\mathbf{\Lambda} \in \mathbb{R}^{D \times D}$  is a diagonal matrix with individual lengthscales  $l_1^2, \dots, l_D^2$  per dimension. In the case that all lengthscales are equal, the SE kernel becomes

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2l^2}\right).$$

As we will see in the next chapter, this kernel corresponds to infinite many basis functions of (2.54).

#### 2.4.2.3 Matérn Kernel

The Matérn class of covariance functions has the following form

$$k(\mathbf{x}, \mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} |\mathbf{x} - \mathbf{x}'|}{l}\right)^\nu K_\nu\left(\frac{\sqrt{2\nu} |\mathbf{x} - \mathbf{x}'|}{l}\right),$$

where  $\nu > 0$ ,  $l > 0$  and  $K_\nu$  is the modified Bessel function. As  $\nu \rightarrow \infty$ , the Matérn kernel converges to the SE kernel. In the case  $\nu = \frac{1}{2}$ , the kernel simplifies to the exponential covariance function

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{|\mathbf{x} - \mathbf{x}'|}{l}\right),$$

where the GP with this kernel is also known as the Ornstein-Uhlenbeck process for  $D = 1$ .

#### 2.4.2.4 Neural Network Kernel

There is also an interesting relationship between neural networks and kernels. Neal [1995] showed, when the number  $M$  of the basis function in (2.55) goes to

infinity, where the bias and the weights are normally distributed with mean zero and variances  $\sigma_b^2, \sigma_w^2$ , then this correspond to the following kernel

$$k(x, y) = \sigma^2 \frac{2}{\pi} \operatorname{asin} \left( \frac{\sigma_w^2 x^\top y + \sigma_b^2}{\sqrt{\sigma_w^2 x^\top x + \sigma_b^2 + 1} \sqrt{\sigma_w^2 y^\top y + \sigma_b^2 + 1}} \right).$$

Note that, this corresponds to a neural network with one hidden layer and infinite many nodes. Recent work shows also that a huge class of neural networks (for instance also deep convolutional neural networks) with infinite many nodes can be formulated as a kernel [Lee et al., 2017].

#### 2.4.2.5 Composed Kernels

Kernels can also be constructed from other kernels. For instance the addition or the multiplication of two kernels

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}'), \\ k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') k_2(\mathbf{x}, \mathbf{x}') \end{aligned}$$

are again valid kernels. Note also that both kernels might be even defined over a different space, which results into direct sum and tensor product kernels. There are many more valid operations for constructing valid kernels, we refer to [Bishop, 2006, Ch. 6] and [Duvenaud, 2014]. In Section 7.3.1, we will see an example with several composed kernels.

# Chapter 3

## Gaussian Processes

In this chapter, we introduce *Gaussian processes* (GPs) tailored to the need for machine learning and in particular for this thesis. Specifically, we first compare GPs with Bayesian linear regression models as discussed in the previous chapter. Thereby, we pointed out that they are limited to represent arbitrary complex functions due to the *finite* number of basis functions. In particular, we explain that GPs can be seen as Bayesian linear models extended to possible *infinite* number of basis functions by the so-called kernel trick in Section 3.1. An alternative view is provided in Section 3.2, where we show that GPs can be seen as the generalization of performing Bayesian inference for a finite parameter to Bayesian inference direct in the space of functions. Moreover, the treatment of the hyperparameters of GPs are discussed in Section 3.3, the Bayesian linear model is revisited in Section 3.4, where the differences to GPs are explained in detail. Finally, the advantages and limitations of full GPs are summarized in Section 3.5.

The content of Sections 3.1-3.3 is based on the standard introduction of GPs from the book by Rasmussen and Williams [2006], however, extended by several illustrative examples and connections. On the other hand, the comparisons between GPs and Bayesian linear models in Section 3.4 cannot be found explicitly in the literature about GPs to the best of our knowledge. In particular, we compare the predictive posterior distribution of GPs and Bayesian linear models and explain in detail the reasons why GPs represent richer models. Specifically, we demonstrate that the predictive posterior mean of a GP can be modeled as particular basis functions, however, the predictive posterior variance of a GP is more flexible due to an additional noise term. These insights are helpful to understand GPs and will be exploited in the subsequent Chapters 5 and 7, where we present novel GP approximation algorithms.

### 3.1 From Bayesian Linear Models to GPs

We start the discussion about GPs informally by highlighting the connection to Bayesian linear models with a finite number  $M$  of basis functions, for which we have discussed several limitations in Section 2.3.5.2. On the one hand, we would like to have as many as possible basis functions to represent arbitrary complex functions, on the other hand, we aim for a mechanism that prevents overfitting for this large basis and provides a good generalization. We reconsider the basis function (2.54) in Example 2.6, that is,  $\phi_j(\mathbf{x}) = \exp\left(-\frac{(x-\theta_j)^2}{\theta_0^2}\right)$  with hyperparameters  $\boldsymbol{\theta} = [\theta_1, \dots, \theta_M]^T \in \mathbb{R}^M$  and  $\theta_0 \in \mathbb{R}$ , so that the  $M$ -dimensional basis is given as  $\boldsymbol{\phi}_\theta(x) = [\phi_1(x), \dots, \phi_M(x)]^T \in \mathbb{R}^M$ . We now define the *infinite* dimensional analogue basis for  $M \rightarrow \infty$ , that is,  $\boldsymbol{\phi}_\mathbb{R}(x) = [\dots, \phi_1(x), \dots, \phi_M(x), \dots]^T \in \mathbb{R}^\infty$  with the infinite dimensional hyperparameters  $\boldsymbol{\theta} = \mathbb{R}$ , so that a Gaussian basis function is placed at every position  $\theta_j$  on the real line  $\mathbb{R}$ . Of course, direct inference in a Bayesian linear model as discussed in Section 2.3.4 cannot be applied to this infinite basis, however, this exactly corresponds to inference with a Gaussian process as illustrated in Figure 3.1.

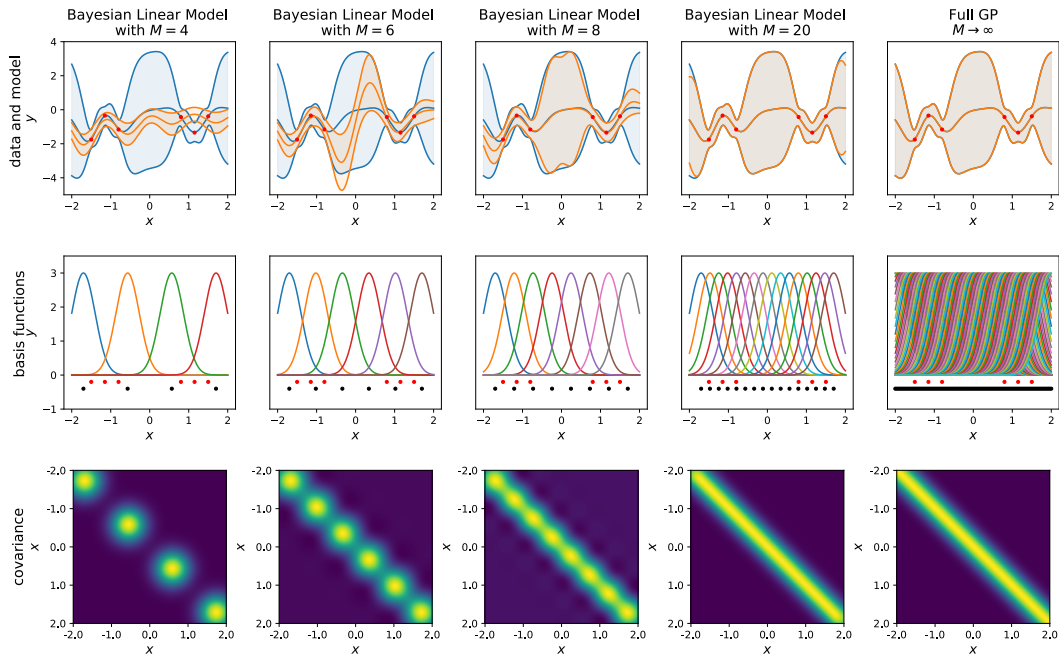


Figure 3.1. From finite Bayesian linear basis function models to Gaussian processes (GPs). For increasing number  $M$  of Gaussian basis functions as depicted in the second row, the corresponding Bayesian linear model and the induced (degenerate) covariance function is depicted in the first and third row, respectively.

**Remark 3.1** *Intuitively, a Gaussian process corresponds to a Bayesian linear basis function model with possibly infinitely many basis functions.*

In order to work with an infinite dimensional basis, the computations are never explicitly performed. Instead, the implicit infinite computations can be achieved by using kernels as discussed in Section 2.4. In particular, we can observe that in the predictive posterior distribution  $p(y(\mathbf{x}_*)|\mathbf{y}) = \mathcal{N}(\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*))$  of the Bayesian linear regression model with mean and covariance given in Equations (2.42) and (2.43), the basis functions  $\phi$  only appear in the form of inner products  $\phi_*^T \Sigma_0 \phi_*$ ,  $\phi_*^T \Sigma_0 \Phi_X^T$  and  $\Phi_X \Sigma_0 \Phi_X^T$ . Computing these products explicitly involves the computation of the possible high-dimensional basis function matrix  $\Phi_X \in \mathbb{R}^{N \times M}$ , which might be expensive or even impossible for  $M \rightarrow \infty$ . However, since  $\phi$  never appears alone but only in the form of inner products, we can *kernelize* these expressions by introducing a kernel

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \Sigma_0 \phi(\mathbf{x}') = \text{Cov}(f(\mathbf{x}), f(\mathbf{x}')). \quad (3.1)$$

Note that the kernel is specified by the inputs  $\mathbf{x}, \mathbf{x}'$ , however, it can be used to model the covariance of the outputs  $f(\mathbf{x}), f(\mathbf{x}')$  of a latent function  $f$ . Using a kernel means that we never explicitly compute the inner products of the basis functions with a possible infinite large inner dimension  $M \rightarrow \infty$ . We instead directly compute the cheap kernel evaluations corresponding implicitly to the same inner product in the possible infinite dimensional space. This is the so-called *kernel-trick*, which is illustrated in Figure 3.2.

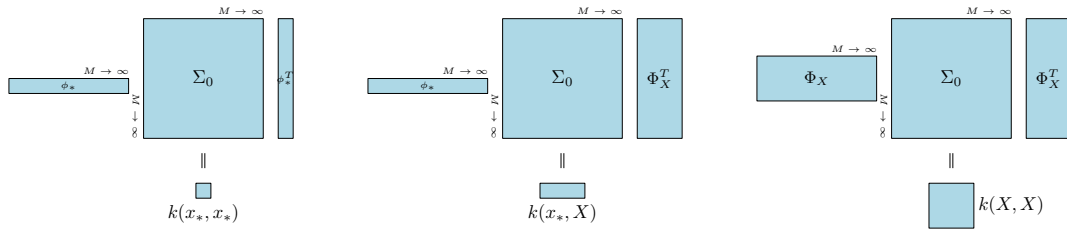


Figure 3.2. Illustration of kernel trick. Instead of computing the inner products with a possible infinite large inner dimension  $M \rightarrow \infty$  as indicated above the equality sign, we can instead compute the small kernel evaluations in the bottom part, which implicitly correspond to the same inner products in a possible infinite dimensional space.

Replacing all terms in (2.42) and (2.43) involving the basis functions  $\phi$  by the corresponding expressions based on the kernel (3.1), leads to the mean and vari-

ance of the posterior predictive distribution  $p(y(\mathbf{x}_*)|y) = \mathcal{N}(y(\mathbf{x}_*)|\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*))$

$$\begin{aligned}\mu(\mathbf{x}_*) &= \mathbf{K}_{\mathbf{x}_*X} (\mathbf{K}_{XX} + \sigma_n^2 \mathbb{I})^{-1} \mathbf{y}; \\ \sigma^2(\mathbf{x}_*) &= \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*X} (\mathbf{K}_{XX} + \sigma_n^2 \mathbb{I})^{-1} \mathbf{K}_{X\mathbf{x}_*} + \sigma_n^2.\end{aligned}\tag{3.2}$$

Note that these predictive moments in (3.2) correspond exactly to the mean and variance of the predictive posterior of a Gaussian process (which will be precisely defined in Definition 3.1) with zero mean and covariance with kernel in Eq. (3.1). However, the kernel in (3.1) induced by a Bayesian linear model with finite basis functions always corresponds to a GP with a degenerate kernel.

**Remark 3.2** *The kernel in (3.1) corresponding to a Bayesian linear model is a **degenerate** covariance function as defined in Definition 2.11, which means that it has a finite number of non-zero eigenvalues.*

From an intuitive point of view, this reflects the fact that the model can only represent a finite number  $M$  of basis functions. This can lead to underfitting, since the model is not flexible enough to capture the data. What might be even worse, it can result in addition to overconfidence of the predictive posterior variance in regions without training data and basis functions as illustrated in Figures 2.9 and 3.1. A possible solution constitutes full Gaussian processes with general kernels.

## 3.2 Function Space View

In this section, we introduce GPs more formally, where the content is based on [Rasmussen and Williams, 2006, Ch. 2].

**Definition 3.1 (Gaussian Process)** *A Gaussian process (GP)*

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')),$$

*with mean and covariance function*

$$\begin{aligned}m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})], \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))^T (f(\mathbf{x}') - m(\mathbf{x}'))],\end{aligned}$$

*is a collection of infinitely many random variables, where any finite number of which have a joint Gaussian distribution. In particular, for any finite index set  $\mathbf{X} \in \mathbb{R}^{N \times D}$ ,*

the marginal distribution  $p(\mathbf{f}(\mathbf{X})) = \mathcal{N}(\mathbf{f} | \mathbf{m}_X, \mathbf{K}_{XX})$  is

$$p(\mathbf{f}(\mathbf{X})) = \mathcal{N} \left( \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix} \middle| \begin{bmatrix} m(\mathbf{x}_1) \\ m(\mathbf{x}_2) \\ \vdots \\ m(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & \cdots & k(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \right).$$

For the sake of simplicity, we assume in the following the mean function  $m(\mathbf{x})$  to be zero, although it is straightforward to include a general mean function [Rasmussen and Williams, 2006, Ch. 2.7].

**Remark 3.3 (Marginalization Property)** Note that a GP is consistent regarding marginalization (2.2), which means that marginalizing a subset of the variables in a joint distribution implied by a GP yields the true marginal distribution. That is, for  $f \sim \mathcal{GP}$  and for any two variables  $\mathbf{A} \in \mathbb{R}^{N_a \times D}$  and  $\mathbf{B} \in \mathbb{R}^{N_b \times D}$ , it holds

$$p(f(\mathbf{A})) = \int p(f(\mathbf{A}), f(\mathbf{B})) \, df(\mathbf{B}). \quad (3.3)$$

This marginalization property is essential for the Definition 3.1, which uses the Kolmogorov extension theorem [Oksendal, 2013].

### 3.2.1 Prior Distribution

Assume a GP prior on  $f$ , therefore all finite function value evaluations follow a joint Gaussian distribution. In particular, the latent function values  $\mathbf{f} = f(\mathbf{X}) \in \mathbb{R}^N$  and  $f_* = f(\mathbf{x}_*) \in \mathbb{R}$  of the training inputs  $\mathbf{X} \in \mathbb{R}^{N \times D}$  and test input  $\mathbf{x}_* \in \mathbb{R}^D$ , respectively, follow a joint Gaussian distribution

$$p(\mathbf{f}, f_*) = \mathcal{N} \left( \begin{bmatrix} \mathbf{f} \\ f_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{XX} & \mathbf{K}_{Xx_*} \\ \mathbf{K}_{x_*X} & K_{x_*x_*} \end{bmatrix} \right), \quad (3.4)$$

where the kernel matrices  $\mathbf{K}_{XX} \in \mathbb{R}^{N \times N}$ ,  $\mathbf{K}_{Xx_*} \in \mathbb{R}^{N \times 1}$  and  $K_{x_*x_*} \in \mathbb{R}$  are defined according to (2.56). Combining  $p(\mathbf{f}, f_*)$  with the likelihood with Gaussian additive noise in (2.21), that is,

$$p(\mathbf{y} | \mathbf{f}) = \mathcal{N}(\mathbf{y} | \mathbf{f}, \sigma_n^2 \mathbb{I}), \quad (3.5)$$

yields the full joint distribution  $p(\mathbf{f}, f_*, \mathbf{y}) = p(\mathbf{y} | \mathbf{f}) p(\mathbf{f}, f_*)$  of the GP regression model

$$p(\mathbf{f}, f_*, \mathbf{y}) = \mathcal{N} \left( \begin{bmatrix} \mathbf{f} \\ f_* \\ \mathbf{y} \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{XX} & \mathbf{K}_{Xx_*} & \mathbf{K}_{XX} \\ \mathbf{K}_{x_*X} & K_{x_*x_*} & \mathbf{K}_{x_*X} \\ \mathbf{K}_{XX} & \mathbf{K}_{Xx_*} & \mathbf{K}_{XX} + \sigma_n^2 \mathbb{I} \end{bmatrix} \right). \quad (3.6)$$

	<b>density</b>	<b>mean</b>	<b>covariance</b>	$\int d\mathbf{f}$	$\int df_*$	$\int d\mathbf{y}$	$\frac{1}{p(\mathbf{f})}$	$\frac{1}{p(\mathbf{y})}$
marginal likelihood	$p(\mathbf{y})$	$\mathbf{0}$	$\mathbf{P}$	x	x			
marginal prior	$p(\mathbf{f})$	$\mathbf{0}$	$\mathbf{K}_{XX}$		x	x		
predictive prior	$p(f_*)$	$\mathbf{0}$	$\mathbf{K}_{x_*x_*}$	x		x		
likelihood	$p(\mathbf{y} \mathbf{f})$	$\mathbf{f}$	$\sigma_n^2\mathbb{I}$		x		x	
predictive conditional	$p(f_* \mathbf{f})$	$\mathbf{H}_*\mathbf{f}$	$\mathbf{V}_*$			x		x
posterior	$p(\mathbf{f} \mathbf{y})$	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$		x			x
predictive posterior	$p(f_* \mathbf{y})$	$\mu_f(\mathbf{x}_*)$	$\sigma_f^2(\mathbf{x}_*)$			x		x

Table 3.1. Overview of inference with GPs, which corresponds to marginalization and conditioning of the joint multivariate Gaussian distribution  $p(\mathbf{f}, f_*, \mathbf{y})$ . In particular, the quantities in each row can be obtained by the corresponding operations indicated by the crosses, where  $\int d\mathbf{z}$  means marginalization (2.2) and  $\frac{1}{p(\mathbf{z})}$  means conditioning (2.3) of the variable  $\mathbf{z}$  in the joint distribution  $p(\mathbf{f}, f_*, \mathbf{y})$ .

### 3.2.2 Inference

Inference with a GP corresponds to conditioning and marginalization of multivariate Gaussian distributions as defined in Equations (2.3) and (2.2). In particular, manipulating the joint distribution  $p(\mathbf{f}, f_*, \mathbf{y})$  according to the indicated operations in Table 3.1 leads to different useful quantities. We start by conditioning the joint distribution  $p(\mathbf{f}, f_*, \mathbf{y})$  on the observed data  $p(\mathbf{y}) > 0$  yielding the *joint posterior* over both the training and test latent function values

$$p(\mathbf{f}, f_*|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}, f_*)}{p(\mathbf{y})} = \frac{p(\mathbf{f}, f_*, \mathbf{y})}{p(\mathbf{y})}, \quad (3.7)$$

which results via (2.3) into the Gaussian distribution

$$\mathcal{N}\left(\begin{bmatrix} \mathbf{f} \\ f_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{K}_{XX} \\ \mathbf{K}_{x_*X} \end{bmatrix} \mathbf{P}^{-1} \mathbf{y}, \begin{bmatrix} \mathbf{K}_{XX} & \mathbf{K}_{Xx_*} \\ \mathbf{K}_{x_*X} & \mathbf{K}_{x_*x_*} \end{bmatrix} - \begin{bmatrix} \mathbf{K}_{XX} \\ \mathbf{K}_{x_*X} \end{bmatrix} \mathbf{P}^{-1} [\mathbf{K}_{XX}, \mathbf{K}_{x_*X}]\right),$$

where we introduced the covariance matrix

$$\mathbf{P} = \mathbf{K}_{XX} + \sigma_n^2 \mathbb{I}. \quad (3.8)$$

#### 3.2.2.1 Predictive Posterior

Prediction in a GP is straight forward, by marginalizing out  $\mathbf{f}$  in  $p(\mathbf{f}, f_*|\mathbf{y})$  in (3.7) yielding the *predictive posterior* distribution

$$p(f_*|\mathbf{y}) = \int p(\mathbf{f}, f_*|\mathbf{y}) d\mathbf{f} = \mathcal{N}(f_* | \mu_f(\mathbf{x}_*), \sigma_f^2(\mathbf{x}_*)), \quad (3.9)$$



with predictive mean and variance of the latent function value  $f_*$

$$\begin{aligned}\mu_f(\mathbf{x}_*) &= \mathbf{K}_{\mathbf{x}_*X} \mathbf{P}^{-1} \mathbf{y}; \\ \sigma_f^2(\mathbf{x}_*) &= \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*X} \mathbf{P}^{-1} \mathbf{K}_{X\mathbf{x}_*}.\end{aligned}\quad (3.10)$$

### 3.2.2.2 Predictive Posterior of Outputs

From the predictive posterior distribution (3.9) of the latent function value  $f_*$ , the predictive distribution  $p(y(\mathbf{x}_*)|\mathbf{y})$  of the outputs  $y(\mathbf{x}_*)$  can be obtained by computing

$$p(y(\mathbf{x}_*)|\mathbf{y}) = \int p(y(\mathbf{x}_*)|f_*)p(f_*|\mathbf{y})df_* = \mathcal{N}(y(\mathbf{x}_*)|\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)), \quad (3.11)$$

involving the likelihood  $p(y(\mathbf{x}_*)|f_*) = \mathcal{N}(y(\mathbf{x}_*)|f_*, \sigma_n^2)$  and the predictive posterior in (3.9). The integral can be explicitly computed (2.2), yielding the predictive mean and variance

$$\begin{aligned}\mu(\mathbf{x}_*) &= \mu_f(\mathbf{x}_*) = \mathbf{K}_{\mathbf{x}_*X} \mathbf{P}^{-1} \mathbf{y}; \\ \sigma^2(\mathbf{x}_*) &= \sigma_f^2(\mathbf{x}_*) + \sigma_n^2 = \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*X} \mathbf{P}^{-1} \mathbf{K}_{X\mathbf{x}_*} + \sigma_n^2.\end{aligned}$$

We can observe that those predictive moments exactly recover the ones of the **kernelized** Bayesian linear model in (3.2).

### 3.2.2.3 Posterior

Marginalizing out  $f_*$  instead of  $\mathbf{f}$  in  $p(\mathbf{f}, f_*|\mathbf{y})$  (3.7) leads via (2.2) to the *posterior* distribution of the latent function values

$$p(\mathbf{f}|\mathbf{y}) = \int p(\mathbf{f}, f_*|\mathbf{y})df_* = \mathcal{N}(\mathbf{f}|\mathbf{K}_{XX} \mathbf{P}^{-1} \mathbf{y}, \mathbf{K}_{XX} - \mathbf{K}_{XX} \mathbf{P}^{-1} \mathbf{K}_{XX}) \quad (3.12)$$

with posterior mean  $\boldsymbol{\mu} = \mathbf{K}_{XX} \mathbf{P}^{-1} \mathbf{y}$  and covariance  $\boldsymbol{\Sigma} = \mathbf{K}_{XX} - \mathbf{K}_{XX} \mathbf{P}^{-1} \mathbf{K}_{XX}$ . Note that these posterior moments have the same structure as those from a Bayesian linear model in (2.39) and can be exactly recovered when inserting for instance  $\Phi_X = \mathbb{I}$  and  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{XX}$  which will be further discussed in Section 3.4.

### 3.2.2.4 Predictive Conditional

From the joint prior in (3.4), we can derive via Gaussian conditioning (2.3) the conditional predictive distribution

$$p(f_*|\mathbf{f}) = \mathcal{N}(f_*|\mathbf{K}_{\mathbf{x}_*X} \mathbf{K}_{XX}^{-1} \mathbf{f}, \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*X} \mathbf{K}_{XX}^{-1} \mathbf{K}_{X\mathbf{x}_*}), \quad (3.13)$$

where we define the projection vector  $H_* = \mathbf{K}_{x_*X} \mathbf{K}_{XX}^{-1}$  and projection variance  $V_* = \mathbf{K}_{x_*x_*} - \mathbf{K}_{x_*X} \mathbf{K}_{XX}^{-1} \mathbf{K}_{Xx_*}$ . This conditional predictive distribution can be used for instance in a two stage procedure to compute predictions in a GP, as discussed in the next section.

### 3.2.2.5 Prediction via Posterior

Instead using the joint prior  $p(\mathbf{f}, f_*)$  directly for inference in (3.7), we can split it into  $p(f_*|\mathbf{f})p(\mathbf{f})$  as the product of the marginal prior  $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}_{XX})$  over the latent function values and the conditional distribution (3.13). The joint posterior (3.7) can be equivalently written as

$$p(\mathbf{f}, f_*|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f})p(f_*|\mathbf{f})p(\mathbf{f})}{p(\mathbf{y})} = p(f_*|\mathbf{f}) \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})}{p(\mathbf{y})}, \quad (3.14)$$

which can be understood as to first compute the posterior over the latent function values

$$p(\mathbf{f}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})}{p(\mathbf{y})}.$$

Note that, this step is independent of the test data and can be done in advance only with the training data. Finally, the predictive posterior distribution can be computed by marginalizing  $\mathbf{f}$  out via (2.2)

$$p(f_*|\mathbf{y}) = \int p(\mathbf{f}, f_*|\mathbf{y}) d\mathbf{f} \int p(f_*|\mathbf{f})p(\mathbf{f}|\mathbf{y}) d\mathbf{f},$$

resulting into the same predictive posterior as in (3.9).

### 3.2.2.6 Noise Free Prediction

For making predictions without noise, that is interpolation, we could set  $\sigma_n^2 = 0$  which has the effect that  $\mathbf{P} = \mathbf{K}_{XX}$  in (3.8). Plugging this into (3.9) leads to the predictive moments

$$\begin{aligned} \mu_f(\mathbf{x}_*) &= \mathbf{K}_{x_*X} \mathbf{K}_{XX}^{-1} \mathbf{y}, \\ \sigma_f^2(\mathbf{x}_*) &= \mathbf{K}_{x_*x_*} - \mathbf{K}_{x_*X} \mathbf{K}_{XX}^{-1} \mathbf{K}_{Xx_*}. \end{aligned}$$

Note that this correspond exactly to the conditional distribution  $p(f_*|\mathbf{f})$  obtained directly from the joint distribution  $p(\mathbf{f}, f_*)$  in (3.4) via Gaussian conditioning (2.3).

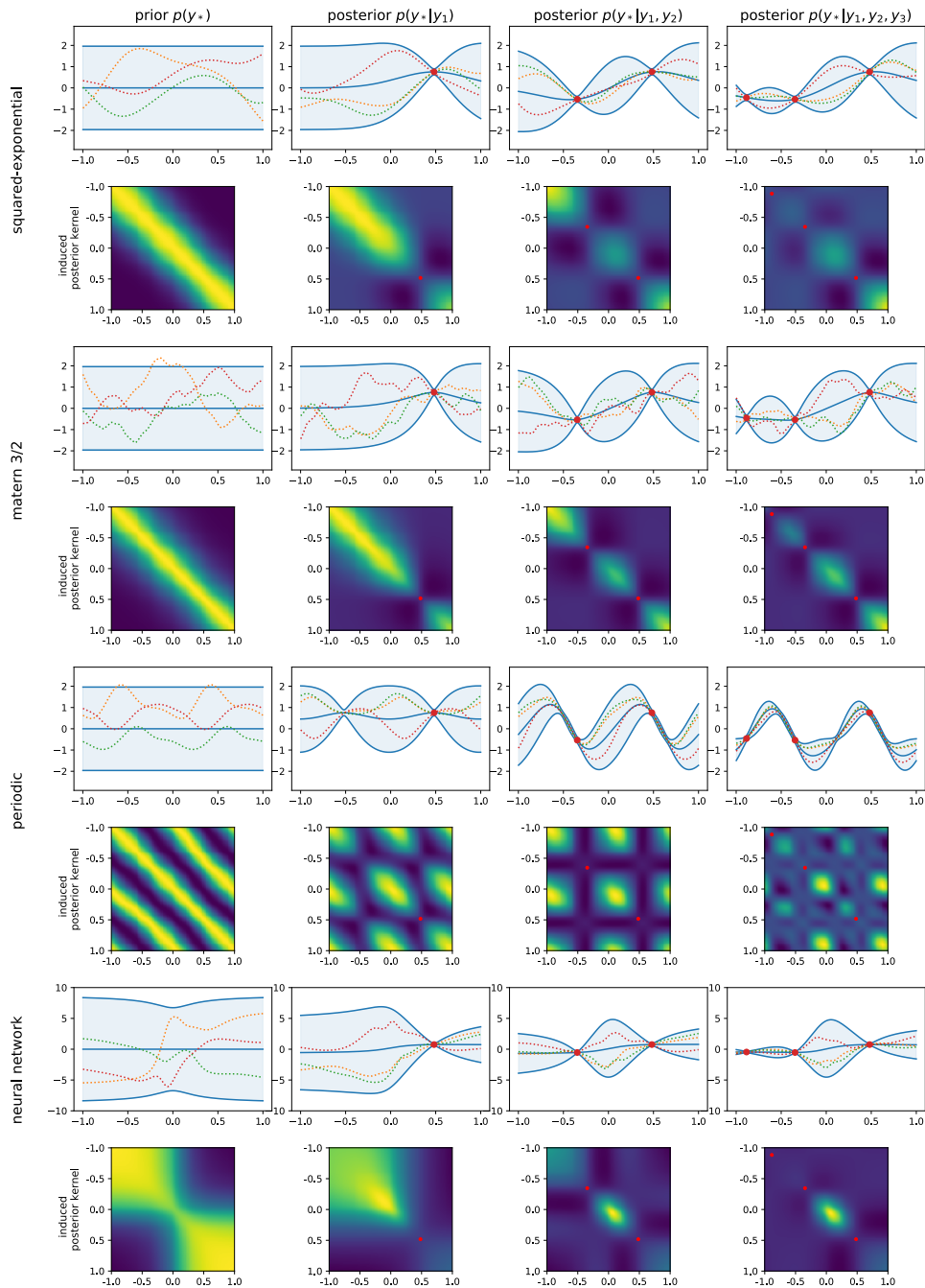


Figure 3.3. Illustration of inference in a GP with different kernels (per rows) and different number of training samples in the posterior (per columns). In particular, the first column shows the prior without any observed data. In each case, on the top row, the predictive distribution is indicated together with three samples functions drawn of it, whereas in the bottom row, the induced prior/posterior kernel is shown. We can observe, as more data is available, the possible functions are constrained such they are conform with the training samples.

## 3.2.2.7 Marginal Likelihood

Direct marginalization of  $f$  and  $f_*$  in  $p(f, f_*, \mathbf{y})$  in (3.6) yields the *marginal likelihood* distribution

$$p(\mathbf{y}) = \int p(f, f_*, \mathbf{y}) \, df \, df_* = \mathcal{N}(\mathbf{y} | \mathbf{0}, \mathbf{P}) \quad (3.15)$$

with covariance  $\mathbf{P} = \mathbf{K}_{XX} + \sigma_n^2 \mathbb{I}$  already defined in (3.8).

## 3.2.2.8 Predictive Posterior Covariance and Induced Kernel

Note that, the predictive distribution in (3.9) is formulated for a single test point  $\mathbf{x}_* \in \mathbb{R}^D$  leading to a scalar predictive mean and variance, which can be used for instance to compute credible intervals as formulated in (2.44). However, the predictive distribution of a GP is not limited to pointwise predictions. When  $\mathbf{x}_*$  is replaced by a general matrix of test inputs  $\mathbf{X}_* \in \mathbb{R}^{N_{test} \times D}$ , the joint predictive distribution  $p(f_* | \mathbf{y})$  of the vector  $f_* \in \mathbb{R}^{N_{test}}$  of test latent function values can be analogously computed to (3.9) as

$$\begin{aligned} \boldsymbol{\mu}_f(\mathbf{X}_*) &= \mathbf{K}_{X_*X} \mathbf{P}^{-1} \mathbf{y}, \\ \boldsymbol{\Sigma}_f(\mathbf{X}_*) &= \mathbf{K}_{X_*X_*} - \mathbf{K}_{X_*X} \mathbf{P}^{-1} \mathbf{K}_{XX_*} \end{aligned} \quad (3.16)$$

with  $\sigma_f^2(\mathbf{x}_*)$  on the diagonal of  $\boldsymbol{\Sigma}_f(\mathbf{X}_*)$  and the off-diagonal entries corresponding to the posterior covariance. Indeed, these predictive posterior moments induce a (degenerate) *posterior GP*

$$\bar{f}(\mathbf{x}) = f(\mathbf{x}) | \mathbf{y} \sim \mathcal{GP}(\bar{m}(\mathbf{x}), \bar{k}(\mathbf{x}, \mathbf{x}'))$$

with mean function corresponding to

$$\bar{m}(\mathbf{x}) = \mathbb{E}[f(\mathbf{x}) | \mathbf{y}] = \mathbb{E}[\bar{f}(\mathbf{x})] = k(\mathbf{x}, \mathbf{X}) \mathbf{P}^{-1} \mathbf{y}$$

and induced posterior kernel

$$\begin{aligned} \bar{k}(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x}) | \mathbf{y}])^T (f(\mathbf{x}') - \mathbb{E}[f(\mathbf{x}') | \mathbf{y}]) | \mathbf{y}] \\ &= \mathbb{E}[(\bar{f}(\mathbf{x}) - \bar{m}(\mathbf{x}))^T (\bar{f}(\mathbf{x}') - \bar{m}(\mathbf{x}'))] \\ &= k(\mathbf{x}, \mathbf{x}') - k(\mathbf{x}, \mathbf{X}) \mathbf{P}^{-1} k(\mathbf{X}, \mathbf{x}'). \end{aligned}$$

Note that these posterior mean and covariance functions can be used for sampling posterior functions as illustrated in Figure 3.3.

### 3.3 Inference of Hyperparameters

In this section, we discuss level II inference for the hyperparameters  $\boldsymbol{\theta}$  including all kernel parameters and the noise variance  $\sigma_n^2$ . In the previous section, we assumed that these hyperparameters are known and all distributions are implicitly conditioned on the hyperparameters as discussed in Section 2.3.2 for Bayesian hierarchical models. Since the marginal likelihood (3.15) depends non-linearly on the hyperparameters, their posterior distribution cannot in general be analytically computed. The most common approach for inferring those hyperparameters in a GP is based on empirical Bayes as thoroughly discussed in Section 2.3.3.1 for the Bayesian linear model. In particular, we can either use the ML-II based approach by optimizing the marginal likelihood (3.15), that is,

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}|\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}|,$$

with covariance matrix  $\mathbf{P}_{\boldsymbol{\theta}} = \mathbf{K}_{XX} + \sigma_n^2 \mathbb{I}$ , or the MAP-II approach also involving some suitable hyper-priors (2.49) leading to

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \\ &= \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_{\boldsymbol{\theta}}^{-1} \mathbf{y} + \log |\mathbf{P}_{\boldsymbol{\theta}}| + (\boldsymbol{\theta} - \boldsymbol{\eta})^T \boldsymbol{\mathcal{V}}^{-1} (\boldsymbol{\theta} - \boldsymbol{\eta}). \end{aligned}$$

These two approaches are illustrated in Figure 3.4, where we generated  $N = 20$  training data samples with a squared-exponential kernel with true hyperparameters  $\sigma_0 = 1$ ,  $l = 0.3$ ,  $\sigma_n = 0.1$ . We fixed the prior variance  $\sigma_0 = 1$  to the true value for illustration purposes and estimated the lengthscale  $l$  and the noise variance  $\sigma_n^2$  via ML-II and MAP-II, respectively, where we used the same hyper-priors as in (2.51) for the latter. Note that in practice, those hyper-priors can be learned for instance from a large collection of datasets as proposed by [Corani et al., 2021] for time series. We can observe that the marginal likelihood as well as the posterior for the hyperparameters correspond to non-Gaussian distributions, which might be even multi-modal. We can observe that the estimated values (blue cross) for both methods converge to the true hyperparameters (red cross) for enough data samples. Further, we see that for a small number of samples, ML-II has some identifiable issues, which can be corrected by suitable priors.

### 3.4 Linear Basis Function Models Revisited

In this section, we briefly compare GPs again with linear basis function models in order to understand the similarities and differences which will be important for

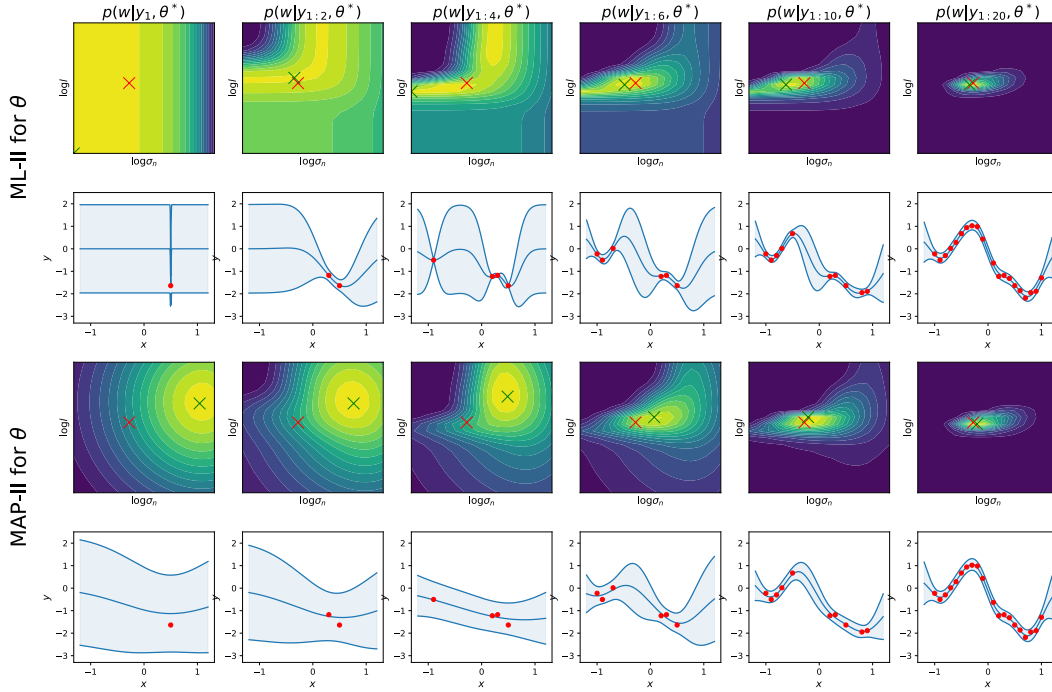


Figure 3.4. Level II inference for the hyperparameters  $\theta$  in GPs with ML-II and MAP-II for increasing number of training samples.

sparse and sequential approximations of GPs. As we have discussed in Section 3.1, a linear Bayesian model with basis function  $\phi$  and prior covariance  $\Sigma_0$  induce a GP with a degenerate kernel.

**Proposition 3.1 (Bayesian Linear Model and GP)** *Inference in a Bayesian linear model with given basis functions  $\phi$  and prior covariance  $\Sigma_0$*

$$\begin{aligned} f(\mathbf{x}) &= \phi(\mathbf{x})^T \mathbf{w} \\ \mathbf{w} &\sim \mathcal{N}(\mathbf{0}, \Sigma_0) \end{aligned} \quad \Leftrightarrow \quad f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \phi(\mathbf{x})^T \Sigma_0 \phi(\mathbf{x}')).$$

is equivalent to exact inference of a GP with a degenerate kernel

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \Sigma_0 \phi(\mathbf{x}').$$

Importantly, the contrary is not true for a GP with a non-degenerate kernel. In fact, Mercer's theorem (2.57) shows that any kernel can be represented by possible *infinite* many basis functions. As a consequence, a zero mean GP prior with general kernel corresponds to a Gaussian prior with infinite-dimensional covariance  $\Sigma_0$  in a Bayesian linear model, which is clearly not feasible. However, in

practice, the prior is not of particular interest, the focus lies mainly on the posterior GP conditioned on a *finite* number data points.

**Remark 3.4** *Even though the GP prior corresponds to infinite many basis functions, the posterior conditioned on finite number of data samples  $N$  can be represented by a  $N$ -variate multivariate Gaussian.*

In the following, we compare the posterior of a GP with the posterior of a Bayesian linear model and try to understand the effect of the infinite dimensional prior on the posterior. We first compare the predictive mean for a Bayesian linear model in (2.42)

$$\mu_L(\mathbf{x}_*) = \phi_* \boldsymbol{\mu} = \phi_* \boldsymbol{\Sigma}_0 \Phi_X^T \mathbf{P}_L^{-1} \mathbf{y},$$

to the predictive posterior mean of a GP in (3.11)

$$\mu_{GP}(\mathbf{x}_*) = \mathbf{K}_{\mathbf{x}_* X} \mathbf{P}_{GP}^{-1} \mathbf{y},$$

where  $\mathbf{P}_L = \Phi_X \boldsymbol{\Sigma}_0 \Phi_X^T + \sigma_n^2 \mathbb{I}$  and  $\mathbf{P}_{GP} = \mathbf{K}_{XX} + \sigma_n^2 \mathbb{I}$ , respectively. Matching terms so that  $\mu_L(\mathbf{x}_*) = \mu_{GP}(\mathbf{x}_*)$ , yields the constraints

$$\phi_* \boldsymbol{\Sigma}_0 \Phi_X^T = \mathbf{K}_{\mathbf{x}_* X} \quad \text{and} \quad \Phi_X \boldsymbol{\Sigma}_0 \Phi_X^T = \mathbf{K}_{XX}.$$

In fact, there are several possible choices for the basis functions and prior covariance.

**Proposition 3.2 (Equal Predictive Mean and Marginal Likelihood)** *A GP with mean zero and non-degenerate kernel  $k(\mathbf{x}, \mathbf{x}')$  and a Bayesian linear model with  $\phi(\mathbf{x})$  and  $\boldsymbol{\Sigma}_0$ , which satisfies*

$$\phi(\mathbf{x})^T \boldsymbol{\Sigma}_0 \phi(\mathbf{x}') = k(\mathbf{x}, X) k(X, X)^{-1} k(X, \mathbf{x}'),$$

*leads to the same predictive posterior mean  $\mu_L(\mathbf{x}_*) = \mu_{GP}(\mathbf{x}_*)$  of the Bayesian linear model as defined in (2.42) and of GP (3.11). Further, also the marginal likelihood covariances  $\mathbf{P}_L = \mathbf{P}_{GP}$  in (2.40) and (3.8) are equal.*

Note that the choice for the  $\phi$  and  $\boldsymbol{\Sigma}_0$  are not unique and the predictive posterior mean can be decomposed into several ways into  $N$  basis functions  $\phi_X(\mathbf{x}_*)$  such that

$$\boldsymbol{\mu}(\mathbf{x}_*) = \phi_X(\mathbf{x}_*)^T \boldsymbol{\mu} = \sum_{i=1}^N \mu_i \phi_X(\mathbf{x}_*)_i.$$

In Table 3.2 and Figure 3.5, there are several interesting choices illustrated. For instance, for the choice  $\mathbf{g}_X(\mathbf{x}_*)$ , the posterior correspond exactly to the output training data  $\mathbf{y}$  so that  $\mathbf{g}_X(\mathbf{x}_*)_i$  shows the influence of each data point, which is also known as linear predictor or linear smoothers [Silverman, 1985]. Other choices as summarized in Table 3.2 lead to cases, in which the basis functions correspond exactly to the kernel or to the eigenfunctions, respectively, or that the prior covariance is independent of the input data.

	basis function $\phi_X(\mathbf{x})$	prior covariance $\Sigma_0$	posterior mean $\boldsymbol{\mu}$	comments
$\mathbf{g}_X(\mathbf{x})$	$\mathbf{P}^{-1} k(\mathbf{X}, \mathbf{x})$	$\mathbf{P}\mathbf{K}_{XX}^{-1}\mathbf{P}$	$\mathbf{y}$	posterior mean of weights correspond to data; linear predictor/smoothen
$\mathbf{k}_X(\mathbf{x})$	$k(\mathbf{X}, \mathbf{x})$	$\mathbf{K}_{XX}^{-1}$	$\mathbf{P}^{-1}\mathbf{y}$	basis function correspond to kernel function; representer theorem
$\mathbf{l}_X(\mathbf{x})$	$\mathbf{L}^{-1} k(\mathbf{X}, \mathbf{x})$	$\mathbb{I}$	$\mathbf{L}^T\mathbf{P}^{-1}\mathbf{y}$	prior covariance correspond to identity via Cholesky
$\mathbf{u}_X(\mathbf{x})$	$\mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T k(\mathbf{X}, \mathbf{x})$	$\mathbb{I}$	$\mathbf{D}^{\frac{1}{2}}\mathbf{U}^T\mathbf{P}^{-1}\mathbf{y}$	prior covariance correspond to identity via eigendecomposition
$\mathbf{d}_X(\mathbf{x})$	$\mathbf{U}^T k(\mathbf{X}, \mathbf{x})$	$\mathbf{D}^{-1}$	$\mathbf{U}^T\mathbf{P}^{-1}\mathbf{y}$	prior covariance = 1/eigenvalues; basis function = eigenfunction; Mercer's theorem
$\mathbf{h}_X(\mathbf{x})$	$\mathbf{K}_{XX}^{-1} k(\mathbf{X}, \mathbf{x})$	$\mathbf{K}_{XX}$	$\mathbf{K}_{XX}\mathbf{P}^{-1}\mathbf{y}$	posterior mean of weights in the Bayesian linear model correspond to posterior in GP

Table 3.2. Possible basis function decomposition so that the predictive posterior mean  $\boldsymbol{\mu}_L(\mathbf{x}_*) = \boldsymbol{\phi}(\mathbf{x}_*)^T \boldsymbol{\mu}$  correspond to them of full GP  $\boldsymbol{\mu}_{GP}(\mathbf{x}_*)$ . It holds  $\mathbf{L}\mathbf{L}^T = \mathbf{K}_{XX}$  with  $\mathbf{L}$  the lower Cholesky matrix. Further,  $\mathbf{U}\mathbf{D}\mathbf{U}^T = \mathbf{K}_{XX}$  with  $\mathbf{U}$  the eigenvector matrix corresponding to the diagonal matrix  $\mathbf{D}$  with the eigenvalues on the diagonal.

**Lemma 3.1 (Equal Posterior Distribution)** *For the particular choice of basis functions and prior covariance*

$$\boldsymbol{\phi}(\mathbf{x}) = \mathbf{h}_X(\mathbf{x}) = \mathbf{K}_{XX}^{-1}k(\mathbf{X}, \mathbf{x}) \quad \text{and} \quad \Sigma_0 = \mathbf{K}_{XX}$$

in Proposition 3.2 yields also to the equivalent posterior moments  $\boldsymbol{\mu}$  and  $\Sigma$  in (2.39) and (3.12), respectively.



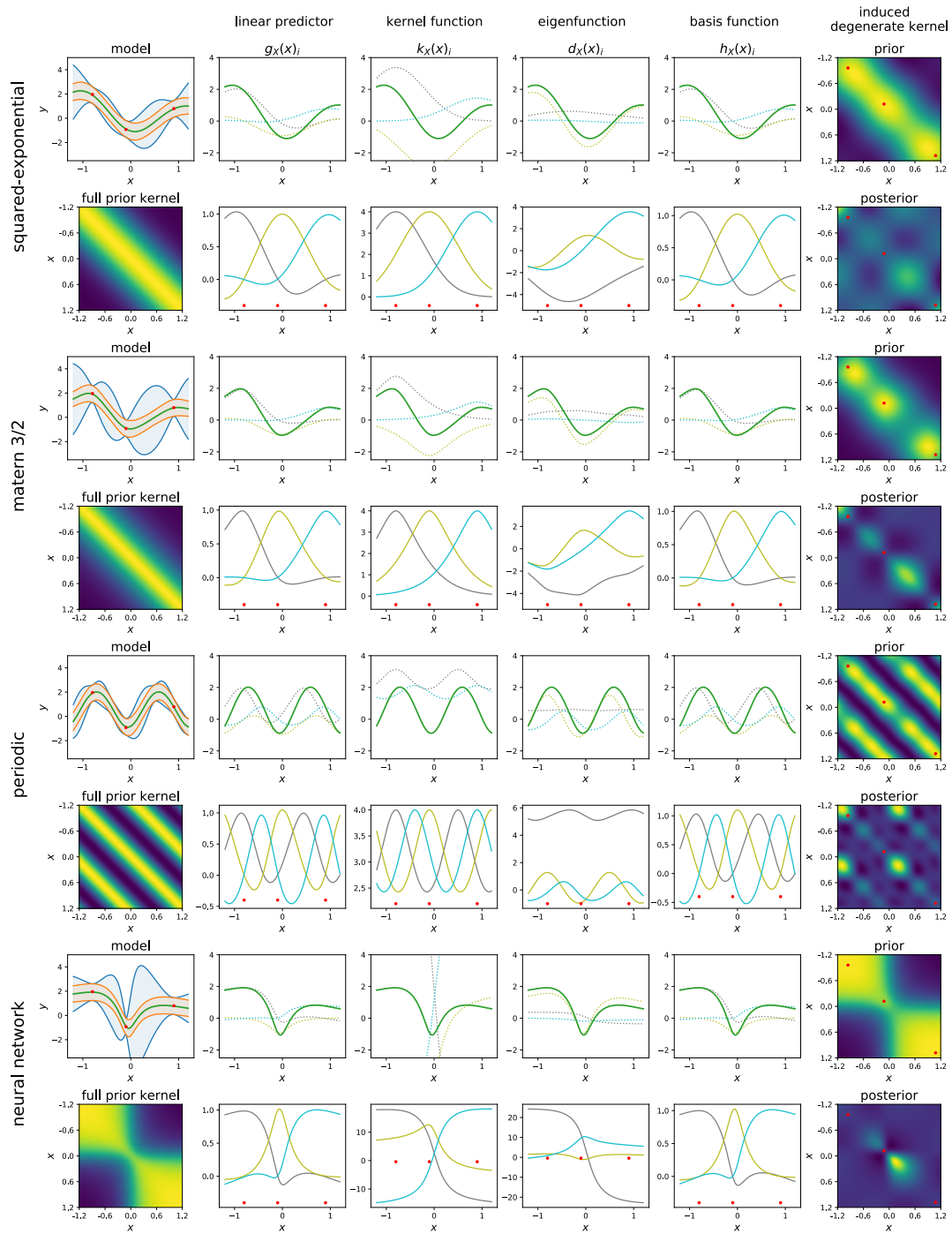


Figure 3.5. Four different basis function decomposition of the predictive mean corresponding to four different kernels. Although the predictive mean is equal, the predictive variance is different outside the training data.

Therefore, the posterior distribution of the latent function values of a GP corresponds exactly to the posterior of the weights of a Bayesian linear model with choice of basis functions and prior covariance as in Lemma 3.1. Note that these basis functions depend on the input data  $\mathbf{X}$  and the number of basis functions corresponds to the number of training data samples  $N$ . Although the predictive mean, the posterior and marginal likelihood are equal, there is a main difference, namely that the predictive variance is different.

**Lemma 3.2 (Different Predictive Variance)** *Consider the same setting as in Proposition 3.2. The predictive posterior variance defined in (2.43) and (3.11) differ by*

$$\sigma_{GP}^2(\mathbf{x}_*) - \sigma_L^2(\mathbf{x}_*) = \mathbf{K}_{\mathbf{x}_*\mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_*\mathbf{X}}\mathbf{K}_{\mathbf{X}\mathbf{X}}^{-1}\mathbf{K}_{\mathbf{X},\mathbf{x}_*}. \quad (3.17)$$

The difference of the predictive variance in (3.17) is illustrated in Figure 3.6 for two squared-exponential kernels with different lengthscales per row. The additional variance term in (3.17) of the GP predictive variance has the effect that outside the observed data the variance can fall back to the prior variance  $\mathbf{K}_{\mathbf{x}_*\mathbf{x}_*}$  and solves the overconfidence in Bayesian linear models. What is missing in a Bayesian linear model is the conditional variance  $p(f_*|\mathbf{f})$  as defined in (3.13). This is the effect of the infinite dimensional prior in a GP so that the basis functions cover all the space also outside of the training data.

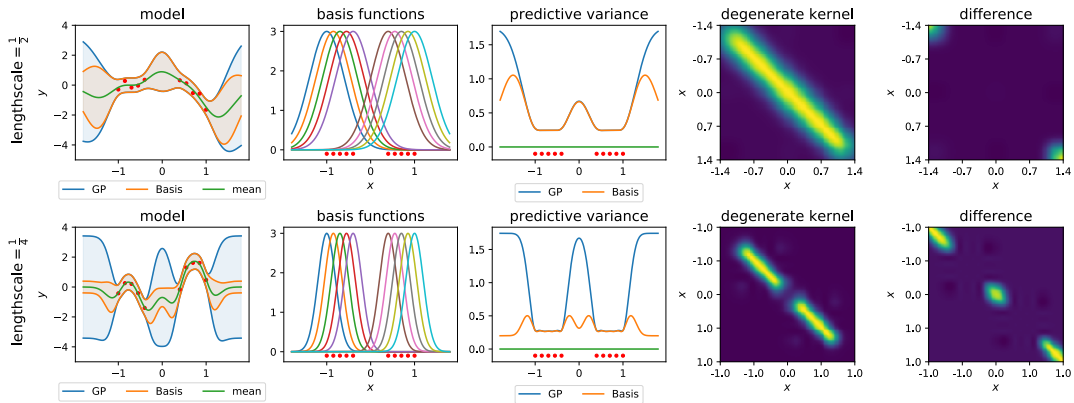


Figure 3.6. Comparison of predictive variance of a Bayesian linear model and a GP. We observe that the predictive posterior variance of a Bayesian linear model is overconfident due to the missing variance outside the training data and the induced kernel is degenerate.

We have seen that a Bayesian linear model leads to a GP with a *degenerate* kernel, which has the effect that the predictive variance outside the training data is missing. If the focus lies on the point-wise predictive distribution, there is an easy way to correct for the missing predictive variance.

**Definition 3.2 (Degenerated and Augmented Kernel)** Given a non-degenerate kernel  $k(\mathbf{x}, \mathbf{x}')$  and a finite index set  $\mathbf{A} \in \mathbb{R}^{M \times D}$ , a **degenerated** kernel of dimension  $M$  can be constructed as

$$\tilde{k}_{\mathbf{A}}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{x}'). \quad (3.18)$$

In order to correct for the missing variance, this kernel can be augmented to an **augmented** kernel of dimension  $M$  by

$$\bar{k}_{\mathbf{A}}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{x}') + \delta_{\mathbf{x}=\mathbf{x}'}v(\mathbf{x}), \quad (3.19)$$

where the corrected variance is

$$v(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - k(\mathbf{x}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{x}) \quad (3.20)$$

and  $\delta_{\mathbf{x}=\mathbf{x}'}$  is the Kronecker delta which is one iff  $\mathbf{x} = \mathbf{x}'$ .

This kernel can be seen as a degenerated kernel by correcting the missing predictive variance (3.17) of a Bayesian linear model compared to the one of a GP using the input data for the finite index set  $\mathbf{A} = \mathbf{X}$ . Note that an augmented kernel is technically non-degenerate since it has infinite many non-zero eigenvalues due to the additional variance  $v(\mathbf{x})$  for  $\mathbf{x} = \mathbf{x}'$ , however, the covariance for  $\mathbf{x} \neq \mathbf{x}'$  correspond to the degenerate kernel induced by the finite index set. We denote a GP with augmented kernel an augmented GP.

**Proposition 3.3 (Inference with Full GPs and augmented GPs)**

Consider a GP  $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}'))$  with a non-degenerate kernel  $k(\mathbf{x}, \mathbf{x}')$  and a GP  $g(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \bar{k}_{\mathbf{X}}(\mathbf{x}, \mathbf{x}'))$  with augmented kernel  $\bar{k}_{\mathbf{X}}(\mathbf{x}, \mathbf{x}')$  as defined in Definition 3.2 with index set  $\mathbf{A} = \mathbf{X}$ . Then the posterior distribution  $p(f(\mathbf{X})|\mathbf{y}) = p(g(\mathbf{X})|\mathbf{y})$ , the marginal likelihood  $p_f(\mathbf{y}) = p_g(\mathbf{y})$  and the (pointwise) predictive distribution  $p(f(\mathbf{x}_*)|\mathbf{y}) = p_g(g(\mathbf{x}_*)|\mathbf{y})$  of  $f$  and  $g$  are equivalent.

This means that GP inference with a non-degenerated kernel is equivalent to GP inference using an augmented kernel with index set  $\mathbf{A} = \mathbf{X}$  from a pointwise prediction point of view. Therefore, augmented GPs are not really interesting for full GPs. However, in the next Chapter 4, we show that exact inference with an augmented kernel with index set  $\mathbf{A} \in \mathbb{R}^{M \times D}$  with  $M < N$  constitutes the starting point for sparse inducing point GPs. Moreover, in Chapter 5, we establish a connection with an extended Bayesian linear regression model with data-dependent observation noise, so that sequential Bayesian inference is equal to exact inference in a GP with augmented kernel.

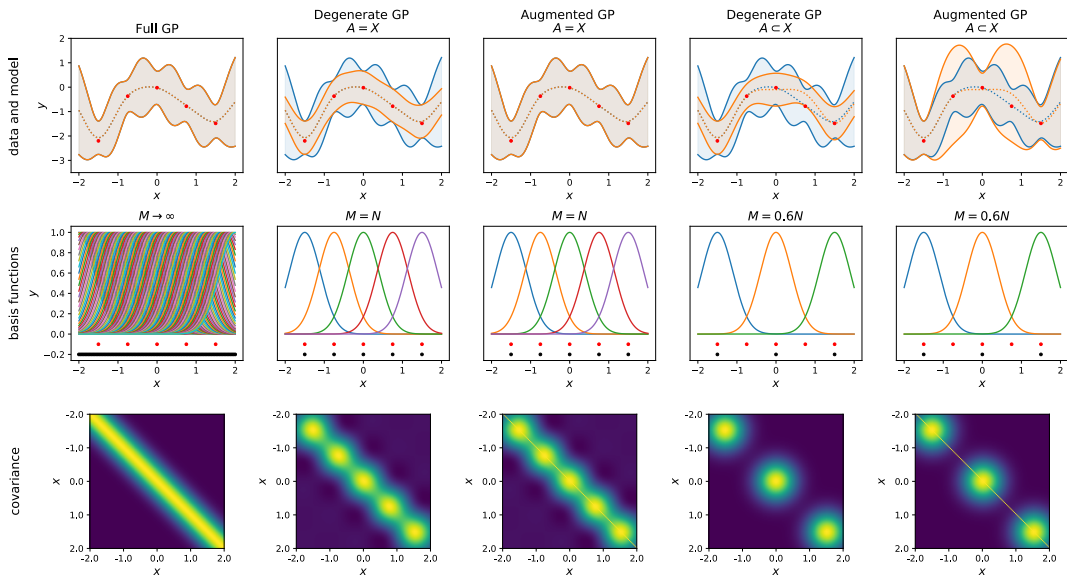


Figure 3.7. Comparison of full GP, degenerate GP and augmented GP, where in the latter the missing covariance on the diagonal is corrected. This has the effect, that the predictive posterior distribution outside the data is also correct.

### 3.5 Advantages and Limitations of GPs

Gaussian processes have many benefits.

- **Analytic Inference:** Given a kernel function with known hyperparameters, the predictive posterior distribution can be computed analytically in closed form which is a rare property for nonparametric models.
- **Expressivity:** With the flexible choice of kernels as covariance functions, GPs allow to express a wide range of arbitrary complex functions. Further, they allow to incorporate the modeling assumptions in a principled way.
- **Inference with Functions:** Inference in a GP can be seen as inference directly in the infinite dimensional space of functions, which is not possible with traditional Bayesian finite dimensional models.
- **Few Hyperparameters:** Compared to parametric models (e.g. neural networks), GPs only have a few kernel hyperparameters which makes the model easy to train and is less prone to overfitting.
- **Credible Intervals:** Since prediction in GPs does not correspond only to point estimates but instead to a whole predictive posterior distribution, also credible intervals and posterior samples are available.

- **Less Prone to Overfitting:** Due to the Bayesian paradigm and that the latent function values can be analytically integrated out, GPs are rather robust against overfitting. As a consequence, the generalization capacity is also good for a small number of data samples.
- **Model Selection:** Since the marginal likelihood of the data given a model can be analytically computed for a GP, it provides a principled way of comparing different models.

However, they also have some drawbacks.

- **Scalability:** The main issue with GPs is the slow inference for large number of data sample  $N$ , basically due to the inversion of the kernel matrix  $\mathbf{K}_{XX}$ . The time complexity for computing the predictive posterior and the marginal likelihood is  $\mathcal{O}(N^3)$  and the required space is  $\mathcal{O}(N^2)$ . There exist already several approaches as it will be discussed in Chapter 4 and we propose new contributions in Chapters 5-7.
- **Non-Gaussian Likelihoods:** For non-Gaussian likelihoods (for instance classification or heavy tails), the posterior is no longer analytically available and requires therefore approximate inference. Several approaches are proposed and we refer to [Rasmussen and Williams, 2006, Ch. 3] and to [Benavoli et al., 2020] for a discussion and a recent approach.
- **Choice of Kernel:** The flexibility of GPs with kernels come with the drawback that the kernel has to be specified. There are some approaches for automatically choose an optimal kernel [Abdessalem et al., 2017; Teng et al., 2020], however, until recently, human experts are required to choose a suitable kernel.



# Chapter 4

## Approximations for Gaussian Processes

In this section, we review existing approximations for Gaussian processes. As we have seen in the previous chapter, full GPs have many advantages, however, they suffer from one main limitation: the cubic time and quadratic space complexity in the number of training samples. In order to scale GPs to larger datasets, several approaches have been proposed in the literature, where we refer to Liu et al. [2020] for a recent review.

In this chapter, we first provide an overview of different approaches for GP approximations in Section 4.1. Subsequently, we focus on two classes of GP approximations. On the one hand, we discuss sparse GP approximations based on so-called *inducing points*, which are auxiliary points on the GP optimally summarizing the dependencies among all training points *globally*. This will be discussed in Section 4.2. On the other hand, we review *local* approaches, where independent and local GP models are combined with averaging methods, as we will discuss in 4.3.

Compared to the standard way GP approximations are introduced in the literature, this chapter provides a unifying framework to present and review GP approximations. In particular, in Section 4.2, we unify several sparse GP models by explaining that they differ only by the training and test conditional covariances, which will also be exploited in Chapter 5. Further, we show that sparse GP models, obtained by exact inference with approximate priors and models obtained via approximate inference, but keeping the exact priors, lead to the same posterior distributions. Further, also in Section 4.3 about local approaches for GPs, we unify many methods by introducing a link function for the predictive posterior distribution of the local experts. Finally, we contrast the advantages and limitations of global and local GP approximation approaches, which constitute the starting points for the development of novel methods in Chapters 5-7.

## 4.1 Overview of GP Approximations

As discussed in the previous chapter, the main limitation of GP regression is the inversion of the matrix  $\mathbf{K}_{XX} + \sigma_n^2 \mathbb{I} \in \mathbb{R}^{N \times N}$  in (3.8), which is not feasible for large number  $N$  of training samples. Consequently, GP approximations are based on different assumptions about the kernel matrix  $\mathbf{K}_{XX}$ . In the following, we briefly provide an overview and refer to Liu et al. [2020] for a recent review.

### Sparse Kernel Matrices

There are several GP approximations approaches, which impose some kind of sparsity directly in the kernel matrix. Some methods [Gneiting, 2002; Melkumyan and Ramos, 2009] rely explicitly on *sparse kernels*, which guarantee that some entries in the kernel matrix are zero. *Structured sparse* approximations exploit other sparse structures in the kernel matrix and iteratively solve the corresponding sparse linear system  $(\mathbf{K}_{XX} + \sigma_n^2 \mathbb{I})^{-1} \mathbf{y}$ . The different assumed sparsity structures range from kd-trees [Shen et al., 2005],  $\mathcal{H}^2$ -matrices [Börm and Garcke, 2007] to Toeplitz-matrices [Cunningham et al., 2008], and Kronecker-structures [Gilboa et al., 2013]. Exploiting these sparsity structures enables to scale these methods to much larger datasets, however, they require grid data inputs. The improved approaches by [Wilson and Nickisch, 2015; Wilson et al., 2015] can deal with arbitrary datasets, by instead imposing the grid structure on auxiliary points and use for the kernel evaluations of the data points local linear interpolation schemes. These approaches can be combined with fast GPU computations [Wang et al., 2019], leading to highly scalable approaches with good accuracy.

### Low-Rank Kernel Matrices

A different research direction for GP approximations is based on a particular low-rank approximation  $\mathbf{K}_{XX} \approx \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX}$  of the kernel matrix, known as *Nyström approximation* [Williams and Seeger, 2000]. It involves  $M$  so-called *inducing points*  $\mathbf{A}$ , that globally summarize the dependencies among the  $N$  data input points  $\mathbf{X}$ , with  $M \ll N$ . Using this low-rank kernel matrix improves the scalability of GPs significantly [Williams and Seeger, 2000], however, it might produce negative and unreliable predictive variances. In order to overcome this issue, several probabilistic models based on this Nyström approximation involving global inducing points have been proposed. We refer for instance to [Quiñero-Candela and Rasmussen, 2005; Titsias, 2009; Bui et al., 2017b] and for an overview to Table 4.1 in Section 4.2, where we will thoroughly discuss these approaches.



### Sparse Precision Kernel Matrices

Several GP approximations exploit sparsity in the precision matrix  $\mathbf{K}_{XX}^{-1}$  instead of in the kernel matrix  $\mathbf{K}_{XX}$ . Sparsity in a Gaussian precision kernel matrix corresponds to conditional independence of certain regions in the input space. Thereby, the structure of these regions can range from band structures [Durand et al., 2019] to Voronoi-tessellations [Kim et al., 2005] and trees [Bui and Turner, 2014], where the structure of the regions in the input space is static, and in particular, independent of the test inputs. This is different in so-called *transductive* approaches [Gramacy and Apley, 2015; Datta et al., 2016; Katzfuss and Guinness, 2021], where dynamic partitions of the input space are employed. All those methods are highly scalable and well suited for spatial data, where independent regions are meaningful and reasonable.

### Block-Diagonal Kernel Matrices with Prediction Aggregation

A particular kind of sparse kernel matrices are block-diagonal matrices, corresponding to mutually independent regions in the input space. However, directly using this kind of kernel approximation is often a too strong assumption, unless the regions are actually independent. In order to compensate for the missing dependencies, probabilistic aggregation schemes of the predictions from the independent and local regions can be used. For aggregating these predictions, there are mainly two research directions: *mixture of experts* (MoE) [Yuksel et al., 2012; Masoudnia and Ebrahimpour, 2014] and *product of experts* (PoE) [Hinton, 2002; Fleet, 2014; Deisenroth and Ng, 2015]. Thereby, the predictive densities are combined with a weighted sum and a weighted product in MoE and PoE, respectively, resulting in efficient and highly scalable GP approximations. We discuss PoEs thoroughly in Section 4.3.

### State Space Approaches

In GP approximations approaches based on so-called *state space models*, the kernel matrix is implicitly modeled. In particular, Hartikainen and Särkkä [2010]; Sarkka et al. [2013] established the connection between GPs and state space models for spatio-temporal regression problems, which allows to apply sequential algorithm such as the *Kalman Filter* [Kalman et al., 1960]. Inspired by this line of research, the authors in [Carron et al., 2016; Todescato et al., 2017; Benavoli and Zaffalon, 2016] focused on efficient implementation and extended the methodology to varying sampling locations over time. These approaches are highly scalable and well suited for spatio-temporal data, however, they rely on particular kernels and are not applicable for higher-dimensional inputs.

## 4.2 Sparse Global GPs

Sparse GP regression approximations based on *global inducing points* reduce the computational complexity by introducing  $M \ll N$  inducing points  $\{a_j, \mathbf{A}_j\}_{j=1}^M$ , that summarize the dependency of the whole training data  $\{y_i, \mathbf{X}_i\}_{i=1}^N$  globally, as illustrated in Figure 4.1. Thereby, the inducing *inputs*  $\mathbf{A}_j \in \mathbb{R}^D$  are in the  $D$ -dimensional input data space and the inducing *outputs*  $a_j = f(\mathbf{A}_j) \in \mathbb{R}$  are the corresponding latent function values on the GP  $f$ . We introduce the notation for the inducing inputs  $\mathbf{A} = [\mathbf{A}_1, \dots, \mathbf{A}_M]^T \in \mathbb{R}^{M \times D}$  and the corresponding inducing outputs  $\mathbf{a} = [a_1, \dots, a_M]^T \in \mathbb{R}^M$  so that  $\mathbf{a} = f(\mathbf{A})$ . The main idea of sparse global inducing point methods is to project the  $N$  latent function values  $\mathbf{f}$  corresponding to the  $N$  data points onto the  $M$  inducing points  $\mathbf{a}$ , where we assume that  $M$  is much smaller than  $N$ . This results in a clever chosen low-rank approximation of  $\mathbf{K}_{XX}$ , so that an approximation for the intractable inverse  $\mathbf{K}_{XX}^{-1}$  can be efficiently computed. This approach leads to a time  $\mathcal{O}(NM^2)$  and space  $\mathcal{O}(NM)$  complexity, instead of  $\mathcal{O}(N^3)$  and space  $\mathcal{O}(N^2)$  for full GP. In the first part of this chapter, we solely follow [Quiñonero-Candela and Rasmussen, 2005], however, we present it in a unifying way, including several works as summarized in Table 4.1.

### 4.2.1 Augmented Model via Inducing Points

In the following section, we describe the sparse GP model augmented with inducing points  $\mathbf{a}$ . In particular, the joint GP prior distribution  $p(\mathbf{f}, \mathbf{f}_*)$  (3.4) over the latent function values of the training  $\mathbf{f} \in \mathbb{R}^N$  and test data  $\mathbf{f}_* \in \mathbb{R}^{N_{test}}$ , can be augmented by any other variable  $\mathbf{a} = f(\mathbf{A}) \in \mathbb{R}^M$  on the GP  $f$ , that is, the exact extended prior can be written as

$$p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*) = \mathcal{N} \left( \begin{bmatrix} \mathbf{a} \\ \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{AA} & \mathbf{K}_{AX} & \mathbf{K}_{AX_*} \\ \mathbf{K}_{XA} & \mathbf{K}_{XX} & \mathbf{K}_{XX_*} \\ \mathbf{K}_{X_*A} & \mathbf{K}_{X_*X} & \mathbf{K}_{X_*X_*} \end{bmatrix} \right). \quad (4.1)$$

Thereby, the original marginal distribution is not affected due to the marginalization property (3.3) of a GP, so that the original prior (3.4) can be recovered by marginalizing out the additional variable  $\mathbf{a}$

$$p(\mathbf{f}, \mathbf{f}_*) = \int p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*) d\mathbf{a}. \quad (4.2)$$

Combining the prior in (4.1) with the Gaussian likelihood  $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I})$  in (3.5), the augmented joint prior

$$p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*) \quad (4.3)$$

is induced similarly to (3.6). From (4.3), for instance the exact marginal likelihood (3.15)

$$p(\mathbf{y}) = \int p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) d\mathbf{f} d\mathbf{f}_* d\mathbf{a} \quad (4.4)$$

and exact posterior distribution (3.12)

$$p(\mathbf{f}|\mathbf{y}) = \int \frac{p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y})}{p(\mathbf{y})} d\mathbf{f}_* d\mathbf{a} \quad (4.5)$$

can be recovered by just integrating also with respect to the inducing points.

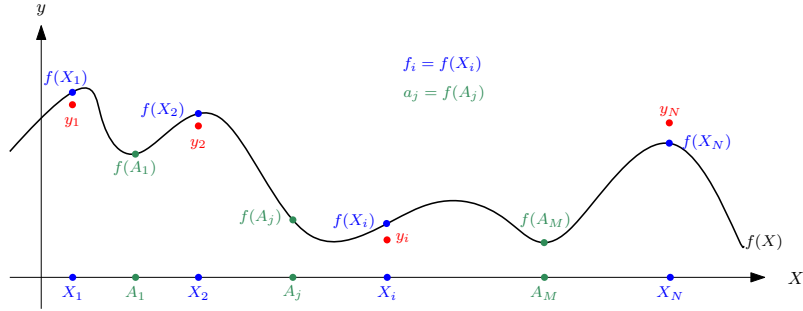


Figure 4.1. Illustration of global inducing points with inducing inputs  $\mathbf{A} \in \mathbb{R}^{M \times D}$  and corresponding outputs  $\mathbf{a} = f(\mathbf{A}) \in \mathbb{R}^M$ .

#### 4.2.1.1 Conditional Independence of Training and Prediction

The sparse GP model including the inducing points  $\mathbf{a}$  as described in the previous section is still exact, however, also no computation benefits are achieved. In order to obtain computational feasible models, some assumptions have to be made. We first rewrite the joint distribution in (4.1) as

$$p(\mathbf{a}, \mathbf{f}, \mathbf{f}_*) = p(\mathbf{f}_*|\mathbf{f}, \mathbf{a})p(\mathbf{f}, \mathbf{a}) = p(\mathbf{f}_*|\mathbf{f}, \mathbf{a})p(\mathbf{f}|\mathbf{a})p(\mathbf{a}) \quad (4.6)$$

based on basic rules of probability theory. One fundamental assumption for sparse GPs based on global inducing points is the conditional independence between the training and test latent function values given the inducing points.

**Assumption 4.1 (Conditional Independence of Training and Prediction)** *The latent function values  $\mathbf{f}$  of the training data are assumed to be conditionally independent of the test latent function values  $\mathbf{f}_*$  given the global inducing points  $\mathbf{a}$ , that is,  $\mathbf{f} \perp\!\!\!\perp \mathbf{f}_* \mid \mathbf{a}$ . Therefore*

$$p(\mathbf{f}_*|\mathbf{f}, \mathbf{a}) \approx q(\mathbf{f}_*|\mathbf{f}, \mathbf{a}) = p(\mathbf{f}_*|\mathbf{a}),$$

where we use  $q$  to indicate that it is an approximate distribution.

As a consequence from the assumption above,  $f$  and  $f_*$  can only interact through  $\mathbf{a}$ , which *induces* implicit dependencies, thus the name *inducing points*. Plugging these assumptions back into (4.6), we get for the joint prior

$$q(\mathbf{f}, \mathbf{f}_*, \mathbf{a}) = p(\mathbf{f}_* | \mathbf{a})p(\mathbf{f}, \mathbf{a}), = p(\mathbf{f}_* | \mathbf{a})p(\mathbf{f} | \mathbf{a})p(\mathbf{a}), \quad (4.7)$$

where the joint distribution  $p(\mathbf{f}, \mathbf{a}) = p(\mathbf{f} | \mathbf{a})p(\mathbf{a})$  of the training values and the inducing points is still exact, only the prediction is affected by Assumption 4.1.

#### 4.2.1.2 Inference Approaches

In the literature, there are at least 2 directions how to proceed next from here. The first approach makes direct further assumptions about the intractable conditional  $p(\mathbf{f} | \mathbf{a})$  leading to an **approximate prior**  $q(\mathbf{f}, \mathbf{a}) = q(\mathbf{f} | \mathbf{a})p(\mathbf{a})$  from which **exact inference** can be applied, which will be discussed in Section 4.2.2. The second approach retains the **exact prior**  $p(\mathbf{f}, \mathbf{a})$  and directly tries to find a posterior distribution via **approximate inference** techniques, as we will discuss in Section 4.2.3. Interestingly, regarding level I inference, i.e. inferring the inducing outputs, those two approaches yield the same models, they only differ how they treat the hyperparameters on level II of the Bayesian hierarchical model. This will be discussed in Section 4.2.4.

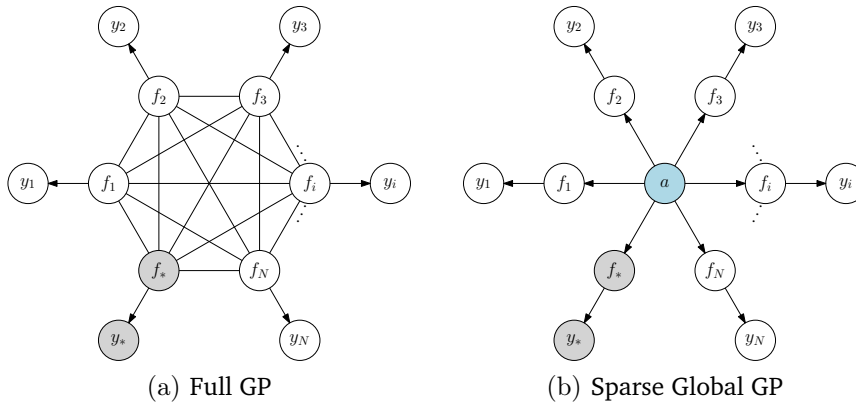


Figure 4.2. Graphical models of full GP and sparse GP with global inducing points  $\mathbf{a}$ .

### 4.2.2 Exact Inference with Approximate Prior

In this Section, we discuss approaches based on an approximate prior  $q(\mathbf{f}, \mathbf{a}) = q(\mathbf{f}|\mathbf{a})p(\mathbf{a})$  combined with exact inference. In particular, further assumptions on the conditionals  $q(\mathbf{f}_*|\mathbf{a}) \approx p(\mathbf{f}_*|\mathbf{a})$  and  $q(\mathbf{f}|\mathbf{a}) \approx p(\mathbf{f}|\mathbf{a})$  in (4.7) are imposed, leading to an approximate prior

$$q(\mathbf{f}, \mathbf{f}_*, \mathbf{a}) = q(\mathbf{f}_*|\mathbf{a})q(\mathbf{f}|\mathbf{a})p(\mathbf{a}), \quad (4.8)$$

from which *exact* Bayesian inference can be applied to obtain an indirect posterior approximation. The true training and test conditional distributions implied by a GP are

$$\begin{aligned} p(\mathbf{f}|\mathbf{a}) &= \mathcal{N}(\mathbf{f} | \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{a}, \mathbf{K}_{XX} - \mathbf{Q}_{XX}); \\ p(\mathbf{f}_*|\mathbf{a}) &= \mathcal{N}(\mathbf{f}_* | \mathbf{K}_{X_*A} \mathbf{K}_{AA}^{-1} \mathbf{a}, \mathbf{K}_{X_*X_*} - \mathbf{Q}_{X_*X_*}), \end{aligned} \quad (4.9)$$

where  $\mathbf{Q}_{XX}$  and  $\mathbf{Q}_{X_*X_*}$  are defined as follows.

**Definition 4.1 (Nyström Matrix)** For a input matrix  $\mathbf{A} \in \mathbb{R}^{M \times D}$  and degenerated kernel  $\tilde{k}_A(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{x}')$  induced by a non-degenerate kernel  $k$  as defined in (3.18), we define the corresponding approximate kernel matrix as

$$\mathbf{Q}_{BC} = \tilde{k}_A(\mathbf{B}, \mathbf{C}) = k(\mathbf{B}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{C}) \in \mathbb{R}^{M_1 \times M_2} \quad (4.10)$$

for any  $\mathbf{B} \in \mathbb{R}^{M_1 \times D}$  and  $\mathbf{C} \in \mathbb{R}^{M_2 \times D}$ . This matrix is also called Nyström matrix, e.g. Williams and Seeger [2000].

Note that, using the true conditionals (4.9) and input matrix  $\mathbf{A} \neq \mathbf{X}$  would lead to a GP with exact distribution  $p(\mathbf{a}, \mathbf{f})$  (and thus also exact posterior distribution  $p(\mathbf{f}|\mathbf{y})$ ), but the predictive distribution would be still different compared to full GP due to Assumption 4.1. We denote this model as *indexed GP*, compare also Table 4.1. Note that, this model is still intractable and does not have any practical advantages, however, it allows interesting comparisons. For computational tractable models, further assumptions about the covariances in the conditionals (4.9) have to be made, where the matrix  $\mathbf{K}_{X_*X_*}$  and in particular  $\mathbf{K}_{XX}$  lead to intractable computations in the inference procedure.

**Assumption 4.2 (Training and Test Conditional)** The intractable full covariances in the true conditionals in (4.9) are approximated by some generic training and test projection covariances  $\bar{\mathbf{V}} \in \mathbb{R}^{N \times N}$  and  $\bar{\mathbf{V}}_* \in \mathbb{R}^{N_{test} \times N_{test}}$ , for which we assume

the inverses  $\bar{\mathbf{V}}^{-1}$  and  $\bar{\mathbf{V}}_*^{-1}$  can be computed efficiently. Therefore, the approximate conditionals have the following structure

$$\begin{aligned} q(\mathbf{f}|\mathbf{a}) &= \mathcal{N}(\mathbf{f}|\mathbf{H}\mathbf{a}, \bar{\mathbf{V}}) \\ q(\mathbf{f}_*|\mathbf{a}) &= \mathcal{N}(\mathbf{f}_*|\mathbf{H}_*\mathbf{a}, \bar{\mathbf{V}}_*) \end{aligned} \quad (4.11)$$

with projection matrices  $\mathbf{H} = \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1} \in \mathbb{R}^{N \times M}$  and  $\mathbf{H}_* = \mathbf{K}_{X_*A}\mathbf{K}_{AA}^{-1} \in \mathbb{R}^{N_{test} \times M}$ , so that the conditional means are correct (4.9).

In order to circumvent the intractable inverses of the covariances, the different approaches in the literature differ basically only by their additional conditional independence assumptions between the training and test latent function values as summarized in Table 4.1. All of those assumptions lead either to a diagonal or block diagonal structure for  $\bar{\mathbf{V}}$  and  $\bar{\mathbf{V}}_*$ . For a relatively small number of  $N_{test}$  points,  $\bar{\mathbf{V}}_*$  is even set to the exact covariance (4.9) in the most methods. However, this has the effect that it cannot be seen as a GP as it will be discussed in Section 4.2.2.7. Note that, when only considering pointwise predictions (for instance in credible intervals), there is no difference whether using the full  $\bar{\mathbf{V}}_*$  or only the diagonal of it.

**Definition 4.2 (Sparse GP Model)** *The sparse GP model for regression based on inducing points can be probabilistically summarized by*

$$\begin{aligned} p(\mathbf{y}|\mathbf{f}) &= \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2\mathbb{I}); \\ p(\mathbf{a}) &= \mathcal{N}(\mathbf{0}, \Sigma_0); \\ q(\mathbf{f}|\mathbf{a}) &= \mathcal{N}(\mathbf{f}|\mathbf{H}\mathbf{a}, \bar{\mathbf{V}}); \\ q(\mathbf{f}_*|\mathbf{a}) &= \mathcal{N}(\mathbf{f}_*|\mathbf{H}_*\mathbf{a}, \bar{\mathbf{V}}_*), \end{aligned} \quad (4.12)$$

so that the approximate joint distribution becomes

$$q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) = p(\mathbf{y}|\mathbf{f}) q(\mathbf{f}_*|\mathbf{a}) q(\mathbf{f}|\mathbf{a}) p(\mathbf{a}). \quad (4.13)$$

The graphical model for a general sparse GP with global inducing points is depicted in Figure 4.2b together with the graphical model of full GP in Figure 4.2a. We can observe, that in full GP, all latent function values  $\mathbf{f}$  and  $\mathbf{f}_*$  are fully connected, whereas in sparse GPs, those dependencies are replaced by the global inducing variables with the assumptions that the latent function values are conditionally independent. Note that, in the case of block-diagonal covariances, the latent function variables in the graphical model can be grouped into blocks where inside the block the full covariances are modeled.

description	training conditional covariance $\bar{V}$	test conditional covariance $\bar{V}_*$	GP?	kernel
<b>full GP</b>	$\mathbf{0}$	$K_{X_*X_*} - K_{X_*X} K_{XX}^{-1} K_{XX_*}$	✓	$k$
<b>indexed GP</b>	$K_{XX} - Q_{XX}$	$K_{X_*X_*} - Q_{X_*X_*}$	✓	$k_A$
<b>degenerated GP</b> SoR [Silverman, 1985], DIC [Smola and Bartlett, 2001]	$\mathbf{0}$	$\mathbf{0}$	✓	$\tilde{k}_A$
<b>augmented training</b>	$Diag[K_{XX} - Q_{XX}]$	$\mathbf{0}$	×	
<b>augmented testing</b>	$\mathbf{0}$	$Diag[K_{X_*X_*} - Q_{X_*X_*}]$	×	
<b>augmented GP</b> FIC [Quiñonero-Candela and Rasmussen, 2005]	$Diag[K_{XX} - Q_{XX}]$	$Diag[K_{X_*X_*} - Q_{X_*X_*}]$	✓	$\bar{k}_A$
<b>deterministic "GP"</b> PP [Csató and Opper, 2002], DTC [Seeger et al., 2003], VFE [Titsias, 2009]	$\mathbf{0}$	$K_{X_*X_*} - Q_{X_*X_*}$	×	
<b>augmented "GP"+</b> SSGP [Snelson and Ghahramani, 2006], FITC [Quiñonero-Candela and Rasmussen, 2005], EP [Bui et al., 2017b]	$Diag[K_{XX} - Q_{XX}]$	$K_{X_*X_*} - Q_{X_*X_*}$	×	
<b>block augmented "GP"+</b> PITC [Quiñonero-Candela and Rasmussen, 2005]	$BlkDiag[K_{XX} - Q_{XX}]$	$K_{X_*X_*} - Q_{X_*X_*}$	×	
<b>power "GP"</b> PEP [Bui et al., 2017b]	$\alpha Diag[K_{XX} - Q_{XX}]$	$K_{X_*X_*} - Q_{X_*X_*}$	×	
<b>block power "GP"</b> PEP-B [Bui et al., 2017b]	$\alpha BlkDiag[K_{XX} - Q_{XX}]$	$K_{X_*X_*} - Q_{X_*X_*}$	×	
<b>block augmented GP</b> PIC [Snelson and Ghahramani, 2007]	$BlkDiag[K_{XX} - Q_{XX}]$	$BlkDiag[K_{X_*X_*} - Q_{X_*X_*}]$	✓	$\hat{k}_A$

Table 4.1. Unifying view of sparse global GP approaches by different choices of the training  $\bar{V}$  and test  $\bar{V}_*$  conditional covariance approximations (4.11), where the Nyström matrices  $Q_{XX}$  and  $Q_{X_*X_*}$  are defined in Definition 4.1. The column *GP?* indicates whether the corresponding sparse GP model can be seen as a GP according to Definition 3.1 with a particular kernel (as indicated in the column *kernel*) as further discussed in Section 4.2.2.7 and Table 4.3.

#### 4.2.2.1 Approximate Prior Distribution

Using the approximate training  $q(\mathbf{f}|\mathbf{a})$  and test conditional  $q(\mathbf{f}_*|\mathbf{a})$  in (4.11) together with the true prior  $p(\mathbf{a}) = \mathcal{N}(\mathbf{a}|\mathbf{0}, K_{AA})$  of the inducing points as summarized in (4.12), the joint induced approximate prior in (4.8) becomes  $q(\mathbf{f}, \mathbf{f}_*, \mathbf{a}) = q(\mathbf{f}_*|\mathbf{a}) q(\mathbf{f}|\mathbf{a}) p(\mathbf{a})$  which can be written as

$$q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*) = \mathcal{N} \left( \begin{bmatrix} \mathbf{a} \\ \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K_{AA} & K_{AX} & K_{AX_*} \\ K_{XA} & Q_{XX} + \bar{V} & Q_{XX_*} \\ K_{X_*A} & Q_{X_*X} & Q_{X_*X_*} + \bar{V}_* \end{bmatrix} \right). \quad (4.14)$$

	density	mean	cov.	$\int d\mathbf{a}$	$\int d\mathbf{f}$	$\int d\mathbf{f}_*$	$\int d\mathbf{y}$	$\frac{1}{p(\mathbf{a})}$	$\frac{1}{q(\mathbf{f})}$	$\frac{1}{q(\mathbf{y})}$
marginal likelihood	$q(\mathbf{y})$	$\mathbf{0}$	$\mathbf{P}$	x	x	x				
prior	$p(\mathbf{a})$	$\mathbf{0}$	$\mathbf{K}_{AA}$		x	x	x			
induced prior	$q(\mathbf{f})$	$\mathbf{0}$	$\mathbf{Q}_{XX} + \bar{\mathbf{V}}$	x		x	x			
predictive prior	$q(\mathbf{f}_*)$	$\mathbf{0}$	$\mathbf{Q}_{X_*X_*} + \bar{\mathbf{V}}_*$		x		x			
likelihood	$p(\mathbf{y} \mathbf{f})$	$\mathbf{f}$	$\sigma_n^2\mathbb{I}$	x		x			x	
projection conditional	$q(\mathbf{f} \mathbf{a})$	$\mathbf{H}\mathbf{a}$	$\bar{\mathbf{V}}$			x	x	x		
induced likelihood	$q(\mathbf{y} \mathbf{a})$	$\mathbf{H}\mathbf{a}$	$\mathbf{V}$		x	x			x	
predictive conditional	$q(\mathbf{f}_* \mathbf{a})$	$\mathbf{H}_*\mathbf{a}$	$\bar{\mathbf{V}}_*$		x		x		x	
posterior	$q(\mathbf{a} \mathbf{y})$	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$		x	x				x
predictive posterior	$q(\mathbf{f}_* \mathbf{y})$	$\boldsymbol{\mu}_f(\mathbf{X}_*)$	$\boldsymbol{\Sigma}_f(\mathbf{X}_*)$			x		x		x

Table 4.2. Derived densities from the joint distribution  $q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y})$  for sparse GPs by marginalization and conditioning. Note that the prior  $p(\mathbf{a})$  and likelihood  $p(\mathbf{y}|\mathbf{f})$  are exact.

Thereby, we used  $\mathbf{H}\mathbf{K}_{AA} = \mathbf{K}_{XA}$ ,  $\mathbf{H}_*\mathbf{K}_{AA} = \mathbf{K}_{X_*A}$ ,  $\mathbf{H}\mathbf{K}_{AA}\mathbf{H}_* = \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX_*} = \mathbf{Q}_{XX_*}$  as defined in (4.10) and similarly for  $\mathbf{H}\mathbf{K}_{AA}\mathbf{H} = \mathbf{Q}_{XX}$  and  $\mathbf{H}_*\mathbf{K}_{AA}\mathbf{H}_* = \mathbf{Q}_{X_*X_*}$ . Comparing (4.14) with (4.1), we can observe directly the approximation on the joint prior. Combining the joint induced prior  $q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*)$  with the Gaussian likelihood  $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2\mathbb{I})$  in (3.5), yields the full joint distribution of the approximate GP model  $q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) = p(\mathbf{y}|\mathbf{f})q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*)$  which can be analytically computed by

$$q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) = \mathcal{N} \left( \begin{bmatrix} \mathbf{a} \\ \mathbf{f} \\ \mathbf{f}_* \\ \mathbf{y} \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{AA} & \mathbf{K}_{AX} & \mathbf{K}_{AX_*} & \mathbf{K}_{AX} \\ \mathbf{K}_{XA} & \mathbf{Q}_{XX} + \bar{\mathbf{V}} & \mathbf{Q}_{XX_*} & \mathbf{Q}_{XX} + \bar{\mathbf{V}} \\ \mathbf{K}_{X_*A} & \mathbf{Q}_{X_*X} & \mathbf{Q}_{X_*X_*} + \bar{\mathbf{V}}_* & \mathbf{Q}_{XX_*} \\ \mathbf{K}_{XA} & \mathbf{Q}_{XX} + \bar{\mathbf{V}} & \mathbf{Q}_{XX_*} & \mathbf{Q}_{XX} + \mathbf{V} \end{bmatrix} \right), \quad (4.15)$$

where we define  $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2\mathbb{I}$ . The analogue joint distribution in the full GP case is formulated in (3.6). Note that from (4.15), exact inference with a sparse GP corresponds again to conditioning and marginalization as summarized in Table 4.2. For instance, the predictive posterior can be derived as described in the next section.



### 4.2.2.2 Predictive Posterior

The predictive posterior distribution  $q(\mathbf{f}_*|\mathbf{y})$  of a sparse GP can be obtained by conditioning on  $\mathbf{y}$  and marginalization of  $\mathbf{f}$  and  $\mathbf{a}$  in (4.15), that is,

$$q(\mathbf{f}_*|\mathbf{y}) = \int \frac{q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y})}{q(\mathbf{y})} d\mathbf{f} d\mathbf{a}, \quad (4.16)$$

which can be obtained with (2.2) and (2.3) leading to the predictive posterior distribution of a sparse GP  $q(\mathbf{f}_*|\mathbf{y}) = \mathcal{N}(\mathbf{f}_*|\mu_f(\mathbf{X}_*), \Sigma_f^2(\mathbf{X}_*))$  with moments

$$\begin{aligned} \mu_f(\mathbf{X}_*) &= \mathbf{Q}_{X_*X} \mathbf{P}^{-1} \mathbf{y}, \\ \Sigma_f(\mathbf{X}_*) &= \mathbf{Q}_{X_*X_*} + \bar{\mathbf{V}}_* - \mathbf{Q}_{X_*X} \mathbf{P}^{-1} \mathbf{Q}_{XX_*}, \end{aligned} \quad (4.17)$$

where we define the covariance matrix

$$\mathbf{P} = \mathbf{Q}_{XX} + \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I} = \mathbf{Q}_{XX} + \mathbf{V} \in \mathbb{R}^{N \times N}. \quad (4.18)$$

Note that (4.17) and (4.18) have a very similar structure as in the full GP case (3.10) and (3.8), respectively. The kernel matrices  $\mathbf{K}_{XX}$ ,  $\mathbf{K}_{X_*X}$  and  $\mathbf{K}_{X_*X_*}$  are replaced by  $\mathbf{Q}_{XX}$ ,  $\mathbf{Q}_{X_*X}$  and  $\mathbf{Q}_{X_*X_*}$  and corrected by the additional training  $\bar{\mathbf{V}}$  and test covariance  $\bar{\mathbf{V}}_*$ . Computing the inverse  $\mathbf{P}^{-1}$  in (4.17) for the matrix  $\mathbf{P}$  in (4.18) requires still  $\mathcal{O}(N^3)$ . However, we can follow a two-stage procedure similarly discussed in Section 4.2.2.4 for full GP which basically means applying the inversion lemma (2.16) and is discussed in Section 4.2.2.4.

The noisy prediction of a sparse GP  $q(y(\mathbf{X}_*)|\mathbf{y}) = \mathcal{N}(y(\mathbf{X}_*)|\mu(\mathbf{X}_*), \Sigma(\mathbf{X}_*))$  can be similarly obtained as described in Section 3.2.2.2, accordingly,  $\mu(\mathbf{X}_*) = \mu_f(\mathbf{X}_*)$  and  $\Sigma(\mathbf{X}_*) = \Sigma_f(\mathbf{X}_*) + \sigma_n^2 \mathbb{I}$  using the sparse predictive moments defined in (4.17).

### 4.2.2.3 Posterior of Inducing Points

The posterior distribution  $q(\mathbf{a}|\mathbf{y})$  of the inducing points can be obtained by conditioning on  $\mathbf{y}$  and marginalization of  $\mathbf{f}$  and  $\mathbf{f}_*$  in (4.15), that is,

$$q(\mathbf{a}|\mathbf{y}) = \int \frac{q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y})}{q(\mathbf{y})} d\mathbf{f} d\mathbf{f}_*, \quad (4.19)$$

which can be computed with (2.2) and (2.3) leading to

$$q(\mathbf{a}|\mathbf{y}) = \mathcal{N}(\mathbf{a}|\mathbf{K}_{AX} \mathbf{P}^{-1} \mathbf{y}, \mathbf{K}_{AA} - \mathbf{K}_{AX} \mathbf{P}^{-1} \mathbf{K}_{XA}) \quad (4.20)$$

$$= \mathcal{N}(\mathbf{a}|\boldsymbol{\Sigma} \mathbf{H}^T \mathbf{V}^{-1} \mathbf{y}, \boldsymbol{\Sigma}), \quad (4.21)$$

where the posterior covariance is defined as  $\Sigma = (\mathbf{K}_{AA}^{-1} + \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H})^{-1}$  and  $\mathbf{H} = \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1}$  defined in (4.11). The second version in (4.21) can be obtained by applying the inversion lemma (2.16) to  $\mathbf{P}$  in the first version (4.20). From a computation point of view, the second version in (4.21) is preferable since it only requires the inversion for  $\Sigma \in \mathbb{R}^{M \times M}$  which needs  $\mathcal{O}(M^3)$  instead of  $\mathbf{P} \in \mathbb{R}^{N \times N}$  which is in  $\mathcal{O}(N^3)$ . Consider also the Remark 2.4 for linear basis function models.

#### 4.2.2.4 Prediction via Posterior

Similarly as discussed in Section 3.2.2.5, predictions can be also obtained in a two-stage procedure by first computing the posterior  $p(\mathbf{a}|\mathbf{y})$  via (4.21) and then combining it with the test conditional  $q(\mathbf{f}_*|\mathbf{a})$  in (4.11), that is,

$$p(\mathbf{f}_*|\mathbf{y}) = \int p(\mathbf{f}_*|\mathbf{a})p(\mathbf{a}|\mathbf{y}) d\mathbf{a}, \quad (4.22)$$

which can be computed via (2.2) leading to

$$\begin{aligned} \mu_f(\mathbf{X}_*) &= \mathbf{H}_* \Sigma \mathbf{H}^T \mathbf{V}^{-1} \mathbf{y}, \\ \Sigma_f(\mathbf{X}_*) &= \bar{\mathbf{V}}_* + \mathbf{H}_* \Sigma \mathbf{H}_*^T, \end{aligned} \quad (4.23)$$

which correspond exactly to the predictive posterior in (4.17) but depending on  $\Sigma$  instead  $\mathbf{P}$  which is computationally less expensive. Further, precomputing the posterior over the inducing points allows to make fast predictions for new prediction input points.

#### 4.2.2.5 Marginal Likelihood

We also formulate the marginal likelihood for sparse GPs, which will be thoroughly discussed in Section 4.2.4. Marginalization of all variables except  $\mathbf{y}$  in (4.15) yields the marginal likelihood distribution

$$q(\mathbf{y}) = \int q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}) d\mathbf{a} d\mathbf{f} d\mathbf{f}_* = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{P}) \quad (4.24)$$

with covariance  $\mathbf{P} = \mathbf{Q}_{XX} + \mathbf{V}$  already defined in (4.18).

#### 4.2.2.6 Induced Prior Distribution

The induced prior distribution  $q(\mathbf{f})$  of the training latent function values  $\mathbf{f}$  can be obtained from (4.15) by marginalization of all other variables, leading to

$$q(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{Q}_{XX} + \bar{\mathbf{V}}).$$

Comparing it with the intractable exact prior distribution  $p(\mathbf{f}) = \mathcal{N}(\mathbf{f} | \mathbf{0}, \mathbf{K}_{XX})$  of a full GP shows

$$\mathbf{K}_{XX} \approx \mathbf{Q}_{XX} + \bar{\mathbf{V}} = \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX} + \bar{\mathbf{V}}, \quad (4.25)$$

which can be understood as a **low-rank** approximation plus a **diagonal** matrix, so that the inverse can be efficiently computed. However, note that we never explicitly compute the full induced covariance matrix in (4.25).

#### 4.2.2.7 Induced Prior Kernel

In this section, we briefly discuss the induced prior kernel by the sparse GP model with approximate priors. First, we assume that the covariances  $\bar{\mathbf{V}}$  and  $\bar{\mathbf{V}}_*$  are set to zero, that is, both conditionals in (4.11) are deterministic. In this case, the approximated joint prior  $q(\mathbf{f}, \mathbf{f}_*)$  can be derived from (4.14) leading to

$$q(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\left[\begin{array}{c} \mathbf{f} \\ \mathbf{f}_* \end{array}\right] \middle| \left[\begin{array}{c} \mathbf{0} \\ \mathbf{0} \end{array}\right], \left[\begin{array}{cc} \mathbf{Q}_{XX} & \mathbf{Q}_{XX_*} \\ \mathbf{Q}_{X_*X} & \mathbf{Q}_{X_*X_*} \end{array}\right]\right), \quad (4.26)$$

which corresponds exactly to a GP prior with degenerated kernel  $\tilde{k}_A(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A})k(\mathbf{A}, \mathbf{A})^{-1}k(\mathbf{A}, \mathbf{x}')$  as defined in (3.18). This means, exact inference for full GP with the degenerated kernel  $\tilde{k}_A(\mathbf{x}, \mathbf{x}')$  is equal to a sparse GP model when the conditional covariances vanish. This allows also to compare sparse GPs with Bayesian linear basis function models.

**Remark 4.1 (Fully Deterministic Sparse GP vs Bayesian Linear Model)** *A fully deterministic sparse GP with covariances  $\bar{\mathbf{V}} = \bar{\mathbf{V}}_* = \mathbf{0}$  is equivalent to a Bayesian linear model of dimension  $M$  with basis function matrix  $\Phi_X = \mathbf{H} = \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1}$  and prior covariance matrix  $\Sigma_0 = \mathbf{K}_{AA}$ .*

**Example 4.1 (Fully Independent Conditionals (FIC))** *Assume that for any input  $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$ , the corresponding latent function values  $f(\mathbf{x})$  and  $f(\mathbf{x}')$  are conditionally independent given the inducing points  $f(\mathbf{A}) = \mathbf{a}$ , that is,*

$$f(\mathbf{x}) \perp f(\mathbf{x}') | f(\mathbf{A}).$$

*For a single input  $\mathbf{x} \in \mathbb{R}^D$ , the conditional distribution implied by the GP is thus  $p(f(\mathbf{x}) | \mathbf{a}) = \mathcal{N}(f(\mathbf{x}) | \mathbf{K}_{xA} \mathbf{K}_{AA}^{-1} \mathbf{a}, \mathbf{K}_{xx} - \mathbf{K}_{xA} \mathbf{K}_{AA}^{-1} \mathbf{K}_{Ax})$  leading to*

$$p(\mathbf{f} | \mathbf{a}) = \prod_{i=1}^N p(f(\mathbf{x}_i) | \mathbf{a}) = \mathcal{N}(\mathbf{f} | \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{a}, \bar{\mathbf{V}}),$$

$$p(\mathbf{f}_* | \mathbf{a}) = \prod_{i=1}^{N_{test}} p(f(\mathbf{X}_*) | \mathbf{a}) = \mathcal{N}(\mathbf{f}_* | \mathbf{K}_{X_*A} \mathbf{K}_{AA}^{-1} \mathbf{a}, \bar{\mathbf{V}}_*),$$

with approximated diagonal covariance matrices  $\bar{\mathbf{V}} = \text{Diag}[\mathbf{K}_{XX} - \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX}]$  and  $\bar{\mathbf{V}}_* = \text{Diag}[\mathbf{K}_{X_*X_*} - \mathbf{K}_{X_*A}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX_*}]$  which can be inverted efficiently. This model was proposed by Quiñonero-Candela and Rasmussen [2005].

The choice of training and test covariances as defined in Example 4.1 with fully independent conditionals  $\bar{\mathbf{V}} = \text{Diag}[\mathbf{K}_{XX} - \mathbf{Q}_{XX}]$  and  $\bar{\mathbf{V}}_* = \text{Diag}[\mathbf{K}_{X_*X_*} - \mathbf{Q}_{X_*X_*}]$ , leads to a joint approximate prior

$$q(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{Q}_{XX} + \bar{\mathbf{V}} & \mathbf{Q}_{XX_*} \\ \mathbf{Q}_{X_*X} & \mathbf{Q}_{X_*X_*} + \bar{\mathbf{V}}_* \end{bmatrix}\right). \quad (4.27)$$

Bayesian inference based on this joint prior can be equivalently seen as inference with a GP prior with augmented kernel  $\bar{k}_A(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A}) k(\mathbf{A}, \mathbf{A})^{-1} k(\mathbf{A}, \mathbf{x}') + \delta_{\mathbf{x}=\mathbf{x}'} \nu(\mathbf{x})$  as defined in (3.19). Note that the most other approaches for sparse global GPs summarized in Table 4.1 *cannot* be seen as a GP according to Definition 3.1 since the training  $\mathbf{f}$  and test  $\mathbf{f}_*$  latent function values are treated differently and thus the joint covariance cannot be described by a kernel function. The indexed GP, degenerated GP, the augmented GP and the block augmented version of the latter, are exceptions, for which the corresponding kernels are summarized in Table 4.3. However, this does not necessarily mean, that those are better GP approximations, as we will see in Section 4.2.3.

In order to understand the effect of the training and test conditional covariances  $\bar{\mathbf{V}}$  and  $\bar{\mathbf{V}}_*$ , we show in Figure 4.3 different combinations of no corrected vs diagonally corrected covariances for the training and testing, respectively. Thereby, the fixed inducing points (black dots) are chosen as a subset of training data points (red dots). In the first column, the corresponding models are depicted, whereas in the second and third columns, the predictive means and the predictive variances, respectively, are shown. In the last two columns, the prior kernels for training and testing are indicated and show whether the variance on the diagonal is corrected or not. We can observe, that using the correction for the test conditional is very important, otherwise the model underestimate the uncertainty outside of training regions significantly. For the training covariance it is less obvious. We can observe that without correction, the mean tries to fit all training data points, with the consequence that all estimates are bad. On the other hand, the diagonally corrected version tries to fit some points very good and ignores point far away due to the additional variance. We will provide another perspective for these observations in Section 4.2.3.

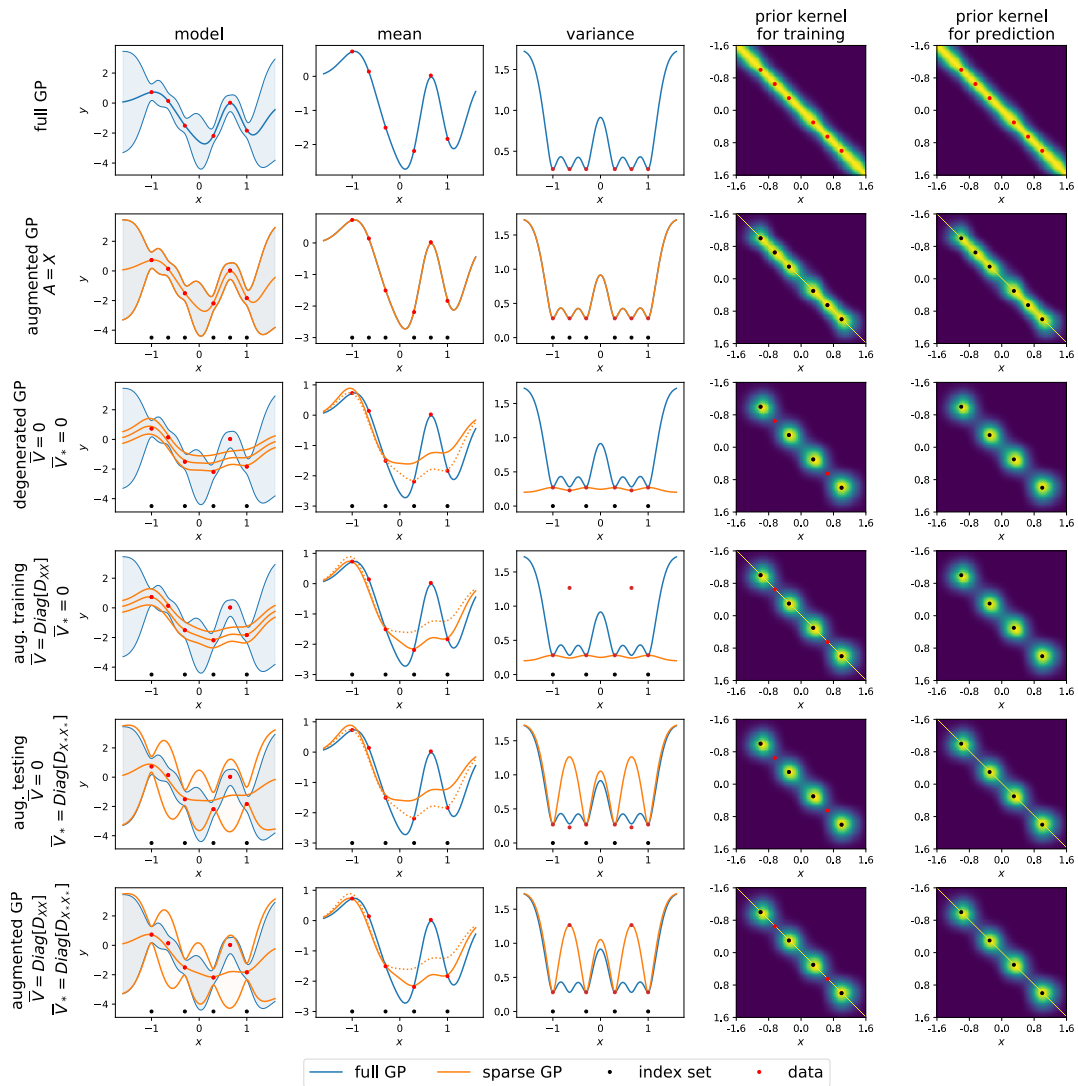


Figure 4.3. Four different versions of sparse GPs which basically summarize the training of all approaches based on inducing points (level I). Note that we indicated the opposite predictive mean as an orange dashed line in the second column in the last four rows for the sake comparison.

model	kernel function	description
full GP	$k(\mathbf{x}, \mathbf{x}')$	exact
degenerated GP	$\tilde{k}_A(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{A})K_{AA}^{-1}k(\mathbf{A}, \mathbf{x}')$	linear basis function model
augmented GP	$\tilde{k}_A(\mathbf{x}, \mathbf{x}') = \tilde{k}_A(\mathbf{x}, \mathbf{x}') + \delta_{\mathbf{x}=\mathbf{x}'}[k(\mathbf{x}, \mathbf{x}') - \tilde{k}_A(\mathbf{x}, \mathbf{x}')] ]$	linear basis function model with corrected variance on the diagonal
block augmented GP	$\tilde{k}_A(\mathbf{x}, \mathbf{x}') = \tilde{k}_A(\mathbf{x}, \mathbf{x}') + \xi_{\mathbf{x}, \mathbf{x}'}[k(\mathbf{x}, \mathbf{x}') - \tilde{k}_A(\mathbf{x}, \mathbf{x}')] ]$	linear basis function model with corrected block-covariance
indexed GP	$k_A(\mathbf{x}, \mathbf{x}') = \tilde{k}_A(\mathbf{x}, \mathbf{x}') + \varrho_{\mathbf{x}, \mathbf{x}'}[k(\mathbf{x}, \mathbf{x}') - \tilde{k}_A(\mathbf{x}, \mathbf{x}')] ]$	linear basis function model with fully corrected training and test covariance

Table 4.3. Different approximate kernel functions corresponding to approximated models of full GP. Although they are approximated models, all of them can be seen as a proper GP with a special structure of the kernel. We used  $\delta_{\mathbf{x}=\mathbf{x}'}$  as the Kronecker delta which is 1 if  $\mathbf{x} = \mathbf{x}'$  and 0 otherwise. Moreover,  $\xi_{\mathbf{x}, \mathbf{x}'}$  is defined to be 1 if  $\mathbf{x}$  and  $\mathbf{x}'$  are in the same block/region and 0 otherwise. Further,  $\varrho_{\mathbf{x}, \mathbf{x}'}$  is 1 if  $\mathbf{x}$  and  $\mathbf{x}'$  are both in the training inputs  $\mathbf{X}$  or both in the test inputs  $\mathbf{X}_*$  and 0 otherwise.

### 4.2.3 Approximate Inference with Exact Prior

In this section, we discuss approaches where the exact prior  $p(\mathbf{a}, \mathbf{f})$  is retained, however, approximate inference techniques are employed to directly find an approximate *posterior* distribution as outlined in Section 4.2.1.2. We start from the decomposition of the joint distribution in (4.7), that is,

$$q(\mathbf{f}, \mathbf{f}_*, \mathbf{a}) = p(\mathbf{f}_* | \mathbf{a})p(\mathbf{f} | \mathbf{a})p(\mathbf{a}) = p(\mathbf{f}_* | \mathbf{a})p(\mathbf{f}, \mathbf{a}),$$

where we noted that the prior  $p(\mathbf{f}, \mathbf{a})$  is still exact but is combined with the conditional  $p(\mathbf{f}_* | \mathbf{a})$  so that  $\mathbf{f}_*$  and  $\mathbf{f}$  only interact via  $\mathbf{a}$ . Instead directly imposing assumptions on the *prior*  $p(\mathbf{f}, \mathbf{a})$  as discussed in Section 4.2.2, the approach in this section is to find an approximate distribution  $q(\mathbf{f}, \mathbf{a})$  which is close to the exact GP *posterior* distribution

$$q(\mathbf{f}, \mathbf{a}) \approx p(\mathbf{f}, \mathbf{a} | \mathbf{y}), \quad (4.28)$$

where the exact posterior can be similarly derived as in (4.5), in particular without integrating out  $\mathbf{a}$ , that is

$$p(\mathbf{f}, \mathbf{a} | \mathbf{y}) = \int \frac{p(\mathbf{f}, \mathbf{f}_*, \mathbf{a}, \mathbf{y})}{p(\mathbf{y})} d\mathbf{f}_* = \frac{p(\mathbf{y} | \mathbf{f})p(\mathbf{f}, \mathbf{a})}{p(\mathbf{y})}. \quad (4.29)$$

The approaches in this section differ by two further choices. The first concerns the additional structural assumptions of the approximate distribution  $q$ . Secondly, the closeness in (4.28) between the approximate distribution  $q$  and the

true distribution  $p$  has to be specified. For the second point, in this section, we use the minimization of the Kullback-Leibler (KL)-divergence as discussed in Section 2.1.2.2. In particular, an approach is discussed in Section 4.2.3.1, where the *forward* Kullback-Leibler-divergence

$$\arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL}[q(\mathbf{f}, \mathbf{a}) || p(\mathbf{f}, \mathbf{a} | \mathbf{y})] \quad (4.30)$$

is minimized, whereas Section 4.2.3.2 describes an approach where the goal is to minimize the *reverse* KL

$$\arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL}[p(\mathbf{f}, \mathbf{a} | \mathbf{y}) || q(\mathbf{f}, \mathbf{a})]. \quad (4.31)$$

The former approach correspond to *variational inference* (VI) and the latter to *expectation propagation* (EP). In Section 4.2.3.3, an approach is discussed which unifies these two approaches, called *power expectation propagation* (PEP).

#### 4.2.3.1 Variational Inference

In this section, the variational approach by Titsias [2009] for global sparse GPs is discussed. The goal is to minimize the forward KL (4.30) between the approximate distribution  $q(\mathbf{f}, \mathbf{a})$  and the exact posterior distribution  $p(\mathbf{f}, \mathbf{a} | \mathbf{y})$ , that is

$$\arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL}[q(\mathbf{f}, \mathbf{a}) || p(\mathbf{f}, \mathbf{a} | \mathbf{y})]. \quad (4.32)$$

Further, the author Titsias [2009] proposed the structure of the approximate distribution

$$q(\mathbf{f}, \mathbf{a}) = p(\mathbf{f} | \mathbf{a})q(\mathbf{a}), \quad (4.33)$$

where the true conditional  $p(\mathbf{f} | \mathbf{a})$  is used so that  $q(\mathbf{a}) \approx p(\mathbf{a} | \mathbf{y})$ .

Instead directly minimize (4.32), in variational inference, the variational lower bound can be maximized.

**Proposition 4.1 (Variational Lower Bound Maximization)** *For a general joint distribution  $p(\mathbf{z}, \mathbf{y})$  with observed variable  $\mathbf{y}$  and unobserved variable  $\mathbf{z}$ , the goal is to find an approximate posterior distribution  $q(\mathbf{z})$  for the intractable true posterior distribution  $p(\mathbf{z} | \mathbf{y})$  as well as an estimate of the intractable log marginal likelihood  $\log p(\mathbf{y})$ . The variational solution is given by the following optimization,*

$$q^*(\mathbf{z}) = \arg \min_{q(\mathbf{z})} \text{KL}[q(\mathbf{z}) || p(\mathbf{z} | \mathbf{y})] = \arg \max_{q(\mathbf{z})} \mathcal{L}(q(\mathbf{z})), \quad (4.34)$$

which means that minimizing the forward KL is equivalent to maximizing the lower bound

$$\mathcal{L}(q(\mathbf{z})) = \int q(\mathbf{z}) \log \frac{p(\mathbf{z}, \mathbf{y})}{q(\mathbf{z})} d\mathbf{z} \leq \log p(\mathbf{y}) \quad (4.35)$$

to the log of the marginal likelihood  $\log p(\mathbf{y})$ .

We directly provide the proof idea for Proposition 4.1. The equivalence in (4.34) directly follows by manipulating the definition of the KL in (2.12), from which the log of the marginal likelihood can be decomposed as

$$\log p(\mathbf{y}) = \mathcal{L}(q(\mathbf{z})) - \text{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{y})], \quad (4.36)$$

and since the KL is always non-negative, it holds  $\log p(\mathbf{y}) \geq \mathcal{L}(q(\mathbf{z}))$ . Since  $\log p(\mathbf{y})$  in Equation (4.36) is independent of  $q(\mathbf{z})$ , it follows that minimizing the KL is equivalent to maximizing the lower bound, which concludes the statement.

Using in Proposition 4.1 for the unobserved variable  $\mathbf{z} = [\mathbf{f}, \mathbf{a}]^T$ , instead of minimizing (4.32), we can therefore equivalently maximize the lower bound

$$\mathcal{L}(q(\mathbf{f}, \mathbf{a})) = \int q(\mathbf{f}, \mathbf{a}) \log \frac{p(\mathbf{f}, \mathbf{a}, \mathbf{y})}{q(\mathbf{f}, \mathbf{a})} d\mathbf{f} d\mathbf{a}. \quad (4.37)$$

Plugging the structural assumption (4.33) into (4.37) yields

$$\mathcal{L}(q(\mathbf{f}, \mathbf{a})) = \int p(\mathbf{f}|\mathbf{a})q(\mathbf{a}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{a})p(\mathbf{a})}{p(\mathbf{f}|\mathbf{a})q(\mathbf{a})} d\mathbf{f} d\mathbf{a}$$

which simplifies to

$$\mathcal{L}(q(\mathbf{a})) = \int p(\mathbf{f}|\mathbf{a})q(\mathbf{a}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{a})}{q(\mathbf{a})} d\mathbf{f} d\mathbf{a}. \quad (4.38)$$

### Posterior Distribution

The approximate posterior distribution  $q(\mathbf{a}) \approx p(\mathbf{a}|\mathbf{y})$  can be obtained by maximizing (4.38), that is,

$$q^*(\mathbf{a}) = \arg \max_{q(\mathbf{a})} \mathcal{L}(q(\mathbf{a})),$$



where we refer to the supplementary material of Titsias [2009] for the complete derivation. The optimal distribution can be analytically obtained

$$q^*(\mathbf{a}) = \mathcal{N}\left(\mathbf{a} \mid \frac{1}{\sigma_n^2} \Sigma \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX} \mathbf{y}, \Sigma\right) \quad (4.39)$$

with  $\Sigma = \left(\mathbf{K}_{AA}^{-1} + \frac{1}{\sigma_n^2} \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX} \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1}\right)^{-1}$ . By comparing the optimal distribution in (4.39) to the posterior distribution of the inducing points in (4.21) indirectly obtained by a prior approximation and performing exact inference, we can notice that  $\mathbf{H} = \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1}$  and  $\mathbf{V} = \frac{1}{\sigma_n^2} \mathbb{I}$  and thus the training conditional covariance in (4.11) vanish  $\bar{\mathbf{V}} = \mathbf{0}$ . This means, regarding training or level I inference, the optimal posterior distribution of the inducing points obtained by variational inference is equivalent to the *deterministic "GP"* discussed in Section 4.2.2, compare also Table 4.1.

### Collapsed Lower Bound

Plugging the optimal distribution (4.39) into (4.38) yields the *collapsed variational lower bound* or the *variational free energy (VFE)*

$$\mathcal{L}_{VFE}(\boldsymbol{\theta}) = \mathcal{L}(q^*(\mathbf{a})) = \log \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{Q}_{XX} + \sigma_n^2) - \frac{1}{\sigma_n^2} \text{tr}[\mathbf{K}_{XX} - \mathbf{Q}_{XX}], \quad (4.40)$$

which correspond to the log marginal likelihood of a *deterministic "GP"* as defined in (4.24) minus an additional term which is the trace of the true training conditional covariance (4.9). This additional term acts as a regularizer when maximizing (4.40) with respect to the inducing inputs as discussed in Section 4.2.4

#### 4.2.3.2 Expectation Propagation

In this section, the *expectation propagation* for sparse global GPs is discussed, where we solely follow Csató and Opper [2002] and Bui et al. [2017b], but provide a different derivation. The goal is to minimize the forward KL (4.31) between the approximate distribution  $q(\mathbf{f}, \mathbf{a})$  and the exact posterior distribution  $p(\mathbf{f}, \mathbf{a} \mid \mathbf{y})$ , that is

$$\arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL}[p(\mathbf{f}, \mathbf{a} \mid \mathbf{y}) \parallel q(\mathbf{f}, \mathbf{a})]. \quad (4.41)$$

Since the Gaussian likelihood  $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2\mathbb{I})$  in (3.5) decomposes as a product

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^N p(y_i|\mathbf{f}_i) = \prod_{i=1}^N \mathcal{N}(y_i|\mathbf{f}_i, \sigma_n^2), \quad (4.42)$$

the exact posterior distribution in (4.29) can be written as

$$p(\mathbf{f}, \mathbf{a}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}, \mathbf{a})}{p(\mathbf{y})} = \frac{\prod_{i=1}^N p(y_i|\mathbf{f}_i)p(\mathbf{f}, \mathbf{a})}{p(\mathbf{y})}. \quad (4.43)$$

The product in (4.43) can be seen as a decomposition of  $N + 1$  factors in the variables  $\mathbf{f}$  and  $\mathbf{a}$ , that is,

$$p(\mathbf{f}, \mathbf{a}|\mathbf{y}) = \frac{1}{p(\mathbf{y})} \prod_{i=0}^N p_i(\mathbf{f}, \mathbf{a}), \quad (4.44)$$

with exact individual factors  $p_0(\mathbf{f}, \mathbf{a}) = p(\mathbf{f}, \mathbf{a})$ ,  $p_i(\mathbf{f}, \mathbf{a}) = p(y_i|\mathbf{f}_i)$  for  $i > 0$  and normalization constant  $p(\mathbf{y}) = \int \prod_{i=0}^N p_i(\mathbf{f}, \mathbf{a}) d\mathbf{f} d\mathbf{a}$ . By observing the structure in (4.44) of the true posterior distribution, we assume the same structure for the approximate distribution  $p(\mathbf{f}, \mathbf{a}|\mathbf{y}) \approx q(\mathbf{f}, \mathbf{a})$ , that is,

$$q(\mathbf{f}, \mathbf{a}) = \frac{1}{Z} \prod_{i=0}^N q_i(\mathbf{f}, \mathbf{a}) \quad (4.45)$$

with  $Z = \int \prod_{i=0}^N q_i(\mathbf{f}, \mathbf{a}) d\mathbf{f} d\mathbf{a} = q(\mathbf{y})$ . The minimization problem in (4.41) becomes then

$$\arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL} \left[ \frac{1}{p(\mathbf{y})} \prod_{i=0}^N p_i(\mathbf{f}, \mathbf{a}) \parallel \frac{1}{Z} \prod_{i=0}^N q_i(\mathbf{f}, \mathbf{a}) \right]. \quad (4.46)$$

Note that minimizing (4.46) is equivalent to minimizing (4.41) with the constraint (4.45). Ideally we would like to determine the approximate factors  $q_i(\mathbf{f}, \mathbf{a})$  by solving (4.46) exactly. However, the reverse KL minimization in general is intractable since the KL-divergence involves averaging with respect to the true distribution on the left in the KL. As a very rough approximation, we could instead minimize the KL divergences between the corresponding pairs  $p_i(\mathbf{f}, \mathbf{a})$  and  $q_i(\mathbf{f}, \mathbf{a})$  of factors. This leads to a much simpler minimization problem but the overall approximation is not satisfying.

### Iterative Moment Matching

*Expectation propagation* (EP) makes a much better approximation by optimizing iteratively each factor in the context of all of the remaining factors. It starts by initializing the approximate posterior

$$q^{(t)}(\mathbf{f}, \mathbf{a}) = \frac{1}{Z_t} \prod_{i=0}^N q_i^{(t)}(\mathbf{f}, \mathbf{a})$$

for some initial factors  $q_i^{(0)}(\mathbf{f}, \mathbf{a})$ . Afterwards it cycles through all factors so that  $t \% N = i$  ( $\%$  is the modulo operator), by removing the previous approximate factor  $q_i^{(t)}(\mathbf{f}, \mathbf{a})$  from the current posterior approximation  $q^{(t)}(\mathbf{f}, \mathbf{a})$  and updates the true factor  $p_i(\mathbf{f}, \mathbf{a})$  leading to the so-called cavity distribution

$$\bar{p}^{(t)}(\mathbf{f}, \mathbf{a}|\mathbf{y}) = \frac{1}{Z_t} \frac{q^{(t)}(\mathbf{f}, \mathbf{a}) p_i(\mathbf{f}, \mathbf{a})}{q_i^{(t)}(\mathbf{f}, \mathbf{a})}.$$

Hence, EP solves iteratively the following minimization task

$$q^{(t+1)} = \arg \min_{q(\mathbf{f}, \mathbf{a})} \text{KL} \left[ \bar{p}^{(t)}(\mathbf{f}, \mathbf{a}|\mathbf{y}) \parallel q(\mathbf{f}, \mathbf{a}) \right], \quad (4.47)$$

Note that this procedure can be applied also to non-Gaussian distributions as long as the multiplication, division and normalization in (4.47) can be computed efficiently. For members of the *exponential-family*, the minimization problem in (4.47) corresponds to matching the expected sufficient statistics of both distributions, see [Bishop, 2006, Section 10.7]. A common situation (e.g. for GP classification) is that the  $p_i(\mathbf{f}, \mathbf{a})$  correspond to a non-Gaussian likelihood and  $q(\mathbf{f}, \mathbf{a})$  is assumed to be Gaussian. In this case, the distribution on the left in (4.47) is projected by its mean and covariance to the mean and covariance of a Gaussian distribution as discussed in [Bishop, 2006, Section 10.7] and intuitively illustrated in in Figure 2.2.

### Posterior Distribution

The setting of GP regression with Gaussian likelihood is even simpler since all involved distributions are Gaussians, including  $p_i(\mathbf{f}, \mathbf{a})$ . In this case, convergence is guaranteed after two passes through the data, independent of the initialization and the moments of the approximate solution  $q(\mathbf{f}, \mathbf{a})$  can be even computed in closed form solutions. In particular, the optimal approximate factors in (4.45) are

$$q_0(\mathbf{f}, \mathbf{a}) = p(\mathbf{a}) \quad \text{and} \quad q_i(\mathbf{f}, \mathbf{a}) = p(\mathbf{y}_i | \mathbf{f}_i) p(\mathbf{f}_i | \mathbf{a}),$$

so that the joint approximate posterior distribution becomes according to (4.45)

$$q(\mathbf{f}, \mathbf{a}) = \frac{1}{Z} \prod_{i=0}^N q_i(\mathbf{f}, \mathbf{a}) = \frac{1}{Z} p(\mathbf{a}) \prod_{i=1}^N p(y_i | f_i) p(f_i | \mathbf{a}) = \frac{p(\mathbf{y} | \mathbf{f}) q(\mathbf{f} | \mathbf{a}) p(\mathbf{a})}{q(\mathbf{y})} \quad (4.48)$$

with approximated training conditional

$$\begin{aligned} q(\mathbf{f} | \mathbf{a}) &= \prod_{i=1}^N p(f_i | \mathbf{a}) = \prod_{i=1}^N \mathcal{N}(f_i | \mathbf{K}_{X_i A} \mathbf{K}_{AA}^{-1} \mathbf{a}, \mathbf{K}_{X_i X_i} - \mathbf{K}_{X_i A} \mathbf{K}_{AA}^{-1} \mathbf{K}_{A X_i}) \\ &= \mathcal{N}(\mathbf{f} | \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{a}, \text{Diag}[\mathbf{K}_{XX} - \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX}]). \end{aligned} \quad (4.49)$$

The approximate posterior distribution  $q(\mathbf{a}) \approx p(\mathbf{a} | \mathbf{y})$  over the inducing points can be obtained by using the exact prior  $p(\mathbf{a})$  and exact likelihood  $p(\mathbf{y} | \mathbf{f})$  together with the approximated training conditional in (4.49) and integrating out  $\mathbf{f}$  in (4.48), yielding

$$q(\mathbf{a}) = \int q(\mathbf{f}, \mathbf{a}) d\mathbf{f} = \mathcal{N}(\mathbf{a} | \Sigma \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX} \mathbf{V}^{-1} \mathbf{y}, \Sigma) \quad (4.50)$$

where the posterior covariance is  $\Sigma = (\mathbf{K}_{AA}^{-1} + \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX} \mathbf{V}^{-1} \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1})^{-1}$  with  $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$  and  $\bar{\mathbf{V}} = \text{Diag}[\mathbf{K}_{XX} - \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1} \mathbf{K}_{AX}]$ . By comparing the optimal distribution in (4.50) to the posterior distribution of the inducing points in (4.21), we can notice that it is equivalent when inserting  $\mathbf{H} = \mathbf{K}_{XA} \mathbf{K}_{AA}^{-1}$  and  $\bar{\mathbf{V}}$  corresponding to the diagonal of training conditional covariance in (4.11). This means that the optimal solution obtained by EP for the posterior distribution is equivalent to the indirect posterior approximation via prior approximation as described in the previous section with an *augmented GP*, compare Table 4.1.

### Marginal Likelihood

The normalization or approximated marginal likelihood  $Z = q(\mathbf{y})$  in the approximated posterior in (4.48) can be computed as

$$q(\mathbf{y}) = \int p(\mathbf{y} | \mathbf{f}) q(\mathbf{f} | \mathbf{a}) p(\mathbf{a}) d\mathbf{f} d\mathbf{a} = \mathcal{N}(\mathbf{y} | \mathbf{Q}_{XX} + \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}). \quad (4.51)$$

Note that this correspond to the marginal likelihood of a *augmented GP* as defined in (4.24). Compared to the lower bound of the variational approach in (4.40), there is no additional term. We denote the log of this marginal likelihood  $\mathcal{L}_{EP}(\boldsymbol{\theta}) = \log q(\mathbf{y})$ .

### 4.2.3.3 Power Expectation Propagation

The two approximate inference approaches for sparse GPs discussed in Sections 4.2.3.1 and 4.2.3.2 have been unified into one general approach by Bui et al. [2017b]. The main idea is to replace the KL-divergence in (4.41) by the more general alpha-divergence [Minka et al., 2005] parametrized by a scalar  $0 < \alpha \leq 1$ . For  $\alpha = 1$ , this divergence correspond to the reverse KL, for  $\alpha \rightarrow 0$ , it converge to the forward KL. When minimising this alpha-divergence instead the KL similarly to the derivations in Section 4.2.3.2 is also known as *power expectation propagation* (PEP) [Minka et al., 2005; Bui et al., 2017b]. It yields the approximated training conditional  $q(\mathbf{f}|\mathbf{a}) = \mathcal{N}(\mathbf{f} \mid \mathbf{H}\mathbf{a}, \bar{\mathbf{V}})$ , with projection matrix  $\mathbf{H} = \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}$  and training covariance

$$\bar{\mathbf{V}} = \alpha \text{Diag}[\mathbf{K}_{XX} - \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX}], \quad (4.52)$$

where we refer to [Bui et al., 2017b] for the full derivation. We note that for  $\alpha = 1$ , this correspond exactly to the training covariance in (4.49) and for  $\alpha \rightarrow 0$  it becomes deterministic as in (4.39). Using this general approximate training covariance in (4.52) together with the true test covariance  $\bar{\mathbf{V}}_* = \mathbf{K}_{X_*X_*} - \mathbf{Q}_{X_*X_*}$  for (4.11) and apply exact Bayesian inference, lead for instance to the predictive distribution

$$\begin{aligned} \mu_f(\mathbf{X}_*) &= \mathbf{Q}_{X_*X} \mathbf{P}^{-1} \mathbf{y}; \\ \Sigma_f(\mathbf{X}_*) &= \mathbf{Q}_{X_*X_*} + \bar{\mathbf{V}}_* - \mathbf{Q}_{X_*X} \mathbf{P}^{-1} \mathbf{Q}_{XX_*}, \end{aligned}$$

with covariance matrix  $\mathbf{P} = \mathbf{Q}_{XX} + \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$ .

**Remark 4.2** For sparse global GP models, approximate inference techniques retaining the exact prior can be equivalently seen as modifying directly the prior and performing exact Bayesian inference regarding level I inference.

However, approximate inference techniques provide different results for level II inference. In particular, they provide a lower bound to the marginal likelihood which will be important when choosing inducing inputs as it will be discussed in Section 4.2.4. In particular, the corresponding lower bound  $q(\mathbf{y}) \geq \mathcal{L}(\boldsymbol{\theta})$  to the marginal likelihood resulting from the PEP optimization can be written as

$$\mathcal{L}_{PEP}(\boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{y} \mid \mathbf{Q}_{XX} + \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}) - \frac{1-\alpha}{2\alpha} \sum_{i=1}^N \log \left( 1 + \frac{\alpha}{\sigma_n^2} \bar{V}_{ii} \right), \quad (4.53)$$

which can be seen as the log marginal likelihood of the corresponding basis function model *minus* a correction term which prevents overfitting as we will discuss

in the next section. Note that for  $\alpha = 1$ , the correction term vanishes and the same log marginal likelihood of EP (4.51) or an augmented GP is recovered. Similarly, for  $\alpha \rightarrow 0$ , the correction term converges to the one in (4.40) corresponding to variational inference.

#### 4.2.4 Choosing Inducing Inputs and Hyperparameters

For global sparse GP models with  $M$  inducing points in  $D$  dimension, there are the kernel hyperparameters, the noise variance and the input locations of the inducing points  $\mathbf{A} = [\mathbf{A}_1, \dots, \mathbf{A}_M]^T \in \mathbb{R}^{M \times D}$  to choose, so that  $MD$  additional parameters have to be estimated compared to full GP. Two efficient ways to choose the inducing inputs  $\mathbf{A}$  are either according to a space-filling design or to the subset of data approach. In the former, the input points are regularly placed in the input space, for instance on a regular grid. In the latter, a subset of  $M$  training data inputs out of  $\mathbf{X} \in \mathbb{R}^{N \times D}$  are randomly chosen for the inducing inputs  $\mathbf{A}$ . For small dimension, i.e.  $D = 1, 2$ , and moderate complex underlying functions, these two ideas constitute often efficient approaches with rather good performance, since the  $M$  chosen inducing points can represent well the input space, the computational time is minimal and there is no danger for overfitting. For instance, the sparse GP model with the random subset approach becomes monotonically more exact with respect to full GP, when increasing the number of inducing points. However, for both approaches, the main disadvantage is that in higher dimension  $D$  or for complex underlying functions (for instance fast varying patterns, e.g. time series), fixed chosen inducing points cannot cover well the input space for a fixed number  $M$  of inducing points. Moreover, increasing the number of inducing points is limited to the fact, that the time complexity is still cubic in the number of inducing points. As a consequence, optimizing the inducing points become cheaper than increasing the number of fixed placed inducing points. Therefore, in the context of sparse global GPs, the inducing inputs together with the kernel hyperparameters and the noise variance are summarized by  $\boldsymbol{\theta}$ , for which we discuss some inference approaches in the following.

Level II inference for  $\boldsymbol{\theta}$  in sparse GPs can also be done based on empirical Bayes from Section 2.3.3.1, similarly as for Bayesian linear models and full GPs as discussed in Sections 2.3.4.6 and 3.3, respectively. From the perspective of Section 4.2.2, where sparse GP models are discussed as exact inference with approximate prior, this is the consequent approach and has been proposed by the corresponding authors from SoR, DIC, FIC, PP, DTC, SSGP, FITC and PIC models as indicated in Table 4.1. In particular, the ML-II approach based on the marginal likelihood distribution  $q(\mathbf{y}|\boldsymbol{\theta}) = q(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{P}_\theta)$  in (4.24) with covariance

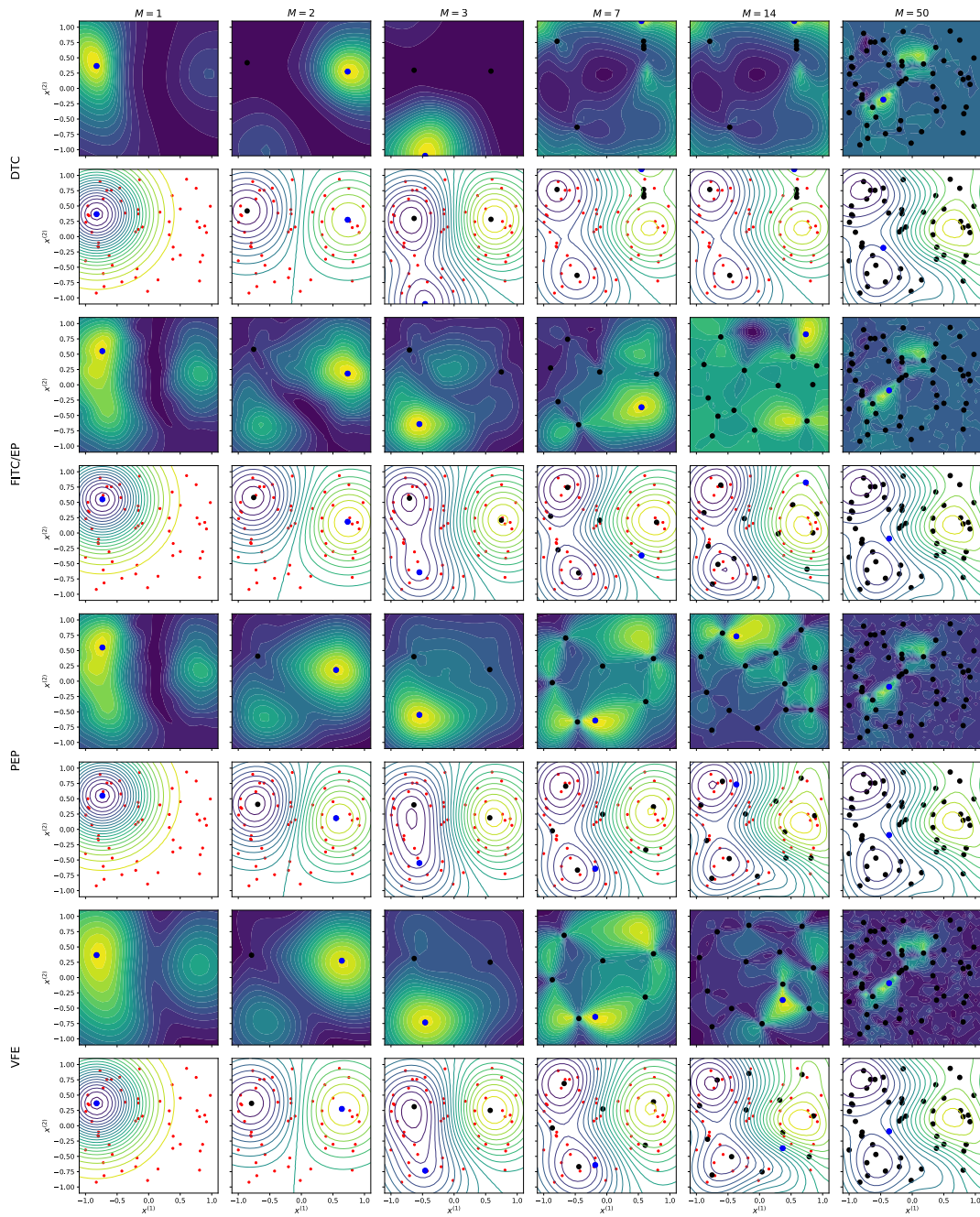


Figure 4.4. Inducing input optimization for different methods (row-wise) and different number  $M$  of inducing points (column-wise). In the rows with odd row numbers, the corresponding objective function is depicted, whereas in rows with even row numbers, the corresponding fit of the sparse GP model with the optimal inducing points is shown. In the last column, all inducing points are placed to the training inputs, so that all sparse GP methods correspond to full GP.

$\mathbf{P}_\theta = \mathbf{Q}_{XX} + \mathbf{V}$  can be applied, that is,

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} q(\mathbf{y}|\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_\theta^{-1} \mathbf{y} + \log |\mathbf{P}_\theta|. \quad (4.54)$$

For moderate number of inducing points  $M$  and dimension  $D$  compared to the number of samples  $N$ , this approach works reasonable, however, due to the many parameters in the model, there is the danger of overfitting. Moreover, what might be even worse, there is no guarantee that the sparse GP model with optimized inducing inputs is a good approximation of full GP.

The approaches discussed in Section 4.2.3 based on approximate inference with exact priors constitute an interesting alternative. Although the predictive posterior can be equivalently explained by exact inference with approximate prior as discussed in Section 4.2.2, they provide a different approach for level II inference for  $\boldsymbol{\theta}$ . In particular, the lower bounds  $\mathcal{L}_{VFE}(\boldsymbol{\theta})$  and  $\mathcal{L}_{PEP}(\boldsymbol{\theta})$  in (4.40) and (4.53), respectively, include a regularization term  $a(\boldsymbol{\theta})$ , so that the optimization task becomes

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \\ &= \arg \max_{\boldsymbol{\theta}} q(\mathbf{y}|\boldsymbol{\theta}) \exp(a(\boldsymbol{\theta})) \\ &= \arg \min_{\boldsymbol{\theta}} \mathbf{y}^T \mathbf{P}_\theta^{-1} \mathbf{y} + \log |\mathbf{P}_\theta| + a(\boldsymbol{\theta}). \end{aligned} \quad (4.55)$$

By optimizing the lower bound (4.55) including the additional regularization term  $a(\boldsymbol{\theta})$  can be seen as introducing an unnormalized prior  $\exp(a(\boldsymbol{\theta}))$ , which is combined with the marginal likelihood  $q(\mathbf{y}|\boldsymbol{\theta})$ . It has the effect, that the posterior distribution  $q(\mathbf{f}, \mathbf{a}|\mathbf{y}, \boldsymbol{\theta}^*)$  of the sparse GP model with the optimized parameters  $\boldsymbol{\theta}^*$  is guaranteed to be close to the posterior distribution  $p(\mathbf{f}, \mathbf{a}|\mathbf{y}, \bar{\boldsymbol{\theta}}^*)$  of full GP as enforced by minimizing a divergence between these two distributions (4.28). In particular, by optimizing the collapsed lower bound  $\mathcal{L}_{VFE}(\boldsymbol{\theta})$  in (4.40), the parameters  $\boldsymbol{\theta}$  including the inducing inputs are variationally safe and there is no danger to overfit with respect to full GP. Moreover, when increasing the number  $M$  of inducing points, the sparse GP posterior converges rigorously to full GP for  $M \rightarrow N$  as demonstrated by Titsias [2009].

These two approaches are illustrated in Figure 4.4, where we generated  $N = 50$  training data samples (red dots) with a squared-exponential kernel in  $D = 2$  dimension. In particular, we show for each method (row-wise) and different number  $M$  of inducing points (column-wise) the corresponding objective functions (in rows with odd row numbers) and the corresponding obtained fit of the sparse GP model with the optimal inducing points (in rows with even row



numbers). We fixed the kernel hyperparameters and the noise variance to the true values for illustration purposes and estimated the  $M$  inducing points with two versions of each approach in (4.54) and (4.55), respectively. In particular, for the pure ML-II approach (4.54), we use a training covariance  $\bar{\mathbf{V}} = \mathbf{0}$  and  $\bar{\mathbf{V}} = \text{Diag}[\mathbf{K}_{XX} - \mathbf{Q}_{XX}]$  corresponding to the DTC [Seeger et al., 2003] and FITC [Quiñonero-Candela and Rasmussen, 2005], respectively. The results are shown in the upper part of Figure 4.4. For the lower bound approaches (4.55), we consider the lower bounds  $\mathcal{L}_{VFE}$  and  $\mathcal{L}_{PEP}$  with  $\alpha = 0.5$  proposed by Titsias [2009] and Bui et al. [2017b], respectively, for which the results are shown in the bottom part of Figure 4.4. Since the input dimension of the objective functions (in rows with odd row numbers) is  $2M$ , we only show the projection to one inducing input (which is 2-dimensional), while the other inducing inputs are fixed to the optimized values. Note that in the last column, the inducing points (black dots) are fixed to the training inputs, so that all fits of the sparse GP models correspond exactly to full GP.

We first note, that the investigated approaches work reasonable well, except the DTC approach shown in the first two rows. Although, the DTC approach provides a decent estimate for the first three inducing inputs, it completely fails for more inducing points since it place them outside the region with training inputs and thus they are completely wasted, resulting in a poor approximation. The differences for the other 3 methods are rather subtle, which will be discussed in the following. The first 3 inducing inputs are placed by all three methods close to the 3 local optima of the function fitted by full GP as depicted in the last column. Thereby, the order which local optima is chosen first correspond to the absolute value of the corresponding local optima. We can notice a small difference in the places of the first inducing points. Whereas FITC tries to fit the function exactly close to the inducing points and cares less about the approximation quality far away from the inducing points. This can be explained by the additional training variance, which has the effect that data points far away from inducing points are treated as noise. On the other hand, the VFE approach makes a compromise, so that all data points are explained relatively well, but no of them exactly. This has the effect, that the method does not overfit with respect to full GP. However, it can be sometimes observed for real data that this approach is over-regularized, so that it has some difficulties to model the observed data well. Thus, in practice, often a good choice constitutes the mix of both approaches, the PEP model with  $\alpha \approx 0.5$ , which corrects a fraction of the training covariance and adds also a fraction of the regularization term. A similar observation is that the inducing points for FITC are preferable placed to regions with many data samples, whereas the VFE approach places the inducing points also in between the samples so that sev-

eral data points can explained relatively well together. This has the effect, that the inducing points for VFE cover better the region with training data and for  $M \rightarrow N$  it converges to full GP which means that the inducing points are placed at the training inputs.

#### 4.2.5 Advantages and Disadvantages of SGPs

Global sparse GPs based on inducing points have several benefits.

- **Time and Space Complexity:** The cubic time  $\mathcal{O}(N^3)$  and quadratic space  $\mathcal{O}(N^2)$  complexity of full GP can be reduced by sparse GP approximations to  $\mathcal{O}(NM^2)$  and  $\mathcal{O}(NM)$ , respectively. This allows to scale GPs to a quite large number of training data samples.
- **Global Patterns:** The probabilistic projection of the training points to the global inducing points constitute a principled way to model the same global pattern and dependencies as full GP.
- **Consistent Uncertainty Information:** With the additional test covariance, the predictive variance of sparse GPs falls back to the prior predictive distribution and can imitate the infinite-dimensional basis functions. This means that in regions without inducing points, the credible intervals are conservative and thus contain the credible intervals of full GP.
- **Low Rank Approximation:** Sparse GPs can be seen as a clever chosen low-rank approximation to circumvent the inversion of the large kernel data matrix so that the main patterns in the covariance can be retained.
- **Convergence to Full GP:** For inducing points chosen as subset of data or by maximizing the variational lower bound [Titsias, 2009], the model converges to full GP as the number  $M$  of inducing points tends to the number of training points  $N$ .

But sparse GP approximations have also some disadvantages.

- **Number of Inducing Points:** The inducing inputs have to cover the regions of the input space where training data is available, otherwise the information of the data is lost. For a fixed number of inducing points, this is particularly difficult for higher dimensional input spaces. Increasing the number of inducing points is limited to the fact, that the time complexity is still cubic in the number of inducing points.

- **Missing Local Patterns:** Due to the finite number of inducing points, the model is limited in modelling arbitrary complex patterns. In particular, fast varying and local patterns are smoothed out and treated as noise.
- **Parametric Approximation of Non-Parametric Method:** The number of additional parameters in the model of  $M$  inducing points in  $D$  dimension is  $\mathcal{O}(MD)$ . Although these parameter can be treated as variational parameters and are thus variationally safe regarding overfitting, sparse GPs lose the character of non-parameteric models since all those parameters have to be estimated.
- **Stochastic Training:** Global sparse GPs as discussed in this section cannot be trained stochastically with sequential mini-batches since the log marginal likelihood does not decompose into a sum of terms. However, for instance the work [Hensman et al., 2013] and the approaches proposed in Chapter 5 make stochastic training for global sparse GPs possible.

## 4.3 Local Approaches

Local approaches for GP approximations are based on the *divide & conquer* principle. In the context of local GPs, this means that the input space  $\mathcal{X}$  is divided into  $J$  regions and to each subregion, an independent full GP is applied. For the combination from the individual subregions to the whole space, different aggregation schemes based on probabilistic averaging techniques have been proposed. In probabilistic aggregation methods, each model from the individual subregions is often called an *expert*, described by some knowledge and uncertainty. The aggregation or averaging is then based on the uncertainty of each expert resulting in a fused knowledge and uncertainty. Since for GP approximations, each expert correspond to a local full GP trained on the training samples in the subregion,  $J$  different predictive distributions according to (3.10) with predictive mean and variance are obtained, which can be used for the aggregation. This basic approach reduces the time complexity drastically from  $\mathcal{O}(N^3) = \mathcal{O}(J^3 B^3)$  to  $\mathcal{O}(JB^3) = \mathcal{O}(NB^2)$ , when assuming that each subregion has equal number of data samples  $B = \frac{N}{J}$  [Liu et al., 2020].

### 4.3.1 Local Models with Single Prediction

In this section, we briefly discuss simple baseline models, where the prediction depends only on a single subregion, without any aggregation step.

#### 4.3.1.1 Naive Local Expert

The most straight forward approach for local GP approximation - here denoted as *naive local expert* (NLE) - is to divide the whole input space into  $J$  disjoint regions or partitions  $\Omega_j$ , i.e.  $\Omega_j \cap \Omega_i = \emptyset$  for all  $i \neq j$ , so that the regions cover the whole space  $\cup_{j=1}^J \Omega_j = \mathcal{X}$ . For given training data  $\mathcal{D} = \{y_i, \mathbf{x}_i\}_{i=1}^N = \{\mathbf{y}, \mathbf{X}\}$ , the training data of each expert is defined as

$$\mathcal{D}_j = \{y_i, \mathbf{x}_i : \mathbf{x}_i \in \Omega_j\} = \{\mathbf{y}_j, \mathbf{X}_j\}.$$

Each expert  $j$  fits locally an independent full GP on  $\mathcal{D}_j$  yielding the individual predictive distributions according to (3.9)

$$p(f_{*j} | \mathbf{y}_j) = \mathcal{N}(f_{*j} | m_j, v_j) \quad (4.56)$$

for the  $j$ th predictive latent function value  $f_{*j} = f(\mathbf{x}_*)_j$  with predictive mean  $m_j = \mu(\mathbf{x}_*)_j$  and variance  $v_j = \sigma^2(\mathbf{x}_*)_j$  in (3.10). The overall predictive distribution is then simply the predictive distribution

$$q(f_* | \mathbf{y}) = \mathcal{N}(f_* | \mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)) = p(f_{*k} | \mathbf{y}_k) = \mathcal{N}(f_{*k} | m_k, v_k), \quad (4.57)$$

of the  $k$ th expert for which  $\mathbf{x}_* \in \Omega_k$ . In practice, defining the partitions  $\Omega_j$  before considering the training data is not straight forward and leads often to experts with very different number of training samples.

#### 4.3.1.2 Nearest Neighbor Expert

A similar method to NLE constitutes the *nearest neighbor expert* (NNE) approach, where instead the training data  $\mathcal{D}$  is directly split into individual datasets  $\mathcal{D}_j = \{\mathbf{y}_j, \mathbf{X}_j\}$  for each expert according to a specified rule. As opposed to NLE, this approach is input data depending, so that for instance clustering techniques (e.g. kmeans, trees) for finding the partition of the training data can be used. The training of each experts yields the individual predictive posterior as in (4.56), however, the overall prediction is based on the training inputs. In particular, for the overall prediction (4.57), the predictive distribution of expert  $k$  with closest mean to the query point  $\mathbf{x}_*$  is used, that is,

$$k = \arg \min_j \|\mathbf{x}_* - M_j\|,$$

where  $M_j = \frac{1}{B} \sum_{i=1}^B \mathbf{X}_j^i$  is the mean of inputs of the  $j$ th expert. Note that we consider here only *inductive* approaches, where the partitions are static and independent of the test points so that the training of all experts can be performed before testing. As opposed to *transductive* approaches [Liu et al., 2020, Section IV], where the neighborhood can depend on other test points.

### 4.3.1.3 Minimal Variance Expert

The variant in this section relies again on a partition of the training data  $\mathcal{D}$  as described for NNE, yielding individual datasets for each expert  $\mathcal{D}_j$ . However, instead of using the expert which is closest to the query point  $\mathbf{x}_*$  as in NNE, the *minimal variance* (minVar) approach uses the expert with minimal variance of the predictive distribution  $p(f_{*j}|\mathbf{y}_j) = \mathcal{N}(f_{*j}|m_j, v_j)$  among the experts  $j = 1, \dots, J$ , that is,

$$k = \arg \min_j v_j.$$

Note that as opposed to NLE or NNE, which only use the input data for determining the optimal expert, the minVar approach uses also the uncertainty of the local GP model. As a consequence, the minVar approach outperforms in the most situations the NLE and NNE due to the use of the predictive variance, which we observed in many experiments and is also confirmed by Rullière et al. [2018]. Therefore, we focus in this work on minVar as comparison for single local prediction experts.

### 4.3.1.4 Limitations of Single Predictions

Models based on the predictions of a single expert provide often surprisingly good performance in practice, in particular, for large enough partitions or relatively small number of experts. However, they suffer from severe discontinuity issues at the border of the partitions or between the experts. This can be observed for instance in the first row of Figure 4.5, where the minVar approach is depicted for  $J = 2, 10, 50$  experts. We notice that the approximation quality compared to full GP is rather good for a small number of experts, since it corresponds locally to the exact full GP model. However, for more experts, the predictive distribution becomes very non-smooth, which is not a desirable behavior since a small change in the input leads to a big change in the output. One approach to overcome these issues are to smooth the predictive distributions of the individual experts, as it will be discussed in the next section.

## 4.3.2 Product of Experts

In order to improve the local independent experts based on single predictions, the *product of expert* (PoE) approach [Hinton, 2002] constitutes a powerful and principled way to combine individual experts, by probabilistically averaging techniques. In particular, it smooths the individual predictive distributions based on

method	link function $g_j(p_j)$	predictive mean $\mu_*$	predictive variance $\sigma_*^2$	remarks
naive local expert	$\begin{cases} p_k & \text{if } j = k \\ 1 & \text{else} \end{cases}$	$\sigma_*^2 \frac{m_k}{v_k}$	$\frac{1}{v_k}$	$k = \{j : \mathbf{x}_* \in \Omega_j\}$ $\Omega_j \cap \Omega_i = \emptyset, \cup_j \Omega_j = \mathcal{X}$
nearest neighbor expert	$\begin{cases} p_k & \text{if } j = k \\ 1 & \text{else} \end{cases}$	$\sigma_*^2 \frac{m_k}{v_k}$	$\frac{1}{v_k}$	$k = \arg \min_j \ \mathbf{x}_* - M_j\ _2$ $M_j = \frac{1}{B} \sum_{i=1}^B \mathbf{X}_j^i$
minimal variance expert	$\begin{cases} p_k & \text{if } j = k \\ 1 & \text{else} \end{cases}$	$\sigma_*^2 \frac{m_k}{v_k}$	$\frac{1}{v_k}$	$k = \arg \min_j v_j$
PoE unnormalized [Hinton, 2002]	$p_j$	$\sigma_*^2 \sum_{j=1}^J \frac{m_j}{v_j}$	$\sum_{j=1}^J \frac{1}{v_j}$	
PoE normalized [Deisenroth and Ng, 2015]	$p_j^{\frac{1}{J}}$	$\frac{\sigma_*^2}{J} \sum_{j=1}^J \frac{m_j}{v_j}$	$\frac{1}{J} \sum_{j=1}^J \frac{1}{v_j}$	
GPoE unnormalized [Fleet, 2014]	$p_j^{\tilde{\beta}_j}$	$\sigma_*^2 \sum_{j=1}^J \tilde{\beta}_j \frac{m_j}{v_j}$	$\sum_{j=1}^J \tilde{\beta}_j \frac{1}{v_j}$	
GPoE normalized [Fleet, 2014]	$p_j^{\beta_j}$	$\sigma_*^2 \sum_{j=1}^J \beta_j \frac{m_j}{v_j}$	$\sum_{j=1}^J \beta_j \frac{1}{v_j}$	$\beta_j = \frac{\tilde{\beta}_j}{\sum_{j=1}^J \tilde{\beta}_j}$
BCM [Tresp, 2000]	$\frac{p_j}{p_0^{\frac{1}{J}-\frac{1}{J}}}$	$\sigma_*^2 \sum_{j=1}^J \frac{m_j}{v_j}$	$\frac{1}{v_0} + \sum_{j=1}^J \left( \frac{1}{v_j} - \frac{1}{v_0} \right)$	
RBCM [Deisenroth and Ng, 2015]	$\frac{p_j^{\tilde{\beta}_j}}{p_0^{\tilde{\beta}_j - \frac{1}{J}}}$	$\sigma_*^2 \sum_{j=1}^J \tilde{\beta}_j \frac{m_j}{v_j}$	$\frac{1}{v_0} + \sum_{j=1}^J \tilde{\beta}_j \left( \frac{1}{v_j} - \frac{1}{v_0} \right)$	
GRBCM [Liu et al., 2018]	$\frac{\tilde{p}_j^{\tilde{\beta}_j}}{\tilde{p}_0^{\tilde{\beta}_j - \frac{1}{J}}}$	$\sigma_*^2 \left[ \frac{m_1}{v_1} + \sum_{j=2}^J \tilde{\beta}_j \left( \frac{m_j}{v_j} - \frac{m_1}{v_1} \right) \right]$	$\frac{1}{v_1} + \sum_{j=2}^J \tilde{\beta}_j \left( \frac{1}{v_j} - \frac{1}{v_1} \right)$	$\tilde{\beta}_j = \begin{cases} 1 & \text{if } j = 1, 2 \\ \tilde{\beta}_j & \text{else} \end{cases}$

Table 4.4. Different aggregation methods for  $q(f_*|\mathbf{y}) = \mathcal{N}(f_*|\mu_*, \sigma_*^2)$  with aggregated predictive mean  $\mu_* = \mu_f(\mathbf{x}_*)$  and  $\sigma_*^2 = \sigma_f^2(\mathbf{x}_*)$  variance, respectively. Thereby, the individual predictive distributions are  $p_j = p(f_{*j}|\mathbf{y}_j) = \mathcal{N}(f_{*j}|m_j, v_j)$  and  $p_0 = p(f_*) = \mathcal{N}(f_*|m_0, v_0)$ , except for GRBCM  $\tilde{p}_j = p(f_{*j}|\mathbf{y}_1, \mathbf{y}_j) = \mathcal{N}(f_{*j}|m_j, v_j)$ .

the estimated uncertainty of the predictive distribution, i.e. the variance of the local GP

#### 4.3.2.1 Predictive Posterior

For PoEs, the individual predictive distributions  $p(f_{*j}|\mathbf{y}_j) = \mathcal{N}(f_{*j}|m_j, v_j)$  from  $J$  experts corresponding to local GPs (3.9) based on the local data  $\mathcal{D}_j = \{\mathbf{y}_j, \mathbf{X}_j\}$ , are aggregated to the final approximate predictive distribution

$$q(f_*|\mathbf{y}_j) = \frac{1}{Z} \prod_{j=1}^J g_j(p(f_{*j}|\mathbf{y}_j)), \quad (4.58)$$

Thereby, we introduce the link function  $g_j$ , which depends on the particular local GP approximation method proposed in the literature as summarized in Table

4.4 in the second column and discussed in Section 4.3.2.5. Note that (4.58) is intractable for general link functions and predictive distributions  $p_j = p(f_{*j} | \mathbf{y}_j)$ , particularly due to the normalization  $Z$ . However, for certain simple choices of  $g_j$  and Gaussian predictive distributions (and for most distributions in the exponential family), the product becomes feasible. Since the individual predictive distributions correspond to local GPs and are thus Gaussian, the aggregated product is again Gaussian distributed and can be explicitly computed as formulated in Section 2.1.1.4.

We want to emphasize the particular choice  $g_j(p_j) = p_j^{\beta_j}$  for some weights  $\beta_j$ , so that the product (4.58) becomes

$$\begin{aligned} q(f_* | \mathbf{y}_j) &= \frac{1}{Z} \prod_{j=1}^J p(f_{*j} | \mathbf{y}_j)^{\beta_j} = \frac{1}{Z} \prod_{j=1}^J \mathcal{N}(f_{*j} | m_j, v_j)^{\beta_j} \\ &= \mathcal{N}(f_* | \boldsymbol{\mu}(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)), \end{aligned} \quad (4.59)$$

for which the aggregated predictive moments can be analytically obtained (2.11)

$$\boldsymbol{\mu}(\mathbf{x}_*) = \sigma^2(\mathbf{x}_*) \sum_{j=1}^J \beta_j \frac{m_j}{v_j} \quad \text{and} \quad \frac{1}{\sigma^2(\mathbf{x}_*)} = \sum_{j=1}^J \beta_j \frac{1}{v_j}.$$

For other choices of the link function, the corresponding aggregated predictive moments are summarized in Table 4.4 in the third and fourth column, respectively.

#### 4.3.2.2 Posterior

All choices of link functions have in common the same approximated posterior distribution  $\mathbf{f} | \mathbf{y}$  over the latent functions values. In particular, the posterior can be written as

$$q(\mathbf{f} | \mathbf{y}) = \frac{q(\mathbf{f}, \mathbf{y})}{q(\mathbf{y})} = \frac{p(\mathbf{y} | \mathbf{f}) q(\mathbf{f})}{q(\mathbf{y})} = \frac{1}{q(\mathbf{y})} \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j), \quad (4.60)$$

where the exact prior  $p(\mathbf{f}) \approx q(\mathbf{f})$  is approximated by the product  $q(\mathbf{f}) = \prod_{j=1}^J p(\mathbf{f}_j)$  corresponding to the assumption that all latent function values of the individual experts are mutually independent. Using the exact Gaussian likelihood  $p(\mathbf{y} | \mathbf{f})$ , the posterior distribution in (4.60) can be formulated as

$$q(\mathbf{f} | \mathbf{y}) = \prod_{j=1}^J \mathcal{N}(\mathbf{f} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \mathcal{N}(\mathbf{f} | \boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad (4.61)$$

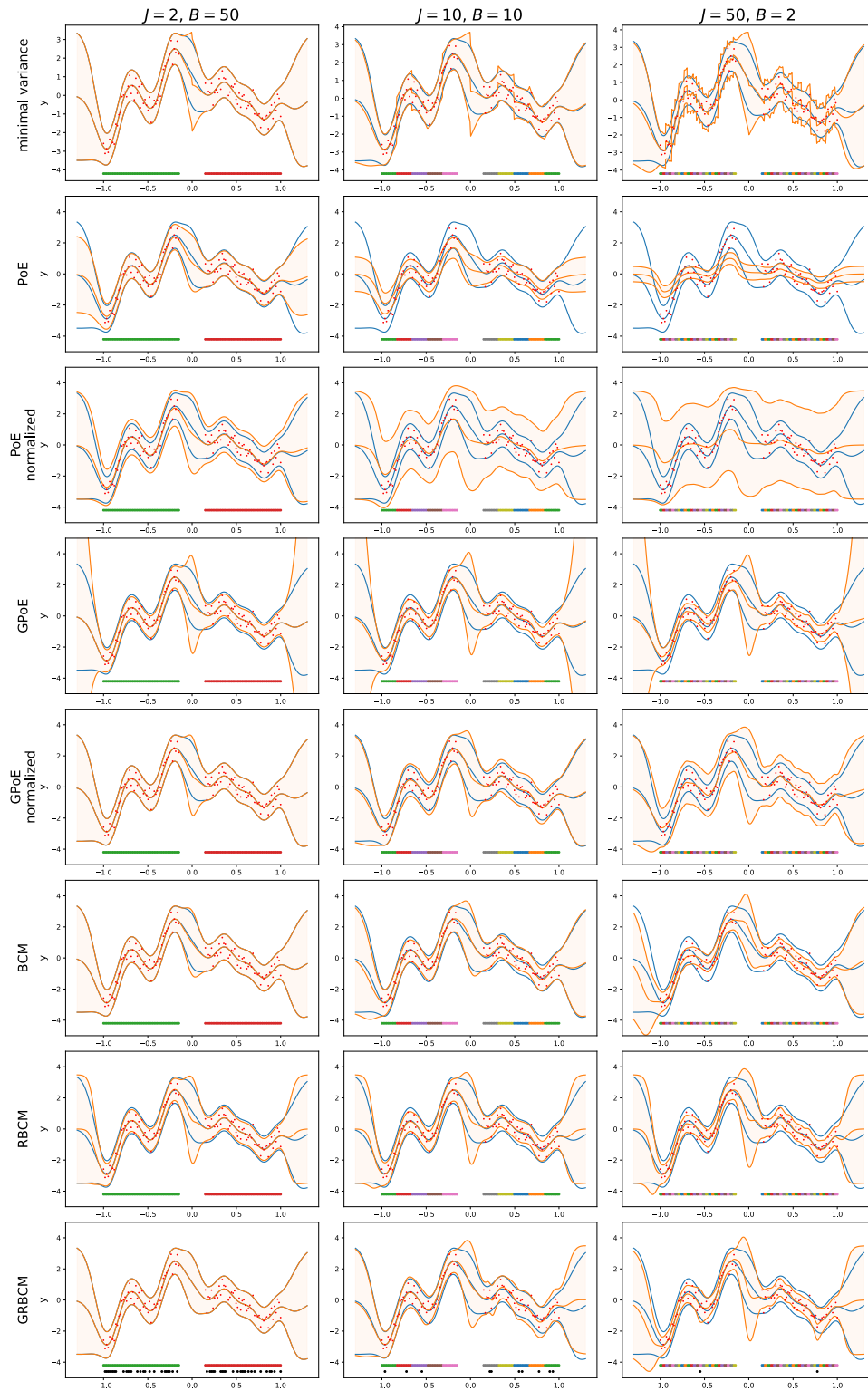


Figure 4.5. Different local GP approaches (per rows in orange) and different number of experts/partitions (per column) compared to full GP (blue).



where the individual local posterior moments  $\boldsymbol{\mu}_j$  and  $\boldsymbol{\Sigma}_j$  are given according to a local full GP (3.12), that is, with posterior mean and covariance

$$\begin{aligned}\boldsymbol{\mu}_j &= \mathbf{K}_{X_j X_j} \mathbf{P}_j^{-1} \mathbf{y}_j; \\ \boldsymbol{\Sigma}_j &= \mathbf{K}_{X_j X_j} - \mathbf{K}_{X_j X_j} \mathbf{P}_j^{-1} \mathbf{K}_{X_j X_j}\end{aligned}$$

and marginal likelihood covariance

$$\mathbf{P}_j = \mathbf{K}_{X_j X_j} + \sigma_n^2 \mathbb{I}. \quad (4.62)$$

Hence, the full posterior mean and covariance in (4.61) are  $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_J]^T$  and  $\boldsymbol{\Sigma}$  a block-diagonal matrix with blocks  $\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_J$ , respectively.

#### 4.3.2.3 Marginal Likelihood

The marginal likelihood for independent PoEs can be written as

$$\begin{aligned}q(\mathbf{y}) &= \int q(\mathbf{f}, \mathbf{y}) d\mathbf{f} = \int q(\mathbf{y}|\mathbf{f}) q(\mathbf{f}) d\mathbf{f} \\ &= \int \prod_{j=1}^J q(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j) d\mathbf{f} = \prod_{j=1}^J \int q(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j) d\mathbf{f}_j \\ &= \prod_{j=1}^J \mathcal{N}(\mathbf{y}_j | \mathbf{0}, \mathbf{P}_j)\end{aligned} \quad (4.63)$$

with marginal likelihood covariance already given in (4.62). Note that also the marginal likelihood decomposes into a product of the independent marginal likelihoods of local full GPs.

#### 4.3.2.4 Level II inference

Using the approximate marginal likelihood  $q(\mathbf{y}|\boldsymbol{\theta}) = q(\mathbf{y})$  in (4.63), inference for the hyperparameters  $\boldsymbol{\theta}$  can be done as explained in Section 2.3.3.1. For instance, the ML-II approach becomes

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} q(\mathbf{y}|\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \sum_{j=1}^J \mathbf{y}_j^T \mathbf{P}_j^{-1} \mathbf{y}_j + \log |\mathbf{P}_j|, \quad (4.64)$$

where  $\mathbf{P}_j$  depends on  $\boldsymbol{\theta}$  through the kernel matrix in (4.62). Note that the objective function (4.64) decomposes as a sum of terms which can be straightforwardly used for stochastic optimization as introduced in Section 2.1.3.2 and also exploited in Sections 6.3.3 and 6.3.7.

#### 4.3.2.5 Discussion about Different Methods

In this section, we briefly discuss several methods proposed in the literature as summarized in Table 4.4 and depicted in Figure 4.5 for a toy data set with  $N = 100$  training samples and different number of experts  $J = 2, 10, 50$  with corresponding sizes  $B = 50, 10, 2$  of the partition of each expert.

##### Product of Experts (PoEs)

PoEs with identity as the link function [Hinton, 2002] provide a continuous aggregated predictive distribution and can be seen as a smoothed version of the single prediction experts. However, the predictive variance is overconfident, in particular outside the training data; as depicted in Figure 4.5 in the second row. For large number of independent experts, the predictive distribution becomes even inside the training data unreliable due to the many non-informative experts. In order to make the predictive variance conservative, Deisenroth and Ng [2015] proposed to scale the predictive distribution by the number of experts  $J$ , which has the effect that the credible intervals of full GP are contained in the provided credible intervals, as depicted in the third row in Figure 4.5. Note that, the mean is not affected by this scaling, since the normalization cancels out. For small number of experts (e.g. first column, third row), the predictive mean and variance look close what we aim for, however, for more experts, the predictive variance becomes very conservative and the mean estimate is not satisfying.

##### Generalized Product of Experts (GPoEs)

In order to improve the shortcomings of pure PoEs, Fleet [2014] proposed to weight the predictions of the experts differently since the individual experts are not equally reliable and many experts are non-informative for a particular query point  $\mathbf{x}_*$  and destroy the overall aggregated distribution. In particular, Fleet [2014] proposed to use some varying weights for each expert  $j$  and query point  $\mathbf{x}_*$ ,

$$\bar{\beta}_{*j} = \bar{\beta}_j(\mathbf{x}_*) = H[p(f_*)] - H[p(f_{*j}|\mathbf{y}_j)] = \frac{1}{2} \log \left( \frac{v_{*0}}{v_{*j}} \right), \quad (4.65)$$

which are set to the difference in entropy  $H$  (2.7) before and after seeing the data. Thereby, the predictive prior is  $p(f_*) = \mathcal{N}(0, v_0)$  with  $v_0 = \mathbf{K}_{\mathbf{x}_*, \mathbf{x}_*}$  and the individual predictive posterior  $p(f_{*j}|\mathbf{y}_j) = \mathcal{N}(m_j, v_j)$  defined above. The link function is set to  $p(f_{*j}|\mathbf{y}_j)^{\bar{\beta}_j(\mathbf{x}_*)}$  which has the effect of increasing or decreasing the importance of the individual experts based on the corresponding prediction uncertainty  $v_0$  and  $v_j$ . As a result, the aggregated predictive distribution is sharper inside the

training data, but produce very underconfident predictive variances outside the training data, as depicted in the forth row of Figure 4.5. This can be corrected by normalizing the weights, such that  $\sum_{j=C}^J \bar{\beta}_{*j} = 1$ , yielding the normalized weights

$$\beta_{*j} = \beta_j(\mathbf{x}_*) = \frac{1}{\sum_{j=C}^J \bar{\beta}_{*j}} \bar{\beta}_{*j}. \quad (4.66)$$

Using these normalized weights instead, the resulting model has again conservative credible intervals, however, they are sharper than the normalized PoE version as depicted in the fifth row of Figure 4.5. With normalized weights, PoEs can be seen as an instance of the *covariance intersection* method [Julier and Uhlmann, 1997], which is useful for combining several estimates of random variables with known mean and variance but unknown correlation between them. This method has some interesting properties, as for instance that the accuracy of the fused estimate outperforms each local one and provides consistent fused estimate, and thus reliable confidence information.

### Bayesian Committee Machine (BCM)

A slightly different approach constitutes the *Bayesian Committee Machine* (BCM, Tresp [2000]), which is based on the following exact factorization of the predictive distribution

$$p(f_*|\mathbf{y}) = \frac{p(f_*, \mathbf{y})}{p(\mathbf{y})} = \frac{p(\mathbf{y}|f_*)p(f_*)}{p(\mathbf{y})} = \frac{p(f_*)}{p(\mathbf{y})} \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{y}_{1:j-1}, f_*), \quad (4.67)$$

where we want to empathize the conditioning on the test latent function value  $f_*$ . Note that (4.67) still corresponds to the exact predictive distribution of full GP in (3.9), however,  $p(\mathbf{y}_j|\mathbf{y}_{1:j-1}, f_*)$  is intractable due to the conditioning on all previous observations. Therefore, the approximate distribution  $q(f_*|\mathbf{y}) \approx p(f_*|\mathbf{y})$  is introduced [Tresp, 2000]

$$\begin{aligned} q(f_*|\mathbf{y}) &= \frac{1}{Z} p(f_*) \prod_{j=1}^J p(\mathbf{y}_j|f_*) = \frac{1}{Z} p(f_*) \prod_{j=1}^J \frac{p(f_*|\mathbf{y}_j) p(\mathbf{y}_j)}{p(f_*)} \\ &= \frac{1}{q(\mathbf{y})} p(f_*)^{1-J} \prod_{j=1}^J p(f_*|\mathbf{y}_j), \end{aligned} \quad (4.68)$$

where  $p(\mathbf{y}_j|\mathbf{y}_{1:j-1}, f_*) \approx p(\mathbf{y}_j|f_*)$  is assumed. We can again observe the product over the individual predictive distributions  $p(f_*|\mathbf{y}_j)$  as in (4.58), however, with

the additional prior  $p(f_*)^{1-J}$  in front of the product. Note that, we can see BCM still as a variant of PoE (4.58) with link function

$$g_j(p_j) = \frac{p_j}{p_0^{1-\frac{1}{j}}},$$

where  $p_0 = p(f_*)$ . Deisenroth and Ng [2015] proposed a robust BCM version (RBCM), which combines BCM with the unnormalized weights (4.65). [Liu et al., 2018] generalized the RBCM further by introducing a global expert, so that some communication between the local experts is possible, by an additional increase of computational time. The corresponding link functions and the aggregated predictive moments are summarized in Table 4.4 in the last two rows. The fit of the three BCM versions discussed in this section are depicted in the last three rows in Figure 4.5. We observe that for small and medium number of experts, they have similar or slightly better performance than normalized GPoE, however, they do not have the consistency properties induced by the covariance intersection method. Without these consistency guarantees, the uncertainty estimates of these methods might be questionable, since the model is always slightly overconfident resulting into too small credible intervals compared to full GP. Note that, normalizing the weights in RBCM and GRBCM is meaningless, since then the prior cancels out and GPoE is recovered. As a conclusive remark, the generalized PoE version is in the most situations preferable, since they provide as sharp as possible but still conservative credible estimates, which makes the model a reliable GP approximation approach.

#### 4.3.2.6 Advantages and Disadvantages of PoE

Local GP approximation based on averaging the local predictions have several benefits.

- **Time and Space Complexity:** The cubic time  $\mathcal{O}(N^3)$  and quadratic space  $\mathcal{O}(N^2)$  complexity of full GP can be reduced by local averaging GP approximations to  $\mathcal{O}(NB^2)$  and  $\mathcal{O}(NB)$ , respectively. This allows to scale these methods to a rather large number of training data samples.
- **Locally Exact:** Each expert corresponds to a local GP, so that it is locally exact. This allows to model quite complicated and fast varying patterns in the data. In particular, many datasets in practice can be well described by local approaches.

- **Smooth and Consistent Aggregation:** PoEs with normalized weights provide a smooth and consistent predictive distribution, so that the confidence information is reliable.
- **Non-Parameteric Method:** PoEs remain still non-parametric and have only the same few hyperparameters as full GP. As opposed to inducing point methods, where several additional parameters are introduced.
- **Distributed Training:** Since the experts are all independent, PoE allow to distribute the training for each expert, followed by the centralized prediction aggregation [Deisenroth and Ng, 2015].
- **Stochastic Training:** Since the log marginal likelihood in (4.64) decomposes as a sum, it can be used for fast stochastic training, which is a useful way to train models for large number of data samples.

On the other hand, local GP approximation methods have some drawbacks.

- **Missing Global Patterns:** The main limitation of PoE is the intrinsic local character. This means, global patterns in the data can not be modeled and important long term dependencies are missing. Note that, whether a local approach makes sense depends strongly on the particular kind of data.
- **Complete Independent Experts:** The complete independence of the experts is a rather strict assumption. Although the predictive distribution is smooth at the borders of the expert, the biggest errors compared to full GP are still between the experts. An novel version of PoEs is given in Chapter 7, where some correlations between the experts are modeled.
- **Unreliable Uncertainty Information:** For the PoE methods, which are not based on the covariance intersection method (i.e. without normalized weights), the provided uncertainty information is not reliable, which is not ideal for a probabilistic method.
- **Non-Informative Aggregation:** The predictive variance for PoE methods with normalized weights might become conservative for many experts. Although it is desirable property that the model knows that it does not know, however, at the same time it becomes also non-informative.



# Chapter 5

## Recursive Estimation for Sparse Gaussian Process Regression

In this chapter, we propose a novel unifying recursive model which allows to train a range of existing sparse GP models in an online and distributed setting. In particular, we introduce an *extended* Bayesian linear model in Section 5.1, which unifies sparse global GP models and explains the differences between sparse GPs and ordinary Bayesian linear models. These connections can be exploited for applying recursive Bayesian inference in Section 5.2. In particular, recursive inference for sparse GPs is discussed in detail in Sections 5.2.1- 5.2.5 and we show how to analytically train sparse GPs in an online and distributed setting in Sections 5.2.6 and 5.2.7, respectively. Moreover, we apply recursive variational inference in Sections 5.2.8, yielding a recursive collapsed lower bound, which constitutes the fundamental for sequential hyperparameter training in the following Chapter 6. Furthermore, we present some examples and more details in Sections 5.2.9- 5.2.11, and finally, we summarize the scientific contributions of this chapter in Section 5.3.

### 5.1 Extended Bayesian Linear Model

We recall the ordinary Bayesian linear regression model with basis functions as discussed in Section 2.3.4. In particular, the observation model is given as  $y(\mathbf{x}) = f(\mathbf{x}) + \varepsilon$ , with Gaussian observation noise  $\varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma_n^2)$ . The latent function values are defined as  $f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$ , where the weights have a Gaussian prior  $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$ . In Proposition 3.1, we showed that Bayesian inference with this model is equivalent to inference with a GP with *degenerate* kernel and thus shows the limitations of this model.

We now introduce an *extended* Bayesian linear model. In a nutshell, it corresponds to the ordinary Bayesian linear model with an additional input-dependent noise term  $\gamma(\mathbf{x})$ , leading to  $f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w} + \gamma(\mathbf{x})$ . However, instead of considering univariate inputs  $\mathbf{x} \in \mathbb{R}^D$ , in following, we consider multivariate inputs  $\mathbf{X}' = [\mathbf{x}'_1, \dots, \mathbf{x}'_{N_b}]^T \in \mathbb{R}^{N_b \times D}$  for some  $N_b \geq 1$ . For instance, we split the  $N$  training data  $\mathcal{D} = \{\mathbf{y}_i, \mathbf{x}_i\}_{i=1}^N = \{\mathbf{y}, \mathbf{X}\}$  into  $J$  blocks of equal size  $B$ , i.e.  $\mathcal{D} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$  with inputs  $\mathbf{X}_j \in \mathbb{R}^{B \times D}$  and outputs  $\mathbf{y}_j \in \mathbb{R}^B$ . Further, we consider test data with inputs  $\mathbf{X}_* \in \mathbb{R}^{N_{test} \times D}$  and outputs  $\mathbf{y}_* \in \mathbb{R}^{N_{test}}$ . Similarly, for an already defined function  $g(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}$ , we adapt the notation for a multivariate function  $g(\mathbf{X}') = [g(\mathbf{x}'_1), \dots, g(\mathbf{x}'_{N_b})]^T : \mathbb{R}^{N_b \times D} \rightarrow \mathbb{R}^{N_b}$ .

**Definition 5.1 (Extended Bayesian Linear Model)** *In the extended Bayesian linear model, the multivariate function  $f(\mathbf{X}') \in \mathbb{R}^{N_b}$  is modeled as a linear combination of the basis functions  $\phi(\mathbf{X}') \in \mathbb{R}^{N_b \times M}$  plus additional **input-dependent** noise  $\gamma(\mathbf{X}') \in \mathbb{R}^{N_b}$ , that is,*

$$f(\mathbf{X}') = \phi(\mathbf{X}')^T \mathbf{w} + \gamma(\mathbf{X}'),$$

where the involved weights  $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$  and the input-dependent noise  $\gamma(\mathbf{X}') \sim \mathcal{N}(\mathbf{0}, V(\mathbf{X}'))$  are both multivariate Gaussian distributed with zero mean and covariances  $\Sigma_0 \in \mathbb{R}^{M \times M}$  and  $V(\mathbf{X}') \in \mathbb{R}^{N_b \times N_b}$ , respectively.

The extended Bayesian model in Definition 5.1 can be combined with the Gaussian additive data generation model  $y(\mathbf{X}) = f(\mathbf{X}) + \boldsymbol{\varepsilon}$  with  $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbb{I})$  and the two noise terms  $\boldsymbol{\varepsilon}$  and  $\gamma$  are assumed to be conditionally independent given the weights.

### 5.1.1 Likelihood

In the extended Bayesian linear model with training block inputs  $\mathbf{X}_j \in \mathbb{R}^{B \times D}$ , the training latent function values  $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$  can be formulated as  $\mathbf{f}_j = \phi(\mathbf{X}_j)^T \mathbf{w} + \gamma(\mathbf{X}_j)$  together with the corresponding observations  $\mathbf{y}_j = y(\mathbf{X}_j) \in \mathbb{R}^B$  given as  $\mathbf{y}_j = \mathbf{f}_j + \boldsymbol{\varepsilon}_j$ , with  $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbb{I})$ . Hence, the likelihood for the observed training samples  $\mathbf{y} = [\mathbf{y}_1, \dots, \mathbf{y}_J]^T \in \mathbb{R}^N$  can be written as

$$p(\mathbf{y} | \mathbf{w}) = \prod_{j=1}^J \mathcal{N}(\mathbf{y}_j | \phi(\mathbf{X}_j)^T \mathbf{w}, V(\mathbf{X}_j) + \sigma_n^2 \mathbb{I}) = \mathcal{N}(\mathbf{y} | \Phi_{\mathbf{X}} \mathbf{w}, \bar{V}_{\mathbf{X}} + \sigma_n^2 \mathbb{I}),$$

where  $\Phi_{\mathbf{X}} = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_N)]^T$  already introduced in (2.22) and  $\bar{V}_{\mathbf{X}}$  a block-diagonal matrix with entries  $V(\mathbf{X}_1), \dots, V(\mathbf{X}_J)$  on the diagonal. Similarly, the



test likelihood for a query input  $\mathbf{X}_* \in \mathbb{R}^{N_{\text{test}} \times D}$  with corresponding output  $\mathbf{y}_* = \mathbf{y}(\mathbf{X}_*)$  can be written as  $p(\mathbf{y}_* | \mathbf{w}) = \mathcal{N}(\mathbf{y}_* | \Phi_* \mathbf{w}, \bar{\mathbf{V}}_* + \sigma_n^2 \mathbb{I})$ , where we introduce  $\Phi_* = \phi(\mathbf{X}_*)^T$  and  $\bar{\mathbf{V}}_* = V(\mathbf{X}_*)$ . As comparison, the likelihood for the ordinary Bayesian model is  $p(\mathbf{y} | \mathbf{w}) = \mathcal{N}(\mathbf{y} | \Phi_X \mathbf{w}, \sigma_n^2 \mathbb{I})$  with constant variance as defined in (2.34).

### 5.1.2 Predictive Posterior

Multiplying the training  $p(\mathbf{y} | \mathbf{w})$  and test likelihood  $p(\mathbf{y}_* | \mathbf{w})$  with the prior  $p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \mathbf{0}, \Sigma_0)$ , yields the joint distribution  $p(\mathbf{w}, \mathbf{y}, \mathbf{y}_*)$

$$\mathcal{N} \left( \begin{bmatrix} \mathbf{w} \\ \mathbf{y} \\ \mathbf{y}_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \Sigma_0 & & \\ \Phi_X \Sigma_0 & \Phi_X \Sigma_0 \Phi_X^T + \bar{\mathbf{V}}_X + \sigma_n^2 \mathbb{I} & \\ \Phi_* \Sigma_0 & \Phi_* \Sigma_0 \Phi_X^T & \Phi_* \Sigma_0 \Phi_*^T + \bar{\mathbf{V}}_* + \sigma_n^2 \mathbb{I} \end{bmatrix} \right). \quad (5.1)$$

Similarly as in the ordinary Bayesian model in (2.41), by conditioning (2.3) on  $\mathbf{y}$  and integrating out (2.2) the weights  $\mathbf{w}$ , yields the predictive posterior distribution

$$p(\mathbf{y}_* | \mathbf{y}) = \int \frac{p(\mathbf{y}_* | \mathbf{w}) p(\mathbf{y} | \mathbf{w}) p(\mathbf{w})}{p(\mathbf{y})} d\mathbf{w} = \mathcal{N}(\mathbf{y}_* | \mu(\mathbf{X}_*), \Sigma(\mathbf{X}_*)),$$

with predictive moments

$$\begin{aligned} \mu(\mathbf{X}_*) &= \Phi_* \Sigma_0 \Phi_X^T \mathbf{P}^{-1} \mathbf{y}, \\ \Sigma(\mathbf{X}_*) &= \Phi_* \Sigma_0 \Phi_*^T - \Phi_* \Sigma_0 \Phi_X^T \mathbf{P}^{-1} \Phi_X \Sigma_0 \Phi_*^T + \bar{\mathbf{V}}_* + \sigma_n^2 \mathbb{I}, \end{aligned} \quad (5.2)$$

and with marginal likelihood covariance  $\mathbf{P} = \Phi_X \Sigma_0 \Phi_X^T + \bar{\mathbf{V}}_X + \sigma_n^2 \mathbb{I}$ . Compared to the predictive mean (2.42) and variance (2.43) of the ordinary Bayesian linear model, the only differences are the additional training  $\bar{\mathbf{V}}_X$  and test  $\bar{\mathbf{V}}_*$  covariances.

### 5.1.3 Sparse GPs Revisited

If we compare the predictive moments in (5.2) with the predictive moments of a generic sparse GP model in (4.17), we can recognize the same structure. We recall here the predictive distribution in (4.17), that is,

$$\begin{aligned} \mu(\mathbf{X}_*) &= \mathbf{Q}_{X_* X} \mathbf{P}^{-1} \mathbf{y}, \\ \Sigma(\mathbf{X}_*) &= \mathbf{Q}_{X_* X_*} - \mathbf{Q}_{X_* X} \mathbf{P}^{-1} \mathbf{Q}_{X X_*} + \bar{\mathbf{V}}_* + \sigma_n^2 \mathbb{I}, \end{aligned} \quad (5.3)$$

with covariance matrix  $\mathbf{P} = \mathbf{Q}_{XX} + \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$ . By matching terms in (5.2) and (5.3), the following proposition can be formulated.

**Proposition 5.1 (Equal Predictive Posterior and Marginal Likelihood)**

Consider a sparse GP model (4.12) as described in Section 4.2 with inducing points  $\mathbf{A}$  and kernel  $k$  and an extended Bayesian linear model in Definition 5.1 with prior covariance  $\Sigma_0$  and basis functions  $\phi(\mathbf{X}')$ . For all choices of prior covariance and basis functions, which satisfy

$$\phi(\mathbf{X}')^T \Sigma_0 \phi(\mathbf{X}'') = k(\mathbf{X}', \mathbf{A}) k(\mathbf{A}, \mathbf{A})^{-1} k(\mathbf{A}, \mathbf{X}''),$$

the predictive distributions in (5.2) and (5.3) are equivalent. Moreover, the marginal likelihood of both models are equal. Thereby, the input-dependent noise  $V(\mathbf{X}')$  of the extended Bayesian linear model depends on the particular version of the sparse GP.

Modifying the input-dependent variance  $V(\mathbf{X}')$  of an extended Bayesian linear model in Proposition (5.1) correspond exactly to modifying the training  $\bar{\mathbf{V}}$  and test variance  $\bar{\mathbf{V}}_*$  as discussed in Section 4.2.2. This means, that all sparse GP models summarized in Table 4.1 can be equivalently modeled as a extended Bayesian linear model by adapting  $\bar{\mathbf{V}}_X$  and  $\bar{\mathbf{V}}_*$  accordingly.

	basis function $\phi(\mathbf{x})$	prior covariance $\Sigma_0$	posterior mean $\mu$	comments
$\mathbf{g}_{\mathbf{x}, \mathbf{A}}(\mathbf{x})$	$\mathbf{P}^{-1} \mathbf{K}_{\mathbf{x}\mathbf{A}} \mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1} k(\mathbf{A}, \mathbf{x})$	$\mathbf{P} \mathbf{Q}_{\mathbf{X}\mathbf{X}}^{-1} \mathbf{P}$	$\mathbf{y}$	posterior of weights correspond to data; linear predictor/smoothing
$\mathbf{k}_{\mathbf{A}}(\mathbf{x})$	$k(\mathbf{A}, \mathbf{x})$	$\mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1}$	$\mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1} \mathbf{K}_{\mathbf{A}\mathbf{X}} \mathbf{P}^{-1} \mathbf{y}$	basis function correspond to kernel; representer theorem
$\mathbf{l}_{\mathbf{A}}(\mathbf{x})$	$\mathbf{L}^{-1} k(\mathbf{A}, \mathbf{x})$	$\mathbb{I}$	$\mathbf{L}^{-1} \mathbf{K}_{\mathbf{A}\mathbf{X}} \mathbf{P}^{-1} \mathbf{y}$	prior covariance correspond to identity via Cholesky
$\mathbf{u}_{\mathbf{A}}(\mathbf{x})$	$\mathbf{D}^{-\frac{1}{2}} \mathbf{U}^T k(\mathbf{A}, \mathbf{x})$	$\mathbb{I}$	$\mathbf{D}^{-\frac{1}{2}} \mathbf{U}^T \mathbf{K}_{\mathbf{A}\mathbf{X}} \mathbf{P}^{-1} \mathbf{y}$	prior covariance correspond to identity via eigendecomposition
$\mathbf{d}_{\mathbf{A}}(\mathbf{x})$	$\mathbf{U}^T k(\mathbf{A}, \mathbf{x})$	$\mathbf{D}^{-1}$	$\mathbf{D}^{-1} \mathbf{U}^T \mathbf{K}_{\mathbf{A}\mathbf{X}} \mathbf{P}^{-1} \mathbf{y}$	prior covariance = 1/eigenvalues; basis function = eigenfunction; Mercer's theorem
$\mathbf{h}_{\mathbf{A}}(\mathbf{x})$	$\mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1} k(\mathbf{A}, \mathbf{x})$	$\mathbf{K}_{\mathbf{A}\mathbf{A}}$	$\mathbf{K}_{\mathbf{A}\mathbf{X}} \mathbf{P}^{-1} \mathbf{y}$	posterior of weights in extended Bayesian linear model correspond to posterior of inducing points in SGP

Table 5.1. Possible choices for basis function and prior covariance in an extended Bayesian linear model, so that the predictive distribution and the marginal likelihood are equivalent to those of sparse GPs.

If the focus lies only on the predictive distribution and marginal likelihood, the choice for the basis function  $\phi$  and prior covariance  $\Sigma_0$  is not unique as illustrated in Table 5.1, similar in Table 3.2 for full GP.

**Proposition 5.2 (Equivalent Posterior)** *For the particular choice of prior covariance  $\Sigma_0 = \mathbf{K}_{AA}$  and basis functions  $\mathbf{h}_A(\mathbf{X}')$  in Table 5.1, which correspond to the training  $\mathbf{H} = \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}$  and test  $\mathbf{H}_* = \mathbf{K}_{X_*A}\mathbf{K}_{AA}^{-1}$  basis function matrices, the posterior of the weights  $p(\mathbf{w}|\mathbf{y})$  in an extended linear model is equivalent to the posterior of the inducing points  $p(\mathbf{a}|\mathbf{y})$  in a sparse GP.*

**Example 5.1 (Extended Bayesian Linear Model for GPs)** *In this example, we illustrate the connection between GP models and extended Bayesian linear models. We first empathize, that also **full GPs** can be explained by an extended Bayesian model. In particular, the basis functions are chosen as  $\phi(\mathbf{X}') = \mathbf{K}_{X'X}\mathbf{K}_{XX}^{-1}$ , which yield identity  $\phi(\mathbf{X}) = \mathbb{I}$  for training inputs and  $\phi(\mathbf{X}_*) = \mathbf{K}_{X_*X}\mathbf{K}_{XX}^{-1}$  for test inputs. Further, the prior covariance  $\Sigma_0 = \mathbf{K}_{XX}$  is set to the kernel matrix of the input points, together with the additional input-dependent noise covariance function  $V(\mathbf{X}') = \mathbf{K}_{X'X'} - \mathbf{K}_{X'X}\mathbf{K}_{XX}^{-1}\mathbf{K}_{XX'}$ , leading to  $V(\mathbf{X}) = \bar{\mathbf{V}}_X = \mathbf{0}$  for the training and  $V(\mathbf{X}_*) = \bar{\mathbf{V}}_* = \mathbf{K}_{X_*X_*} - \mathbf{K}_{X_*X}\mathbf{K}_{XX}^{-1}\mathbf{K}_{XX_*}$  for the test inputs. These choices yield an extended Bayesian model with the same predictive posterior, marginal likelihood and posterior distribution as full GP. The important difference to an ordinary Bayesian linear model is the additional input-dependent noise for the test inputs, which imitate the infinite-dimensional basis functions. Note that, if only the predictive posterior and marginal likelihood distributions of the extended Bayesian model and full GP should match, then there are several choices for the basis function and prior covariance as summarized in Table 3.2, for which some examples of basis functions are given in Figure 5.1 in the first column.*

Similarly, for **sparse GPs**, an extended Bayesian model with basis functions  $\phi(\mathbf{X}') = \mathbf{K}_{X'A}\mathbf{K}_{AA}^{-1}$  and prior covariance  $\Sigma_0 = \mathbf{K}_{AA}$ , together with the additional noise covariances  $\bar{\mathbf{V}}_X$  and  $\bar{\mathbf{V}}_*$  according to Table 5.2, lead to an equivalent sparse GP model. In particular, in Figure 5.1, we show 3 examples for different training covariances  $\bar{\mathbf{V}}_X = \alpha \text{Diag}[\mathbf{K}_{XX} - \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX}]$ , with  $\alpha \in \{0, 0.5, 1\}$  together with the test covariance is set to  $\bar{\mathbf{V}}_* = \mathbf{K}_{X_*X_*} - \mathbf{K}_{X_*A}\mathbf{K}_{AA}^{-1}\mathbf{K}_{AX_*}$ . In the first row, we show the corresponding predictive distribution indicated with the mean and 95%-credible interval. In the second row, we show the mean alone, where we note that for  $\alpha = 0$ , the mean correspond somehow to an average between all data points, whereas for  $\alpha = 1$ , the data points far away from the inducing points are explained by the additional variance and do not have a big influence. On the other hand, the posterior mean of the data points close to the inducing points are basically exact compared to them of full

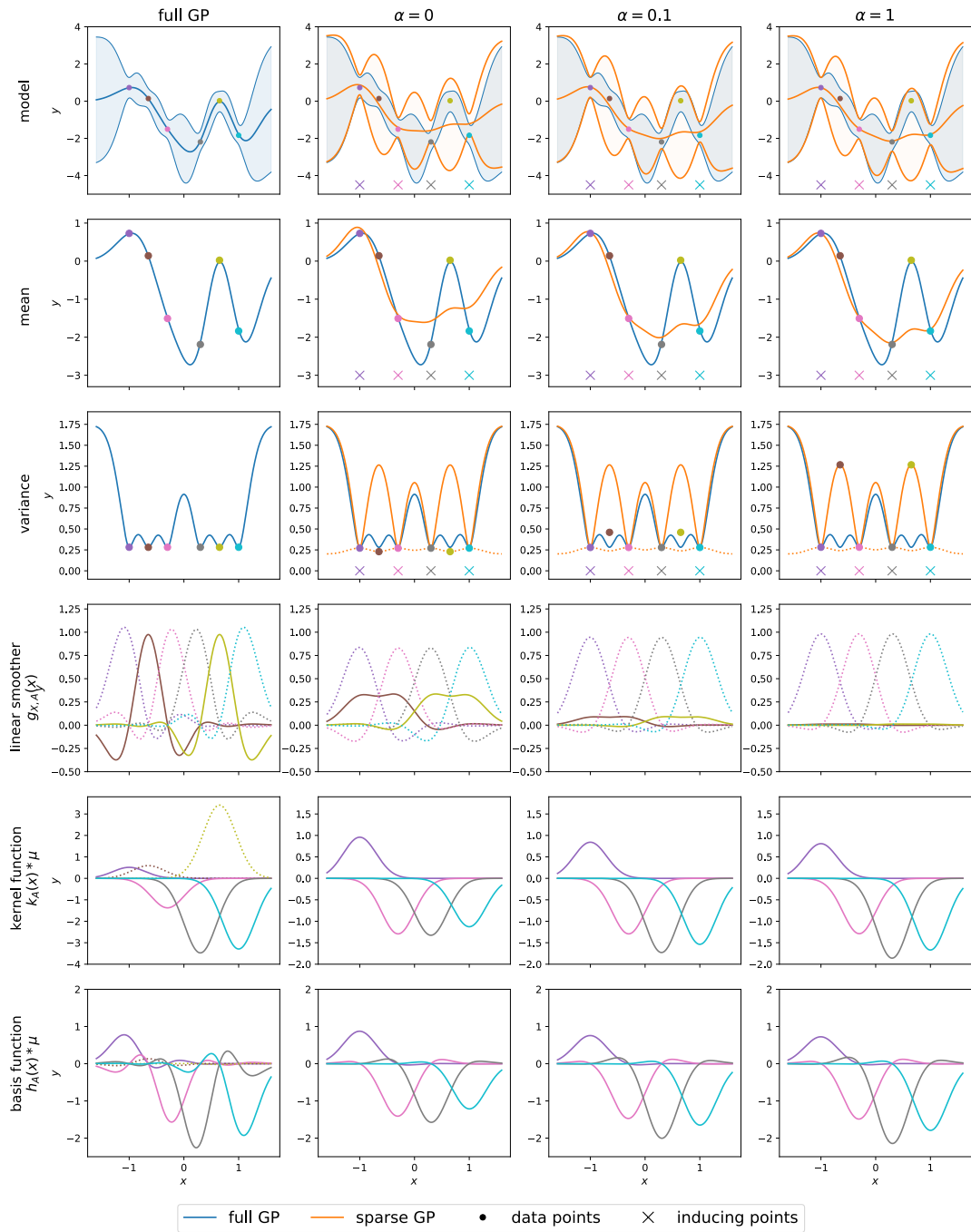


Figure 5.1. Extended Bayesian linear basis function model with additional input-varying noise. Consider Example 5.1 for details.

GP. In the third row, the predictive variance of an **ordinary** Bayesian linear model (dotted orange line) together with the variance of an **extended** Bayesian linear model (solid orange line) is depicted. We observe, that in regions, where no inducing points are available, the predictive variance is rather conservative compared to them of full GP, taking into account that it cannot model the function properly. In the fourth row, we show the influence of each data point, by the corresponding basis function, which is weighted by the observation (compare 1st row in Table 5.1). In particular, the basis function corresponding to the two data points, where no inducing point is placed, are indicated as solid lines, whereas the others are depicted as dotted lines. We can observe that for full GP, the two data points have a significant influence. On the other hand, for  $\alpha = 1$ , they have basically no influence, since they are treated completely as noise. For  $\alpha = 0$ , they still have an influence, however, at the same time, the influence of the others is decreased, resulting in an average-like mean. In the last two rows, two choices for weighted basis functions are shown (compare also Table 5.1 for more details), for which the predictive distributions of the extended Bayesian linear model is equal to them of sparse GP.

## 5.2 Recursive Sparse GP Model

In the previous section, we established the connection between sparse GP models and extended Bayesian linear models. These connections allow to exploit recursive Bayesian inference techniques, so that we can train sparse GPs sequentially. In particular, we consider an extended Bayesian linear model with basis functions  $\phi(\mathbf{X}') = H(\mathbf{X}') = \mathbf{K}_{\mathbf{X}'\mathbf{A}}\mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1} \in \mathbb{R}^{B \times M}$  and prior covariance  $\Sigma_0 = \mathbf{K}_{\mathbf{A}\mathbf{A}} \in \mathbb{R}^{M \times M}$ , which correspond to a generic sparse GP regression model with kernel  $k$  and inducing inputs  $\mathbf{A} \in \mathbb{R}^{M \times D}$  with inducing outputs  $\mathbf{a} \in \mathbb{R}^M$  as discussed in Proposition 5.2. For blocks of training inputs  $\mathbf{X}_j$  and test input  $\mathbf{X}_*$ , the extended Bayesian linear model can be therefore formulated as

$$\begin{aligned} f(\mathbf{X}_j) &= H(\mathbf{X}_j) \mathbf{a} + \gamma(\mathbf{X}_j) & \text{and} & & y(\mathbf{X}_j) &= f(\mathbf{X}_j) + \varepsilon_j; \\ f(\mathbf{X}_*) &= H(\mathbf{X}_*) \mathbf{a} + \gamma(\mathbf{X}_*) & \text{and} & & y(\mathbf{X}_*) &= f(\mathbf{X}_*) + \varepsilon_*, \end{aligned}$$

with input-varying noise  $\gamma(\mathbf{X}_j) \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{V}}_j)$  and  $\gamma(\mathbf{X}_*) \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{V}}_*)$ , where the approximate covariances  $\bar{\mathbf{V}}_j$  and  $\bar{\mathbf{V}}_*$  are given in Table 5.2 depending on the particular model. Further, we have  $\varepsilon_j \sim \varepsilon_* \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbb{I})$ , where we assume that  $\gamma(\mathbf{X}_j)$  and  $\varepsilon_j$ , and similarly  $\gamma(\mathbf{X}_*)$  and  $\varepsilon_*$ , are conditionally independent given the inducing points and we abbreviate  $\mathbf{f}_j = f(\mathbf{X}_j)$ ,  $\mathbf{f}_* = f(\mathbf{X}_*)$ ,  $\mathbf{y}_j = y(\mathbf{X}_j)$ ,  $\mathbf{y}_* = y(\mathbf{X}_*)$ ,  $\mathbf{H}_j = H(\mathbf{X}_j)$  and  $\mathbf{H}_* = H(\mathbf{X}_*)$ . Similarly to the batch case in (4.12), this sparse recursive GP model can be probabilistically formulated:

**Prior:**

$$p(\mathbf{a}) = \mathcal{N}(\mathbf{0}, \Sigma_0) \quad (5.4)$$

**Recursive Training:**

$$\begin{aligned} q(\mathbf{f}_j|\mathbf{a}) &= \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}, \bar{\mathbf{V}}_j) \\ p(\mathbf{y}_j|\mathbf{f}_j) &= \mathcal{N}(\mathbf{y}_j|\mathbf{f}_j, \sigma_n^2\mathbb{I}) \end{aligned} \quad (5.5)$$

**Recursive Prediction:**

$$\begin{aligned} q(\mathbf{f}_*|\mathbf{a}) &= \mathcal{N}(\mathbf{f}_*|\mathbf{H}_*\mathbf{a}, \bar{\mathbf{V}}_*) \\ p(\mathbf{y}_*|\mathbf{f}_*) &= \mathcal{N}(\mathbf{y}_*|\mathbf{f}_*, \sigma_n^2\mathbb{I}) \end{aligned} \quad (5.6)$$

Note that the prior  $p(\mathbf{a})$  and the likelihoods  $p(\mathbf{y}_j|\mathbf{f}_j)$ ,  $p(\mathbf{y}_*|\mathbf{f}_*)$  are assumed to be exact with respect to full GP, however, the covariance of the recursive conditionals  $q(\mathbf{f}_j|\mathbf{a})$  and  $q(\mathbf{f}_*|\mathbf{a})$  are approximated by the covariances  $\bar{\mathbf{V}}_j = V(\mathbf{X}_j)$  and  $\bar{\mathbf{V}}_* = V(\mathbf{X}_*)$  according to Table 5.2, so that different sparse GP methods can be recovered. This is similar to Section 4.2.2, where batch sparse GP models with exact Bayesian inference together with approximate priors are discussed, however, in this section, we discuss exact *recursive* Bayesian inference with approximate sequential conditional distributions.

The probabilistic equations (5.4), (5.5) and (5.6) give rise to the joint distribution of the recursive sparse GP model

$$q(\mathbf{a}, \mathbf{f}, \mathbf{f}_*, \mathbf{y}, \mathbf{y}_*) = p(\mathbf{a}) p(\mathbf{y}_*|\mathbf{f}_*) q(\mathbf{f}_*|\mathbf{a}) \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) q(\mathbf{f}_j|\mathbf{a}), \quad (5.7)$$

which is equivalent to the joint distribution for batch sparse GP in (4.13). Since the latent function values  $\mathbf{f}$  and  $\mathbf{f}_*$  are not of particular interest, they can be integrated out leading to the induced likelihood conditioned on the inducing points

$$q(\mathbf{y}_j|\mathbf{a}) = \int p(\mathbf{y}_j|\mathbf{f}_j) q(\mathbf{f}_j|\mathbf{a}) d\mathbf{f}_j = \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \mathbf{V}_j),$$

with  $\mathbf{V}_j = \bar{\mathbf{V}}_j + \sigma_n^2\mathbb{I}$  and similarly for the test case  $q(\mathbf{y}_*|\mathbf{a}) = \int p(\mathbf{y}_*|\mathbf{f}_*) q(\mathbf{f}_*|\mathbf{a}) d\mathbf{f}_* = \mathcal{N}(\mathbf{y}_*|\mathbf{H}_*\mathbf{a}, \mathbf{V}_*)$  with  $\mathbf{V}_* = \bar{\mathbf{V}}_* + \sigma_n^2\mathbb{I}$ . Hence, the joint distribution of the recursive sparse GP model (5.7) simplifies to

$$q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}) = p(\mathbf{a}) q(\mathbf{y}_*|\mathbf{a}) \prod_{j=1}^J q(\mathbf{y}_j|\mathbf{a}).$$

description	training conditional covariance $\tilde{V}_j$	test conditional covariance $\tilde{V}_*$	marginal likelihood correction term $a_j$
<b>degenerated GP</b> SoR [Silverman, 1985], DIC [Smola and Bartlett, 2001]	$\mathbf{0}$	$\mathbf{0}$	0
<b>augmented GP</b> FIC [Quiñonero-Candela and Rasmussen, 2005]	$\text{Diag}[\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}]$	$\text{Diag}[\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}]$	0
<b>deterministic "GP"</b> PP [Csató and Opper, 2002], DTC [Seeger et al., 2003] VFE [Titsias, 2009]	$\mathbf{0}$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	$\frac{0}{\frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}]}$
<b>augmented "GP"+</b> SSGP [Snelson and Ghahramani, 2006], FITC [Quiñonero-Candela and Rasmussen, 2005], EP [Bui et al., 2017b]	$\text{Diag}[\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}]$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	0
<b>block augmented "GP"+</b> PITC [Quiñonero-Candela and Rasmussen, 2005]	$\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	0
<b>power "GP"</b> PEP [Bui et al., 2017b]	$\alpha \text{Diag}[\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}]$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	$\frac{1-\alpha}{2\alpha} \sum_{i=1}^B \log \left( 1 + \frac{\alpha}{\sigma_n^2} (\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j})_{ii} \right)$
<b>block power "GP"</b> PEP-B [Bui et al., 2017b]	$\alpha (\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j})$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	$\frac{1-\alpha}{2\alpha} \log \left  \mathbb{I} + \frac{\alpha}{\sigma_n^2} (\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}) \right $
<b>block augmented GP</b> PIC [Snelson and Ghahramani, 2007]	$\mathbf{K}_{X_j, X_j} - \mathbf{Q}_{X_j, X_j}$	$\mathbf{K}_{X_*, X_*} - \mathbf{Q}_{X_*, X_*}$	0

Table 5.2. Summary of parameters for recursive sparse GP models. In particular, using the indicated choices of recursive training  $\tilde{V}_j$  and test  $\tilde{V}_*$  covariances together with the marginal likelihood correction term  $a_j$  (which is discussed in Section 5.2.8.1), allows to train a range of well-known sparse GP models in a recursive way, so that after considering all  $J$  blocks of data, the corresponding batch sparse GP model, indicated in the first column, is recovered (compare also Table 4.1).

### 5.2.1 Bayesian Recursive Updating

Instead directly computing the batch posterior  $q(\mathbf{a}|\mathbf{y})$  of the inducing points conditioned on all output variables  $\mathbf{y} = [\mathbf{y}_1, \dots, \mathbf{y}_J]^T$  as in the batch case (4.15), we propose here a recursive procedure based on Bayesian recursive updating. In this first subsection, we provide an overview over the involved operations, which are explained in detail in the subsequent sections. We assume by recursion that at step  $j$  the *previous posterior* distribution, conditioned on the previous observations  $\mathbf{y}_{1:j-1}$ , is available

$$q(\mathbf{a}|\mathbf{y}_{1:j-1}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_{j-1}).$$

The recursion starts from  $j = 1$  with  $p(\mathbf{a}|\mathbf{y}_{1:0}) = p(\mathbf{a}) = \mathcal{N}(\mathbf{a}|\mathbf{0}, \boldsymbol{\Sigma}_0)$ . In general, this previous posterior can be interpreted as prior for the current step, which can be combined with the likelihood, yielding the *joint prior* at step  $j$

$$q(\mathbf{a}, \mathbf{y}_* \mathbf{y}_j | \mathbf{y}_{1:j-1}) = p(\mathbf{y}_* | \mathbf{a}) p(\mathbf{y}_j | \mathbf{a}) q(\mathbf{a} | \mathbf{y}_{1:j-1}).$$

By conditioning on the current observations  $\mathbf{y}_j$ , the *updated joint posterior*  $j$  can be computed by

$$q(\mathbf{a}, \mathbf{y}_* | \mathbf{y}_{1:j}) = \frac{q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_j | \mathbf{y}_{1:j-1})}{p(\mathbf{y}_j)},$$

which is conditioned on the data  $\mathbf{y}_{1:j}$  including the current observations  $\mathbf{y}_j$ . From this distribution, the next posterior can be obtained by integrating out  $\mathbf{y}_*$

$$q(\mathbf{a} | \mathbf{y}_{1:j}) = \int q(\mathbf{a}, \mathbf{y}_* | \mathbf{y}_{1:j}) d\mathbf{y}_* = \mathcal{N}(\mathbf{a} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),$$

which acts as prior in the next step. Moreover, the *predictive distribution* at step  $j$  can be obtained by integrating out  $\mathbf{a}$ , that is,

$$q(\mathbf{y}_* | \mathbf{y}_{1:j}) = \int q(\mathbf{a}, \mathbf{y}_* | \mathbf{y}_{1:j}) d\mathbf{a} = \mathcal{N}(\mathbf{y}_* | \boldsymbol{\mu}_j(\mathbf{X}_*), \boldsymbol{\Sigma}_j(\mathbf{X}_*)).$$

Since for sparse GPs, all these recursive distributions are Gaussian, they can be analytically computed, which will be demonstrated in the following.

### 5.2.2 Recursive Joint Prior Distribution

The recursion is started involving the first output  $\mathbf{y}_1$ , that is,  $q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_1) = q(\mathbf{y}_* | \mathbf{a}) p(\mathbf{y}_1 | \mathbf{a}) p(\mathbf{a})$ . In the general case at step  $j$ , the previous posterior distribution  $p(\mathbf{a} | \mathbf{y}_{1:j-1})$  is interpreted as the next prior, so that the joint prior at step  $j$  can be formulated as the product of the Gaussians  $q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_j | \mathbf{y}_{1:j-1}) = p(\mathbf{y}_* | \mathbf{a}) p(\mathbf{y}_j | \mathbf{a}) p(\mathbf{a} | \mathbf{y}_{1:j-1}) = \mathcal{N}(\mathbf{y}_* | \mathbf{H}_* \mathbf{a}, \mathbf{V}_*) \mathcal{N}(\mathbf{y}_j | \mathbf{H}_j \mathbf{a}, \mathbf{V}_j) \mathcal{N}(\mathbf{a} | \boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_{j-1})$ , which can be explicitly computed as

$$\mathcal{N} \left( \begin{bmatrix} \mathbf{a} \\ \mathbf{y}_* \\ \mathbf{y}_j \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_{j-1} \\ \mathbf{H}_* \boldsymbol{\mu}_{j-1} \\ \mathbf{H}_j \boldsymbol{\mu}_{j-1} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{j-1} & \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T & \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \\ \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} & \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T + \mathbf{V}_* & \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \\ \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} & \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T & \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \mathbf{V}_j \end{bmatrix} \right). \quad (5.8)$$

By comparing this joint recursive sparse GP prior in (5.8) to the batch sparse GP prior in (4.14), we note that the recursive prior mean is no longer zero and  $\mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T$  is the recursive analogous to the Nyström matrix  $\mathbf{Q}_{XX_*}$  in the batch case.



### 5.2.3 Recursive Updated Posterior

The updated posterior distribution over the inducing points, conditioned on all previous data  $p(\mathbf{y}_{1:j})$ , can be obtained by integrating out  $\mathbf{y}_*$  in (5.8) via (2.2) and conditioning (2.3) on the current observation  $\mathbf{y}_j$ , that is,

$$q(\mathbf{a}|\mathbf{y}_{1:j}) = \int \frac{q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_j | \mathbf{y}_{1:j-1})}{p(\mathbf{y}_j)} d\mathbf{y}_* = \mathcal{N}(\mathbf{a} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad (5.9)$$

with recursive posterior mean and covariance

$$\begin{aligned} \boldsymbol{\mu}_j &= \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \mathbf{P}_j^{-1} (\mathbf{y}_j - \mathbf{H}_j \boldsymbol{\mu}_{j-1}), \\ \boldsymbol{\Sigma}_j &= \boldsymbol{\Sigma}_{j-1} - \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \mathbf{P}_j^{-1} \mathbf{H}_j \boldsymbol{\Sigma}_{j-1}, \end{aligned} \quad (5.10)$$

where  $\mathbf{P}_j = \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \mathbf{V}_j$ . The corresponding batch posterior moments are formulated in (4.20). The matrix inversion lemma (2.16) can be applied to  $\mathbf{P}_j$  (or directly applying (2.6) to (5.8)), leading to a second version of the recursive posterior moments

$$\begin{aligned} \boldsymbol{\mu}_j &= \boldsymbol{\Sigma}_j \left( \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j + \boldsymbol{\Sigma}_{j-1}^{-1} \boldsymbol{\mu}_{j-1} \right), \\ \boldsymbol{\Sigma}_j &= \left( \boldsymbol{\Sigma}_{j-1}^{-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j \right)^{-1}, \end{aligned} \quad (5.11)$$

which correspond to (4.21) in the batch case. This second version requires the inversion for the posterior covariance  $\boldsymbol{\Sigma}_j \in \mathbb{R}^{M \times M}$  instead the inversion of  $\mathbf{P}_j \in \mathbb{R}^{B \times B}$ .

### 5.2.4 Recursive Predictive Posterior

The recursive predictive posterior can be obtained by instead integrating out  $\mathbf{a}$  via (2.2) and conditioning (2.3) on the current observation  $\mathbf{y}_j$  in the recursive joint prior (5.8), that is,

$$q(\mathbf{y}_* | \mathbf{y}_{1:j}) = \int \frac{q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_j | \mathbf{y}_{1:j-1})}{p(\mathbf{y}_j)} d\mathbf{a} = \mathcal{N}(\mathbf{a} | \boldsymbol{\mu}_j(\mathbf{X}_*), \boldsymbol{\Sigma}_j(\mathbf{X}_*)), \quad (5.12)$$

with recursive predictive posterior mean and covariance

$$\begin{aligned} \boldsymbol{\mu}_j(\mathbf{X}_*) &= \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \mathbf{P}_j^{-1} (\mathbf{y}_j - \mathbf{H}_j \boldsymbol{\mu}_{j-1}), \\ \boldsymbol{\Sigma}_j(\mathbf{X}_*) &= \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T + \mathbf{V}_* - \mathbf{H}_* \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \mathbf{P}_j^{-1} \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_*^T, \end{aligned} \quad (5.13)$$

which has a similar structure as the batch predictive distribution (4.17). Alternatively, we can again follow a two-stage procedure based on the current updated posterior  $q(\mathbf{a}|\mathbf{y}_{1:j})$  in (5.9), that is,

$$q(\mathbf{y}_*|\mathbf{y}_{1:j}) = \int p(\mathbf{y}_*|\mathbf{a})q(\mathbf{a}|\mathbf{y}_{1:j})d\mathbf{a} = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}_j(\mathbf{X}_*), \boldsymbol{\Sigma}_j(\mathbf{X}_*)), \quad (5.14)$$

leading to the predictive moments  $\boldsymbol{\mu}_j(\mathbf{X}_*) = \mathbf{H}_*\boldsymbol{\mu}_j$  and  $\boldsymbol{\Sigma}_j(\mathbf{X}_*) = \mathbf{H}_*\boldsymbol{\Sigma}_j\mathbf{H}_*^T + \mathbf{V}_*$ , which are equivalent to (5.13).

### 5.2.5 Recursive Marginal Likelihood

The recursive marginal likelihood for the current output  $\mathbf{y}_j$ , conditioned on all previous observations  $\mathbf{y}_{1:j-1}$ , can be obtained by integrating out  $\mathbf{y}_*$  and  $\mathbf{a}$  via (2.2) in the recursive joint prior (5.8), that is,

$$q(\mathbf{y}_j|\mathbf{y}_{1:j-1}) = \int q(\mathbf{a}, \mathbf{y}_*, \mathbf{y}_j|\mathbf{y}_{1:j-1})d\mathbf{y}_*d\mathbf{a} = \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\boldsymbol{\mu}_{j-1}, \mathbf{P}_j), \quad (5.15)$$

where the recursive marginal likelihood covariance  $\mathbf{P}_j = \mathbf{H}_j\boldsymbol{\Sigma}_{j-1}\mathbf{H}_j^T + \mathbf{V}_j$  already defined above. The product over this recursive marginal likelihood distribution satisfies

$$q(\mathbf{y}) = \prod_{j=1}^J q(\mathbf{y}_j|\mathbf{y}_{1:j-1}) = \prod_{j=1}^J \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\boldsymbol{\mu}_{j-1}, \mathbf{P}_j) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{P}), \quad (5.16)$$

with batch marginal likelihood  $\mathbf{P} = \mathbf{H}\boldsymbol{\Sigma}_0\mathbf{H}^T + \mathbf{V} = \mathbf{Q}_{\mathbf{X}\mathbf{X}} + \mathbf{V}$  in (4.24).

**Proposition 5.3 (Equivalence of Batch and Recursive Estimation)** *At step  $j = J$ , the recursive posterior  $q(\mathbf{a}|\mathbf{y}_{1:J})$  in (5.9), the recursive predictive posterior  $q(\mathbf{y}_*|\mathbf{y}_{1:J})$  in (5.9) and the product of the recursive marginal likelihood  $\prod_{j=1}^J q(\mathbf{y}_j|\mathbf{y}_{1:j-1})$  in (5.16) formulated in this section for recursive sparse GPs are equivalent to the corresponding distributions for batch sparse GP as discussed in Section 4.2.2.*

### 5.2.6 Kalman Filter Like Updating

By carefully inspecting the recursive posterior moments in (5.10) over the inducing points and introducing the temporary variables  $\mathbf{r}_j$  and  $\mathbf{G}_j$ , lead to the following efficient updates

$$\begin{aligned} \mathbf{r}_j &= \mathbf{y}_j - \mathbf{H}_j\boldsymbol{\mu}_{j-1}; & \boldsymbol{\mu}_j &= \boldsymbol{\mu}_{j-1} + \mathbf{G}_j\mathbf{r}_j; \\ \mathbf{P}_j &= \mathbf{H}_j\boldsymbol{\Sigma}_{j-1}\mathbf{H}_j^T + \mathbf{V}_j; & \boldsymbol{\Sigma}_j &= \boldsymbol{\Sigma}_{j-1} - \mathbf{G}_j\mathbf{P}_j\mathbf{G}_j^T, \\ \mathbf{G}_j &= \boldsymbol{\Sigma}_{j-1}\mathbf{H}_j^T\mathbf{P}_j^{-1}; & & \end{aligned} \quad (5.17)$$

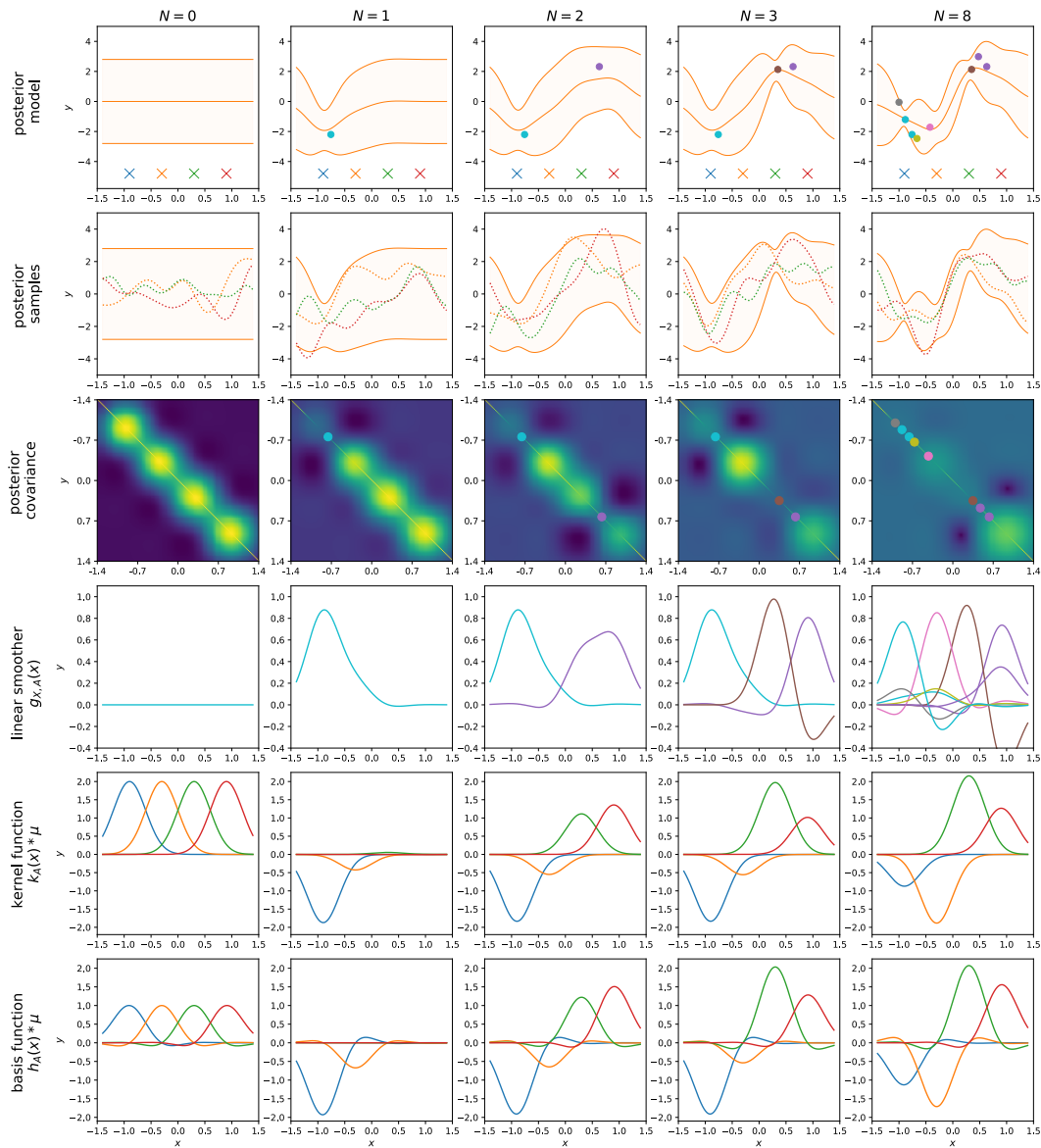


Figure 5.2. Recursive updating for sparse GP models. In the top row, the sequential posterior model is illustrated with the predictive mean and the 95%-credible interval, together with the used data points (dots) and inducing points (crosses). In the second row, the recursive posterior credible intervals with each 3 posterior samples are depicted. In the next row, the induced recursive posterior covariance is depicted together with the training data points. In the bottom 3 rows, the sequential basis functions  $\mathbf{h}_{X,A}(\mathbf{x})$ ,  $\mathbf{k}_A(\mathbf{x})$  and  $\mathbf{g}_A(\mathbf{x})$  are depicted, which are defined analogously to Table 5.1. We notice that recursive sparse GPs can be explained by manipulating sequentially the weights of certain basis functions.

where the recursion starts with  $\boldsymbol{\mu}_0 = \mathbf{0}$  and  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{AA}$ . Computing the posterior moments with these recursive equations in (5.17) yields the same posterior moments as in (5.10) and (5.11). At step  $j = J$ , they are again equivalent to the batch posterior distribution of a sparse GP in (4.21), and the different versions can be obtained by varying the additional noise  $\mathbf{V}_j = \bar{\mathbf{V}}_j + \sigma_n^2 \mathbb{I}$ .

Note that these equations are also known as the update equation of the *Kalman filter* (KF) [Kalman et al., 1960], which is a Bayesian filter for estimating the optimal state distribution of a state space system. A state space system is a compact representation for probabilistically modeling a dynamical system. It represents a more general model as the extended Bayesian linear model by also incorporating transitions as we will exploit in Chapter 7.

### 5.2.7 Information Filter Like Updating

Alternatively, by investigating the recursive posterior moments in (5.11) in more detail, we notice that the corresponding natural moments of the Gaussian distribution reduce to a simply form. In particular, the precision matrix, that is the inverse of the covariance matrix  $\boldsymbol{\Lambda}_j = \boldsymbol{\Sigma}_j^{-1}$ , can be formulated as

$$\boldsymbol{\Lambda}_j = \boldsymbol{\Lambda}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j$$

and similarly the natural mean  $\boldsymbol{\eta}_j = \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j$ , which is the transformed mean by the inverse covariance matrix, is given as

$$\boldsymbol{\eta}_j = \boldsymbol{\eta}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j.$$

Thus, the previous natural mean and previous precision matrix are updated by the additive factors  $\Delta \boldsymbol{\Lambda}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j$  and  $\Delta \boldsymbol{\eta}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j$ , respectively, so that

$$\boldsymbol{\eta}_J = \boldsymbol{\eta}_0 + \sum_{j=1}^J \Delta \boldsymbol{\eta}_j \quad \text{and} \quad \boldsymbol{\Lambda}_J = \boldsymbol{\Lambda}_0 + \sum_{j=1}^J \Delta \boldsymbol{\Lambda}_j.$$

The recursion starts with  $\boldsymbol{\eta}_0 = \mathbf{0}$  and  $\boldsymbol{\Lambda}_0 = \mathbf{K}_{AA}^{-1}$ . At any step  $j$ , the equivalent ordinary posterior moments (5.17) can be obtained by transforming back  $\boldsymbol{\mu}_j = \boldsymbol{\Lambda}_j^{-1} \boldsymbol{\eta}_j$  and  $\boldsymbol{\Sigma}_j = \boldsymbol{\Lambda}_j^{-1}$ . These equations in the natural space are known as the update equations of the *information filter* (IF) [Kalman et al., 1960], which can be equivalently used as the KF for estimating the optimal filtered state distribution of a state space system.

### 5.2.8 Recursive Variational Inference

In this section, we provide a more theoretical perspective to the model presented in the previous sections by applying recursive variational inference. This also leads to a lower bound of the log marginal likelihood, which can be used for stochastic training of the hyperparameters, as it will be discussed in Chapter 6.

In the previous section, the true recursive conditional distribution  $p(\mathbf{f}_j|\mathbf{a})$  is directly approximated by  $q(\mathbf{f}_j|\mathbf{a}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}, \tilde{\mathbf{V}}_j)$  in (5.5) with covariance  $\tilde{\mathbf{V}}_j$  according to Table 5.2. This is then combined with exact Bayesian inference as discussed in the previous sections. In this section instead, the exact recursive training conditional  $p(\mathbf{f}_j|\mathbf{a}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}, \mathbf{K}_{\mathbf{x}_j\mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j\mathbf{x}_j})$  is kept for which approximate inference is applied. In particular, we use variational inference, similarly discussed in Section 4.2.3.1 for the batch case, however, here we apply it recursively. Concretely, the forward KL between a recursive approximate distribution  $q_j(\mathbf{f}_j, \mathbf{a})$  and the exact recursive posterior distribution  $p(\mathbf{f}_j, \mathbf{a}|\mathbf{y}_{1:j})$  is minimized, that is,

$$\arg \min_{q_j(\mathbf{f}_j, \mathbf{a})} \text{KL}[q(\mathbf{f}_j, \mathbf{a})||p(\mathbf{f}_j, \mathbf{a}|\mathbf{y}_{1:j})]. \quad (5.18)$$

Similar to Section 4.2.3.1, we assume the following structure of the approximate variational distribution

$$q_j(\mathbf{f}_j, \mathbf{a}) = p(\mathbf{f}_j|\mathbf{a}) q_j(\mathbf{a}), \quad (5.19)$$

where the true conditional  $p(\mathbf{f}_j|\mathbf{a}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}, \mathbf{K}_{\mathbf{x}_j\mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j\mathbf{x}_j})$  is used, so that only the variational distribution over the inducing points  $q_j(\mathbf{a}) \approx p(\mathbf{a}|\mathbf{y}_{1:j})$  is implicitly used. Instead minimizing (5.18) directly, a recursive variational lower bound to the recursive marginal likelihood

$$\log q(\mathbf{y}_j|\mathbf{y}_{1:j-1}) \geq \mathcal{L}(q_j(\mathbf{f}_j, \mathbf{a})) = \mathcal{L}(q_j(\mathbf{a})) \quad (5.20)$$

can be maximized as discussed in Proposition 4.1 for the batch case. In the recursive case, this can be analytically done as shown in the next Proposition with Proof given in the Appendix B.1.

**Proposition 5.4 (Recursive Variational Lower Bound Maximization; Pr. B.1)**  
*Maximizing recursively a sequential lower bound to the recursive marginal likelihood (5.20), that is,*

$$q_j^*(\mathbf{a}) = \arg \max_{q_j(\mathbf{a})} \mathcal{L}(q_j(\mathbf{a})),$$

lead to the optimal recursive distribution

$$q_j^*(\mathbf{a}) = \mathcal{N}\left(\mathbf{a} \mid \frac{1}{\sigma_n^2} \boldsymbol{\Sigma}_j \mathbf{H}_j^T \mathbf{y}_j, \left(\boldsymbol{\Sigma}_{j-1}^{-1} + \frac{1}{\sigma_n^2} \mathbf{H}_j^T \mathbf{H}_j\right)^{-1}\right) = \mathcal{N}(\mathbf{a} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j).$$

Moreover, the corresponding optimal lower bound is given as

$$\mathcal{L}(q_j^*(\mathbf{a})) = \log \mathcal{N}(\mathbf{y}_j \mid \mathbf{H}_j \boldsymbol{\mu}_{j-1}, \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \sigma_n^2 \mathbb{I}) - \frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{\mathbf{x}_j \mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j \mathbf{x}_j}].$$

A proof is given in Proof B.1.

This recursive lower bound has a similar structure as the fully collapsed lower bound in (4.40). We recognize the log of the marginal likelihood in (5.15) with an approximate training covariance  $\bar{\mathbf{V}}_j = \mathbf{0}$  plus a correction term involving the trace of the true training conditional covariance  $\mathbf{K}_{\mathbf{x}_j \mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j \mathbf{x}_j}$ . Differently to the fully collapsed lower bound in (4.40), however, the recursive structure of the recursive bound leads to a sum of  $J$  terms. In particular, using the factorization of the log marginal likelihood in (5.16) together with (5.20), the overall log marginal likelihood can be lower bounded by the sum of the  $J$  optimal variational lower bounds.

**Proposition 5.5 (Recursive Collapsed Lower Bound)** *The sum of the recursive lower bounds in Proposition 5.4 is a lower bound to the overall log marginal likelihood (5.16), that is,*

$$\log q(\mathbf{y}) \geq \mathcal{L}_{\text{REC}}(q_1^*(\mathbf{a}), \dots, q_J^*(\mathbf{a})) = \sum_{j=1}^J \mathcal{L}(q_j^*(\mathbf{a})), \quad (5.21)$$

which we call the **recursive collapsed** lower bound to the log marginal likelihood.

### 5.2.8.1 Generalized Recursive Collapsed Lower Bound

The recursive variational approach in the previous Section 5.2.8 can be generalized to a range of sparse batch GP models. In particular, using the generic training covariance  $\bar{\mathbf{V}}_j$  according to Table 5.2, yields the posterior distribution  $p(\mathbf{a} \mid \mathbf{y}_{1:j}) = \mathcal{N}(\mathbf{a} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$  in (5.10). Further, the recursive log of the marginal likelihood  $\log q(\mathbf{y})$  in (5.15) can be adapted by a positive correction term  $a_j \in \mathbb{R}_+$  as indicated in the first column of Table 5.2, that is,

$$\Psi^{(J)} = \sum_{j=1}^J \log q(\mathbf{y}_j \mid \mathbf{y}_{1:j-1}) - a_j = \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j \mid \mathbf{H}_j \boldsymbol{\mu}_{j-1}, \mathbf{P}_j) - a_j. \quad (5.22)$$

We refer to this as the *generalized recursive collapsed bound* and  $\Psi^{(J)}$  decomposes into a sum of  $J$  terms  $\psi_j$  so that  $\Psi^{(J)} = \sum_{j=1}^J \psi_j$  with  $\psi_j = \log \mathcal{N}(\mathbf{y}_j | \mathbf{H}_j \boldsymbol{\mu}_{j-1}, \mathbf{P}_j) - a_j$ . This recursive computation together with the model specific quantities  $\bar{\mathbf{V}}_j$ ,  $\bar{\mathbf{V}}_*$  and  $a_j$  in Table 5.2 yields the equivalent posterior distribution, predictive distribution and marginal likelihood as the corresponding batch sparse models as discussed in Section 4.2.

The main advantage of the application of recursive variational inference are the recursive collapsed bounds in Proposition 5.4 and 5.5, which decompose into a sum of additive terms. As a consequence, it can be used for stochastic training of the hyperparameters as it will be discussed in Chapter 6.

### 5.2.9 Examples

In this Section, we discuss some examples to illustrate the connection between the batch sparse GP model and the proposed recursive estimation approach, which allows to train sparse GP models analytically either in online or distributed setting for fixed hyperparameters.

**Example 5.2 (Full Batch Setting)** *In this example, we consider an illustrative example with  $N = 100$  data samples in  $D = 1$  for which the batch sparse GP solution is discussed. In particular, we consider a PEP model with  $\alpha = 0.5$  with  $M = 15$  inducing points as introduced in Section 4.2.3.3 with the corresponding lower bound  $\mathcal{L}_{PEP}$  in (4.53). We assume that the data is available as full batch and the computational resources are large enough to fit this model. In Figure 5.3, the  $N = 100$  data samples  $\{\mathbf{x}_i, y_i\}_{i=1}^J$  with  $\mathbf{x}_j, y_j \in \mathbb{R}$  (red dots), the predictive mean and 95%-credible interval of full GP (black dotted lines) as well as the mean (blue solid line), the 95%-credible interval (blue shaded area) together with the  $M = 15$  equidistantly placed inducing inputs  $\mathbf{A} \in \mathbb{R}^{15}$  with corresponding outputs  $\mathbf{a} \in \mathbb{R}^{15}$  (black dots) of the sparse batch model are depicted. For illustration purposes, a slightly smaller than optimal lengthscale was selected. The numbers in the left and right corner indicate the lower bound to the marginal log likelihood in (4.53) and its derivative with respect to the lengthscale, respectively.*

**Example 5.3 (Online Setting)** *We continue the Example 5.2, however, in this example we assume that the data samples  $\{\mathbf{x}_j, y_j\}_{j=1}^J$  with  $\mathbf{x}_j, y_j \in \mathbb{R}$  arrive online in a stream of  $J = 100$  blocks with batch size  $B = 1$ . Therefore, we have the basis function matrix  $\mathbf{H}_j = \mathbf{K}_{\mathbf{x}_j \mathbf{A}} \mathbf{K}_{\mathbf{A} \mathbf{A}}^{-1} \in \mathbb{R}^{1 \times 15}$  and the prior covariance  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{\mathbf{A} \mathbf{A}} \in \mathbb{R}^{15 \times 15}$ .*

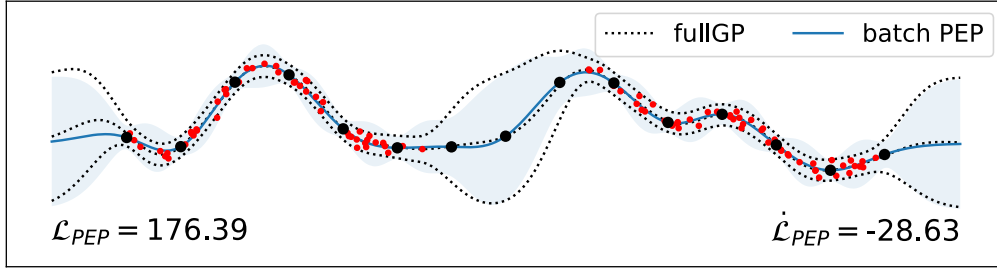


Figure 5.3. Full GP and batch sparse GP regression with PEP model ( $\alpha = 0.5$ ).

The sparse GP model in the previous example corresponds to PEP with  $\alpha = 0.5$ , thus according to Table 5.2 the recursive training covariance is  $\bar{\mathbf{V}}_j = \alpha(\mathbf{K}_{\mathbf{x}_j, \mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j, \mathbf{x}_j}) \in \mathbb{R}$  and test variance  $\bar{\mathbf{V}}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{x}_*} - \mathbf{Q}_{\mathbf{x}_* \mathbf{x}_*}$  together with the marginal likelihood correction  $a_j = \frac{1-\alpha}{2\alpha} \log\left(1 + \frac{\alpha}{\sigma_n^2}(\mathbf{K}_{\mathbf{x}_j, \mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j, \mathbf{x}_j})\right) \in \mathbb{R}$ . We apply the KF like update equations (5.17), in particular, the initial state moments are initialized to  $\boldsymbol{\mu}_0 = \mathbf{0} \in \mathbb{R}^{15}$  and  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{AA} \in \mathbb{R}^{15 \times 15}$  and recursively updated. This is done by computing the residual  $\mathbf{r}_j = \mathbf{y}_j - \mathbf{H}_j \boldsymbol{\mu}_{j-1} \in \mathbb{R}$ , the variance  $\mathbf{P}_j = \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \mathbf{V}_j \in \mathbb{R}$  and the Kalman gain  $\mathbf{G}_j = \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T \mathbf{P}_j^{-1} \in \mathbb{R}^{15 \times 1}$ , followed by the recursive moment updates

$$\begin{aligned} \boldsymbol{\mu}_j &= \boldsymbol{\mu}_{j-1} + \mathbf{G}_j \mathbf{r}_j \in \mathbb{R}^{15} \\ \boldsymbol{\Sigma}_j &= \boldsymbol{\Sigma}_{j-1} - \mathbf{G}_j \mathbf{P}_j \mathbf{G}_j^T \in \mathbb{R}^{15 \times 15}, \end{aligned} \quad (5.23)$$

where the previous mean is updated by  $\Delta \boldsymbol{\mu}_j = \mathbf{G}_j \mathbf{r}_j$  and the covariance reduced by  $\Delta \boldsymbol{\Sigma}_j = -\mathbf{G}_j \mathbf{P}_j \mathbf{G}_j^T$ . For test data  $\mathbf{X}_*$ , predictions at any time point can be obtained by applying (5.14) with  $\mathbf{H}_* = \mathbf{K}_{\mathbf{x}_* A} \mathbf{K}_{AA}^{-1}$  and  $\mathbf{V}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{x}_*} - \mathbf{Q}_{\mathbf{x}_* \mathbf{x}_*}$  yielding the predictive mean and 95%-credible intervals as illustrated in 5.4 with the blue solid line and blue shaded area, respectively. After processing all  $N$  samples, the predictive distribution correspond to the batch version in Figure 5.3. Moreover, the value of the generalized collapsed lower bound  $\Psi^{(j)}$  in (5.22) is depicted in the lower left corner (together with its derivatives in the right corner, which will be discussed in Section 6.2.2). We note that also the cumulative lower bound at step  $j = 100$  is equal to the corresponding batch version in Figure 5.3.

**Example 5.4 (Distributed Setting)** We continue the Examples 5.2 and 5.3, however, here we assume that the data is available in a full batch, but we want to distribute the computations among  $J = 4$  computational nodes, thus we split the data into four blocks of size  $B = 25$  as illustrated in Figure 5.5. Note that the



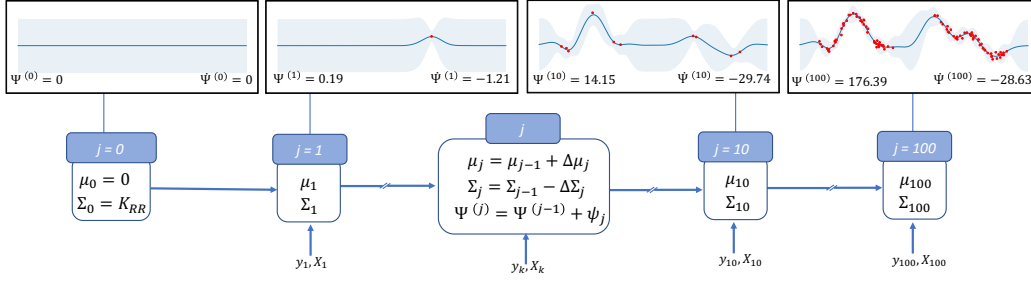


Figure 5.4. Online updating for the toy example in Example 5.3 with batch size  $B = 1$ .

samples are only sorted for illustration purposes but the splits of the data into the blocks can be arbitrary. We consider the Information Filter like updates in Section 5.2.7, where the natural moments  $\boldsymbol{\eta}_j$  and  $\boldsymbol{\Lambda}_j$  are computed. In this example, we have the sequential basis function matrix  $\mathbf{H}_j = \mathbf{K}_{X_j A} \mathbf{K}_{AA}^{-1} \in \mathbb{R}^{25 \times 15}$ , the diagonal training covariance  $\bar{\mathbf{V}}_j = \alpha \text{Diag}[\mathbf{K}_{x_j x_j} - \mathbf{Q}_{x_j x_j}] \in \mathbb{R}^{25 \times 25}$  and the prior covariance  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{AA} \in \mathbb{R}^{15 \times 15}$ , leading to the initial precision matrix  $\boldsymbol{\Lambda}_0 = \mathbf{K}_{AA}^{-1}$ . Each node computes in parallel

$$\Delta \boldsymbol{\eta}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j \in \mathbb{R}^{15 \times 1} \quad \text{and} \quad \Delta \boldsymbol{\Lambda}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j \in \mathbb{R}^{15 \times 15}.$$

Afterwards, the individual natural posterior moments are summed at a central node

$$\boldsymbol{\eta}_J = \sum_{j=1}^4 \Delta \boldsymbol{\eta}_j \quad \text{and} \quad \boldsymbol{\Lambda}_J = \boldsymbol{\Lambda}_0 + \sum_{j=1}^4 \Delta \boldsymbol{\Lambda}_j,$$

which can be transformed back to the original posterior moments  $\boldsymbol{\Sigma}_J = \boldsymbol{\Lambda}_J^{-1}$  and  $\boldsymbol{\mu}_J = \boldsymbol{\Sigma}_J \boldsymbol{\eta}_J$ . Prediction can be done according to (5.14) with  $\mathbf{H}_* = \mathbf{K}_{X_* A} \mathbf{K}_{AA}^{-1}$  and  $\mathbf{V}_* = \mathbf{K}_{X_* X_*} - \mathbf{Q}_{X_* X_*}$ . This procedure is illustrated in Figure 5.5 and we note that the aggregated posterior distribution in the end equals the batch sparse solution depicted in Figure 5.3.

### 5.2.10 Transformed Basis Functions

If only the recursive predictive distribution (5.13) and the marginal likelihood (5.15) are of particular interest and not the recursive posterior distribution of the inducing points, there are several choices for the sequential basis function matrices  $\mathbf{H}_j = \mathbf{K}_{X_j A} \mathbf{K}_{AA} \in \mathbb{R}^{B \times M}$ ,  $\mathbf{H}_* = \mathbf{K}_{X_* A} \mathbf{K}_{AA}$  and the prior covariance  $\boldsymbol{\Sigma}_0 = \mathbf{K}_{AA} \in \mathbb{R}^{M \times M}$ , as discussed in Proposition 5.1 and summarized in Table 5.1. For

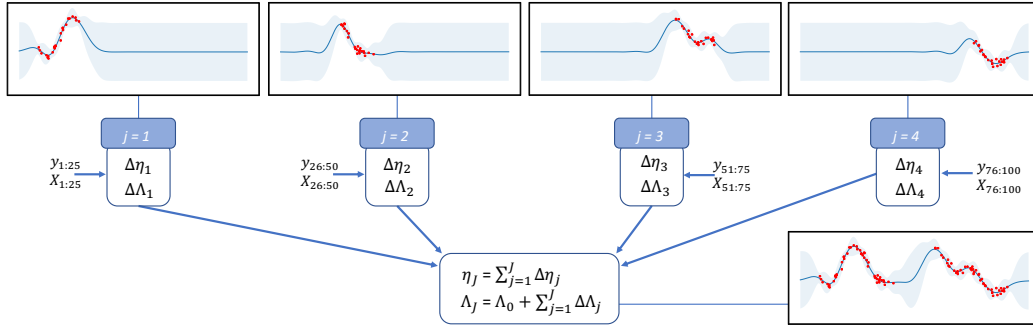


Figure 5.5. Distributed computation among  $J = 4$  computational nodes for the toy example 5.4 with block size  $B = 25$ .

instance, using the basis functions and prior covariance indicated in the second row, that is,  $\tilde{\Sigma}_0 = \mathbf{K}_{AA}^{-1}$  and  $\tilde{\mathbf{H}}_j = \mathbf{K}_{X_jA}$  together with  $\tilde{\mathbf{H}}_* = \mathbf{K}_{X_*A}$ , leads to the equivalent predictive distribution and marginal likelihood. Implicitly, thereby the transformed posterior moments  $\tilde{\boldsymbol{\mu}}_j = \mathbf{K}_{AA}^{-1}\boldsymbol{\mu}_j$ ,  $\tilde{\boldsymbol{\Sigma}}_j = \mathbf{K}_{AA}^{-1}\boldsymbol{\Sigma}_j\mathbf{K}_{AA}^{-1}$  are computed with reverse mapping  $\boldsymbol{\mu}_j = \mathbf{K}_{AA}\tilde{\boldsymbol{\mu}}_j$ ,  $\boldsymbol{\Sigma}_j = \mathbf{K}_{AA}\tilde{\boldsymbol{\Sigma}}_j\mathbf{K}_{AA}$ , respectively. This parametrization constitutes a computational shortcut, since the basis functions are very easy to interpret and do not include any matrix multiplication. Similarly, for other choices of basis functions and prior covariance matrix as summarized in Table 5.1 lead to different transformed posterior moments.

### 5.2.11 Connection to Dynamical Systems

The extended Bayesian linear model with Gaussian observation noise as discussed in this chapter for sparse GPs, that is,

$$\begin{aligned} \mathbf{y}_j &= \mathbf{f}_j + \boldsymbol{\varepsilon}_j; \\ \mathbf{f}_j &= \mathbf{H}_j\mathbf{a} + \boldsymbol{\gamma}_j, \end{aligned}$$

shares similarities with a Gaussian dynamical state space systems. Thereby, the state of the system corresponds to the global inducing points  $\mathbf{a}$ . However, a dynamical system can represent more general models by varying the state over time described by a dynamic of the state. In the context of sparse GPs, this corresponds to the following model

$$\begin{aligned} \mathbf{y}_j &= \mathbf{f}_j + \boldsymbol{\varepsilon}_j; \\ \mathbf{f}_j &= \mathbf{H}_j\mathbf{a}_j + \boldsymbol{\gamma}_j; \\ \mathbf{a}_j &= \mathbf{F}_j\mathbf{a}_{j-1} + \mathbf{q}_j, \end{aligned}$$

where the next state  $\mathbf{a}_j$  depends on the previous state  $\mathbf{a}_{j-1}$  by some transition matrix  $F_j$  and noise of the dynamics  $q_j$ . The time varying inducing points  $\mathbf{a}_j$  can represent either global inducing points but indexed with different hyperparameters, which change over time (for which an approach is outlined in Section 8.2) or local inducing points associated with the data  $\mathcal{D}_j$ , as thoroughly discussed in Chapter 7, consider particularly Section 7.2.6. These connections open up a range of new models for sparse GPs.

### 5.3 Summary of Contributions

This Chapter 5 is based on the first part of the paper "*Recursive Estimation for Sparse Gaussian Process Regression*" [Schürch et al., 2020], with focus on Bayesian recursive estimation for **known** hyperparameters. On the other hand, estimation procedures for the hyperparameters as discussed in the second part of [Schürch et al., 2020] are discussed in Chapter 6. Thereby, the content presented here in this thesis is an extended and slightly adapted version from [Schürch et al., 2020]. In particular, Section 5.1 about the "*Extended Bayesian Linear Model*" and the subsections about the recursive inference in Section 5.2 "*Recursive Sparse GP Model*" include many more explanations and details which make some connections more clearer without changing the main messages significantly. In the following, we summarize the novel scientific contributions contained in this chapter:

- **Extended Bayesian Linear Model:** We extend the usual Bayesian linear regression model by introducing an additional input-varying noise and establish a connection to a class of known sparse GP regression models as formulated in Propositions 5.1 and 5.2. This unifying view explains the difference between sparse GPs and Bayesian linear regression and the insights can be used for applying recursive Bayesian inference.
- **Recursive Bayesian Inference for Sparse GPs:** Based on the extended Bayesian linear model, we show how to analytically apply recursive Bayesian inference for the inducing points. In particular, the posterior distribution after considering all recursive updates equals the batch solution of sparse GPs as stated in Proposition 5.3.
- **Online Training:** The recursive character of our model allows to train sparse GP models in an online fashion for fixed hyperparameters. This means, the posterior belief of the model can be analytically updated by a new mini-batch of sequentially arriving data.

- **Distributed Training:** The Gaussian posterior moments of the inducing points in the natural parameter space decomposes into a sum of independent terms, which can be efficiently used for parallel computations in a distributed setting.
- **Recursive Collapsed Bound:** We apply recursive variational inference, yielding the recursive collapsed bound (Proposition 5.4 and 5.5), which decomposes into a sum of additive terms. As a consequence, it can be used for stochastic training of the hyperparameters as it will be discussed in Chapter 6.
- **Connection to Dynamical State Space Systems:** We show the connections between inference in sparse GPs and inference in state space systems, for instance by using the Kalman Filter and Information Filter. These insights can be further exploited in Chapter 7.

## Chapter 6

# Sequential Hyperparameter Learning for Sparse Gaussian Processes

In this chapter, we propose two novel methods for sequential hyperparameter learning for sparse GPs. They correspond to inference at level II in a Bayesian hierarchical model, as opposed to the previous Chapter 5, where the focus was on inference for the parameters on level I, that is, inference for the global inducing points with fixed hyperparameters. In particular, we provide briefly the necessary background including state-of-the-art methods for hyperparameter learning for sparse GPs in Section 6.1. Subsequently, we present in Section 6.2 and 6.3 the two novel approaches for sequential hyperparameter learning for a range of sparse GP models. The first approach is based on propagating analytically the gradients, so that the correlations between the mini-batches are taken into account. On the other hand, the second approach, is based on an independence assumptions for the mini-batches, resulting in a very efficient and scalable method for hyperparameter learning. Both methods are extensively investigated on several synthetic, as well as real world datasets, and compared to the state-of-the-art methods for sequential hyperparameter learning for sparse GPs. In Section 6.4, we summarize our scientific contributions of the presented content of this chapter in detail.

### 6.1 Background

In the previous Chapter 5, the focus was on inference for the parameters on level I, that is, inference for the global inducing points with fixed hyperparameters. In particular, we introduced algorithms based on recursive Bayesian inference, which can be used to train a range of well known sparse GP regression models

analytically either in an online or distributed setting based on mini-batches of data  $\mathcal{D}_j$ , where the full data batch is  $\mathcal{D} = \bigcup_{j=1}^J \mathcal{D}_j$ . In this chapter, however, we propose novel approaches for inference at level II, that is, estimating the hyperparameters  $\theta$  of sparse GP models. Beside full batch training, we distinguish between the *online*, *sequential* and *distributed* setting for hyperparameter learning as illustrated in Figure 6.1. In the former two cases, we assume that the data arrives as a stream of mini-batches  $\mathcal{D}_j$ . However, in the online setting, each mini-batch can be only considered once, whereas in the sequential setting, it is allowed to consider each block of data several times. In the distributed setting, we assume that there is a central node and each mini-batch is associated with a computational node working in parallel. In this section we focus on hyperparameter estimation in the sequential setting, however, we outline also a distributed and online approach.

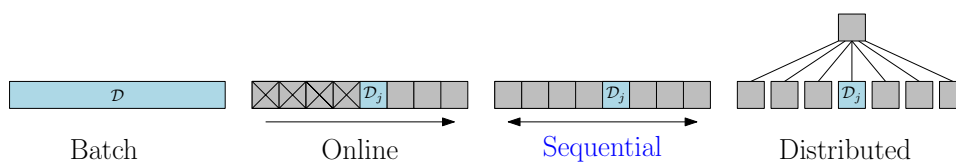


Figure 6.1. Different ways to train global sparse GP models.

In Table 6.1, we summarize different training approaches for level I and II inference for global sparse GP regression models. As discussed in the previous chapters, inference for the inducing points  $\mathbf{a}$  can be performed analytically for the full batch case (Section 4.2), as well as for the online and distributed setting, discussed in Sections 5.2.6 and 5.2.7, respectively. However, for level II inference of the hyperparameters  $\theta$ , we have to resort even in the batch case to numerical approximations as discussed in Sections 2.3.3.1 and 4.2.4. Thereby, the inference is based on numerically optimizing a loss function, as described in Section 2.1.3. In particular, in the full batch case, this is based on optimizing deterministically a loss function involving the full batch of data  $\mathcal{D}$ . However, for big data, where the number of samples  $N = |\mathcal{D}|$  can be in the order of a million, keeping all data in memory is not possible or the data might even arrive sequentially. In order to speed up the optimization part, the general idea is to update the hyperparameters more frequently for a subset of data  $\mathcal{D}_j$  instead the whole dataset  $\mathcal{D}$ . This is often done by *stochastic* optimization, which can be applied if the loss function can be decomposed into a sum of loss functions depending only on the blocks of data  $\mathcal{D}_j$ , for which more details are given in Section 2.1.3.2. In the remainder of this chapter, we first provide a brief summary of the state-of-the-art for hyperparameter optimization of global sparse GPs in Section 6.1.1. In

	Batch	Online	Sequential	Distributed
Level I $a$	analytic Bayesian inference Section 4.2	analytic recursive Bayesian inference Section 5.2.6	not needed; same as online	analytic recursive Bayesian inference Section 5.2.7
Level II $\theta$	deterministic optimization Section 4.2.4	deterministic optimization [Bui et al., 2017a]	stochastic optimization Chapter 6	deterministic optimization Section 6.3.7

Table 6.1. Overview of level I and II inference approaches for global sparse GP regression models. In this Chapter, we focus on the sequential hyperparameters optimization with stochastic optimization on level II. In Section 8.2, we outline also an approach for online hyperparameter estimation.

the subsequent Sections 6.2 and 6.3, we propose two new sequential hyperparameter estimation approaches for sparse GPs and conclude the work in Section 8.2.

### 6.1.1 State of the Art

We briefly discuss the state-of-the-art for hyperparameter learning in sparse GPs. Since the most stochastic approaches are based on the VFE model [Titsias, 2009], as introduced in Section 4.2.3.1, we focus here on this model, however, our method is more general and will be introduced for a range of sparse GP models. We first recall the corresponding batch method based on deterministic optimization, followed by the discussion about the state-of-the-art method using sequential/stochastic optimization.

#### 6.1.1.1 Deterministic Variational Inference

We recall the *variational free energy (VFE)* or the *collapsed variational lower bound* in (4.40), that is,

$$\mathcal{L}_{VFE}(\theta) = \log \mathcal{N}(\mathbf{y} | \mathbf{0}, \mathbf{Q}_{XX}^\theta + \sigma_n^2) - \frac{1}{\sigma_n^2} \text{tr}[\mathbf{K}_{XX}^\theta - \mathbf{Q}_{XX}^\theta],$$

which corresponds to the log marginal likelihood of a *deterministic "GP"* with training conditional  $\tilde{\mathbf{V}} = \mathbf{0}$ , as defined in (4.24), minus an additional term which is the trace of the true training conditional covariance  $\mathbf{K}_{XX}^\theta - \mathbf{Q}_{XX}^\theta$ . Note that, this lower bound  $\mathcal{L}_{VFE}(\theta)$  depends implicitly on the hyperparameters  $\theta$  through  $\mathbf{Q}_{XX}^\theta = \mathbf{K}_{XA}^\theta (\mathbf{K}_{AA}^\theta)^{-1} \mathbf{K}_{AX}^\theta$  and  $\mathbf{K}_{XX}^\theta$ . This bound can be optimized deterministically as described in Section 2.1.3.1, involving the full training dataset  $\mathcal{D} = \{\mathbf{y}, \mathbf{X}\}$ . This constitutes a principled way to obtain good estimates  $\theta^*$  for moderate sample size  $N = |\mathcal{D}|$ , whereby, the order of training samples depends on the number

of inducing points  $M$  and the available computational resources, but it is typically around  $N \approx 20'000 - 50'000$ . This bound is called *collapsed variational lower bound*, since the optimal variational posterior distribution (4.39), which is analytically available, is plugged into (4.38), which collapses the bound. This has the effect, that global dependencies between all training data samples in  $\mathcal{D}$  are introduced, which makes this bound unsuitable for stochastic training.

### 6.1.1.2 Stochastic Variational Inference

The collapsed lower bound above requires to process the whole dataset  $\mathcal{D}$  in a batch sense, which is very inefficient and not feasible for large  $N$ . The aim is to update the hyperparameters in the optimization more frequently based only on a subset of data  $\mathcal{D}_j = \{\mathbf{y}_j, \mathbf{X}_j\}$ , known as stochastic optimization as discussed in Section 2.1.3.2. However, this requires a loss function, which decomposes into a sum of loss functions only depending on the mini-batch  $\mathcal{D}_j$ . For this setting, an elegant solution is proposed by Hensman et al. [2013] with *Stochastic Variational Gaussian Process* (SVGP), where they applied stochastic optimization to an *uncollapsed lower bound* to the log marginal likelihood, in particular,

$$\mathcal{L}_{SVGP}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\theta}) = \sum_{j=1}^J \mathbb{E}_{q(\mathbf{a})} p_{\theta}(f_j|\mathbf{a}) [p_{\theta}(\mathbf{y}_j|\mathbf{f}_j)] - \text{KL}[q(\mathbf{a})||p_{\theta}(\mathbf{a})]. \quad (6.1)$$

Note that this bound is explicitly parametrized by the approximate posterior moments  $\boldsymbol{\mu}, \boldsymbol{\Sigma}$  of the variational distribution  $q(\mathbf{a}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ , which means that there is no analytic solution for the Gaussian posterior distribution of the inducing points  $\mathbf{a}$ , instead all entries in the mean vector  $\boldsymbol{\mu}$  and the covariance matrix  $\boldsymbol{\Sigma}$  have to be estimated numerically. This uncollapsed bound satisfies  $\mathcal{L}_{SVGP}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\theta}) \leq \mathcal{L}_{VFE}(\boldsymbol{\theta})$ , with equality when inserting the optimal mean and covariance of the variational distribution of the batch VFE (4.39). The key property of this bound is that it can be written as a sum of  $J$  terms, which allows *stochastic variational inference* (SVI, [Hoffman et al., 2013]). Note that collapsing the bound, i.e. inserting the optimal distribution, reintroduces dependencies between the observations, and eliminates the global parameter  $\mathbf{a}$  which is needed for SVI. For this reason, all variational parameters are numerically estimated by following the noisy gradients of a stochastic estimate of the lower bound  $\mathcal{L}_{SVGP}$ . By passing through the training data a sufficient number of times, the variational distribution converges to the batch solution of VFE method. It allows to apply approximate GP inference to training samples in the order of millions of data. However, this approach requires a large number of parameters: in addition to the  $T$



Method	Symbol	Equation	Reference	Stochastic Optimization	Analytic Posterior	Number of Hyperparameters
collapsed variational lower bound	$\mathcal{L}_{VFE}$	(4.40)	Titsias [2009]	×	✓	$\mathcal{O}(MD + T)$
uncollapsed variational lower bound	$\mathcal{L}_{SVGP}$	(6.1)	Hensman et al. [2013]	✓	×	$\mathcal{O}(MD + M^2 + T)$
recursive collapsed lower bound	$\mathcal{L}_{REC}$	(5.21)	Schürch et al. [2020]	✓	✓	$\mathcal{O}(MD + T)$
hybrid lower bound	$\mathcal{L}_{IA \rightarrow PP}$	(6.9)	Kania et al. [2021]	✓	✓	$\mathcal{O}(MD + T)$

Table 6.2. Different methods for estimating the hyperparameters of sparse GP models. The former two correspond to state-of-the-art methods, whereas the latter two are our novel proposed algorithm described in Sections 6.2 and 6.3, respectively. As a reminder about the notation,  $T$  is the number of kernel hyperparameters,  $M$  is the number of inducing points, and  $D$  is the data dimension.

kernel hyperparameters, all entries in the mean vector  $\boldsymbol{\mu}$  and the covariance matrix  $\boldsymbol{\Sigma}$  have to be estimated numerically, which is in order  $\mathcal{O}(M^2 + MD + T)$ . Note that, in particular the quadratic number  $\mathcal{O}(M^2)$  of parameters to be numerically optimized is limiting, and we will propose new methods with less parameters to be optimized, as indicated in the last row of Table 6.2.

An improved version of this algorithm has been proposed by Shi et al. [2020] with their work about *sparse orthogonal variational* GP inference (SOLVEGP), which allows to use more inducing points at the same computational complexity. The main idea is to decompose the prior as the sum of a low-rank approximation using the global inducing points, and a full-rank residual process parametrized by a second orthogonal set of inducing points. As a consequence, the method can model richer functions and the resulting lower bound is tighter with respect to the collapsed lower bound (4.40), which makes this method even more scalable as the standard stochastic variational approach SVGP.

## 6.2 Recursive Gradient Propagation

In the previous Chapter 5, we presented an analytic and online procedure for training sparse GP models at inference level I, that is, inferring the posterior distribution of the inducing points  $\mathbf{a}$  with fixed hyperparameters  $\boldsymbol{\theta}$ . In this section, we focus on sequential inference for the hyperparameters  $\boldsymbol{\theta}$  at level II, for which we present a novel approach based on recursive gradient propagation. In particular, we note that the *recursive collapsed bound* in (5.21)

$$\mathcal{L}_{REC}(\boldsymbol{\theta}) = \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j | \mathbf{H}_j^\theta \boldsymbol{\mu}_{j-1}^\theta, \mathbf{P}_j^\theta) - \frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{\mathbf{x}_j, \mathbf{x}_j}^\theta - \mathbf{Q}_{\mathbf{x}_j, \mathbf{x}_j}^\theta],$$

decomposes into a recursive sum over the mini-batches  $\mathcal{D}_j = \{\mathbf{y}_j, \mathbf{X}_j\}$ , which allows to optimize the hyperparameters  $\boldsymbol{\theta}$  sequentially as opposed to the collapsed bound (4.40). Similarly to the uncollapsed bound of SVGP in (6.1), our recursive collapsed bound (5.21) enables the application of stochastic optimization, however, due to the recursive character of our bound, the posterior distribution over the inducing points is analytically available. As a consequence, the entries in the posterior mean vector and covariance matrix do not have to be estimated numerically, which reduces the number of parameters to be numerically estimated drastically from  $\mathcal{O}(MD + M^2 + T)$  to  $\mathcal{O}(MD + T)$  and makes the bound tighter compared to the collapsed lower bound. Since the number  $M$  of inducing points determines the quality of the approximation to full GP, this reduction in number of parameters from  $M^2$  to  $M$  is crucial and results in more accurate and faster convergence than state-of-the-art approaches such as SVGP. For example, in the application to learn the input output behavior of a non-linear plant, as it will be presented in Section 6.2.5.3, the number of parameters estimated by SVGP is  $\approx 10500$ , while our approach only estimates  $\approx 500$  parameters due to the analytic updates.

### 6.2.1 Generalized Recursive Collapsed Bound

The above recursive collapsed bound corresponds to the VFE model, however, it can be generalized to a range of well-known sparse GP models by varying the training conditional  $\bar{\mathbf{V}}_j^\theta$  in the marginal likelihood covariance  $\mathbf{P}_j^\theta = \mathbf{H}_j^\theta \boldsymbol{\Sigma}_{j-1}^\theta (\mathbf{H}_j^\theta)^T + \bar{\mathbf{V}}_j^\theta + \sigma_n^2 \mathbb{I}$  and using the generalized correction term  $a_j(\boldsymbol{\theta})$  according to Table 5.2, as discussed in Section 5.2. In particular, we recall the *generalized recursive col-*

*lapsed bound* in (5.22), that is,

$$\begin{aligned}\Psi^{(J)}(\boldsymbol{\theta}) &= \sum_{j=1}^J \log q_{\boldsymbol{\theta}}(\mathbf{y}_j | \mathbf{y}_{1:j-1}) - a_j(\boldsymbol{\theta}) \\ &= \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j | \mathbf{H}_j^{\boldsymbol{\theta}} \boldsymbol{\mu}_{j-1}^{\boldsymbol{\theta}}, \mathbf{P}_j^{\boldsymbol{\theta}}) - a_j(\boldsymbol{\theta}) = \sum_{j=1}^J \log \mathcal{N}(\mathbf{r}_j^{\boldsymbol{\theta}} | \mathbf{0}, \mathbf{P}_j^{\boldsymbol{\theta}}) - a_j(\boldsymbol{\theta}),\end{aligned}$$

where we explicitly show the dependencies on  $\boldsymbol{\theta}$ . By introducing new notation  $d_j(\boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{r}_j^{\boldsymbol{\theta}} | \mathbf{0}, \mathbf{P}_j^{\boldsymbol{\theta}})$  and  $\psi_j(\boldsymbol{\theta}) = d_j(\boldsymbol{\theta}) - a_j(\boldsymbol{\theta})$ , the recursive collapsed lower bound can be written as

$$\Psi^{(J)}(\boldsymbol{\theta}) = \sum_{j=1}^J d_j(\boldsymbol{\theta}) - a_j(\boldsymbol{\theta}) = \sum_{j=1}^J \psi_j(\boldsymbol{\theta}).$$

The important property of this recursive collapsed bound is that it decomposes into a sum of  $J$  terms  $\psi_j(\boldsymbol{\theta}) = \psi_j(\boldsymbol{\theta}, \mathcal{D}_j)$ , which only depends on mini-batch  $\mathcal{D}_j = \{\mathbf{y}_j, \mathbf{X}_j\}$ , so that stochastic optimization can be applied as described in Section 2.1.3.2. In particular, maximizing this bound with respect to the hyper-parameters  $\boldsymbol{\theta}$  is equivalent to find a minimizer of the negative objective function, that is,

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta}} -\Psi^{(J)}(\boldsymbol{\theta}, \mathcal{D}) = \arg \min_{\boldsymbol{\theta}} -\sum_{j=1}^J \psi_j(\boldsymbol{\theta}, \mathcal{D}_j),$$

where we show explicitly the dependency on the data. An iterative solution is obtained by following the stochastic negative gradient direction (2.15), that is,

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \gamma_t \left. \frac{\partial \psi_j(\boldsymbol{\theta}, \mathcal{D}_j)}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}, \quad (6.2)$$

with learning rate  $\gamma_t$ , for which we use in this section the particular stochastic optimization method ADAM [Kingma and Ba, 2014]. Note that, usually more than one pass over all mini-batches is required, whereby, we denote one pass over all  $J$  mini-batches as an epoch. Therefore, we use the notation  $\boldsymbol{\theta}^{(e,j)} = \boldsymbol{\theta}^{(t)}$  for  $t = J(e-1) + j$  corresponding to the estimate of  $\boldsymbol{\theta}$  in epoch  $e$  for mini-batch  $j$ .

## 6.2.2 Derivatives of Recursive Collapsed Bound

In order to apply SGD, the stochastic gradient is needed, which can be computed by exploiting the chain rule of derivatives

$$\frac{\partial \psi_j(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial d_j(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} - \frac{\partial a_j(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}},$$

where for instance the derivative of  $d_j(\boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{r}_j^\theta | \mathbf{0}, \mathbf{P}_j^\theta)$  can be further rewritten as

$$\frac{\partial d_j(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\frac{1}{2} \frac{\partial \log |\mathbf{P}_j^\theta|}{\partial \boldsymbol{\theta}} - \frac{1}{2} \frac{\partial (\mathbf{r}_j^\theta)^T (\mathbf{P}_j^\theta)^{-1} \mathbf{r}_j^\theta}{\partial \boldsymbol{\theta}}$$

with  $\mathbf{r}_j^\theta = \mathbf{y}_j - \mathbf{H}_j^\theta \boldsymbol{\mu}_{j-1}^\theta$  and  $\mathbf{P}_j^\theta = \mathbf{H}_j^\theta \boldsymbol{\Sigma}_{j-1}^\theta (\mathbf{H}_j^\theta)^T + \mathbf{V}_j^\theta$ . Those derivatives can be again computed by applying the chain rule for derivatives, for which more details and efficient pseudo-code is given in the Appendix B.2. We want to emphasize that ignoring naively the dependency of  $\boldsymbol{\theta}$  through the previous posterior moments  $\boldsymbol{\mu}_{j-1}^\theta$  and  $\boldsymbol{\Sigma}_{j-1}^\theta$  completely forgets the past and thus results in overfitting the current mini-batch. In order to compute the derivatives of the posterior moments, we exploit the chain rule for derivatives recursively and propagate the gradients of the mean and the covariance over time, that is

$$\begin{aligned} \frac{\partial \boldsymbol{\mu}_j}{\partial \boldsymbol{\theta}} &= \frac{\partial \boldsymbol{\mu}_{j-1}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{G}_j}{\partial \boldsymbol{\theta}} \mathbf{r}_j + \mathbf{G}_j \frac{\partial \mathbf{r}_j}{\partial \boldsymbol{\theta}}, \\ \frac{\partial \boldsymbol{\Sigma}_j}{\partial \boldsymbol{\theta}} &= \frac{\partial \boldsymbol{\Sigma}_{j-1}}{\partial \boldsymbol{\theta}} - \frac{\partial \mathbf{G}_j}{\partial \boldsymbol{\theta}} \mathbf{P}_j \mathbf{G}_j^T - \mathbf{G}_j \frac{\partial \mathbf{P}_j}{\partial \boldsymbol{\theta}} \mathbf{G}_j - \mathbf{G}_j \mathbf{P}_j \frac{\partial \mathbf{G}_j^T}{\partial \boldsymbol{\theta}}, \end{aligned} \quad (6.3)$$

where  $\frac{\partial \mathbf{G}_j}{\partial \boldsymbol{\theta}}$ ,  $\frac{\partial \mathbf{r}_j}{\partial \boldsymbol{\theta}}$  and  $\frac{\partial \mathbf{P}_j}{\partial \boldsymbol{\theta}}$  are computed recursively according to the definitions in (5.17).

**Proposition 6.1 (Recursive Gradient Propagation)** *By recursively propagating the gradients, as described in this section, the batch gradient involving the whole dataset  $\mathcal{D}$  is equivalent to the cumulative gradients, only depending on the individual mini-batch  $\mathcal{D}_j$ , that is,*

$$\frac{\partial \Psi^{(j)}(\boldsymbol{\theta}, \mathcal{D})}{\partial \boldsymbol{\theta}} = \sum_{j=1}^J \frac{\partial \psi_j(\boldsymbol{\theta}, \mathcal{D}_j)}{\partial \boldsymbol{\theta}}.$$

Since we know that the generalized recursively collapsed lower bound  $\Psi^{(j)}(\boldsymbol{\theta}, \mathcal{D})$  in (5.22) corresponds to a range of sparse GP models, we can conclude that also the recursive derivatives obtained by gradient propagation are equivalent to the batch GP models after considering all mini-batches.

The toy example in Fig. 5.4 also shows this equivalence. The numbers in the bottom left and right corners show the cumulative recursive collapsed bound  $\Psi^{(k)}$  and its cumulative derivative  $\frac{\partial \Psi^{(j)}}{\partial l}$  (abbreviated as  $\dot{\Psi}^{(j)}$ ) with respect to the lengthscale  $l$ . The lower bound of the marginal likelihood as well as its derivatives are equivalent to the value of the corresponding batch counterpart in Fig.

5.3. However, note that this requires that the hyperparameters are fixed for all mini-batch updates. In the following, we describe an approach with varying hyperparameters for each mini-batch.

For *Stochastic Recursive Gradient Propagation* (SRGP), in each epoch  $e$  and mini-batch  $j$ , we interleave the SGD update (6.2) of the parameters  $\boldsymbol{\theta}^{(e,j)}$  and the update step of the recursive posterior moments of the inducing points in (5.10), that is, we compute

$$p(\mathbf{a}|\mathbf{y}_{1:j}, \boldsymbol{\theta}^{(e,j)}) \propto p(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}^{(e,j)}) p(\mathbf{a}|\mathbf{y}_{1:j-1}, \boldsymbol{\theta}^{(e,j-1)}).$$

In particular, assume by recursion that the previous posterior  $p(\mathbf{a}|\mathbf{y}_{1:j-1}, \boldsymbol{\theta}^{(e,j-1)})$  and its gradient involving the old hyperparameters  $\boldsymbol{\theta}^{(e,j-1)}$  are available. Those quantities are used to compute the recursive gradients, so that the new hyperparameters  $\boldsymbol{\theta}^{(e,j)}$  can be updated via (6.2). Afterwards, the analytic new posterior  $p(\mathbf{a}|\mathbf{y}_{1:j}, \boldsymbol{\theta}^{(e,j)})$  is computed with (5.10) together with the gradient propagation in (6.3), which then acts as prior information for the next mini-batch. Detailed and efficient pseudo-code is given in Appendix B.2, where we exploited several matrix derivative rules which simplify the computation significantly.

### 6.2.3 Computational Complexity

In the following, we assume that the mini-batch size  $B = |\mathcal{D}_j|$  is larger than the number of inducing points  $M$ . For one mini-batch, the time complexity to update the posterior (5.17) is dominated by matrix multiplications of size  $B$  and  $M$ , thus  $\mathcal{O}(B^2M)$ . In order to propagate the gradients of the posterior and to compute the derivative of the bound needs  $\mathcal{O}(BM^2)$  for a mini-batch and a parameter  $\boldsymbol{\theta}$ . Thus, updating a mini-batch including all  $\mathcal{O}(MD)$  parameters costs  $\mathcal{O}(BM^3D + B^2M)$  for the SRGP method. Since SRGP stores the gradients of the posterior, it requires  $\mathcal{O}(M^3D + BM)$  storage.

On the other hand, SVGP (as discussed in Section 6.1.1.2) needs  $\mathcal{O}(M^2 + BM)$  storage and  $\mathcal{O}(BM^3 + B^2M)$  time per mini-batch, where the latter can be broken down into once  $\mathcal{O}(B^2M)$  and  $\mathcal{O}(BM)$  for each of the  $\mathcal{O}(M^2)$  parameters. This means, for moderate dimensions, our algorithm has the same time complexity as state-of-the-art method SVGP. However, due to the analytic updates of the posterior we achieve a higher accuracy and less epochs are needed as shown in Fig. 6.2 and in Sect. 6.2.5 empirically. Figure 6.2 shows the convergence of SRGP on a 1-D toy example with  $N = 1000$  data samples and  $M = 15$  inducing points. The parameters are sequentially optimized with our recursive approach (blue)

and as comparison with SVGP (green) with a mini-batch size of  $B = 100$  over several epochs. The root-mean-squared-error (RMSE) computed on test points, the bound of the log marginal likelihood (LML) as well as the hyper-parameters converge in a few iterations to the corresponding batch values of VFE (red). Due to the analytic updates of the posterior, the accuracy is higher and SRGP needs much less epochs until convergence.

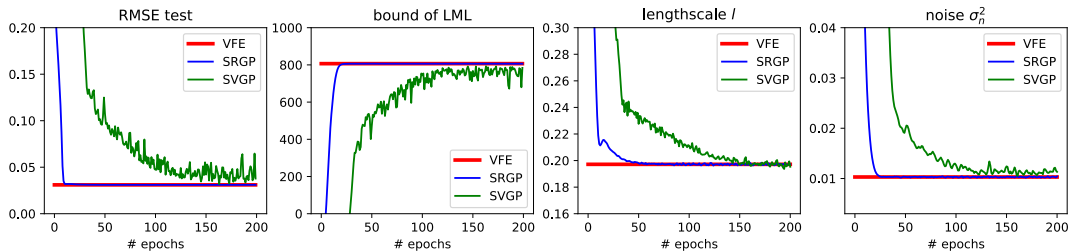


Figure 6.2. Convergence of SRGP (blue) on a 1-D toy to batch version VFE (red). Compared to SVGP (green), the convergence of the root-mean-squared-error (RMSE) of test points, the bound of the log marginal likelihood (LML) as well as the hyper-parameters is faster and more accurate.

#### 6.2.4 Mini-batch size

The size  $B$  of the mini-batches has an impact on the speed of convergence of the algorithm. Proposition 6.1 tells us that if we use a full batch, that is  $B = N$ , our algorithm requires the same number of gradient updates as a full batch method to converge. On the other hand smaller batches should require more updates and should lead to a higher variance in the results. Fig. 6.3 shows a comparison of different mini-batch sizes on a 1-D toy example with  $N = 10'000$  data samples generated with the same parameters as in Sect. 6.2.5.1. The convergence to the full batch value is slower as the batch size decreases. Moreover the variance of the error, over the repetitions, is much larger for smaller batch sizes: in the last 10 normalized gradient updates, the standard deviation of the error is on average  $3.8 \times 10^{-3}$  for  $B = 100$  and  $8.1 \times 10^{-4}$  for  $B = 5'000$ , denoting a more stable procedure for higher batch sizes. As the mini-batch size increases the computational cost for each gradient update also increases. In this example one gradient update requires on average  $4.9 \times 10^{-3}$  sec and  $5.8 \times 10^{-2}$  sec with  $B = 100$  and  $B = 5'000$  respectively.<sup>1</sup> These considerations suggest that a reasonable choice is a large mini-batch size within the computational and time budgets.

<sup>1</sup>The times are measured on a laptop with a Intel i5-7300U CPU @ 2.6 GHz.

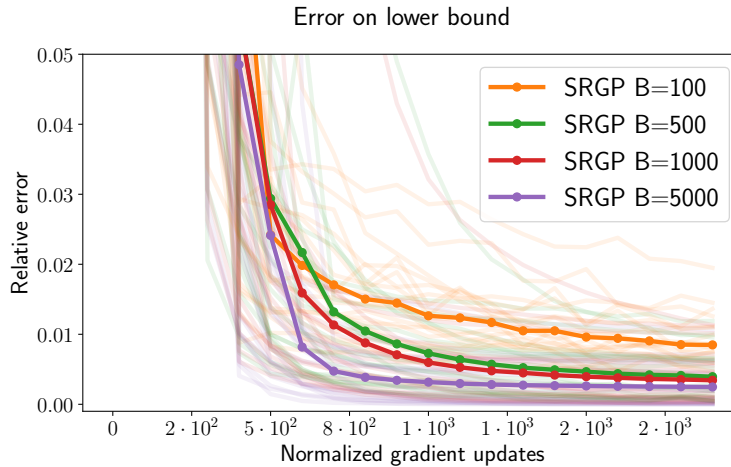


Figure 6.3. Relative error in the bound of the log marginal likelihood between full batch and SRGP. Average over 20 repetitions.

## 6.2.5 Experiments

We first benchmark our method with  $N = 100'000$  synthetic data samples generated by a GP in several dimensions. Next, we apply our approach to the Airline data used in Hensman et al. [2013] with a million of data samples. Finally a more realistic setup is presented where we use up to a million data samples to train a nonlinear plant. We compare our SRGP method to full GP and sparse batch method VFE for a subset of data (using the implementation in GPy [since 2012]) and to the state-of-the-art stochastic parameter estimation method SVGP implemented in GPflow [Matthews et al., 2017]. Our algorithm works also for many other sparse models, however, only large-scale implementations of standard SVGP are available (corresponding to the VFE model), thus we restrict the investigation to this model.

### 6.2.5.1 GP Simulation

In this section we test our proposed learning procedure on simulated GP data. We generate  $N = 100'000$  data samples from a zero-mean (sparse) GP with SE covariance kernel with hyper-parameters  $\sigma_0 = 1$ ,  $\sigma_n = 0.1$  and  $l = \{0.1, 0.2, 0.5\}$  in  $D = \{1, 2, 5\}$  dimensions. The initial  $M = \{20, 50, 100\}$  inducing points are randomly selected points from the data and the hyper-parameters of a SE kernel with individual lengthscales for each dimension are initialized to the same values for both algorithms ( $\sigma_0 = 1$ ,  $\sigma_n = 1$ ,  $l_1, \dots, l_D = 1$ ). All parameters are sequentially optimized with our recursive approach and with SVGP with a mini-batch

size of  $B = 5000$ . The stochastic gradient descent method ADAM [Kingma and Ba, 2014] is employed for both methods with learning rates  $\{0.001, 0.005, 0.005\}$  for SVGP and  $\{0.0001, 0.001, 0.005\}$  for SRGP (based on some preliminary experiments). Each experiment is replicated 10 times.

Fig. 6.4 shows the bound to the log marginal likelihood, the RMSE and the coverage of 10'000 test points for the data dimensions  $D = \{1, 2, 5\}$  of both methods over 50 epochs. The shaded lines indicates the 10 repetitions and the thick line correspond to the mean. The recursive propagation of the gradients achieves faster convergence and more accurate performance regarding mean RMSE and smaller values for the log marginal likelihood. The higher accuracy and faster convergence can possibly be explained by the analytic updates of the posterior mean and covariance which leads to less parameters to be optimized numerically.

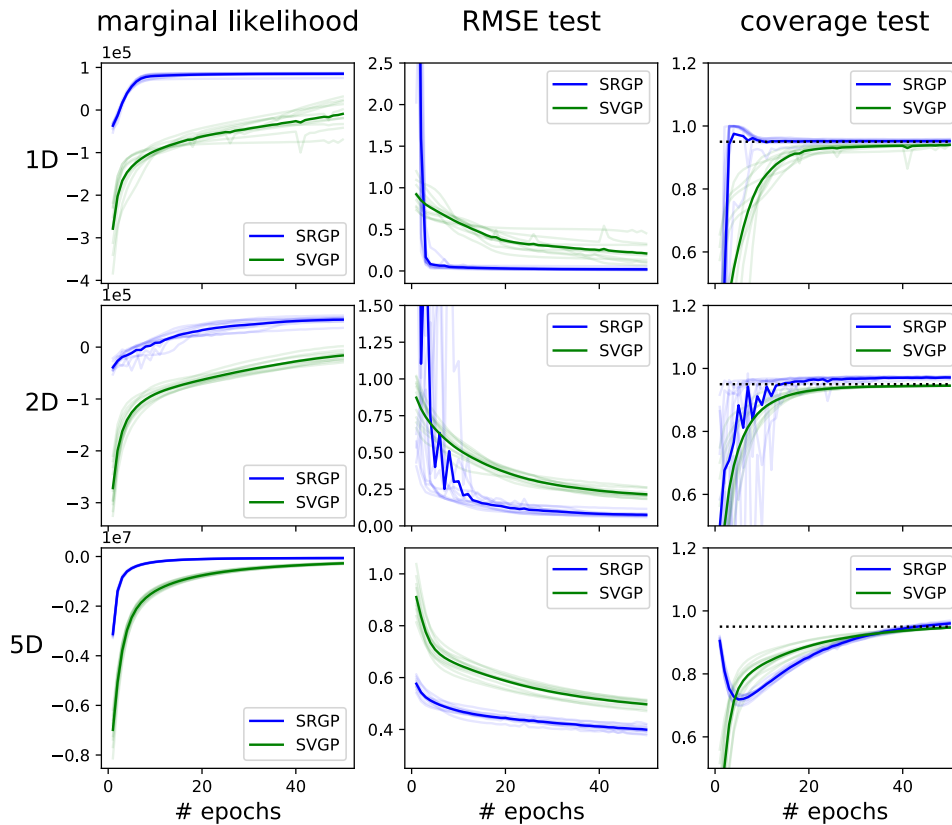


Figure 6.4. Results of simulation study. In particular, the convergence over 50 epochs for  $N = 100'000$  synthetic GP data samples in several dimensions obtained by SVGP and our proposed method SRGP are depicted.



## 6.2.5.2 Airline Data

For the second example we apply our recursive method to the Airline Data used in Hensman et al. [2013]. It consists of flight arrival and departure times for more than 2 millions flights in the USA from January 2008 and April 2008. We preprocessed the data as similar as possible as described in Hensman et al. [2013] resulting in 8 variables: age of the aircraft, distance that needs to be covered, airtime, departure time, arrival time, day of the week, day of the month and month. We trained our recursive method as well as SVGP with an SE kernel on  $N = 1'000'000$  data samples with  $M = 500$  inducing points randomly selected from the data and a mini-batch size of  $B = 10'000$ . The ADAM learning rates are set to 0.005 for both methods and the size of the test set is 50'000. For 5 different repetitions, the RMSE as a function of epochs is depicted in Fig. 6.5. The mean coverage on test data (at 95%) is comparable for both methods with values of 0.92 and 0.97 for SVGP and SRGP respectively. The overall performance of SRGP is superior to SVGP.

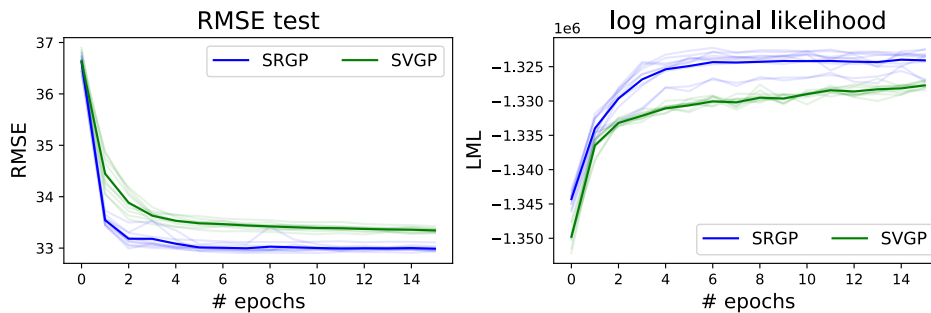


Figure 6.5. Convergence over several epochs of RMSE and bound to log marginal likelihood for  $N = 1'000'000$  samples from the Airline data for SRGP and SVGP.

## 6.2.5.3 Non-Linear Plant

GPs are a powerful non-parametric model to describe complex functions, thus they are suitable to learn the complex input output behavior of a non-linear plant. However, with full or even sparse batch GP methods the use is restricted to a few thousands of samples. With our sequential learning method, we are able to exploit the huge amount of available data by training with up to a million of samples. We consider a Continuous Stirred Tank Reactor (CSTR). The dynamic

model of the plant is

$$\begin{aligned}\frac{d}{dt}h(t) &= w_1(t) + w_2(t) - 0.2\sqrt{h(t)} \\ \frac{d}{dt}C_b(t) &= (C_{b1} - C_b(t))\frac{w_1(t)}{h(t)} + (C_{b2} - C_b(t))\frac{w_2(t)}{h(t)} - \frac{k_1 C_b(t)}{(1+k_2 C_b(t))^2},\end{aligned}$$

where  $C_b(t)$  is the product concentration at the output of the process,  $h(t)$  is the liquid level,  $w_1(t)$  is the flow rate of concentrated feed  $C_{b1}$ , and  $w_2(t)$  is the flow rate of the diluted feed  $C_{b2}$ . The input concentrations are  $C_{b1} = 24.9$  and  $C_{b2} = 0.1$ . The constants associated with the rate of consumption are  $k_1 = k_2 = 1$ . The objective of the controller is to maintain the product concentration by changing the flow  $w_1(t)$ . To simplify the example, we assume that  $w_2(t) = 0.1$  and that the level of the tank  $h(t)$  is not controlled. We denote the controlled outputs  $C_b(t), C_b(t-1), \dots, C_b(t-p)$  as  $f_t, f_{t-1}, \dots, f_{t-p}$  and the control variables as  $w_t, w_{t-1}, \dots, w_{t-p}$ . Therefore, the plant identification problem can be shaped into the problem of estimating the non-linear function defined as  $f_t = g(f_{t-1}, \dots, f_{t-p}, w_t, w_{t-1}, \dots, w_{t-p})$ , which depends on the  $p$  previous values as well as on the current and the  $p$  past control values  $w$ . However, we can

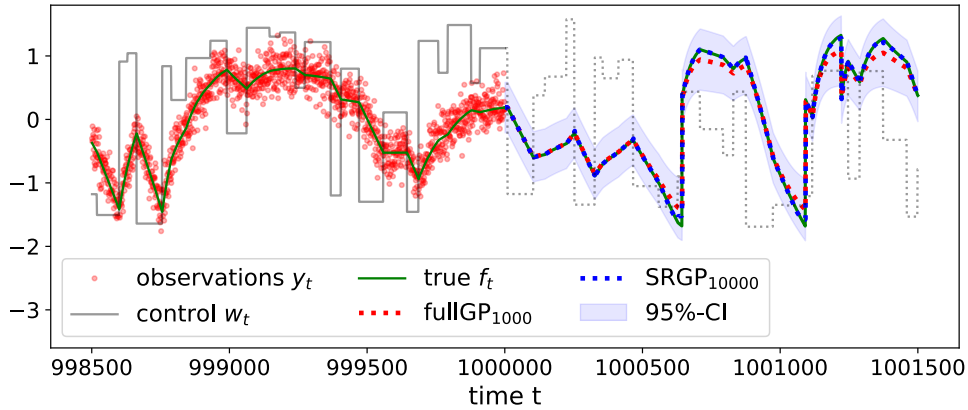


Figure 6.6. Training and prediction phases for non-linear plant.

only observe a noisy version of the controlled response, that is  $y_t = f_t + \varepsilon_t$  with  $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$ . Using a sampling rate of  $0.2s$ , we have generated  $1'200'000$  observations (about 3 days of observations). The plant input is a series of steps, with random height (in the interval  $[0, 4]$ ), occurring at random intervals (in the interval  $[5, 20]s$ ). For different numbers  $N_{train}$ , we use the samples  $y_{10^6-N_{train}}, \dots, y_{10^6}$  for training and the last  $200'000$  are used as a test set. The goal is to learn a model for the controlled response  $y_t$  given  $\mathbf{x}_t = [y_{t-1}, y_{t-2}, w_t, w_{t-1}, w_{t-2}]^T \in \mathbb{R}^5$  for the particular choice of  $p = 2$ . We model the non-linear function  $g$  with a GP with a SE kernel. For comparison, we train full GP and sparse batch GP (with

100 inducing points) on a time horizon  $N_{train}$  of up to 10'000 and 50'000 past values, respectively. With the sequential version SVGP and our recursive gradient propagation method SRGP (both with 100 inducing points and mini-batch size of 1'000), we use a time horizon of up to a million. Note that in this example, the number of parameters numerically estimated by SVGP is  $\approx 10500$ , while our approach only estimates  $\approx 500$  parameters due to the analytic updates. This situation is depicted in Fig. 6.6, where for 1500 training samples  $y_t$  (red dots), the true (unknown) function  $f_t$  (green) and the control input  $w_t$  (grey) is shown together with the predicted values with full GP (red dotted) and recursive GP (blue dotted) trained on a time horizon of 1'000 and 10'000, respectively. In Fig. 6.7, the RMSE and the median computed on the test set (with 10 repetitions) is depicted for full GP, sparse GP (VFE), SVGP and SRGP trained with varying time horizons. For small and medium training sizes, when the batch methods are applicable, our recursive method achieves the same performance as the batch counterpart (VFE) and is comparable to full GP. Due to the analytic updates of the posterior, SRGP outperforms SVGP regarding both RMSE and median for all training sizes. By exploiting more than several thousand past values, a significant increase in performance of SRGP can be still observed, thus it constitutes an approach to accurately scale GPs up to a million of past values.

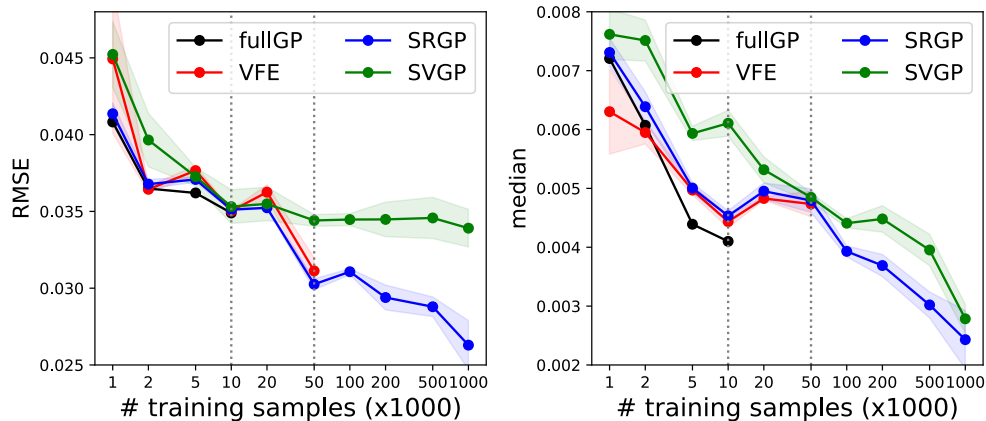


Figure 6.7. RMSE and median for full GP, batch sparse GP (VFE), sequential SVGP, and our recursive method (SRGP) trained on varying time horizons (logarithmic scale). The grey dotted vertical lines at 10'000 and 50'000 indicate the maximal samples used for training full GP and sparse batch GP (VFE), respectively.

### 6.2.6 Conclusion

In this section, we introduced a recursive hyperparameter estimation method called *stochastic recursive Gaussian process* (SRGP) for a general class of sparse GP approximations. In particular, we proposed a recursive collapsed bound to the log marginal likelihood that matches exactly the batch version, but can be used for stochastic estimation. Due to the analytic updates of the posterior, our method has much less parameters to be estimated numerically. For example, in the application to learn the input output behavior of a non-linear plant, as presented in Section 6.2.5.3, the number of parameters estimated by SVGP is  $\approx 10500$ , while our approach only estimates  $\approx 500$  parameters due to the analytic updates. As a consequence, the experimental section showed that our recursive method needs less epochs and has superior accuracy compared to state-of-the-art, thus constitutes an efficient methodology for scaling GPs to big data problems.

## 6.3 Sparse Information Filter for Fast Sparse GPs

In this section, we propose a very efficient method for estimating the hyperparameters  $\theta$  for a range of well-known sparse GPs, which enables to scale GP inference up to several millions of training samples. As discussed in Section 6.1.1, the state-of-the-art for hyperparameter optimization is based on variational inference. In particular the VFE method proposed by Titsias [2009] and discussed in Section 6.1.1.1, which is based on deterministically optimizing a collapsed variational lower bound, constitutes a principled way to find the hyperparameters. However, this approach is not appropriate for big data since the optimization of the lower bound involves the whole training data and cannot be split into mini-batches. In order to address this issue, Hensman et al. [2013] proposed a stochastic variational method (SVGP), as discussed in Section 6.1.1.2, employing an uncollapsed version of the variational lower bound, which decomposes into a sum over mini-batches and allows for stochastic optimization. Unfortunately, this has the effect, that the posterior moments are no longer analytically available, consequently, the bound is less tight and increases the size of the parameter space and can lead to unstable results. An improved stochastic variational inference algorithm for sparse GPs has been proposed by Shi et al. [2020] with their method SOLVEGP, which is based on two orthogonal sets of inducing points, which has slightly better performance than the original method SVGP. In Schürch et al. [2020] and Section 6.2, we proposed a recursive collapsed lower bound that exploits analytic updates for the posterior distribution and decom-

poses into a sum over mini-batches. Consequently, the method can be scaled to millions of data points by stochastic optimization. This recursive approach provides a performance competitive with SVGP both in terms of accuracy and computational time, however, it requires to store the past gradients in memory. When the input space dimension or the number of inducing points is very large, this method becomes problematic memory-wise as the past Jacobian matrices are cumbersome to store.

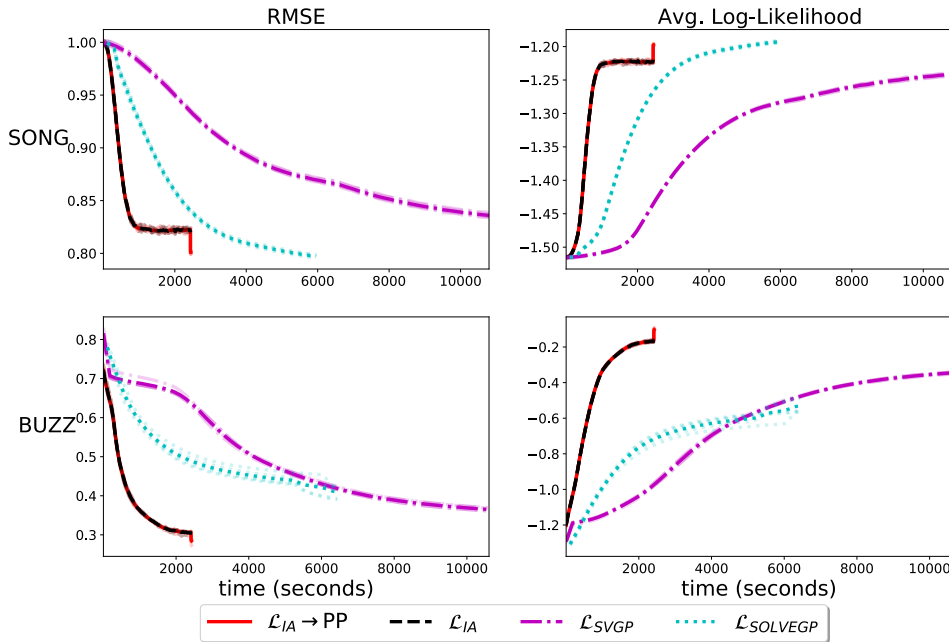


Figure 6.8. The main method presented in this paper,  $\mathcal{L}_{IA} \rightarrow PP$ , compared against  $\mathcal{L}_{SVGP}$  and  $\mathcal{L}_{SOLVEGP}$  on the high dimensional UCI datasets SONG (89 dimensions) and BUZZ (77 dimensions). All methods run for the same number of iteration, plotted against wall-clock time. This behaviour is also observed in the other experiments, see Section 6.3.8.

In this approach, we address this issue and propose a simple and cheap training method that efficiently achieves state-of-the-art performance in practice. It builds on the recursive approach by Schürch et al. [2020], so that the posterior moments are still analytically available, but we circumvent to store the past gradients. This leads to a highly scalable method for hyperparameter learning for sparse GPs. In particular, our method is based on the following hybrid approach. First, we stochastically train the hyperparameters on independent mini-batches and then, we compute the recursive posterior on the whole dataset with analytical updates for the fixed optimal hyperparameters. Note that the independence approximation of the mini-batches only affects the level II inference

for the hyperparameters, the level I inference is exact for fixed hyperparameters. We benchmark our method on several real datasets with millions of data points against the state-of-the-art stochastic variational GP (SVGP) and sparse orthogonal variational inference for GPs (SOLVEGP). As a preview, in Figure 6.8, there is an example of the behavior of our main method  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  compared against SVGP and SOLVEGP depicted. The plots show the root mean squared error (RMSE) and average log likelihood as a function of computational time. We observe, that our method achieves comparable performances to SVGP and SOLVEGP while providing considerable speed-ups.

### 6.3.1 Recursive Estimation for Sparse GPs

We briefly recall the main findings from Chapter 5 [Schürch et al., 2020], where a recursive way to train a range of sparse GP models is presented. In particular, the posterior distribution  $q(\mathbf{a}|\mathbf{y}_{1:j})$  of the inducing points can be recursively updated

$$\begin{aligned} q(\mathbf{a}|\mathbf{y}_{1:j}) &= \int \frac{q(\mathbf{y}_j|\mathbf{a}) q(\mathbf{a}|\mathbf{y}_{1:j-1})}{q(\mathbf{y}_j)} d\mathbf{a} \\ &\propto \int \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \mathbf{V}_j) \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_{j-1}) d\mathbf{a} \\ &= \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \end{aligned}$$

based on the previous posterior distribution  $q(\mathbf{a}|\mathbf{y}_{1:j-1})$ , which acts as prior in the next step. Thereby, the recursive posterior moments  $\boldsymbol{\mu}_j$  and  $\boldsymbol{\Sigma}_j$  are analytically available (5.11), namely

$$\begin{aligned} \boldsymbol{\mu}_j &= \boldsymbol{\Sigma}_j \left( \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j + \boldsymbol{\Sigma}_{j-1}^{-1} \boldsymbol{\mu}_{j-1} \right), \\ \boldsymbol{\Sigma}_j &= \left( \boldsymbol{\Sigma}_{j-1}^{-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j \right)^{-1}, \end{aligned} \tag{6.4}$$

which exploit the previous posterior moments  $\boldsymbol{\mu}_{j-1}$  and  $\boldsymbol{\Sigma}_{j-1}$ , for which [Schürch et al., 2020] proposed a Kalman-Filter-(KF)-like updating as described in Section 5.17. These recursive posterior updates give rise to the recursive collapsed variational lower bound

$$\mathcal{L}_{\text{REC}}(\boldsymbol{\theta}) = \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j | \mathbf{H}_j \boldsymbol{\mu}_{j-1}, \mathbf{P}_j) - a_j(\boldsymbol{\theta}), \tag{6.5}$$

where we recall  $\mathbf{P}_j = \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \mathbf{V}_j$ ,  $\mathbf{H}_j = \mathbf{K}_{\mathbf{X}_j \mathbf{A}} \mathbf{K}_{\mathbf{A} \mathbf{A}}^{-1}$  and  $\mathbf{V}_j = \bar{\mathbf{V}}_j + \sigma_n^2 \mathbb{I}$  with the model specific training covariance  $\bar{\mathbf{V}}_j$  and correction term  $a_j$  according to

Table 5.2. By keeping the hyperparameters fixed, the posterior distribution, as well as the lower bound, are equal to the batch version. For instance, using  $V_j = \sigma_n^2 \mathbb{I}$  and  $a_j(\boldsymbol{\theta}) = \frac{\text{Tr}(\mathbf{K}_{X_j X_j} - \mathbf{Q}_{X_j X_j})}{2\sigma_n^2}$  yields the VFE model, and it holds in particular  $\mathcal{L}_{\text{VFE}} = \mathcal{L}_{\text{REC}}$ . For hyperparameter estimation, the proposed method *stochastic recursive gradient propagation* (SRGP) is based on a stochastically approximated gradient by propagating the gradients of the posterior moments. This method has two main drawbacks. First, it is constrained to use small learning rates for the approximation to be valid and stable in practice. Second, the gradients of the last iteration must be stored. Such storage becomes problematic when the input space dimension or the number of inducing points is very large.

### 6.3.2 Information Filter for Sparse GPs

In order to overcome the previous issues, while keeping the advantage of analytic posterior updates, we propose an *Information-Filter*-(IF)-like update for the posterior moments, which constitutes an efficient and easy to interpret method for computing the recursive posterior  $p(\mathbf{a} | \mathbf{y}_{1:j-1})$  of the inducing points  $\mathbf{a}$  for sparse GPs. Instead working with the posterior moments in (6.4), the posterior over the inducing points can be more efficiently propagated using a natural parameterization  $\mathcal{N}^{-1}(\mathbf{a} | \boldsymbol{\eta}_j, \boldsymbol{\Lambda}_j)$  with  $\boldsymbol{\eta}_j = \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j$  and  $\boldsymbol{\Lambda}_j = \boldsymbol{\Sigma}_j^{-1}$ . The advantage of this formulation are the compact updates

$$\boldsymbol{\eta}_j = \boldsymbol{\eta}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j \quad \text{and} \quad \boldsymbol{\eta}_0 = \mathbf{0}, \quad (6.6)$$

$$\boldsymbol{\Lambda}_j = \boldsymbol{\Lambda}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j \quad \text{and} \quad \boldsymbol{\Lambda}_0 = \mathbf{K}_{AA}^{-1}, \quad (6.7)$$

involving  $\mathbf{H}_j = \mathbf{K}_{X_j A} \mathbf{K}_{AA}^{-1}$  and  $\mathbf{V}_j = \bar{\mathbf{V}}_j + \sigma_n^2 \mathbb{I}$ . For instance, for the VFE model, the training covariance is  $\bar{\mathbf{V}}_j = \mathbf{0}$  so that  $\mathbf{V}_j = \sigma_n^2 \mathbb{I}$ . Particularly, this IF-like propagation is more efficient than the KF formulation used by Schürch et al. [2020], when the mini-batch size is greater than the number of inducing points, which is usually the case in practice.

### 6.3.3 Generalized Lower Bound based on IF

Based on the IF-like update, the corresponding lower bound can be derived by

$$\begin{aligned}
\mathcal{L}_{IF}(\boldsymbol{\theta}) &= \log q(\mathbf{y}|\boldsymbol{\theta}) - \sum_{j=1}^J a_j(\boldsymbol{\theta}) \\
&= \log \prod_{j=1}^J \int q(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}) q(\mathbf{a}|\mathbf{y}_{1:j-1}, \boldsymbol{\theta}) d\mathbf{a} - \sum_{j=1}^J a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \int q(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}) q(\mathbf{a}|\mathbf{y}_{1:j-1}, \boldsymbol{\theta}) d\mathbf{a} - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \int \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \mathbf{V}_j) \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_{j-1}) d\mathbf{a} - a_j(\boldsymbol{\theta}) \quad (6.8) \\
&= \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\boldsymbol{\mu}_{j-1}, \mathbf{H}_j\boldsymbol{\Sigma}_{j-1}\mathbf{H}_j^T + \mathbf{V}_j) - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \mathcal{N}(\mathbf{r}_j|\mathbf{0}, \mathbf{P}_j) - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \mathcal{N}^{-1}(\mathbf{r}_j|\mathbf{0}, \mathbf{P}_j^{-1}) - a_j(\boldsymbol{\theta}),
\end{aligned}$$

where  $\mathbf{V}_j = \bar{\mathbf{V}}_j + \sigma_n^2 \mathbb{I}$ ,  $\mathbf{r}_j = \mathbf{y}_j - \mathbf{H}_j\boldsymbol{\mu}_{j-1}$  and  $\mathbf{P}_j = \mathbf{H}_j\boldsymbol{\Sigma}_{j-1}\mathbf{H}_j^T + \mathbf{V}_j$ . For instance, for the VFE model, by plugging-in  $\bar{\mathbf{V}}_j = \mathbf{0}$  and the corresponding correction term  $a_j$  according to Table 5.2, we get

$$\mathcal{L}_{IF}(\boldsymbol{\theta}) = \sum_{j=1}^J \log \mathcal{N}^{-1}(\mathbf{r}_j|\mathbf{0}, \mathbf{P}_j^{-1}) - \frac{\text{Tr}(\mathbf{K}_{X_j X_j} - \mathbf{Q}_{X_j X_j})}{2\sigma_n^2}, \quad (6.9)$$

where  $\mathbf{P}_j^{-1} = \frac{\mathbb{I}}{\sigma_n^2} - \frac{1}{\sigma_n^4} \mathbf{K}_{X_j A} \boldsymbol{\Lambda}_j^{-1} \mathbf{K}_{A X_j}$  can be obtained by applying the inversion lemma to  $\mathbf{P}_j$ . Further computational efficiency could be gained by using the equivalent transformed parameterization as described in Section 5.2.10 leading to an equivalent predictive distribution and marginal likelihood. We refer the reader to the supplementary material of Kania et al. [2021] for efficient computation of the posterior propagation and the lower bound in terms of the transformed posterior mean and covariance.

Note that the value of  $\mathcal{L}_{IF}$  is equivalent to  $\mathcal{L}_{REC}$ , however, it is based on the IF-like updating, which makes it more efficient. Moreover, it constitutes a starting point for an even more efficient approach, as it will be discussed in Section 6.3.5.



### 6.3.4 Stochastic Hyperparameter Optimization

The additive structure of the lower bound  $\mathcal{L}_{\text{IF}}$ , together with the efficient IF posterior propagation, allow for more frequent parameter updates. Analogously to the approach by Schürch et al. [2020],  $\mathcal{L}_{\text{IF}}$  can be stochastically optimized with respect to all the kernel parameters and the inducing point locations. We denote  $\mathcal{L}_{\text{IF},j}$  the  $j$ th term in the sum of  $\mathcal{L}_{\text{IF}} = \sum_{j=1}^J \mathcal{L}_{\text{IF},j}$  and explicitly show the dependencies on the previous posterior moments and the current hyperparameters  $\boldsymbol{\theta}_j$ , that is,

$$\mathcal{L}_{\text{IF},j}(\boldsymbol{\theta}_j, \boldsymbol{\Lambda}_{j-1}, \boldsymbol{\eta}_{j-1}) = d_j(\boldsymbol{\theta}_j, \boldsymbol{\Lambda}_{j-1}, \boldsymbol{\eta}_{j-1}) - a_j(\boldsymbol{\theta}_j),$$

where  $d_j = \log \mathcal{N}^{-1}(\mathbf{r}_k | \mathbf{0}, \mathbf{P}_j^{-1})$ . The gradient of  $\mathcal{L}_{\text{IF},j}$  w.r.t. the hyperparameters  $\boldsymbol{\theta}_j$ , can be approximated using Jacobian matrices of the previous iteration

$$\frac{\partial \mathcal{L}_{\text{IF},j}}{\partial \boldsymbol{\theta}_j} \approx \frac{\partial d_j}{\partial \boldsymbol{\theta}_j} + \frac{\partial d_j}{\partial \boldsymbol{\Lambda}_{j-1}} \frac{\partial \boldsymbol{\Lambda}_{j-1}}{\partial \boldsymbol{\theta}_{j-1}} + \frac{\partial d_j}{\partial \boldsymbol{\eta}_{j-1}} \frac{\partial \boldsymbol{\eta}_{j-1}}{\partial \boldsymbol{\theta}_{j-1}} - \frac{\partial a_j}{\partial \boldsymbol{\theta}_j}. \quad (6.10)$$

This recursive propagation of the gradients of the posterior mean and covariance indirectly takes into account all the past gradients via  $\frac{\partial \boldsymbol{\Lambda}_{j-1}}{\partial \boldsymbol{\theta}_{j-1}}$  and  $\frac{\partial \boldsymbol{\eta}_{j-1}}{\partial \boldsymbol{\theta}_{j-1}}$ . It would be exact and optimal if the derivatives with respect to the current parameters were available. However, when changing the parameters too fast between iterations, e.g. due to a large learning rate, this approximation is too rough and leads to unstable optimization results in practice. Furthermore, the performance gain provided by the recursive gradient propagation does not compensate for the additional storage requirements. For instance, if  $M = 500$  inducing points are used in a  $D = 10$ -dimensional problem, only storing the Jacobian matrices requires 10 GB, under double float precision, i.e. around the memory limit of the GPUs used for this work.

### 6.3.5 Generalized Independent Lower Bound

In order to circumvent these instabilities and inefficiencies in the optimization part based on  $\mathcal{L}_{\text{IF}}$ , we propose a fast and efficient method that ignores the approximated correlations between the mini-batches in the stochastic gradient computation in the beginning. In particular, we approximate  $\mathcal{L}_{\text{IF}}(\boldsymbol{\theta}) \approx \mathcal{L}_{\text{IA}}(\boldsymbol{\theta})$  by using the prior instead of the recursive posterior, that is,  $q(\mathbf{a} | \mathbf{y}_{1:j-1}, \boldsymbol{\theta}) \approx p(\mathbf{a} | \boldsymbol{\theta})$ ,

which leads to the following derivation

$$\begin{aligned}
\mathcal{L}_{IF}(\boldsymbol{\theta}) &= \log q(\mathbf{y}|\boldsymbol{\theta}) - \sum_{j=1}^J a_j(\boldsymbol{\theta}) \\
&= \log \prod_{j=1}^J \int q(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}) q(\mathbf{a}|\mathbf{y}_{1:j-1}, \boldsymbol{\theta}) d\mathbf{a} - \sum_{j=1}^J a_j(\boldsymbol{\theta}) \\
&\approx \log \prod_{j=1}^J \int q(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}) p(\mathbf{a}|\boldsymbol{\theta}) d\mathbf{a} - \sum_{j=1}^J a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \int q(\mathbf{y}_j|\mathbf{a}, \boldsymbol{\theta}) p(\mathbf{a}|\boldsymbol{\theta}) d\mathbf{a} - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \int \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \mathbf{V}_j) \mathcal{N}(\mathbf{a}|\mathbf{0}, \boldsymbol{\Sigma}_0) d\mathbf{a} - a_j(\boldsymbol{\theta}) \tag{6.11} \\
&= \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j|\mathbf{0}, \mathbf{H}_j\boldsymbol{\Sigma}_0\mathbf{H}_j^T + \mathbf{V}_j) - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \log \mathcal{N}(\mathbf{y}_j|\mathbf{0}, \mathbf{P}_j) - a_j(\boldsymbol{\theta}) \\
&= \sum_{j=1}^J \mathcal{L}_{IA,j}(\boldsymbol{\theta}) \\
&= \mathcal{L}_{IA}(\boldsymbol{\theta}),
\end{aligned}$$

where  $\mathbf{P}_j = \mathbf{H}_j\boldsymbol{\Sigma}_0\mathbf{H}_j^T + \mathbf{V}_j = \mathbf{Q}_{\mathbf{x}_j\mathbf{x}_j} + \mathbf{V}_j$ . Note that each mini-batch is assumed to be mutually independent, which simplifies for instance for the VFE lower bound of the marginal likelihood to

$$\mathcal{L}_{IA} = \sum_{k=1}^K \log \mathcal{N}(\mathbf{y}_j|\mathbf{0}, \mathbf{Q}_{\mathbf{x}_j\mathbf{x}_j} + \sigma_n^2\mathbb{I}) - \frac{\text{Tr}(\mathbf{K}_{\mathbf{x}_j\mathbf{x}_j} - \mathbf{Q}_{\mathbf{x}_j\mathbf{x}_j})}{2\sigma_n^2}. \tag{6.12}$$

Note that maximizing this lower bound leads to slightly different hyperparameters as obtained by  $\mathcal{L}_{IF}$ , since the correlations between the mini-batches in the bound are not taken into account. An iterative solution can be obtained by following the stochastic negative gradient direction as explained in Section 2.1.3.2, that is,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma_t \left. \frac{\partial \mathcal{L}_{IA,j}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}, \tag{6.13}$$

with learning rate  $\gamma_t$ , for which we use the particular stochastic optimization method ADAM [Kingma and Ba, 2014].

---

**Algorithm 1**  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  Algorithm
 

---

```

Choose number of epochs  $E_{\text{IA}}$ 
Split  $\mathcal{D}$  into  $J$  mini-batches  $\mathcal{D}_1, \dots, \mathcal{D}_J$  each of size  $B = |\mathcal{D}_j|$ 
for  $e = 1 : E_{\text{IA}}$  do                                      $\triangleright$  loop over epochs
  for  $j = 1 : J$  do                                        $\triangleright$  loop over mini-batches
    compute gradients for the  $j$ th mini-batch  $\mathcal{D}_j$  in (6.12)
    update the hyperparameters (6.13)
  end for
end for
for  $j = 1 : J$  do                                        $\triangleright$  loop over mini-batches
  compute posterior moments with (6.6) and (6.7)
end for

```

---

### 6.3.6 Recovery of the Full Posterior

In order to incorporate all the dependencies once reasonable hyperparameters are achieved with the optimization of  $\mathcal{L}_{\text{IA}}$ , we could switch to optimize the bound  $\mathcal{L}_{\text{IF}}$  during the last few epochs. Eventually, all the ignored data dependencies are taken into account with this method, which is denoted as  $\mathcal{L}_{\text{IA}} \rightarrow \mathcal{L}_{\text{IF}}$ . This has much faster convergence as optimizing  $\mathcal{L}_{\text{IF}}$  directly, however, the memory for propagating the past gradients might be still problematic.

A further computational shortcut is to fix the hyperparameters to the last obtained value, however, computing afterwards the exact posterior moments according to Equations (6.6) and (6.7) without any optimization in the last epoch. This method, denoted  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$ , constitutes a practical alternative when the computation and storage of the Jacobian matrices is costly, for instance in high dimensional problems where a large number of inducing points are needed. For clarity, a pseudo-code of the method is provided in Algorithm 1. Note that with this method is exact for level I inference for the inducing points, it only approximates the inference on level II for the hyperparameters. This means, the independence assumption influences only the optimization of the hyperparameters and no approximation is used for computing the posterior distribution of the inducing points.

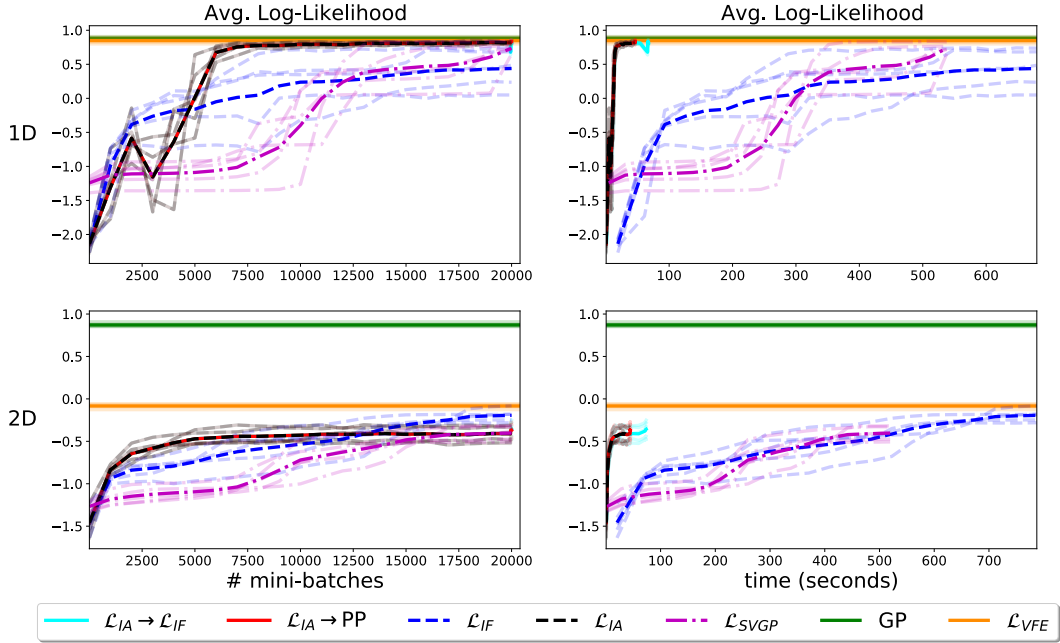


Figure 6.9. Toy experiments: 1D and 2D generated GP data. Performances of GP and  $\mathcal{L}_{VFE}$  in horizontal lines as their optimization does not use mini-batches. On the left comparison in number of mini-batches, on the right runtime comparison.

### 6.3.7 Distributed Hyperparameter Optimization

A further advantage of the Information Filter formulation is that it allows to compute the posterior moments in parallel. Note that, Equations (6.6) and (6.7) can be rewritten as

$$\boldsymbol{\eta}_J = \boldsymbol{\eta}_0 + \sum_{j=1}^J \Delta \boldsymbol{\eta}_j \quad \text{and} \quad \boldsymbol{\Lambda}_J = \boldsymbol{\Lambda}_0 + \sum_{j=1}^J \Delta \boldsymbol{\Lambda}_j,$$

with the additive factors  $\Delta \boldsymbol{\Lambda}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j$  and  $\Delta \boldsymbol{\eta}_j = \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j$ , respectively. Since the sums terms are functionally independent, the computation of the posterior distribution can be easily distributed to  $J$  nodes and aggregated at a central node afterwards. Moreover, the independent lower bound  $\mathcal{L}_{IA}(\boldsymbol{\theta}) = \sum_{j=1}^J \mathcal{L}_{IA,j}(\boldsymbol{\theta})$  can be also straight-forwardly distributed and each node computes the value and its gradient, which can be aggregated at the central node. The aggregated gradient can then be used with deterministic optimization as discussed in Section 2.1.3.1 to find optimal values for the hyperparameters. Note again, the independence assumption only affects the level II inference for the hyperparameters. However, level I inference still takes into account all recursive dependencies and

corresponds exactly to level I inference as in the full batch case.

### 6.3.8 Experiments

Each experiment presented in this section, is repeated 5 times by using different random splits for the datasets. We always use the same hardware equipped with a GeForce RTX 2080 Ti and an Intel(R) Xeon(R) Gold 5217 (3GHz). Furthermore, we switch from optimizing  $\mathcal{L}_{\text{IA}}$  to the posterior propagation, denoted  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$ , in the last 5 epochs of training for all the synthetic and real datasets. Although only one epoch is needed to incorporate all the information of a dataset, we do 5 epochs so the effect is noticeable in the figures. Alternatively, an adaptive switch could be used which is briefly discussed in the supplementary material of Kania et al. [2021]. Moreover, we did not exploit the parallelization of the posterior propagation in order to be able to compare all training algorithms with the same number of iterations in the stochastic optimization. Note that  $\mathcal{L}_{\text{IF}}$  correspond to  $\mathcal{L}_{\text{REC}}$  [Schürch et al., 2020], however, we used a new implementation based on Tensorflow with automatic gradients computations exploiting GPU, so that the computational times are comparable. For full GP, VFE and SVGP, we used the GPflow 2.0 library [Matthews et al., 2017], which is based on Tensorflow 2.0 [Abadi et al., 2015]. We further compare with the recently introduced SOLVEGP [Shi et al., 2020], implemented in GPflow 1.5.1 and Tensorflow 1.15.4. Our algorithms, namely the training of  $\mathcal{L}_{\text{IF}}$ ,  $\mathcal{L}_{\text{IA}} \rightarrow \mathcal{L}_{\text{IF}}$  and  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  are implemented in Tensorflow 2.0<sup>2</sup>. All stochastic methods were optimized with the ADAM optimizer [Kingma and Ba, 2014] using the default settings ( $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e^{-7}$ ) with a learning rate  $\gamma = 0.1$  for the stochastic optimization. We compare each method in terms of average log-likelihood, root mean squared error (RMSE) and computational time. The comparison in computational time is fair because all methods were run on the same hardware which ran exclusively the training procedure.

#### 6.3.8.1 Toy Data

We start by showcasing the methods on a simple dataset generated by a sparse GP with  $M = 200$  random inducing points, and a SE kernel with variance  $\sigma_0^2 = 1$ , Gaussian noise  $\sigma_n^2 = 0.01$  and lengthscales  $l \in \{0.1, 0.2\}$  for dimensions  $D = 1$  and  $D = 2$ , respectively. We consider  $N = 5000$  training points and 500 test points. In all experiments, the initial parameters for the SE kernel were 1 for the lengthscales and Gaussian noise, and 2 for the variance. Moreover, the 200 initial

<sup>2</sup>The code is available at <https://github.com/lkania/Sparse-IF-for-Fast-GP>

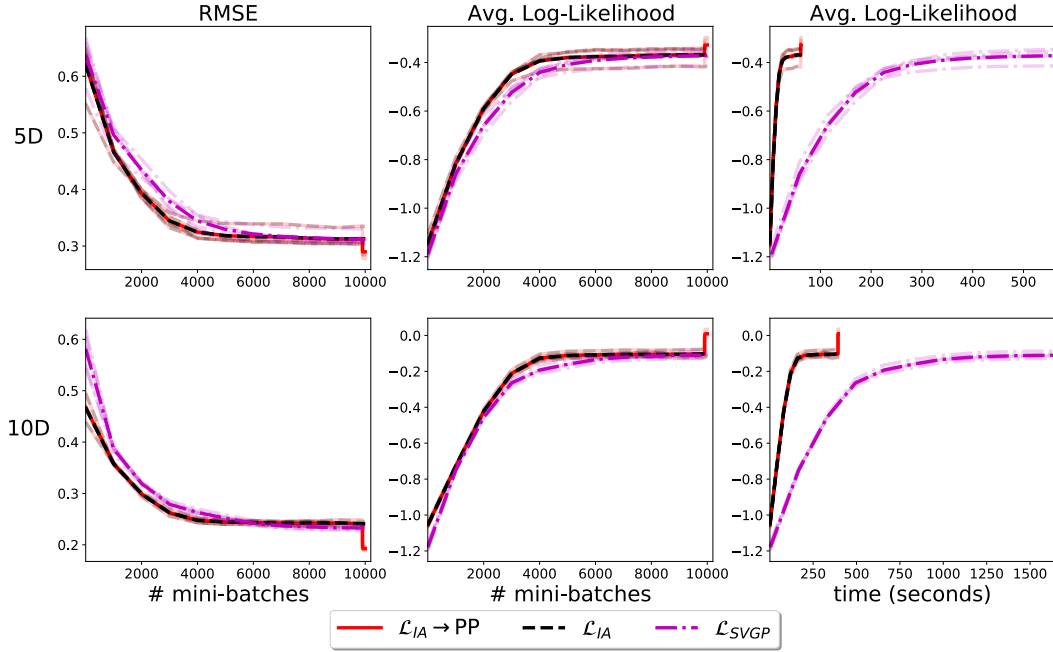


Figure 6.10. Synthetic experiments: 5D and 10D data generated from GPs. RMSE vs number of mini-batches (left), average log-likelihood vs number of mini-batches (center), and average log-likelihood vs runtime (right).

inducing points were randomly selected from the training data. In this example, the switch for  $\mathcal{L}_{IA} \rightarrow \mathcal{L}_{IF}$  and  $\mathcal{L}_{IA} \rightarrow \text{PP}$  was done in the last 200 epochs to make the difference between them visually noticeable. All methods were run for 20K iterations with a learning rate of  $\gamma = 0.001$  and mini-batches of size  $B = 500$  data points. Figure 6.9 shows that in 1D all sparse training methods converge to the  $\mathcal{L}_{VFE}$  solution, which itself converges to the GP solution. However, this is not the case in the 2D example.  $\mathcal{L}_{IF}$  comes closer to the VFE solution, followed by  $\mathcal{L}_{IA} \rightarrow \mathcal{L}_{IF}$ . Note that the convergence of  $\mathcal{L}_{IA} \rightarrow \mathcal{L}_{IF}$  is expected since using  $\mathcal{L}_{IA}$  at the beginning is just a way of *warm-starting* the parameters for  $\mathcal{L}_{IF}$ .

### 6.3.8.2 Synthetic Data

Using the same data generating process used in the previous experiments, we produced two datasets with  $N = 100\text{K}$  points for  $D = 5$  and  $D = 10$  dimensions using an SE kernel with variance  $\sigma_0^2 = 1$ , Gaussian noise  $\sigma_n^2 = 0.01$ , and lengthscales  $l = 0.5$  and  $l = 1$ , respectively. All SGPs models were initialized with Gaussian noise of 1, kernel variance equal to 2 and the lengthscales all equal to 1 in the 5-dimensional case, and 2 in the 10-dimensional problem. All methods

D	Metric	$\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$	$\mathcal{L}_{\text{IA}}$	$\mathcal{L}_{\text{SVGP}}$
5	avg log-lik	<b>-0.33 ± 0.03</b>	-0.45±0.14	-0.48±0.15
10	avg log-lik	<b>0.01 ± 0.01</b>	-0.21±0.20	-0.24±0.20
5	RMSE	<b>0.29 ± 0.01</b>	0.34±0.05	0.36±0.06
10	RMSE	<b>0.19 ± 0.003</b>	0.26±0.04	0.27±0.05
5	runtime	<b>1.03 ± 0.11</b>	1.05±0.11	9.37±1.16
10	runtime	<b>6.59 ± 0.06</b>	6.65±0.09	27.40±0.79

Table 6.3. Average log-likelihood of last epoch (higher is better), average RMSE of the last epoch (lower is better), and average runtime in minutes (lower is better). The algorithm with the best metric is highlighted in each case.

run for 10K iterations in all the datasets using  $M = 100$  and  $M = 500$  inducing points for 5 and 10 dimensions respectively. In all cases, we used learning rates equal to 0.001 and mini-batches of  $B = 5000$  points. Note, that  $\mathcal{L}_{\text{IF}}$  and  $\mathcal{L}_{\text{IA}} \rightarrow \mathcal{L}_{\text{IF}}$  were not run due to the memory constraints of our equipment. Figure 6.10 displays the log-likelihood and RMSE for all the methods. In 5 and 10 dimensions, we can clearly see the effect of a few posterior updates, for  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$ , after fixing the parameters with  $\mathcal{L}_{\text{IA}}$ . Additionally, Table 6.3 displays the average runtimes for each method. Note that  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  offers a speed-up of 9x and 4x over  $\mathcal{L}_{\text{SVGP}}$  for the 5- and 10-dimensional datasets respectively.

Name	Train	Test	D	Learning rate	Iterations
AIRLINES	1.6M	100K	8	0.0001	60000
GAS	1.4M	100K	17	0.0001	40000
BUZZ	0.46M	100K	77	0.0001	60000
SONG	0.41M	100K	89	0.0001	60000
SGEMM	0.19M	48K	14	0.001	15000
PROTEIN	0.036M	9K	9	0.001	5000
BIKE	0.013M	3.4K	12	0.0001	40000

Table 6.4. Overview of used datasets. In particular, the size of the training and test as well as the dimension of the benchmarked datasets are indicated. The last two columns show the learning rate and the number of iterations for all training methods run in each dataset.

### 6.3.8.3 Real Data

We trained the sequential sparse SGP methods exploiting the lower bounds  $\mathcal{L}_{\text{SVGP}}$ ,  $\mathcal{L}_{\text{SOLVEGP}}$ ,  $\mathcal{L}_{\text{IA}}$  and  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  on several real-world datasets as summarized in Table 6.4. All datasets were downloaded from the UCI repository *dua* except for AIRLINES, where we follow the original example in Hensman et al. [2013]. For each training algorithm, we present average log-likelihood, RMSE and runtime for the same number of iterations. Figure 6.8 shows the results corresponding to the datasets SONG and BUZZ. Tables 6.5, 6.6 and 6.7 report the average log-likelihood, RMSE and runtime of all the algorithms. Note that training with  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  always provides comparable results to the state-of-the-art. However, such results are achieved, on average, approximately 4 times faster than  $\mathcal{L}_{\text{SVGP}}$  and 2.5 times faster than  $\mathcal{L}_{\text{SOLVEGP}}$ , see the last two columns of table 6.7. While  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  is uniformly around 4 times faster than  $\mathcal{L}_{\text{SVGP}}$ , its performance with respect to  $\mathcal{L}_{\text{SOLVEGP}}$  strongly depends on the dataset. In general,  $\mathcal{L}_{\text{SOLVEGP}}$  is faster on smaller and simpler datasets, while it becomes computationally costly on noisy datasets like AIRLINE or BUZZ. The results demonstrate that the proposed method  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  constitutes an efficient method for hyperparameter estimation in sparse GPs in practice, in particular for high dimensional and large datasets. Note that the hybrid approach  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$  with the additional posterior propagation step always increases performances (lower RMSE, higher log-likelihood) compared to  $\mathcal{L}_{\text{IA}}$ .



Dataset	$\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$	$\mathcal{L}_{\text{IA}}$	$\mathcal{L}_{\text{SVGP}}$	$\mathcal{L}_{\text{SOLVEGP}}$
AIRLINES	<b>-1.31 ± 0.01</b>	-1.33±0.01	-1.33±0.01	-1.32±0.01
GAS	-0.13±0.02	-0.24±0.02	-0.25±0.01	<b>-0.05 ± 0.05</b>
BUZZ	<b>-0.10 ± 0.01</b>	-0.17±0.01	-0.36±0.01	-0.55±0.04
SONG	<b>-1.20±0.001</b>	-1.22±0.002	-1.25±0.004	<b>-1.20 ± 0.002</b>
SGEMM	0.47±0.002	0.33±0.02	0.28±0.05	<b>0.63 ± 0.17</b>
PROTEIN	<b>-1.04 ± 0.01</b>	-1.13±0.05	-1.14±0.08	-1.06±0.02
BIKE	<b>0.08 ± 0.02</b>	0.03±0.02	0.04±0.02	0.01±0.04

Table 6.5. Average Log-Likelihood of the last epoch, higher is better. The algorithm with the highest average mean log-likelihood is highlighted for each dataset.

Dataset	$\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$	$\mathcal{L}_{\text{IA}}$	$\mathcal{L}_{\text{SVGP}}$	$\mathcal{L}_{\text{SOLVEGP}}$
AIRLINES	<b>0.89 ± 0.01</b>	0.92±0.01	0.92±0.01	0.90±0.01
GAS	0.28±0.01	0.32±0.01	0.32±0.01	<b>0.25 ± 0.01</b>
BUZZ	<b>0.28 ± 0.01</b>	0.31±0.004	0.37±0.01	0.42±0.02
SONG	<b>0.80±0.001</b>	0.82±0.002	0.84±0.003	<b>0.80 ± 0.002</b>
SGEMM	0.14±0.001	0.17±0.004	0.17±0.01	<b>0.13 ± 0.03</b>
PROTEIN	<b>0.68 ± 0.005</b>	0.74±0.01	0.74±0.04	0.69±0.01
BIKE	<b>0.22 ± 0.004</b>	0.24±0.005	0.23±0.004	0.23±0.01

Table 6.6. Average RMSE of the last epoch, lower is better. The algorithm with the lowest average mean RMSE is highlighted for each dataset.

Dataset	$\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$	$\mathcal{L}_{\text{IA}}$	$\mathcal{L}_{\text{SVGP}}$	$\mathcal{L}_{\text{SOLVEGP}}$	$\Delta_{\text{SV}}$	$\Delta_{\text{SOLVE}}$
AIRLINES	<b>0.65 ± 0.004</b>	0.66±0.004	2.78±0.07	3.72±0.16	4.29	5.75
GAS	<b>0.43 ± 0.001</b>	0.44±0.004	1.84±0.04	2.20±0.11	4.28	5.12
BUZZ	<b>0.67 ± 0.004</b>	0.68±0.005	2.95±0.04	1.79±0.08	4.38	2.66
SONG	<b>0.68 ± 0.01</b>	0.69±0.01	3.00±0.11	1.65±0.02	4.41	2.43
SGEMM	<b>0.16 ± 0.001</b>	0.16±0.001	0.69±0.02	0.30±0.01	4.22	1.84
PROTEIN	<b>0.05 ± 0.0002</b>	0.05±0.0001	0.23±0.004	0.07±0.001	4.20	1.33
BIKE	<b>0.44 ± 0.005</b>	0.44±0.003	1.88±0.07	0.56±0.02	4.27	1.28

Table 6.7. Average runtimes in hours. The algorithm with the lowest average mean runtime is highlighted for each dataset.  $\Delta_{\text{SV}}$  is the ratio between the average runtimes of  $\mathcal{L}_{\text{SVGP}}$  and  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$ , and similarly,  $\Delta_{\text{SOLVE}}$  between  $\mathcal{L}_{\text{SOLVEGP}}$  and  $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$ .

## 6.4 Summary of Contributions

In this chapter, we proposed 2 novel algorithms for sequential hyperparameter estimation for sparse GPs.

The Section 6.2 about "*Recursive Gradient Propagation*" is based on the second part of the paper "*Recursive Estimation for Sparse Gaussian Process Regression*" [Schürch et al., 2020] for which we present here in this thesis an extended and slightly adapted version without significant changes. This work provide the following novel contributions:

- **Stochastic Optimization:** Based on a recursive collapsed lower bound, a unifying stochastic optimization procedure for estimating the hyperparameters for a general class of sparse GP models is presented.
- **Analytical Posterior:** The recursive posterior of our model is analytically available, which has the effect that the resulting lower bound is tighter and there are less parameters to numerically optimize compared to competitor methods.
- **Scaling to Large Datasets:** Compared to the full batch case, our presented method can be scaled to large data sets with up to a million of training data samples with comparable performance.
- **Faster Convergence:** Due to the analytic posterior, our method shows in practice that the convergence to the full batch sparse GP is faster than state-of-the-art methods, i.e. it needs less number of iterations in the optimization part.
- **Superior Accuracy:** Since the optimal analytic posterior is used in the lower bound, our method shows in many practical experiments superior accuracy compared to state-of-the-art methods.

The Section 6.3 about "*Sparse Information Filter for Fast Sparse GPs*" is based on the paper "*Sparse Information Filter for Fast Gaussian Process Regression*" [Kania, Schürch, Azzimonti, and Benavoli, 2021] for which in this thesis an extended and slightly adapted version without significant changes is presented. This work provides the following novel contributions:

- **Fast Stochastic Optimization:** Based on an independent lower bound, we propose a very efficient method for optimizing hyperparameters for sparse

GPs. Thereby, the independence assumption between the mini-batches only affects the inference at level II (hyperparameters) and not inference at level I (inducing points).

- **Analytical Posterior:** We combine the fast optimization for the hyperparameters with an analytic Information-Filter-like updating for the posterior of the inducing points, which is equivalent to the batch posterior for fixed hyperparameters.
- **Scaling to Big Data:** Due to the hybrid approach with the analytic posterior and the fast stochastic optimization, the resulting method is highly scalable up to several millions of training data samples.
- **Significant Speed-Up:** Based on several experiments on real world data, our proposed method achieves competitive performance to state-of-the-art techniques in a fraction of the running time.
- **Distributed Learning:** The same hybrid approach as used for sequential learning can also be used to efficiently compute the optimal hyperparameters, together with the analytic posterior in parallel in a distributed setting.



## Chapter 7

# Correlated Product of Experts for Sparse Gaussian Process Regression

In this chapter, we propose a novel approach for GP approximation based on local and correlated experts. In particular, our model is a generalization of the independent Product of Experts (PoE) and sparse global GPs (SGPs), as discussed in Section 4.2 and 4.3, respectively. In particular, we introduce local correlations between the originally independent experts. Thereby, each of these experts correspond to a local and sparse GP model represented by a set of *local inducing points*, which are points on the GP summarizing locally the dependencies of the training data. The degree of correlation between the experts can vary between independent up to fully correlated experts in a consistent way, so that our model recovers independent PoEs, sparse global GPs and full GP in the limiting cases. In particular, in this chapter, the necessary background is discussed in Section 7.1, the novel model *Correlated Product of Experts* (CPoE) is presented in Section 7.2, followed by a discussion about experiments to state-of-the-art methods in Section 7.3. Moreover, we present in Section 7.4 an alternative inference approach of the same model based on local message propagation. Finally, we provide a conclusion and an explicit list of the contributions in Section 7.5.

### 7.1 Background

As discussed in previous chapters, full GP inference is limited to datasets with a few thousand data points  $N$ , because of their computational complexity  $\mathcal{O}(N^3)$  and memory complexity  $\mathcal{O}(N^2)$  due to the inversion of a  $N \times N$  kernel matrix. For this reason, many GP approximation techniques have been developed over the past years, as thoroughly discussed in Chapter 4. For the sake of clarity, we

briefly provide here the necessary context again. There are at least two different approaches to circumvent the computational limitation of full GP. On the one hand, there are *sparse and global* methods as discussed in Section 4.2 based on  $M_g \ll N$  so-called global inducing points, which cover sparsely the input space and summarizing the dependencies of the training points. This results in a low-rank approximation of the kernel matrix of size  $M_g \times M_g$ , which is less expensive to invert. These methods consistently approximate full GP, for instance the authors in Titsias [2009] have shown that it converges to full GP as  $M_g \rightarrow N$ . However, all these methods are still cubic in the number of global inducing points  $M_g$  and for many applications - in particular in higher dimensions - the amount of inducing points has to be rather large to capture the pattern of the function properly. On the other hand, there are *independent and local* models, as discussed in Section 4.3, based on averaging predictions from  $J$  independent local experts/models, resulting in a block-diagonal approximation of the kernel matrix. The final probabilistic aggregation is then based on a product of the individual predictive densities, that is why they are called *Product of Experts (PoEs)*. PoE methods provide fast and rather accurate predictions, because they have fewer hyperparameters than inducing point methods and are locally exact. However, the predictive aggregation of independent experts leads to unreliable uncertainty estimates and less accurate predictions in regions between experts.

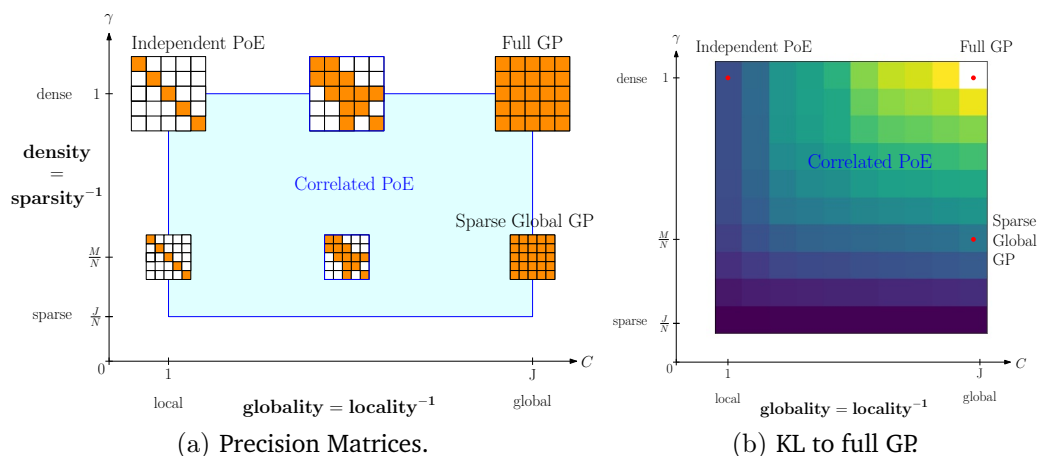


Figure 7.1. Unifying view of our proposed method.

Our approach has the aim to overcome these limitations by introducing a framework based on  $J$  correlated experts, so that it approximates full GP in two orthogonal directions: sparsity and locality. Thereby, our model is a generalization of the independent PoEs and sparse global GPs by introducing local correlations

between experts. These experts correspond to local and sparse GP models represented by a set of *local inducing points*, which are points on the GP summarizing locally the dependencies of the training data. The degree of correlation  $C$  between the experts can vary between independent up to fully correlated experts in a consistent way, so that our model recovers independent PoEs, sparse global GP and full GP in the limiting cases. Our method exploits the conditional independence between the experts, resulting in a sparse and low-rank prior as well as posterior precision (inverse of covariance) matrix, which can be used to efficiently obtain local and correlated predictions from each expert. These correlated predictions are aggregated by the *covariance intersection* method [Julier and Uhlmann, 1997], which is useful for combining consistently several estimates with unknown correlations. The resulting predictive distribution is a smooth weighted average of the predictive distributions of the individual experts. Our algorithm works with a general kernel function and performs well with variables in higher dimensions. The number of hyperparameters to optimize for our method is the same as for full GP, which are just a few parameters (depending on the kernel). These parameters can be similarly estimated via the log marginal likelihood, which is analytically and efficiently computable for our model. In our inference, also (log-normal) priors can be incorporated leading to maximum-a-posteriori estimates for the hyperparameters.

Compared to the independent Product of Experts, the performance can already significantly improve by modelling just a few of the pairwise correlations between the experts. Compared to the number of *global* inducing point  $M_g$ , which is usual much smaller than the number of data points  $N$ , our approach allows a much higher of total *local* inducing points in the order of  $N$ , which helps to cover the space and therefore to model more complicated functions.

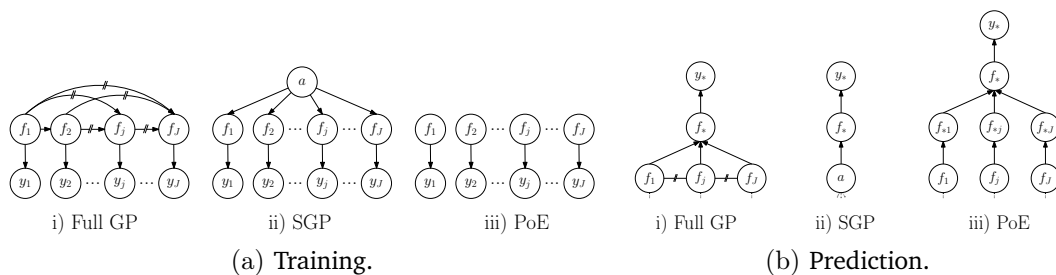


Figure 7.2. Graphical models of different GP approaches.

### 7.1.1 GP Regression

In this section, we briefly contrast the posterior distributions of full GPs, global sparse GPs and local Product of Experts (PoEs) in order to highlight the connections to our model, which will be introduced in the subsequent sections.

We consider data  $\mathcal{D} = \{\mathbf{y}, \mathbf{X}\} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$  split into  $J$  mini-batches of size  $B$  with inputs  $\mathbf{X}_j \in \mathbb{R}^{B \times D}$ , outputs  $\mathbf{y}_j \in \mathbb{R}^B$  and the corresponding latent function values  $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$ , as similarly used in previous chapters. The posterior distribution of full GP over the latent variables given the data as introduced in (3.12) can be alternatively formulated as

$$p(\mathbf{f}|\mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j)p(\mathbf{f}_j|\mathbf{f}_{1:j-1}), \quad (7.1)$$

where the joint prior  $p(\mathbf{f})$  is rewritten using the product rule for probabilities and the other quantities defined in Section 3.2. Similarly, for sparse global methods based on global inducing points  $\mathbf{a}$ , the posterior over the inducing points  $p(\mathbf{a}|\mathbf{y}) \propto \int p(\mathbf{a}, \mathbf{f}, \mathbf{y}) d\mathbf{f}$  as introduced in (4.19) can be derived from the joint distribution

$$p(\mathbf{a}, \mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{a})p(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j)p(\mathbf{f}_j|\mathbf{a})p(\mathbf{a}), \quad (7.2)$$

where more details are given in Section 4.2. The posterior distribution of local independent experts in (4.60) can be also formulated as

$$p(\mathbf{f}|\mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j)p(\mathbf{f}_j). \quad (7.3)$$

For all three approaches, the corresponding graphical model is depicted in Figure 7.2a together with the corresponding prediction procedures in 7.2b, which are defined in the corresponding previous sections. Moreover, we provide a toy example as a preview, which compares the different GP approximations (with comparable time complexity) in Figure 7.3. Thereby, for each of the GP approximation method, the predictive mean (solid blue) and 95%-credible interval (dotted blue) are indicated and compared to full GP (black and shaded blue area). The number in the right bottom corner indicates the KL-divergence (2.12) to full GP. In the right bottom plot, our method *Correlated Product of Expert (CPoE)* is presented for a degree of correlation  $C = 2$  and sparsity  $\gamma = 1$ . We can notice, that sparse GPs can model the underlying function well as long as the global



inducing points (green) can represent the input space properly. Further, both purely local approaches, *minimal variance expert (minVar)* and the *generalized PoE (GPoE)*, as discussed in Sections 4.3.1.3 and 4.3.2.5, respectively, are local good approximation methods, however, they have difficulties between the independent experts. In particular, we note in the right top plot in Figure 7.3, that the method *minVar* has severe discontinuities. On the other hand, our method CPoE includes correlations between the local experts, resulting in an approximation, which can represent well local and global patterns in the data.

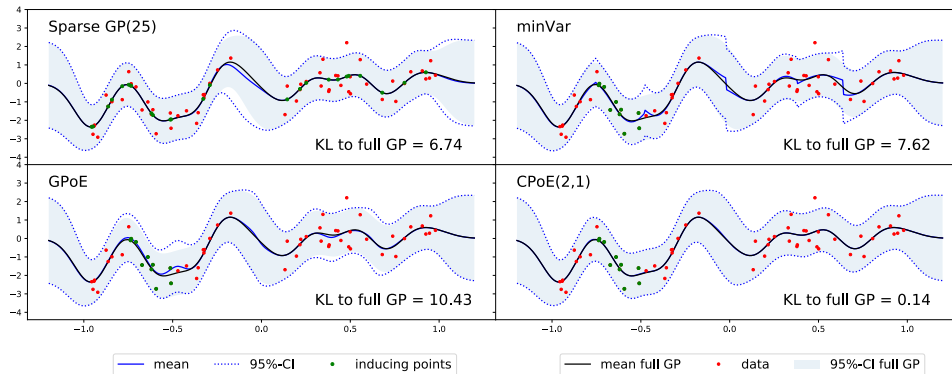


Figure 7.3. Different GP approximations methods for toy example.

## 7.2 Correlated Product of Experts

In this section, we present our GP regression method *Correlated Product of Expert CPoE*( $C, \gamma$ ), which is a generalization of the independent PoEs and sparse global GPs. The first generalization is the introduction of correlations between the experts, which can be adjusted by the parameter  $1 \leq C \leq J$  and allows to interpolate between local and global models. Secondly, similar to the sparse global approximation, our method allows to sparsify the inducing points by sparsity parameter  $0 < \gamma \leq 1$ , as illustrated in Figure 7.1.

In particular, we introduce the graphical model for our method in Section 7.2.1 and explain the local and sparse character of the prior approximation in Section 7.2.2. Further, we discuss how to perform inference and prediction with our model in Section 7.2.3 and 7.2.4, respectively. In Section 7.2.5, we show that the quality of our approximation consistently improves in terms of Kullback-Leibler-(KL)-divergence (2.12) w.r.t. full GP for increasing degree of correlation. Moreover, we present connections to Bayesian linear models in 7.2.6, deterministic and stochastic hyperparameter optimization techniques in 7.2.7, and com-

parisons against state-of-the-art GP approximation methods in a time versus accuracy sense, for synthetic as well as several real-world datasets, in Section 7.3. Thereby, we demonstrate superior performance of our proposed method for different kernels in multiple dimensions. Finally, Section 7.5 concludes the work.

### 7.2.1 Graphical Model

Assuming  $N = BJ$  data samples, which are divided into  $J$  ordered partitions (or experts) of size  $B$ , i.e.  $\mathcal{D} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$  with inputs  $\mathbf{X}_j \in \mathbb{R}^{B \times D}$  and outputs  $\mathbf{y}_j \in \mathbb{R}^B$ . We denote  $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$  the corresponding latent function values on the GP  $f$ . We abbreviate  $\mathbf{y} = \mathbf{y}_{1:J} \in \mathbb{R}^N$ ,  $\mathbf{X} = \mathbf{X}_{1:J} \in \mathbb{R}^{N \times D}$  and  $\mathbf{f} = \mathbf{f}_{1:J} \in \mathbb{R}^N$ .

**Definition 7.1 (Local Inducing Points)** *We introduce a set of local inducing points  $\{\mathbf{a}_j, \mathbf{A}_j\}_{j=1}^J$  with inducing inputs  $\mathbf{A}_j \in \mathbb{R}^{L \times D}$  and the corresponding inducing outputs  $\mathbf{a}_j = f(\mathbf{A}_j) \in \mathbb{R}^L$  of size  $L = \lfloor \gamma B \rfloor$  with  $0 < \gamma \leq 1$ .*

These  $L$  local inducing points  $(\mathbf{a}_j, \mathbf{A}_j)$  of expert  $j$  serve as local summary points for the data  $(\mathbf{y}_j, \mathbf{X}_j)$  where the sparsity level can be adjusted by  $\gamma$ . If  $\gamma = 1$ , the inducing inputs  $\mathbf{A}_j$  are exactly to  $\mathbf{X}_j$  and correspondingly  $\mathbf{a}_j = \mathbf{f}_j$ . We abbreviate  $\mathbf{a} = \mathbf{a}_{1:J} \in \mathbb{R}^M$  with  $M = LJ$  for all local inducing outputs with the corresponding local inducing inputs  $\mathbf{A} = \mathbf{A}_{1:J} \in \mathbb{R}^{M \times D}$ . Next, we model connections between the experts by a set of neighbour experts according to the given ordering.

**Definition 7.2 (Predecessor and Correlation Index Sets)** *We introduce  $\phi_i(j) \in \{1, \dots, j-1\}$ , corresponding to the index of the  $i$ th predecessor of the  $j$ th expert. For a given correlation parameter  $1 \leq C \leq J$ , we introduce the predecessor set  $\pi_C(j) = \bigcup_{i=1}^C \phi_i(j)$  satisfying*

$$\pi_C(j) \subset \{1, \dots, j-1\} \quad \text{and} \quad \pi_{C+1}(j) = \pi_C(j) \cup \phi_{C+1}(j),$$

*such that the size of the set  $I_j = |\pi_C(j)| = \min(j-1, C-1)$ . Further, we define the region of correlation with the correlation indices as  $\psi_C(j) = \pi_C(j) \cup j$  if  $j > C$  and  $\psi_C(j) = \psi_C(C) = \{1, \dots, C\}$  otherwise, so that  $|\psi_C(j)| = C$  for all  $j$ .*

The purpose of these predecessor and correlation indices are to model the local correlations among the experts of degree  $C$ . If for all  $j$  the indices  $\pi_C(j)$  are the  $C-1$  previous indices, we say that the predecessors are *consecutive* and *non-consecutive* otherwise. If  $C$  is clear from the context,  $\pi_C(j)$  and  $\psi_C(j)$  are abbreviated by  $\pi(j)$  and  $\psi(j)$ , respectively. Details about the specific choices of the ordering, partition, inducing points and predecessor indices are given in Section 7.2.7.1.

**Definition 7.3 (Graph)** We define a directed graph  $\mathcal{G}(V, E)$  with nodes  $V = \mathbf{a} \cup \mathbf{f} \cup \mathbf{y}$  and directed edges

$$E = \{ \{(\mathbf{a}_{\pi_c^i(j)}, \mathbf{a}_j)\}_{i=1}^{I_j} \cup \{(\mathbf{a}_{\psi_c^i(j)}, \mathbf{f}_j)\}_{i=1}^C \cup (\mathbf{f}_j, \mathbf{y}_j) \}_{j=1}^J,$$

where  $\pi_c^i(j)$  and  $\psi_c^i(j)$  denote the  $i$ th element in the corresponding set.

The directed graph  $\mathcal{G}$  is depicted in Fig. 7.5a(ii), where the local inducing points of the  $j$ th expert are connected with the inducing points of the  $I_j$  experts in  $\pi_c(j)$ . Further, the function values  $\mathbf{f}_j$  are connected in the region of correlation  $\psi_c(j)$  to the local inducing points. The graph  $\mathcal{G} = (V, E)$  can be equipped with a probabilistic interpretation, in particular, each node  $\mathbf{v} \in V$  and each incoming edge  $(\mathbf{v}_i, \mathbf{v}) \in E$  for all predecessors  $i = 1, \dots, I$  can be interpreted as a conditional probability density  $p(\mathbf{v} | \mathbf{v}_1, \dots, \mathbf{v}_I)$ .

**Proposition 7.1 (Graphical Model; Proof C.1)** We define a graphical model corresponding to the graph  $\mathcal{G}(V, E)$  with the conditional probability distributions

$$p(\mathbf{y}_j | \mathbf{f}_j) = \mathcal{N}(\mathbf{y}_j | \mathbf{f}_j, \sigma_n^2 \mathbb{I}), \quad (7.4)$$

$$p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j | \mathbf{H}_j \mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j), \quad (7.5)$$

$$p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \mathcal{N}(\mathbf{a}_j | \mathbf{F}_j \mathbf{a}_{\pi(j)}, \mathbf{Q}_j), \quad (7.6)$$

where (7.4) is the usual Gaussian likelihood for GP regression with noise variance  $\sigma_n^2$ , (7.5) the projection conditional and (7.6) the prior transition. Thereby, the matrices are defined as

$$\begin{aligned} \mathbf{H}_j &= \mathbf{K}_{X_j \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \in \mathbb{R}^{B \times LC}, \\ \bar{\mathbf{V}}_j &= \text{Diag}[\mathbf{K}_{X_j X_j} - \mathbf{K}_{X_j \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\psi(j)} X_j}] \in \mathbb{R}^{B \times B}, \\ \mathbf{F}_j &= \mathbf{K}_{A_j \mathbf{A}_{\pi(j)}} \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \in \mathbb{R}^{L \times LI_j}, \\ \mathbf{Q}_j &= \mathbf{K}_{A_j A_j} - \mathbf{K}_{A_j \mathbf{A}_{\pi(j)}} \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\pi(j)} A_j} \in \mathbb{R}^{L \times L}, \end{aligned}$$

with  $\mathbf{F}_1 = \mathbf{0}$  and  $\mathbf{Q}_1 = \mathbf{K}_{A_1 A_1}$ .

The two conditional distributions (7.5) and (7.6) can be derived from the true joint prior distribution  $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$ , as shown in Proof C.1. Alternatively, a generalization of this model can be obtained when using a modified projection distribution  $p(\mathbf{f}_j | \mathbf{a}_{\psi(j)})$  so that for  $C \rightarrow J$  and  $\gamma < 1$ , our model recovers a range of well known global sparse GP methods as described in Section 7.2.8 and Proposition 7.7. In any case, these local conditional distributions lead to the following joint distribution.

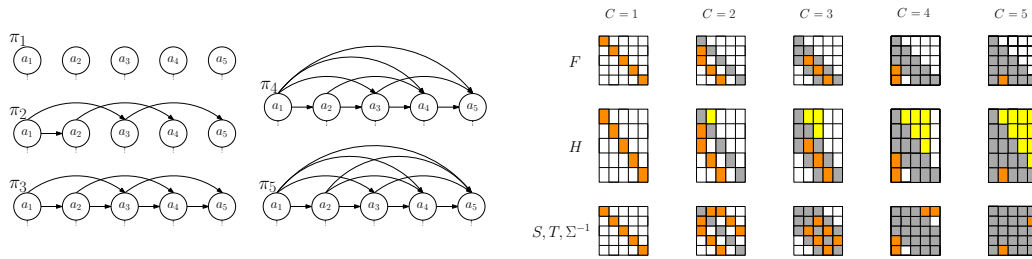


Figure 7.4. Correlation structure  $\pi_C$  between the  $J = 5$  experts for different degrees of correlation  $1 \leq C \leq J$ . Left: Graphical model among the local inducing points  $\mathbf{a}_j$ . Right: Structure of sparse transition matrix  $F$ , projection matrix  $H$ , prior precision  $S$ , likelihood precision  $T$  and posterior precision  $\Sigma^{-1}$ . Note that  $\pi_C$  does not have to be consecutive, e.g.  $2 \notin \pi_2(3)$ .

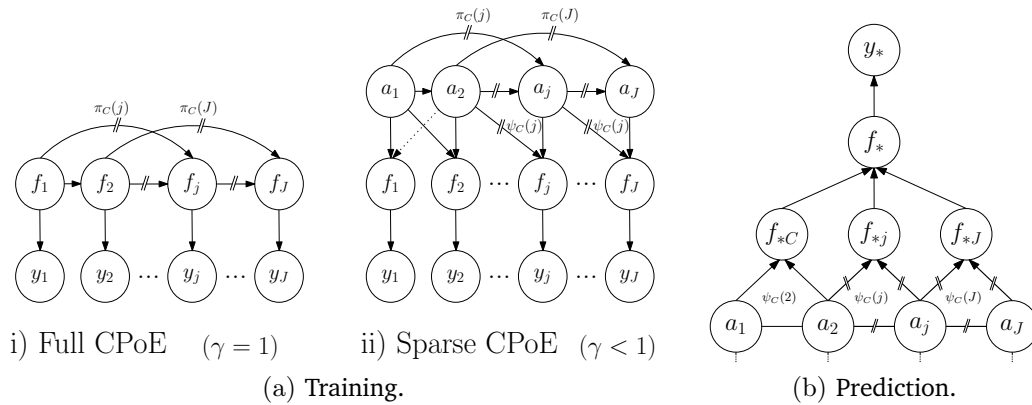


Figure 7.5. Graphical model for training and prediction of CPoE( $C, \gamma$ ).

**Definition 7.4 (Joint Distribution)** For the graphical model corresponding to graph  $\mathcal{G}$ , the joint distribution over all variables  $\mathbf{f}, \mathbf{a}, \mathbf{y}$  can be written as

$$q_{c,\gamma}(\mathbf{f}, \mathbf{a}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}). \quad (7.7)$$

In the case  $\gamma = 1$  and thus  $\mathbf{a} = \mathbf{f}$ , the joint distribution simplifies (Proof C.2) to

$$q_{c,1}(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}). \quad (7.8)$$

We use  $q = q_{c,\gamma}$  instead of  $p$  in order to indicate that it is an approximate distribution. The joint distributions in Definition 7.4 and the corresponding graphical model in Fig. 7.5a allow interesting comparisons to other GP models in Fig. 7.2 and the corresponding formulas (7.1), (7.2), (7.3). Whereas the conditioning set for full GP are all the previous latent values  $\mathbf{f}_{1:j-1}$ , for sparse GPs some global inducing points  $\mathbf{a}$  and for local independent experts the empty set, we propose to condition on the  $C - 1$  predecessors  $\mathbf{f}_{\pi(j)}$  (or a sparsified version in the general case). From this point of view, we can notice that our probabilistic model is equal to full GP, sparse GP and PoEs under certain circumstances, which are more precisely formulated in Proposition 7.7.

### 7.2.2 Sparse and Local Prior

The conditional independence assumptions between the experts, induced by the predecessor structure  $\pi_C$ , lead to an approximate prior  $q_{c,\gamma}(\mathbf{a})$  and approximate projection  $q_{c,\gamma}(\mathbf{f} | \mathbf{a})$ , yielding a sparse and local joint prior  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ .

**Proposition 7.2 (Joint Prior Approximation, Proof C.4)** The prior over all local inducing points  $\mathbf{a}$  in our CPoE model is

$$q_{c,\gamma}(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \mathcal{N}(\mathbf{a} | \mathbf{0}, \mathbf{S}_C^{-1}),$$

with the prior precision  $\mathbf{S}_C = \mathbf{S} = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F} \in \mathbb{R}^{M \times M}$ , where  $\mathbf{Q} = \text{Diag}[\mathbf{Q}_1, \dots, \mathbf{Q}_J] \in \mathbb{R}^{M \times M}$  and  $\mathbf{F} \in \mathbb{R}^{M \times M}$  is given as the sparse lower triangular matrix in Fig. 7.6. Moreover, the projection conditional is

$$q_{c,\gamma}(\mathbf{f} | \mathbf{a}) = \prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f} | \mathbf{H}\mathbf{a}, \bar{\mathbf{V}}),$$

where  $\mathbf{H} \in \mathbb{R}^{N \times M}$  defined in Figure 7.6 and  $\bar{\mathbf{V}} = \text{Diag}[\bar{\mathbf{V}}_1, \dots, \bar{\mathbf{V}}_J] \in \mathbb{R}^{N \times N}$ . Together with the exact likelihood  $p(\mathbf{y}|\mathbf{f}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I})$  determines the joint approximate prior

$$q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f}) q_{c,\gamma}(\mathbf{f}|\mathbf{a}) q_{c,\gamma}(\mathbf{a}).$$

$$F = \begin{bmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ -F_2^1 & I & 0 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ -F_3^1 & -F_3^2 & I & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & -F_4^1 & -F_4^2 & I & 0 & \ddots & 0 & 0 & 0 \\ 0 & -F_5^1 & 0 & -F_5^2 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots & I & 0 & 0 & 0 \\ j - 0 & 0 & \ddots & \dots & -F_j^1 & \dots & I & 0 & 0 \\ \vdots & \ddots & 0 & 0 & \vdots & \ddots & 0 & I & 0 \\ 0 & \dots & 0 & -F_j^1 & \dots & -F_j^{C-2} & 0 & -F_j^{C-1} & I \end{bmatrix}$$

$$H = \begin{bmatrix} H_1^1 & H_1^2 & H_1^3 & 0 & 0 & 0 & 0 & \dots & 0 \\ H_2^1 & H_2^2 & H_2^3 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ H_3^1 & H_3^2 & H_3^3 & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & H_4^1 & H_4^2 & H_4^3 & 0 & \ddots & 0 & 0 & 0 \\ 0 & H_5^1 & 0 & H_5^2 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots & H_{j-1}^C & 0 & 0 & 0 \\ j - 0 & 0 & \ddots & \dots & H_j^1 & \dots & H_j^C & 0 & 0 \\ \vdots & \ddots & 0 & 0 & \vdots & \ddots & 0 & H_{j+1}^C & 0 \\ 0 & \dots & 0 & H_j^1 & \dots & H_j^{C-2} & 0 & H_j^{C-1} & H_j^C \end{bmatrix}$$

Figure 7.6. Sparse transition matrix  $\mathbf{F} \in \mathbb{R}^{M \times M}$  and sparse projection matrix  $\mathbf{H} \in \mathbb{R}^{N \times M}$ , where  $\mathbf{F}_j^i \in \mathbb{R}^{L \times L}$  and  $\mathbf{H}_j^i \in \mathbb{R}^{B \times L}$  are the  $i$ th part of  $\mathbf{F}_j \in \mathbb{R}^{L \times L(C-1)}$  and  $\mathbf{H}_j \in \mathbb{R}^{B \times LC}$ , respectively, corresponding to the contribution of the  $i$ th predecessor  $\pi^i(j)$  and  $\psi^i(j)$ .

Note that the joint prior  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$  is Gaussian  $\mathcal{N}(\mathbf{0}, \mathbf{W})$  with dense covariance  $\mathbf{W}$  and sparse precision  $\mathbf{Z} = \mathbf{W}^{-1}$ , as shown in Fig. 7.13 in subsequent sections. If the predecessor set is consecutive, the matrix  $\mathbf{F}$  is a lower band (block)matrix with bandwidth  $C$  and in the non-consecutive case each row has exactly  $C$  non-zero blocks. The sparsity pattern of  $\mathbf{F}$  is inherited to the prior precision  $\mathbf{S} = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}$ , which is also a sparse matrix (see Fig. 7.4). For the consecutive case,  $\mathbf{S}$  is a block-band matrix with bandwidth  $2C - 1$ . Note that the inverse  $\mathbf{S}^{-1}$  is dense. The likelihood matrix  $\mathbf{H}$  is exact in the corner up to indices  $C$ , which ensures that we recover sparse global GP in the limiting case  $C = J$ . The quality of the approximation of our CPoE( $C, \gamma$ ) model is discussed in Section 7.2.5, where we show that  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$  converges to the true prior  $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$  for  $C \rightarrow J$ .

### 7.2.3 Inference

For our model, it is possible to infer analytically the posterior  $q_{c,\gamma}(\mathbf{a}|\mathbf{y})$  and the marginal likelihood  $q_{c,\gamma}(\mathbf{y})$  used later for prediction and for hyperparameter estimation, respectively.

**Proposition 7.3 (Posterior Approximation; Proof C.12)** *From the joint distribution, the latent function values  $\mathbf{f}$  can be integrated out yielding*

$$q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \int q_{c,\gamma}(\mathbf{f}, \mathbf{a}, \mathbf{y}) d\mathbf{f} = q_{c,\gamma}(\mathbf{y}|\mathbf{a})q_{c,\gamma}(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V})\mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1}),$$

with  $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbf{I} \in \mathbb{R}^{N \times N}$ . The posterior can be analytically computed by

$$q_{c,\gamma}(\mathbf{a}|\mathbf{y}) = \frac{q_{c,\gamma}(\mathbf{a}, \mathbf{y})}{q_{c,\gamma}(\mathbf{y})} \propto q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}^{-1}(\mathbf{a}|\boldsymbol{\eta}, \boldsymbol{\Lambda}), \quad (7.9)$$

with  $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Lambda} = \mathbf{T} + \mathbf{S} \in \mathbb{R}^{M \times M}$ ,  $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta} \in \mathbb{R}^M$ ,  $\boldsymbol{\eta} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{y} \in \mathbb{R}^M$  and  $\mathbf{T} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H} \in \mathbb{R}^{M \times M}$ .

The posterior precision matrix  $\boldsymbol{\Sigma}^{-1} = \mathbf{T} + \mathbf{S}$  inherits the sparsity pattern of the prior, since the addition of the projection precision  $\mathbf{T} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H}$  has the same sparsity structure, as depicted in Figs. 7.4 and 7.7. On the other hand, the posterior covariance  $\boldsymbol{\Sigma}$  is dense, therefore it will be never explicitly fully computed. Instead, the sparse linear system of equations  $\boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} = \boldsymbol{\eta}$  can be efficiently solved for  $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta}$ , as shown in Section 7.2.7.2.

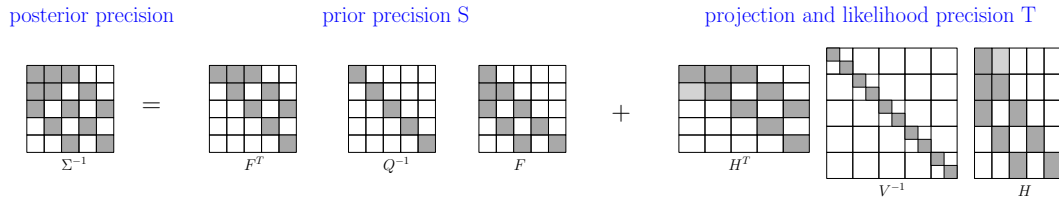


Figure 7.7. Sparse posterior precision approximation.

**Proposition 7.4 (Marginal Likelihood; Proof C.13)** *The marginal likelihood is*

$$q_{c,\gamma}(\mathbf{y}|\boldsymbol{\theta}) = q_{c,\gamma}(\mathbf{y}) = \int q_{c,\gamma}(\mathbf{y}, \mathbf{a}) d\mathbf{a} = \mathcal{N}(\mathbf{0}, \mathbf{P})$$

with  $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V} \in \mathbb{R}^{N \times N}$ , where all dependencies on  $\boldsymbol{\theta}$  of the matrices are omitted.

Further, in our CPoE model, the marginal likelihood  $q_{c,\gamma}(\mathbf{y}|\boldsymbol{\theta})$  can be analytically computed (Proposition 7.4) with the dense covariance matrix  $\mathbf{P}$ , which can be used for hyperparameter optimization, as shown in in Section 7.2.7.3. The posterior approximation  $q_{c,\gamma}(\mathbf{a}|\mathbf{y})$  as well as the approximate marginal likelihood  $q_{c,\gamma}(\mathbf{y})$  converge to the true distributions  $p(\mathbf{a}|\mathbf{y})$  and  $p(\mathbf{y})$ , respectively, for  $C \rightarrow J$ . In particular, they correspond exactly to the posterior and marginal likelihood of full GP and sparse global GP with  $\lfloor \gamma N \rfloor$  inducing points for  $C = J$ ,  $\gamma = 1$  and  $C = J$ ,  $\gamma < 1$ , respectively.

### 7.2.4 Prediction

The final predictive posterior distribution is obtained by an adaptation of the PoE aggregation, as presented in Section 4.3.2. The main idea is to consistently aggregate weighted local predictions from the experts, such that the correlations between them are taken into account, resulting in a smooth and continuous predictive distribution.

**Proposition 7.5 (Prediction Aggregation; Proof C.17)** *Similarly to the PoE aggregation in (4.58), we define the final predictive posterior distribution  $q_{c,\gamma}(f_*|\mathbf{y})$  for a query point  $\mathbf{x}_* \in \mathbb{R}^D$  as*

$$q_{c,\gamma}(f_*|\mathbf{y}) = \prod_{j=C}^J q_{c,\gamma}(f_{*j}|\mathbf{y})^{\beta_{*j}},$$

involving the local predictions  $q_{c,\gamma}(f_{*j}|\mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$  and weights  $\beta_{*j} \in \mathbb{R}$  defined in Proposition 7.6 and Definition 7.5, respectively. Moreover, the distribution  $q_{c,\gamma}(f_*|\mathbf{y}) = \mathcal{N}(m_*, v_*)$  with  $m_* = v_* \sum_{j=C}^J \beta_{*j} \frac{m_{*j}}{v_{*j}}$  and  $\frac{1}{v_*} = \sum_{j=C}^J \frac{\beta_{*j}}{v_{*j}}$  is analytically available. The final noisy prediction is  $p(y_*|\mathbf{y}) = \mathcal{N}(m_*, v_* + \sigma_n^2)$ .

The graphical model corresponding to this prediction procedure is depicted in Fig. 7.5b. Note that the first  $C - 1$  experts are only implicitly considered since  $\psi(j) = \psi(C)$  for  $j \leq C$ , resulting in  $J_2 = J - C + 1$  predictive experts, so that the proposed prediction aggregation interpolates between predictions from  $J_2 = J$  completely independent experts and predictions from  $J_2 = 1$  fully correlated expert, which is depicted in Figure 7.8.

**Proposition 7.6 (Local Predictions, Proof C.18)** *The local predictive distribution  $q_{c,\gamma}(f_{*j}|\mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$  of the  $j$ th expert are based on the region  $\psi(j)$  where the correlations are modelled and can be computed as*

$$q_{c,\gamma}(f_{*j}|\mathbf{y}) = \int p(f_{*j}|\mathbf{a}_{\psi(j)}) q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) d\mathbf{a}_{\psi(j)} = \mathcal{N}(\mathbf{h}_* \boldsymbol{\mu}_{\psi(j)}, \mathbf{h}_*^T \boldsymbol{\Sigma}_{\psi(j)} \mathbf{h}_* + v_*),$$

involving the local posteriors  $q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$  and the predictive conditional  $p(f_{*j}|\mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{h}_* \mathbf{a}_{\psi(j)}, v_*)$  (which is exactly defined in Proof C.18).

The local posteriors with mean  $\boldsymbol{\mu}_{\psi(j)}$  and covariance entries  $\boldsymbol{\Sigma}_{\psi(j)}$  could be obtained from the corresponding entries  $\psi(j)$  of  $\boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$ . However, computing explicitly some entries in the dense covariance  $\boldsymbol{\Sigma}$  based on the sparse precision



$\Sigma^{-1}$  is not straightforward, since in the inverse the blocks are no longer independent. However, we can exploit the particular sparsity and block-structure of our precision matrix and obtain an efficient implementation of this part, which is key to achieve a competitive performance of our algorithm. More details are given in Section 7.2.7.2.

**Definition 7.5 (Aggregation Weights)** *The input depending weights  $\beta_{*j} = \beta_j(X_*)$  at query point  $X_*$  models the influence of expert  $j$ . In particular, the unnormalized weights*

$$\bar{\beta}_{*j} = H[p(f_*)] - H[p(f_{*j}|y)] = \frac{1}{2} \log \left( \frac{v_{*0}}{v_{*j}} \right),$$

are set to the difference in entropy  $H$  (2.7) before and after seeing the data similarly proposed by Fleet [2014]. Thereby, the predictive prior is  $p(f_*) = \mathcal{N}(0, v_{*0})$  with  $v_{*0} = \mathbf{k}_{X_*X_*}$  and the predictive posterior defined in Proposition 7.6. The normalized weights are then obtained by  $\beta_{*j} = b^{-1} \bar{\beta}_{*j}^Z$  where  $b = \sum_{j=C}^J \bar{\beta}_{*j}^Z$  and  $Z = \log(N)C$ .

These weights bring the flexibility of increasing or decreasing the importance of the experts based on the predictive uncertainty. However, independent of the particular weights, our aggregation of the predictions is consistent since it is based on the *covariance intersection* method [Julier and Uhlmann, 1997], which is useful for combining several estimates of random variables with known mean and variance but unknown correlation between them. For the covariance method, the authors in Li et al. [2015] showed that (with normalized weights) (a) the accuracy of the fused estimate outperforms each local one and (b) provide consistent fused estimate, and thus reliable confidence information. The  $Z$  in the exponent of the normalization of the weights has a sharpening effect, so that the informative experts have even more weight compared to the non-informative experts for more data  $N$  and higher correlations  $C$ . This is a heuristic, but showed quite robust performance in experiments. Moreover, the consistency properties are more relevant than the particular weights.

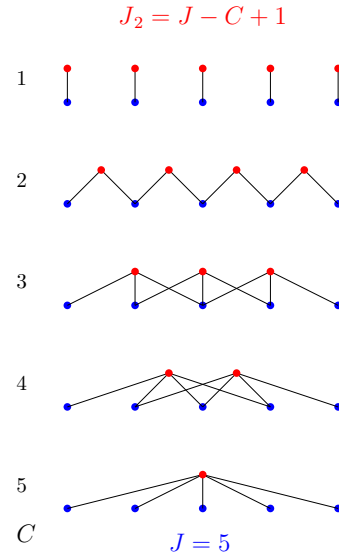


Figure 7.8. Prediction Aggregation in CPoE( $C, \gamma$ ) model with  $J$  base experts and  $J_2 = J - C + 1$  predictive experts.

### 7.2.5 Properties

**Proposition 7.7 (Equality; Proof C.3)** *Our model correlated Product of Experts  $CPoE(C, \gamma)$  is equal to full GP for  $C = J$  and  $\gamma = 1$ . For  $\gamma < 1$ , our model corresponds to sparse global GP with  $M_g = \lfloor \gamma N \rfloor$  inducing points. Further, with  $C = 1$  and  $\gamma = 1$ , our model is equivalent to independent PoEs. That is, we have*

$$CPoE(J, 1) = GP; \quad CPoE(J, \gamma) = SGP(\lfloor \gamma N \rfloor); \quad CPoE(1, 1) = GPoE^*,$$

where SGP refers to the FTIC model [Snelson and Ghahramani, 2006] and  $GPoE^*$  corresponds to  $GPoE$  [Fleet, 2014] with slightly different weights ( $Z = 1$ ) in the prediction.

In Section 7.2.8, we present a generalization of our model, so that  $CPoE(J, \gamma)$  corresponds to a range of other well known versions of sparse global GP, by changing the projection distribution and adding a correction term in the log marginal likelihood, similarly discussed in Schürch et al. [2020] and Chapter 5 for the global case. For instance, we can extend our model analogously to the variational version of Titsias [2009].

For correlations between the limiting cases  $C = 1$  and  $C = J$ , we investigate the difference in KL of the true GP model with  $CPoE(C, \gamma)$  and  $CPoE(C_2, \gamma)$  for  $1 \leq C \leq C_2 \leq J$ . For that reason, we define the difference in KL between the true distribution of  $\mathbf{x}$  and two different approximate distributions, i.e.

$$\mathbb{D}_{(C, C_2)}[\mathbf{x}] = KL[p(\mathbf{x}) \parallel q_{c, \gamma}(\mathbf{x})] - KL[p(\mathbf{x}) \parallel q_{c_2, \gamma}(\mathbf{x})]. \quad (7.10)$$

Similarly, we define the difference in KL of a conditional distribution  $\mathbf{x}|\mathbf{y}$  to be

$$\begin{aligned} \mathbb{D}_{(C, C_2)}[\mathbf{x}|\mathbf{y}] &= KL[p(\mathbf{x}|\mathbf{y}) \parallel q_{c, \gamma}(\mathbf{x}|\mathbf{y})] - KL[p(\mathbf{x}|\mathbf{y}) \parallel q_{c_2, \gamma}(\mathbf{x}|\mathbf{y})] \\ &= \mathbb{E}_{p(\mathbf{y})} \left[ \mathbb{E}_{p(\mathbf{x}|\mathbf{y})} \left[ \log \frac{q_{c_2, \gamma}(\mathbf{x}|\mathbf{y})}{q_{c, \gamma}(\mathbf{x}|\mathbf{y})} \right] \right], \end{aligned} \quad (7.11)$$

which follows from the the definition of KL (2.12). Using these definitions, we show that the approximation quality of the prior  $q_{c, \gamma}(\mathbf{a})$  and projection approximation  $q_{c, \gamma}(\mathbf{f}|\mathbf{a})$  monotonically improves for  $C \rightarrow J$ , so that the KL between the true  $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$  and our approximate joint distribution  $q_{c, \gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ , is decreasing for  $C \rightarrow J$ .

**Proposition 7.8 (Decreasing KL; Proof C.6)** *For any predecessor structure  $\pi_C$  and any  $0 < \gamma \leq 1$  and  $1 \leq C \leq C_2 \leq J$ , the difference in KL of the marginal prior, projection and data likelihood are non negative, i.e.*

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}] \geq 0, \quad \mathbb{D}_{(C, C_2)}[\mathbf{f}|\mathbf{a}] \geq 0, \quad \mathbb{D}_{(C, C_2)}[\mathbf{y}|\mathbf{f}] = 0,$$

so that the joint difference in KL is also non-negative

$$\mathbb{D}_{(C,C_2)}[\mathbf{a}, \mathbf{f}, \mathbf{y}] = \mathbb{D}_{(C,C_2)}[\mathbf{a}] + \mathbb{D}_{(C,C_2)}[\mathbf{f}|\mathbf{a}] + \mathbb{D}_{(C,C_2)}[\mathbf{y}|\mathbf{f}] \geq 0.$$

Moreover, we can quantify the approximation quality, in particular  $\mathbb{D}_{(C,C_2)}[\mathbf{a}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|}$  and  $\mathbb{D}_{(C,C_2)}[\mathbf{f}|\mathbf{a}] = \frac{1}{2} \log \frac{|\tilde{\mathbf{V}}_C|}{|\tilde{\mathbf{V}}_{C_2}|}$ .

The last statement demonstrates that our CPoE model is a sound GP prior precision approximation, which converges monotonically to the true prior for  $C \rightarrow J$ . Moreover, we can quantify the relative approximation quality of our model, which constitutes an approach of estimating the needed  $C$ , since it is independent of the true (and non-calculable) full GP distribution. The decreasing KL of the joint prior is depicted in Fig. 7.9 together with the decreasing KL of the posterior, marginal likelihood and predictive posterior.

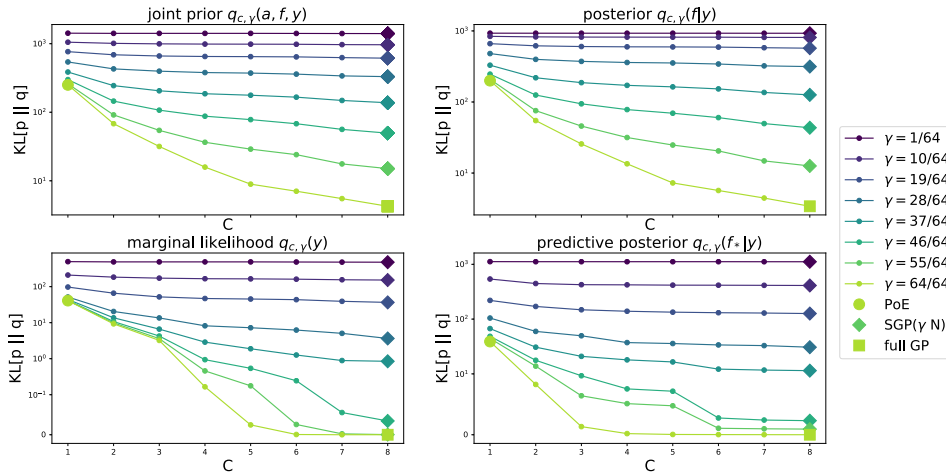


Figure 7.9. Decreasing  $\text{KL}[p||q]$  between true distribution  $p$  of full GP and approximate distribution  $q = q_{c,\gamma}$  of CPoE for increasing values of  $C$  and  $\gamma$  for the joint prior, posterior, marginal likelihood and predictive posterior for synthetic GP data ( $N = 1024, D = 2$ , SE kernel).

Note that, the accuracy of our proposed model is increasing for larger values of  $C$ , as theoretically shown in Proposition 7.8 and illustrated in Figure 7.9. However, there is a trade-off between accuracy and time. This means, a reasonable value for the degree of correlation  $C$  has to be determined in practice. In our empirical experiments, we found values  $C \in \{2, 3, 4\}$  a good choice for competitive performance. However, this value could be easily increased in our algorithm.

### 7.2.6 Generalizations of Bayesian Linear Models

In this section, we briefly summarize and establish some connections to the Bayesian linear model and their use for GP approximations. These models are summarized in Table 7.1 and depicted in Figure 7.10. In the previous chapters, the connections between GP models and Bayesian linear model with basis functions and Gaussian observation noise are discussed. In particular, we show in Chapter 5, that the ordinary Bayesian linear model can be extended with an additional noise term, so that it can represent sparse global GP models. These two models are shown in Figure 7.10 in i) and ii), where the deterministic relations between the weights  $\mathbf{a}$  and  $\mathbf{f}_j$  in i) are depicted as red arrows, whereas the probabilistic relation in ii) are shown as green arrows. These two models both exploit some global weights, or inducing points  $\mathbf{a}$ , which summarize the relationship among the latent function values  $\mathbf{f}_j$ . In this chapter instead, we investigate models with a different dependency structure among the latent function values. A possible structure would be to directly introduce a local dynamics among the latent function values  $\mathbf{f}_j$ , as shown in iii), which is also known as a *dynamical Bayesian linear model* in the literature. This model can be further improved, by instead modeling a dynamic among some hidden variables/weights  $\mathbf{a}_j$ , as indicated in model iv). The dependency structure of the latter two models are restricted to a chain structure. In this chapter, we even generalize this model to arbitrary dependency structures among the hidden variable, as depicted with the blue arrows in the model vi) in Figure 7.10, which we call *hidden conditional Bayesian linear model*. In particular, in this chapter, we introduce a novel model, which unifies all these generalized dependency structures among the latent functions values, resulting in a novel framework for GP approximations.

### 7.2.7 Computational Details

#### 7.2.7.1 Graph

The graphical model in Section 7.2.1 is generically defined and several choices are left for completely specifying the graph  $\mathcal{G}(V, E)$  for a particular dataset: the partition method, the ordering of the partition, the selection of the predecessors and the local inducing points. We tried to make these choices as simple and straightforward as possible with focus on computational efficiency, however, there might be more sophisticated heuristics. Concretely, we use KD-trees [Maneewongvatana and Mount, 2001] for partitioning the data  $\mathcal{D}$  into  $J$  regions and the ordering starts with a random partition, which is then greedily extended by the closest partition in euclidean distance (represented by the mean of the induc-

Model	Reference	Equations	Assumptions	GP Model
Gaussian Observ. Model		$\mathbf{y}_j = \mathbf{f}_j + \varepsilon_j$	$\varepsilon_j \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbb{I})$	
i) Bayesian Linear Model	Section 2.3.4	$\mathbf{f}_j = \mathbf{H}_j \mathbf{a}$	$\mathbf{a} \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$	Degenerate GP
ii) Extended Bayesian Linear Model	Definition 5.1	$\mathbf{f}_j = \mathbf{H}_j \mathbf{a} + \gamma_j$	$\mathbf{a} \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$ $\gamma_j \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{V}}_j)$	Sparse Global GP
iii) Dynamical Bayesian Linear Model	special case of Proposition 7.1	$\mathbf{f}_j = \mathbf{F}_j \mathbf{f}_{j-1} + \mathbf{q}_j$	$\mathbf{a}_0 \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$ $\mathbf{q}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_j)$	CPoE with $\gamma = 1$ and $C = 1$
iv) Hidden Dynamical Bayesian Linear Model	special case of Proposition 7.1	$\mathbf{f}_j = \mathbf{H}_j \mathbf{a}_j + \gamma_j$ $\mathbf{a}_j = \mathbf{F}_j \mathbf{a}_{j-1} + \mathbf{q}_j$	$\mathbf{a}_0 \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$ $\gamma_j \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{V}}_j)$ $\mathbf{q}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_j)$	CPoE with $\gamma < 1$ and $C = 1$
vi) Hidden Conditional Bayesian Linear Model	Proposition 7.1	$\mathbf{f}_j = \mathbf{H}_j \mathbf{a}_{\psi_j} + \gamma_j$ $\mathbf{a}_j = \mathbf{F}_j \mathbf{a}_{\pi_j} + \mathbf{q}_j$	$\mathbf{a}_0 \sim \mathcal{N}(\mathbf{0}, \Sigma_0)$ $\gamma_j \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{V}}_j)$ $\mathbf{q}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_j)$	CPoE with $\gamma < 1$ and $C > 1$

Table 7.1. Summary of Bayesian linear models and their use for GP approximations.

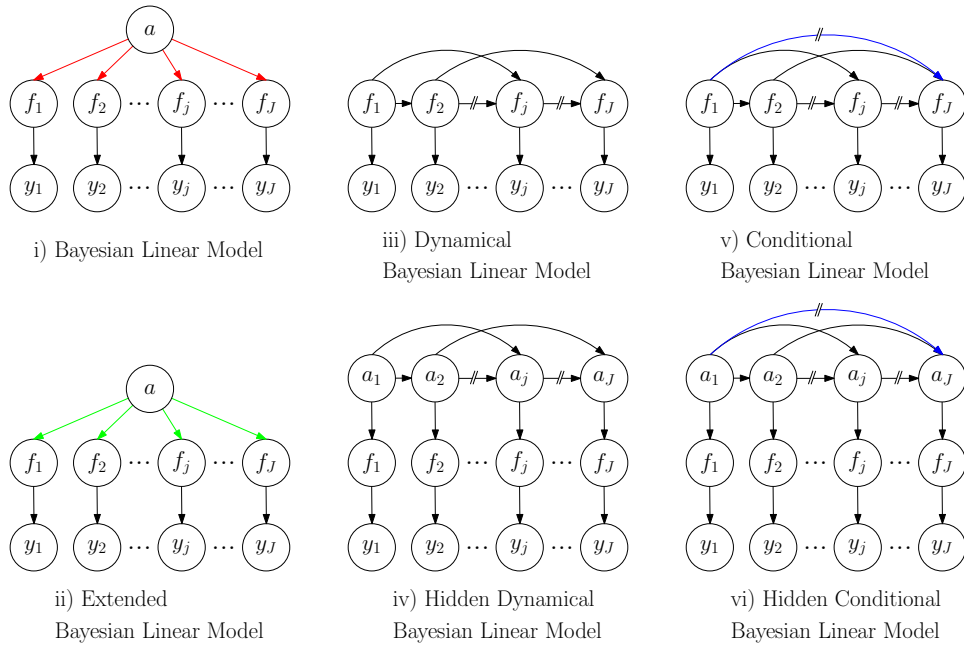


Figure 7.10. The Bayesian linear model with basis functions and possible generalizations. Our unifying model in this chapter can represent all of these models.

ing points). The  $L \leq B$  inducing inputs  $A_j \in \mathbb{R}^{L \times D}$  of the  $j$ th partition (or expert) can be in principle arbitrary, however, in this work they are chosen as a random subset of the data inputs  $X_j \in \mathbb{R}^{B \times D}$  of the  $j$ th expert for the sake of simplicity. For the predecessors (block-)indices  $\pi_c$ , the  $C - 1$  closest partitions among the previous (according to the ordering) predecessors in euclidean distance are greedily selected. These explained concepts are illustrated for a toy example in Fig. 7.11.

Note that, there might be more sophisticated methods to find in particular an appropriate partition. We have also investigated partitions based on *K-Means* and splits according to a particular or random data dimension. We found that KD-trees provide rather robust results in a range of synthetic and real-world data sets in different dimensions. However, the investigation of improved partitions in our algorithm is postpone to future work. An interesting data-dependent partition approach is presented by Buschjäger et al. [2019].

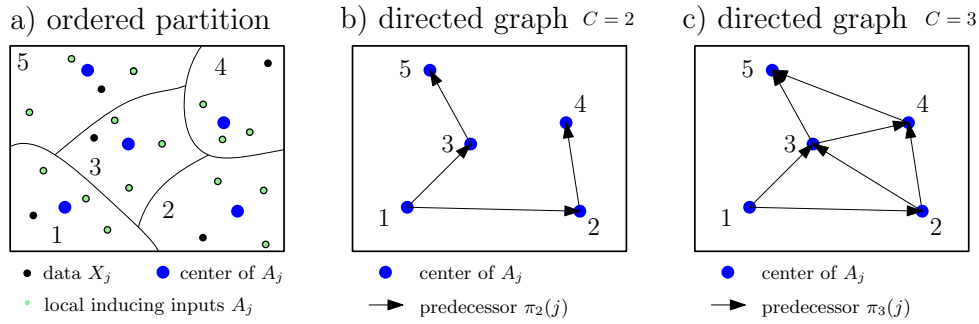


Figure 7.11. Toy example for partition, local inducing points, predecessors and directed graph illustrated for  $D = 2$  with  $J = 5$  experts/partitions each with  $B = 4$  samples,  $\gamma = 0.75$  and thus  $L = 3$  local inducing points. In a) the ordered partition with the data (black), local inducing points (green) and their mean (blue) are depicted. In b) and c) the directed graph for  $C = 2$  and  $C = 3$  are shown with corresponding predecessors  $\pi_2(1) = \{\}$ ,  $\pi_2(2) = \{1\}$ ,  $\pi_2(3) = \{1\}$ ,  $\pi_2(4) = \{2\}$ ,  $\pi_2(5) = \{3\}$  and  $\pi_3(1) = \{\}$ ,  $\pi_3(2) = \{1\}$ ,  $\pi_3(3) = \{1, 2\}$ ,  $\pi_3(4) = \{2, 3\}$ ,  $\pi_3(5) = \{3, 4\}$ , respectively. In the previous example,  $\pi_3$  is consecutive and  $\pi_2$  is non-consecutive.

### 7.2.7.2 Solving Linear System & Partial Inversion

For solving the sparse linear system  $\Sigma^{-1}\mu = \eta$  in Proposition 7.3, sparse Cholesky decomposition is exploited. In particular,  $M\Sigma^{-1}M^T = LL^T =: Y$  is computed so that  $\nu$  and  $\bar{\mu}$  can be efficiently obtained via solving  $L\nu = \eta$  and  $L^T\bar{\mu} = \nu$ , respectively. Thereby, the matrix  $M$  is a so-called fill-reduction permutation matrix,

such that the Cholesky matrix  $\mathbf{L}$  is as sparse as possible and thus  $\boldsymbol{\mu} = \mathbf{M}^{-1}\bar{\boldsymbol{\mu}}$ . Note that,  $\mathbf{M}$  is computed only via the structure on the block level, which is only  $J$  dimensional instead of  $JL$ .

Additionally to the mean  $\boldsymbol{\mu}$ , also some entries  $\Sigma_{\psi(j)}$ , in the covariance matrix  $\boldsymbol{\Sigma}$ , has to be explicitly computed, which are needed for computing local predictions (Section 7.2.4) and (derivatives) of the marginal likelihood (Section C.2), respectively. Therby, the needed entries correspond to the non-zeros in the precision matrix  $\boldsymbol{\Sigma}^{-1}$ . Computing efficiently these entries is not straightforward, since in the inverse the blocks are no longer independent. However, we can exploit the particular sparsity and block-structure of our precision matrix and obtain an efficient implementation of this part, which is key to achieve a competitive performance of our algorithm. Computing some entries in  $\mathbf{Z} = \mathbf{Y}^{-1}$  is also known as *partial inversion*. We adapted the approach in Takahashi [1973], where the recursive equations with  $J$  blocks for computing the full inverse  $\mathbf{Z}$  are provided

$$\mathbf{Z}_{B_j} = -\mathbf{Z}_{C_j}\mathbf{L}_{B_j}\mathbf{L}_{A_j}^{-1} \quad \text{and} \quad \mathbf{Z}_{A_j} = \mathbf{L}_{A_j}^{-T}\mathbf{L}_{A_j}^{-1} - \mathbf{Z}_{B_j}^T\mathbf{L}_{B_j}\mathbf{L}_{A_j}^{-1},$$

where the recursion starts from  $j = J$  with  $\mathbf{Z}_{A_j} = \mathbf{L}_{A_j}^{-T}\mathbf{L}_{A_j}^{-1}$ .

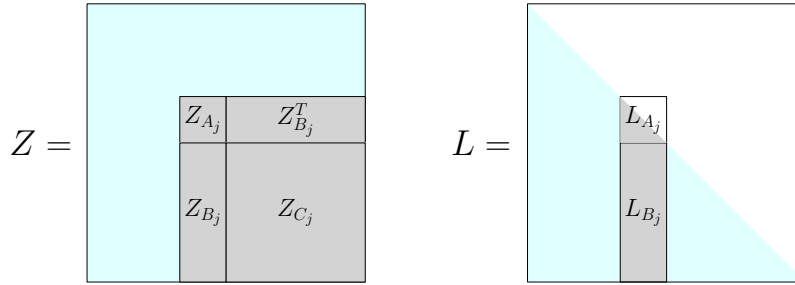


Figure 7.12. Shape of  $\mathbf{Z}$  and  $\mathbf{L}$ .

Instead of computing the full inverse using this recursion, we exploited the block-sparsity structure of our posterior precision matrix in order to gain significant speed-up. We only computed the entries in the inverse  $\mathbf{Z}$ , which are symbolically non-zero in  $\mathbf{L}$ . In Algorithm 2, we provide efficient pseudo-code using sparse-block-matrices in the block-sparse-row format.

Alternatively, for computing the Cholesky factor of  $\boldsymbol{\Sigma}^{-1} = \mathbf{S} + \mathbf{H}\mathbf{V}^{-1}\mathbf{H}$ , we could directly exploit that the prior precision  $\mathbf{S} = \mathbf{F}^T\mathbf{Q}^{-1}\mathbf{F}$  is already decomposed into an upper/lower-triangular form, since  $\mathbf{F}$  lower triangular. However, when updating the Cholesky factor with  $\mathbf{H}\mathbf{V}^{-1}\mathbf{H}$  needs quadratic time in the number of nonzeros for each expert.

**Algorithm 2** Partial Sparse Block Inversion

**Require:** Cholesky matrix  $L$  of size  $JB \times JB$  in block-sparse-row (bsr) format with  $J \times J$  total blocks, block-size  $B$  and  $N$  non-zero blocks. Data array  $d$  of size  $N \times B \times B$ , the column-block-indices  $r$  of size  $N$ , row-block-pointer  $p$  of length  $J + 1$ , and lookup table  $M$  of dimension  $J \times J$ .

**Ensure:** The lower part of the symmetric partial inversion is computed in bsr-format with the same row-block-indices  $r$  and row-block-pointer  $p$  and data array  $z$  of size  $N \times B \times B$ .

```

for  $i \in \{J, \dots, 1\}$  do
   $L_A^{-1} \leftarrow d[M[i, i], :, :]^{-1}$ 
   $z[M[i, i]] \leftarrow (L_A^{-1})^T \cdot L_A^{-1}$ 
  for  $j \in \{r[p[i + 1]], r[p[i + 1] - 1], \dots, r[p[i]]\}$  do
     $Q \leftarrow 0$ 
    for  $l \in \{r[p[i]], r[p[i] + 1], \dots, r[p[i + 1]]\}$  do
       $R \leftarrow z[M[j, l]]$ 
      if  $l > j$  then
         $R \leftarrow R^T$ 
      end if
       $Q \leftarrow Q + R \cdot d[M[l, i]] \cdot L_A^{-1}$ 
    end for
     $z[M[i, j]] \leftarrow z[M[i, j]] - Q$ 
  end for
end for

```

## 7.2.7.3 Hyperparameter Estimation

In Section 7.2, we introduced CPoE for fixed hyperparameters  $\theta$ , where implicitly all distributions are conditioned on  $\theta$ . However, we omitted these dependencies on  $\theta$  in the most cases for the sake of brevity. Similar to full GP, sparse GP or PoEs, we can use the ML-II approach, where the *log marginal likelihood (LML)* can be used as an objective function for optimizing the few hyperparameters  $\theta$ , as discussed in Section 2.3.3. The log of the marginal likelihood of our model, formulated in Section 7.2.3, is  $\log q(\mathbf{y}|\theta) = \log \mathcal{N}(\mathbf{0}, \mathbf{P})$  with  $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}$ , which can be efficiently computed, as detailed in the Appendix of [Schürch et al., 2022], and can be used for *deterministic optimization* with full batch  $\mathbf{y}$  for moderate sample size  $N$ . However, in order to scale this parameter optimization part to larger number of samples  $N$  in a competitive time, *stochastic optimization* techniques exploiting subsets of data have to be developed similarly done for the global



sparse GP model (SVI Hensman et al. [2013]; REC Schürch et al. [2020]; IF Kania et al. [2021]). We adapt the hybrid IF approach of Kania et al. [2021], where we can also exploit an independent factorization of the log marginal likelihood, which decomposes into a sum of  $J$  terms, so that it can be used for stochastic optimization. This constitutes a very fast and accurate alternative for our method, as shown in the Appendix of [Schürch et al., 2022], and will also be exploited in Section 7.3 for large data sets.

### Prior on Hyperparameters

Alternatively to the ML-II approach, we can use the MAP-II approach, as introduced in Section 2.3.3. The MAP-II approach is based on the log of the posterior distribution  $p(\boldsymbol{\theta}|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ , where  $p(\boldsymbol{\theta})$  is a suitable prior on the hyperparameters, yielding  $\log p(\boldsymbol{\theta}|\mathbf{y}) = \log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$  as objective function. In the following, we assume  $p(\boldsymbol{\theta}) = \prod_j p(\boldsymbol{\theta}_j)$  and a log-normal prior for each hyperparameter  $p(\boldsymbol{\theta}_j) = \log \mathcal{N}(\boldsymbol{\theta}_j | \nu_j, \lambda_j^2)$  for means  $\nu_j$  and variances  $\lambda_j^2$ , similarly introduced in Section 2.3.4.6. For the deterministic case, the MAP estimator can be straightforwardly computed by just adding the log prior on  $\boldsymbol{\theta}$  to the batch log marginal likelihood, i.e.  $\log p(\boldsymbol{\theta}|\mathbf{y}) = \mathcal{L}(\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$ . Similarly for the stochastic case, the stochastic MAP can be decomposed as  $\log p(\boldsymbol{\theta}|\mathbf{y}) \approx \sum_{j=1}^J (l_j(\boldsymbol{\theta}) + \frac{1}{J} \log p(\boldsymbol{\theta}))$ , where  $l_j(\boldsymbol{\theta})$  is the  $j$ th term in the stochastic marginal likelihood (defined properly in the Appendix of Schürch et al. [2022]), so that it can be used again for stochastic mini-batch optimization. An example using priors for the hyperparameters is presented in Section 7.3.1.

#### 7.2.7.4 Complexity

The time complexity for computing the posterior and the marginal likelihood in our algorithm is dominated by  $J$  operations which are cubic in  $LC$  (inversion, matrix-matrix multiplication, determinants). This leads to  $\mathcal{O}(NB^2\alpha^3)$  and  $\mathcal{O}(NB\alpha^2)$  for time and space complexity, respectively, where we define the approximation quality parameter  $\alpha = C\gamma$ . Similarly, for  $N_t$  testing points, the time and space complexities are  $\mathcal{O}(NB\alpha^2N_t)$  and  $\mathcal{O}(N\alpha N_t)$ . Note that, an approach to remove the dependency on  $N$  is outlined in the Appendix of [Schürch et al., 2022]. In Table 7.2, the asymptotic complexities of our model together with other GP algorithms are indicated. It is interesting that for  $\alpha = 1$ , our algorithm has the same asymptotic complexity for training as sparse global GP with  $M_g = B$  global inducing points, but we can have  $M = LJ = \gamma BJ = \gamma N$  total local inducing points! Thus, our approach allows much more total local inducing points  $M$  in the order of  $N$  (e.g.  $M = 0.5N$  with  $C = 2$ ), whereas for sparse global GP

	full GP	sparse GP	PoE	CPoE
time	$\mathcal{O}(N^3)$	$\mathcal{O}(NM_g^2)$	$\mathcal{O}(NB^2)$	$\mathcal{O}(NB^2\alpha^3)$
space	$\mathcal{O}(N^2)$	$\mathcal{O}(NM_g)$	$\mathcal{O}(NB)$	$\mathcal{O}(NB\alpha^2)$
time <sub>t</sub>	$\mathcal{O}(N^2N_t)$	$\mathcal{O}(M_g^2N_t)$	$\mathcal{O}(NBN_t)$	$\mathcal{O}(NBN_t\alpha^2)$
space <sub>t</sub>	$\mathcal{O}(NN_t)$	$\mathcal{O}(M_gN_t)$	$\mathcal{O}(NN_t)$	$\mathcal{O}(NN_t\alpha)$
#pars	$ \boldsymbol{\theta} $	$MD +  \boldsymbol{\theta} $	$ \boldsymbol{\theta} $	$ \boldsymbol{\theta} $

Table 7.2. Complexity for training, pointwise predictions for  $N_t$  points and number of optimization parameters for different GP algorithms.

usually  $M_g \ll N$ . This has the consequence that the local inducing points can cover the input space much better and therefore represent much more complicated functions. As a consequence, there is also no need to optimize the local inducing points resulting in much fewer parameters to optimize. Consider the following example with  $N = 10'000$  in  $D = 10$  dimensions. Suppose a sparse global GP model with  $M_g = 500$  global inducing points. A CPoE model with the same asymptotic complexity has a batch size  $B = M_g = 500$  and  $\alpha = 1$ . Therefore, we have  $J = \frac{N}{B} = 20$  experts and we choose  $C = 2$  and  $\gamma = \frac{1}{2}$  such that we obtain  $L = \gamma B = 250$  local inducing points per experts and  $M = \gamma N = 5'000$  total local inducing points! Further, the number of hyperparameters to optimize with a SE kernel is for global sparse GP  $M_g D + |\boldsymbol{\theta}| = 5012$ , whereas for CPoE there are only  $|\boldsymbol{\theta}| = 12$ . For an extended version of this section consider Appendix of Schürch et al. [2022].

### 7.2.8 Generalized CPoE

In this section, we present a generalization of our CPoE model. The graphical model defined in Definition 7.4 recovers the sparse global GP model FITC [Snelson and Ghahramani, 2006] in the limiting case  $C \rightarrow J$ , as shown in Proposition 7.7. In this section, we present a generalization of our CPoE model, such that it recovers other sparse global GP models, such as VFE [Titsias, 2009] or PEP [Bui et al., 2017b]. As shown in Chapter 5 based on [Schürch et al., 2020] for the global case, these models differ in the training only by the choice of the projection matrix  $\bar{\mathbf{V}}_j$  in Definition 7.4 and in the hyperparameter optimization by a modification of the log marginal likelihood  $\log q(\mathbf{y}|\boldsymbol{\theta})$  in Section 7.2.7.3. These two changes can also be made for our local sparse CPoE model. In particular, we

modify  $\bar{\mathbf{V}}_j$  and  $a_j$  according to the values in Table 5.2 in the projection conditional

$$p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j | \mathbf{H}_j \mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j),$$

and in the lower bound to the log marginal likelihood

$$\mathcal{L}(\boldsymbol{\theta}) = \log q(\mathbf{y} | \boldsymbol{\theta}) - \sum_{j=1}^J a_j(\boldsymbol{\theta}).$$

This generalizes the CPoE method, such that for  $C \rightarrow J$ , we recover the mentioned global methods in Table 5.2. Thereby, the involved Nyström matrix  $\mathbf{Q}_{X_j X_j}$  is in this context slightly adapted, namely,

$$\mathbf{Q}_{X_j X_j} = \mathbf{K}_{X_j A_{\psi(j)}} \mathbf{K}_{A_{\psi(j)} A_{\psi(j)}}^{-1} \mathbf{K}_{A_{\psi(j)} X_j}.$$

The setting in VFE [Titsias, 2009] is particularly interesting, since it constitutes in the global case a direct posterior approximation derived via a variational maximization of the lower bound of the log marginal likelihood as shown in Section 4.2.3.1. Moving a bit away from the true marginal likelihood of full GP has the effect that overfitting (w.r.t. full GP) can not happen when optimizing the hyperparameters with the lower bound. This is particularly important when all inducing inputs are optimized, as it is usually recommended in sparse global methods. Note that, this is not the case for our model, since it allows to have a number of inducing points in the order of the number of data samples. In the adapted "local VFE" model, when using  $\bar{\mathbf{V}}_j = \mathbf{0}$  and minimize also  $a_j = \text{tr}\{\mathbf{K}_{X_j X_j} - \mathbf{Q}_{X_j X_j}\}$ , has the effect that the model is locally variationally optimal. However, it would be interesting to directly derive a lower bound analogously to Titsias [2009], so that the posterior of our CPoE model is rigorously connected to full GP. Since this is not a straight-forward extension, we postpone this task to future work. Below, we present the connection to full GP for this adapted model in the joint prior sense analogously to Proposition 7.8 for the local FITC model.

**Proposition 7.9 (Local VFE)** *Using the deterministic conditional  $q(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j | \mathbf{H}_j \mathbf{a}_{\psi(j)}, \mathbf{0})$  in the graphical model in Definition 7.4 and Proposition 7.1, that is, setting the covariance  $\bar{\mathbf{V}}_j = \mathbf{0}$  in the projection step, recovers global VFE for  $C \rightarrow J$ . Moreover, the difference in KL to full GP of the joint prior is also decreasing. In particular, the difference in KL of the prior of the local VFE model for  $1 \leq C \leq C_2 \leq J$  is*

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|} \geq 0.$$

Further, the difference in KL of the projection is

$$\mathbb{D}_{(C,C_2)}[\mathbf{y}|\mathbf{a}] = \frac{1}{2\sigma_n^2} \text{tr}\{\tilde{\mathbf{V}}_C - \tilde{\mathbf{V}}_{C_2}\} \geq 0.$$

The overall prior approximation quality is

$$\mathbb{D}_{(C,C_2)}[\mathbf{a}, \mathbf{y}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|} + \frac{1}{2\sigma_n^2} \text{tr}\{\tilde{\mathbf{V}}_C - \tilde{\mathbf{V}}_{C_2}\},$$

where

$$\text{tr}\{\tilde{\mathbf{V}}_C\} = \sum_{i=1}^N K_{\mathbf{x}_i \mathbf{x}_i} - K_{\mathbf{x}_i \mathbf{A}_{\psi(j_i)}} K_{\mathbf{A}_{\psi(j_i)} \mathbf{A}_{\psi(j_i)}}^{-1} K_{\mathbf{A}_{\psi(j_i)} \mathbf{x}_i}.$$

Compared to the FITC model, note the difference in the trace instead of the fraction of the log-determinants.

### 7.2.9 Additional Results

In this section, we present some additional theoretical results for our CPoE model.

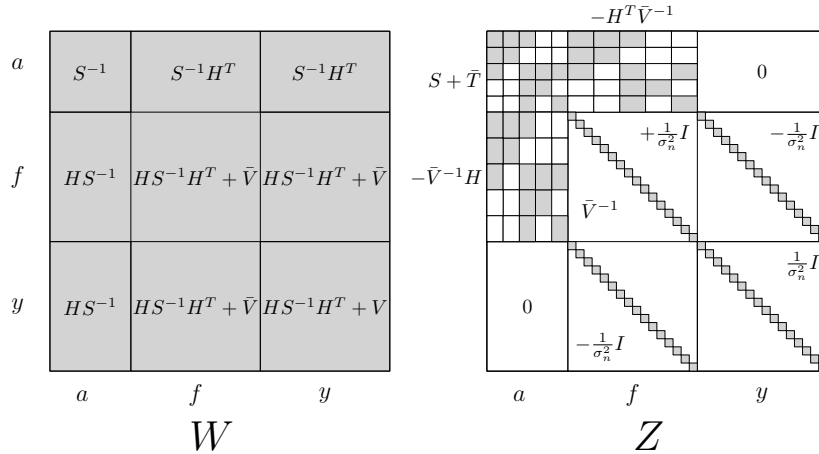


Figure 7.13. Covariance  $W$  and precision  $Z = W^{-1}$  of joint prior approximation  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y}) = \mathcal{N}(\mathbf{0}, W)$  of CPoE model. Compare Proof C.19.

**Proposition 7.10 (Prior Approximation II; Proof C.15)** *Alternatively to Proposition 7.2, the prior approximation  $q(\mathbf{a}) = \mathcal{N}(\mathbf{a}|\mathbf{0}, S^{-1})$  can be equivalently written as*

$$q(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{S}_{(j)}^{-1}),$$

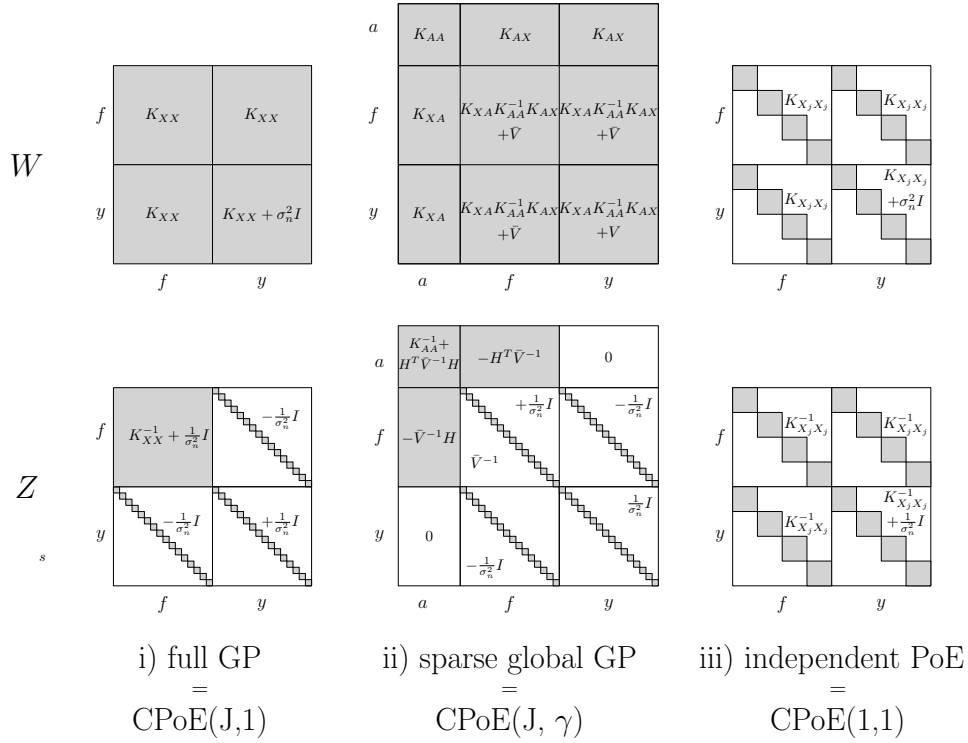


Figure 7.14. Covariance  $\mathbf{W}$  and precision  $\mathbf{Z} = \mathbf{W}^{-1}$  of joint prior of different GP models. Compare Figure 7.13 for the corresponding covariance and precision matrices for CPoE model. Note that we used  $\mathbf{H} = \mathbf{K}_{XA}\mathbf{K}_{AA}^{-1}$  and  $\bar{\mathbf{V}}$  the same as in the local CPoE.

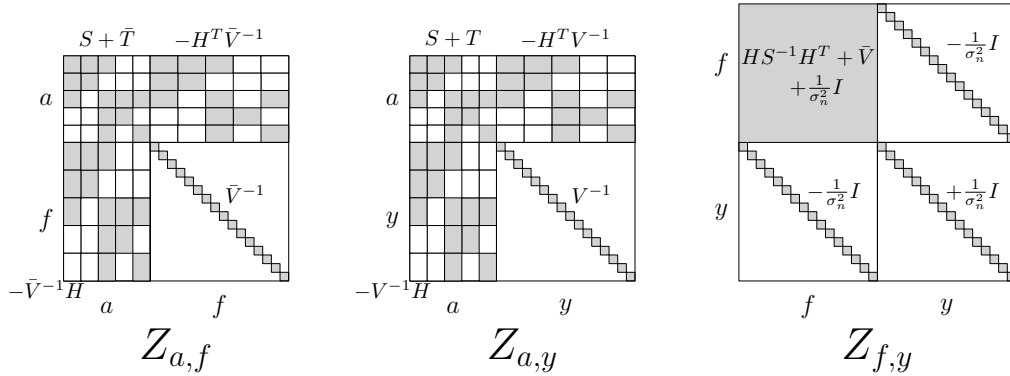


Figure 7.15. Marginalized precision corresponding to  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{a},\mathbf{f}}^{-1})$ ,  $q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{a},\mathbf{y}}^{-1})$  and  $q_{c,\gamma}(\mathbf{f}, \mathbf{a}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{f},\mathbf{y}}^{-1})$ , respectively. Thereby we used the notation  $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$ ,  $\bar{\mathbf{T}} = \mathbf{H}^T \bar{\mathbf{V}}^{-1} \mathbf{H}$  and  $\mathbf{T} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H}$ . Note that the corresponding dense covariance matrices are directly obtained from  $\mathbf{W}$  in Fig. 7.13, by selecting the corresponding entries.

with  $\mathbf{S}_{(j)} = \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j$ ,  $\tilde{\mathbf{F}}_j = [-\mathbf{F}_j \quad \mathbb{I}]$  and  $\pi^+(j) = \pi(j) \cup j$ . Further, the prior precision matrix can also be written as

$$\mathbf{S} = \sum_{j=1}^J \bar{\mathbf{S}}_{(j)},$$

where  $\bar{\mathbf{S}}_{(j)} \in \mathbb{R}^{M \times M}$  is the augmented matrix consisting of  $\mathbf{S}_{(j)} \in \mathbb{R}^{LC \times LC}$  at the entries  $[\pi^+(j), \pi^+(j)]$  and 0 otherwise.

**Proposition 7.11 (Prior Approximation III; Proof C.16)** *Alternatively to Proposition 7.2 and Proposition 7.10, the prior approximation  $q(\mathbf{a})$  can be equivalently written as*

$$q(\mathbf{a}) = \prod_{j=1}^J \frac{p(\mathbf{a}_j, \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_{\pi(j)})} = \prod_{j=C}^J \frac{p(\mathbf{a}_{\pi^+(j)})}{p(\mathbf{a}_{\pi(j)})},$$

which is a Gaussian  $\mathcal{N}(\mathbf{a} | \mathbf{0}, \mathbf{S}^{-1})$  with prior precision

$$\mathbf{S} = \sum_{j=1}^J \bar{\mathbf{K}}_{A_{\pi^+(j)} A_{\pi^+(j)}}^{-1} - \bar{\mathbf{K}}_{A_{\pi(j)} A_{\pi(j)}}^{-1},$$

where  $\bar{\mathbf{K}}_{A_{\phi} A_{\phi}}^{-1} \in \mathbb{R}^{M \times M}$  is the augmented matrix consisting of  $\mathbf{K}_{A_{\phi} A_{\phi}}^{-1} \in \mathbb{R}^{T \times T}$  at the entries  $[\phi, \phi]$  and 0 otherwise with  $T = |\phi|$ .

**Proposition 7.12 (Exact Diagonal of Prior; Proof C.14)** *The precision matrix  $\mathbf{S}_C$  of the prior approximation  $q_C(\mathbf{a})$  is exact on the diagonal, that is,*

$$\text{tr}(\mathbf{S}_C \mathbf{K}_{AA}) = JL,$$

where  $JL$  is the dimension of the matrices.

**Proposition 7.13 (Band-Diagonal Approximation)** *In the consecutive case, i.e.  $\psi(j) = \{j - C + 1, \dots, j\}$ , the block-entries*

$$\mathbf{S}_{[\psi(j), \psi(j)]}^{-1} = \mathbf{K}_{AA[\psi(j), \psi(j)]},$$

are equal, which means that the block-band-diagonals  $-C + 1, \dots, 0, \dots, C - 1$  of the both matrices are the same. For the case  $C = J$  it holds  $\mathbf{S}^{-1} = \mathbf{K}_{AA}$ .

**Proposition 7.14 (Decreasing Prior Entropy; Proof C.10)** *For any predecessor structure  $\pi_C$  as in Definition 7.2, the entropy  $H$  of the approximate prior  $q_C(\mathbf{a})$  is decreasing for  $C \rightarrow J$ , in particular*

$$H[q_1(\mathbf{a})] \geq \dots \geq H[q_j(\mathbf{a})] \geq \dots \geq H[q_J(\mathbf{a})],$$

where it holds  $H[q_J(\mathbf{a})] = H[p(\mathbf{a})]$  and

$$H[q_j(\mathbf{a})] = \frac{1}{2} \log |\mathbf{Q}_C| + \frac{M}{2} (1 + \log 2\pi).$$

Similar results can be obtained for the joint prior  $q_C(\mathbf{a}, \mathbf{f}, \mathbf{y})$ .

From the last proposition we know, that increasing the degree of correlation  $C$  add always more information to the prior. In particular, the prior of complete independent PoEs (i.e.  $C = 1$ ) encodes the least of information, since all correlations between the experts are missing, whereas the prior of full GP incorporates the most information, since all correlations are modeled.

### 7.3 Comparison

In this section, we compare the performance with competitor methods for GP approximations using several *synthetic* and *real world* datasets as summarized in Table 7.5a. In this thesis, we provided a slightly shortened versions of this section, for more details about the experiments we refer to the Appendix of Schürch et al. [2022].

First, we examine the accuracy vs. time performance of different GP algorithms for fixed hyperparameters in a simulation study with *synthetic* GP data. We generated  $N = 8192$  data samples in  $D = 2$  with 5 repetitions from the sum of two SE kernels with a shorter and longer lengthscale, such that both global and local patterns are present in the data, as indicated in Figure 7.16. In Figure 7.17, the mean results are shown for the KL and RMSE to full GP, the 95%-coverage and the log marginal likelihood against time in seconds. The results for sparse GP with increasing number of global inducing points  $M_g$  are shown in blue, the results for minVar, GPoE and BCM for increasing number of experts  $J$  are depicted in red, cyan and magenta, respectively. For CPoE, the results for increasing correlations  $C$  are shown in green. We observe superior performance of our method compared to competitors in terms of accuracy compared to full GP vs. time. Moreover, one can observe that the confidence information of our model are reliable

	KL					time				
	concrete	mg	space	abalone	kin	concrete	mg	space	abalone	kin
fullGP	0.0	0.0	0.0	0.0	0.0	7.3	25.5	114.8	237.9	161.5
SGP(100)	352.9	9.9	108.1	15.6	603.7	36.4	14.4	46.6	58.9	42.2
minVar	122.2	19.4	63.6	25.1	211.0	1.5	2.0	7.2	6.4	9.3
GPoE	174.4	54.2	98.0	50.3	342.3	1.4	1.9	7.2	6.3	9.4
GRBCM	224.6	69.1	105.6	36.4	129.8	1.7	2.3	6.5	7.6	11.9
<b>CPoE(1)</b>	111.1	12.2	63.0	16.8	152.4	1.5	2.1	7.8	6.4	9.2
<b>CPoE(2)</b>	89.6	8.4	36.5	8.1	79.9	2.1	2.8	10.6	7.5	12.9
<b>CPoE(3)</b>	82.2	7.8	36.3	6.2	46.9	2.5	3.1	12.9	9.3	19.8
<b>CPoE(4)</b>	<b>79.5</b>	<b>7.6</b>	<b>36.0</b>	<b>4.7</b>	<b>32.8</b>	2.8	3.3	14.9	10.4	27.8

Table 7.3. Results with deterministic optimization. Average KL to full GP (left) and time (right) for different GP methods and 5 datasets with 10 repetitions. More results are provided in Appendix C.3.

already for small approximation orders, since it is based on the consistent covariance intersection method. More details about the experiment is provided in the supplementary material of Schürch et al. [2022].

Second, we benchmark our method with 10 real world datasets as summarized in Table 7.5a. For the 5 smaller datasets in the first block we use deterministic parameter optimization, for which the average results over 10 training/testing splits are depicted in Table 7.3. In particular, the KL to full GP (left) and time (right) for different GP methods are shown. Similarly, the average accuracy and computational times for the 4 larger datasets in the second block, where stochastic parameter optimization is exploited, can be found in Table 7.4.

In general, the local methods perform better than the global sparse method. Further, the performance of our correlated PoEs is superior to the one of independent PoEs for all datasets. In particular, the KL to full GP can be continuously improved for increasing degree of correlation, i.e. larger  $C$  values. The time for CPoE(1) is comparable with the independent PoEs and for increasing  $C$ , our approxima-

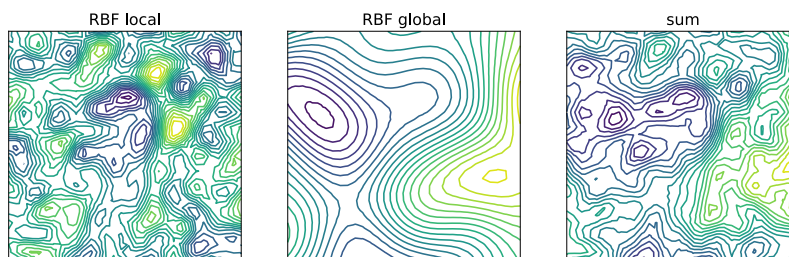


Figure 7.16. Generated data by a sum of 2 SE-kernels with local and global lengthscales.



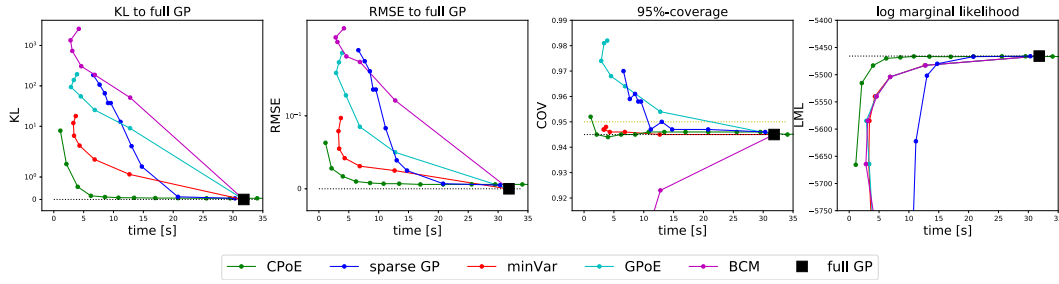


Figure 7.17. Average accuracy vs. time performance of different GP algorithms.

	CRPS				time			
	kin2	cadata	sarcos	casp	kin2	cadata	sarcos	casp
SGP(500)	0.183	0.253	0.069	0.329	112.1	346.9	730.1	632.9
SGP(1000)	0.166	0.252	0.063	0.325	244.1	727.6	1718.5	1362.5
minVar	0.173	0.257	0.052	0.294	14.4	28.2	71.3	45.8
GPoE	0.193	0.289	0.086	0.302	14.4	28.3	71.4	45.6
GRBCM	0.164	0.262	0.060	0.310	16.5	33.5	84.6	59.4
CPoE(1)	0.163	0.259	0.052	0.289	13.8	24.5	45.4	45.1
CPoE(2)	0.155	0.251	0.051	0.287	18.9	33.4	67.3	70.3
CPoE(3)	<b>0.151</b>	<b>0.249</b>	<b>0.051</b>	<b>0.282</b>	31.7	52.0	134.3	123.8

Table 7.4. Results with stochastic optimization. Average CRPS (left) and time (right) for different GP methods and 4 datasets with 5 repetitions. More results are provided in Section C.3 in the Appendix.

tion has a moderate increase in time with a significant decrease in KL. For more details about the experiments we refer to the Appendix of Schürch et al. [2022], and for more results, e.g. tables including standard deviations, are provided in Appendix C.3.

### 7.3.1 Application

In this Section, our method is applied on time series data with covariates using a rather complicated and non-stationary kernel together with priors on the hyperparameters, as discussed in Section 7.2.7.3. In recent work [Corani et al., 2021], the authors have shown that GPs constitute a competitive method for modelling time series using a sum of several kernels, including priors on the hyperparameters, which are previously learnt from a large set of different time series. We adapt their idea by using a slightly modified kernel and the same priors. In particular, for two data points  $\mathbf{x}_1 = [t_1, x_{1,2}, \dots, x_{1,D}]$  and  $\mathbf{x}_2 = [t_2, x_{2,2}, \dots, x_{2,D}]$

	$N$	$D$	$N_{test}$	$J$		KL	KL IN	KL OUT	time
concrete	927	8	103	4	full GP	0.0	0.0	0.0	404.3
mg	1247	6	138	8	SGP(100)	120.9	110.5	146.7	56.3
space	2797	6	310	8	SGP(200)	114.9	65.6	238.3	75.2
abalone	3760	8	417	16	minVar	503.0	406.5	744.5	20.7
kin	5192	8	3000	16	GPoE	328.0	336.0	307.9	20.4
kin2	7373	8	819	16	GRBCM	393.4	382.1	421.8	28.2
cadata	19640	8	1000	64	<b>CPoE(1)</b>	289.5	255.1	375.5	20.5
sarcos	43484	21	1000	128	<b>CPoE(2)</b>	113.1	108.5	124.3	36.8
casp	44730	9	1000	128	<b>CPoE(3)</b>	86.4	61.9	147.6	39.7
elecdemand	2184	3	15288	13	<b>CPoE(4)</b>	<b>58.3</b>	<b>59.4</b>	<b>55.5</b>	52.9

(a) Description of datasets. (b) KL to full GP and time of different methods.

Table 7.5. Summary of used datasets and results for the *elecdemand* time series.

we model the kernel as the sum of 4 components

$$k_{\theta}(\mathbf{x}_1, \mathbf{x}_2) = k_{p_1}(t_1, t_2) + k_{p_2}(t_1, t_2) + k_{SM}(t_1, t_2) + k_{SE}(\mathbf{x}_1, \mathbf{x}_2),$$

where  $k_{p_1}$  and  $k_{p_2}$  are standard periodic kernels with period  $p_1$  and  $p_2$ , respectively, to capture the seasonality components of the time series. Further,  $k_{SM}$  is a spectral-mixture kernel and  $k_{SE}$  a squared-exponential kernel. Note that, the former 3 kernels only depend on the first variable, which corresponds to time, whereas the SE-kernel depends on all variables, thus models the influence of the additional covariates. With our CPoE model it is straightforward to handle time series with covariates, as opposed to other time series methods [Benavoli and Corani; Corani et al., 2021; Sarkka et al., 2013; Hyndman and Athanasopoulos, 2018]. The kernel  $k_{\theta}$  depends on several hyperparameters  $\theta$  for which we use the parametrization in Corani et al. [2021]. We assume a log-normal prior on  $\theta$  as described in Section 7.2.7.3 in which the corresponding means and variances are taken from Table 1 in Corani et al. [2021]. We demonstrate the MAP estimation for  $\theta$  on the *elecdemand* time series ([Hyndman, 2020], Table 7.5a), which contains the electricity demand as response  $y$  together with the time as the first variable  $x_1$ , the corresponding temperature as  $x_2$  and the indicator variable whether it is a working day as  $x_3$ , which is depicted in the plots in Fig. 7.18 on the left, where we shifted the first and third variable in the second plot for the sake of clarity. Similarly as in the previous section, we run full GP, SGP, PoEs and CPoE and optimized the hyperparameter deterministically, using the MAP as objective function taking into account the priors. The results are provided in Table 7.5b and in Fig. 7.18 on the right, which again show very competitive perfor-

mance also for a general kernel with priors on the hyperparameters. A complete description of the experiment is given in the Appendix of Schürch et al. [2022]. Overall, the experiments in this section demonstrate superior performance of CPoE compared to state-of-the-art GP approximation algorithms. In particular, competitive results can be experimentally verified for synthetic GP data as well as for real world data in multiple dimensions, using different kernels, and including priors on the hyperparameters.

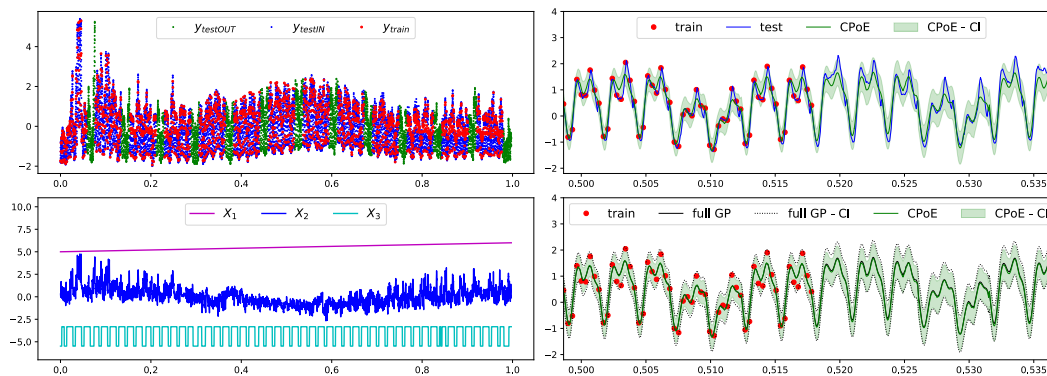


Figure 7.18. Time series data with covariates and prior on hyperparameters.

## 7.4 CPoE via Gaussian Belief Propagation

In this section, we present an alternative inference approach for the model *Correlated Product of Experts (CPoE)*, as presented in Section 7.2 and evaluated in Section 7.3. Although the underlying model is equivalent, the inference approach is different, so that new insights and connections to a range of other probabilistic methods and algorithms can be established. The CPoE algorithm presented in Section 7.2 is based on the exploitation of local conditional independence assumptions, as illustrated in Figure 7.19. This figure constitutes a different visualization of the graphical models of full GP, sparse global GPs, and CPoE in Figures 7.2a.i)-ii) and 7.5a.i), respectively.

This local representation leads to a block-sparse precision matrix  $\mathbf{\Lambda} = \mathbf{\Sigma}^{-1}$  and natural mean  $\boldsymbol{\eta}$  of the posterior distribution  $q(\mathbf{a}|\mathbf{y}) = \mathcal{N}^{-1}(\mathbf{a}|\boldsymbol{\eta}, \mathbf{\Lambda}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \mathbf{\Sigma})$  of the local inducing points  $\mathbf{a}$ , as presented in Proposition 7.3. For inference in this model, the posterior mean  $\boldsymbol{\mu}$  and some entries in the posterior covariance  $\mathbf{\Sigma}_{\psi(j)}$  are required. In the previous sections, our approach was to solve the corresponding block-sparse linear system  $\mathbf{\Lambda}\boldsymbol{\mu} = \boldsymbol{\eta}$  based on sparse block Cholesky

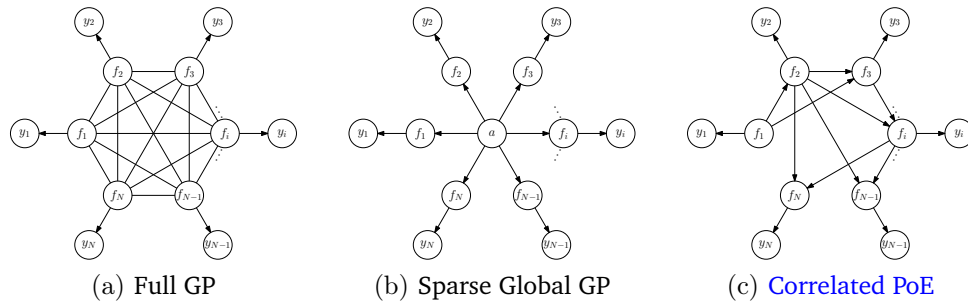


Figure 7.19. Fully connected latent function values in full GP (a), global inducing points in sparse GPs (b), and local correlation structure of CPoE (c).

decomposition of the precision matrix for obtaining  $\boldsymbol{\mu}$ . Further, the needed entries  $\boldsymbol{\Sigma}_{\psi(j)}$  are computed via *partial inversion* based on an adapted approach by Takahashi [1973], as discussed in Section 7.2.7.2.

Another approach for inference in this model constitutes *Belief Propagation (BP)* [Koller and Friedman, 2009, Section 11.3.2], or, in particular in our case *Gaussian Belief Propagation (GaBP)* [Koller and Friedman, 2009, Section 14.2.3]; [Murphy, 2012, Section 20.2.3], which can be used for computing the marginal posterior distributions for each factor in the product of the joint posterior distribution. BP can be implemented as iterative message passing procedure on the posterior factor graph and operates by sending and receiving local messages from neighboring nodes. We will see, that the resulting local posterior marginal distributions in the factor graph correspond exactly to the needed entries  $\mathcal{N}(\mathbf{a}_{\psi(j)} | \boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$ .

The GaBP is exact if the underlying graph is a tree, otherwise, *loopy BP* [Koller and Friedman, 2009, Chapter 11] can be applied, where several iterations are performed. If the loopy GaBP message passing process converges, which can be proven for certain conditions [Weiss and Freeman, 1999; Malioutov et al., 2006], the resulting posterior beliefs encode the correct marginal mean  $\boldsymbol{\mu}_{\psi(j)}$  of the joint distribution, however, the estimated marginal posterior covariances  $\boldsymbol{\Sigma}_{\psi(j)}$  are generally not correct. In particular, they are underestimates of the true covariances, so that the resulting posteriors are "overconfident", which is not a desired property. An exact solution for general underlying graphs can be achieved by constructing as so-called *junction tree* or *clique tree* [Koller and Friedman, 2009, Chapter 10], and to perform BP on this graph instead. Since this is a tree, all marginal distributions are then exact.

We first discuss the construction of the junction tree based on the initial directed graph of the experts in Section 7.4.1. Next, the particular factorization of the

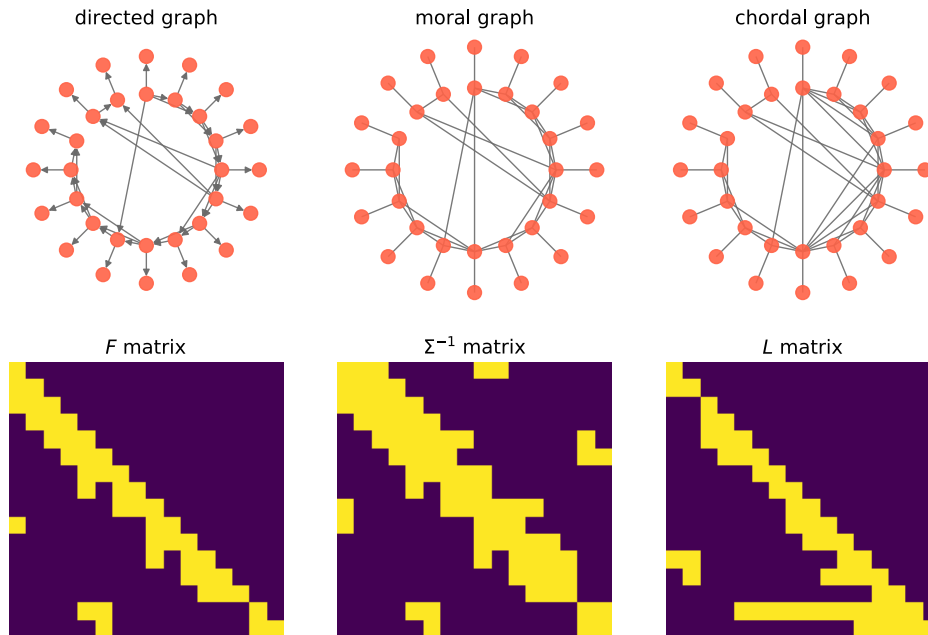


Figure 7.20. Connection between sparse matrices and graphs. In particular, we show the correlation structure between the experts in the first column in form of a directed graph and the transition matrix  $F$ . The resulting posterior precision matrix  $\Sigma^{-1}$  can be encoded as the corresponding moral graph, as shown in the second column. Finally, the structure of the Cholesky factor  $L$  of the posterior precision matrix has the same structure as the chordal completion.

joint distribution is discussed in 7.4.2, together with the Gaussian belief propagation in Section 7.4.3. Finally, we outline some future work and provide concluding remarks in Section 7.4.4.

### 7.4.1 Construction of Junction Tree

First, the construction of a junction tree from a given directed graph over the sparse experts is discussed. This general procedure is described in [Koller and Friedman, 2009, Section 10.4], summarized in Algorithm 3, and illustrated in Figures 7.20-7.22. We start from a given directed graph  $\mathcal{G} = (V, E)$  with nodes  $V = \mathbf{f} \cup \mathbf{y} \cup \mathbf{a} = \{\mathbf{v}_1, \dots, \mathbf{v}_{2N+M}\}$  and edges  $E$  according to Definition 7.3. From this graph  $\mathcal{G}$ , we compute the corresponding undirected *moral* graph  $\mathcal{M} = (V, E_M)$ . In particular, all directed edges  $E$  are interpreted as undirected and some additional edges are added to the original edges to obtain  $E_M$ . Intuitively, those additional edges are added, so that all experts in the predecessor set are

connected. Based on this moralized graph  $\mathcal{M}$ , a *chordal* graph  $\mathcal{C} = (V, E_C)$  is computed by adding further edges. This is also known as *triangulation* of a graph. Note that, exact algorithms for a chordal completion with minimal number of additional edges are not feasible in general, since this problem is NP-complete. There are several useful heuristics, e.g. *min-fill*, which show often good performance for computing chordal completions with additional edges as few as possible. These steps are illustrated in Figure 7.20 for an example with  $C = 2$  and  $\gamma = 1$ . In particular, the latent function values  $f_j$  and the observed data  $y_j$  are depicted in the inner and outer circle, respectively. In the lower row, we depict also the corresponding sparse matrices from Section 7.2, since the task of computing a minimum chordal completion of a moral graph and the problem of finding an optimal permutation matrix in the Cholesky decomposition with minimal number of fill-in entries is closely related [Barfoot, 2020].

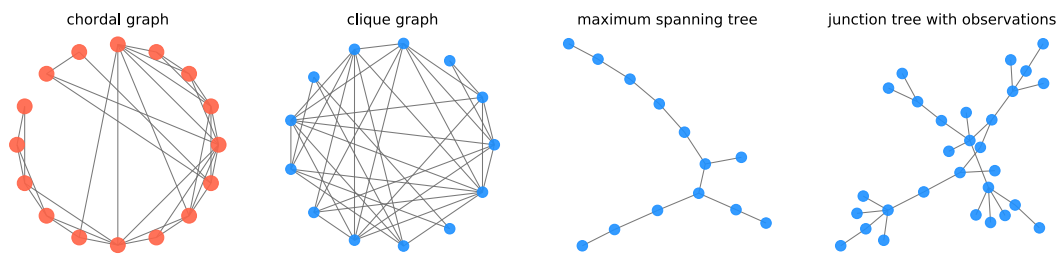


Figure 7.21. Construction of junction tree from chordal graph via clique graph.

Based on the obtained chordal graph  $\mathcal{C}$ , all cliques  $Q = \{q_1, \dots, q_T\}$  are computed. These cliques are fully connected subgraphs in  $\mathcal{C}$ , and thus, each clique contains some variables from the original set of variable  $V$ . Note that, the size of the largest clique is also called *treewidth*. These cliques can be used to compute the clique graph  $\mathcal{CG} = (Q, E_Q)$ , in which an edge is present in  $E_Q$ , iff two cliques  $q_i$  and  $q_j$  have at least one common variables and the edge weight correspond to the number of common variables. From this clique graph  $\mathcal{CG}$ , a maximum spanning tree is computed, which corresponds then to the final junction tree  $\mathcal{JT} = (Q, E_J)$ . This is illustrated in Figure 7.21, where in the first 3 plots only the unobserved nodes, i.e. the ones involving  $f_j$ , are depicted for the sake of simplicity. In particular, we depict the chordal graph over the (unobserved) variables in  $V$  (red) in the first plot, the corresponding clique graph over the cliques  $Q$  (blue) in the second plot, and the maximum spanning tree of the clique graph in the third plot. This graph corresponds to the junction tree over the cliques, however, involving only the latent values  $f_j$ . In the last plot, we depict the corresponding junction tree including the observed variables  $y_j$ , where the base structure is the same as in previous graph. This junction tree structure can then

be used to pass the local messages among the edges, so that the local marginal posteriors can be computed. In Figure 7.22, we present several examples of this construction for increasing degree of correlation  $C$  between the experts.

---

**Algorithm 3 - Construction of Junction Tree**

 adapted from [Koller and Friedman, 2009, Section 10.4]
 

---

**Require:** directed graph  $\mathcal{G} = (V, E)$  as in Def. 7.3

**Ensure:** compute junction tree  $\mathcal{JT} = (Q, E_J)$ 

- ▷ **compute moralized graph**  $\mathcal{M} = (V, E_M)$ : construct the undirected moral graph from the directed graph  $\mathcal{G}$
  - ▷ **compute chordal graph**  $\mathcal{C} = (V, E_C)$ : make triangulation of moral graph  $\mathcal{M}$  with as few additional edges, e.g. with *min-fill* heuristics
  - ▷ **compute clique graph**  $\mathcal{CG} = (Q, E_Q)$ : compute all cliques  $Q$  of the chordal graph  $\mathcal{C}$ , that is, find all fully connected subgraphs
  - ▷ **compute junction tree**  $\mathcal{JT} = (Q, E_J)$ : compute maximum spanning tree of clique graph  $\mathcal{CG}$ , weights are number of common variables
- 

## 7.4.2 Factorization of Joint Distribution

In the previous section, the construction of a junction tree is discussed. This is independent of the particular associated distribution to the initial directed graph. In this section, we explain the connection between the factors in this distribution and the nodes in the clique graph.

### 7.4.2.1 Initial Factors of Variables

We first define the initial factors in the undirected graph  $\mathcal{G} = (V, E)$  over the initial variables  $V = \mathbf{f} \cup \mathbf{y} \cup \mathbf{a} = \{\mathbf{v}_1, \dots, \mathbf{v}_{2N+M}\}$ . In particular, we introduce the general formulation for the joint distribution  $q(V) = q(\mathbf{f}, \mathbf{y})$  of the graphical model in (7.7), that is,

$$q(V) = \prod_{\mathbf{v}_i \in V} \phi_0(\mathbf{v}_i) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}). \quad (7.12)$$

Therefore, the initial factors are  $\phi_0(\mathbf{v}_j) = p(\mathbf{f}_j | \mathbf{f}_{\pi(j)})$  for  $1 \leq j \leq N$ ,  $\phi_0(\mathbf{v}_j) = p(\mathbf{y}_j | \mathbf{f}_j)$  for  $N < j \leq 2N$ , and  $\phi_0(\mathbf{v}_j) = p(\mathbf{a}_j | \mathbf{a}_{\pi(j)})$  for  $j > 2N$ .

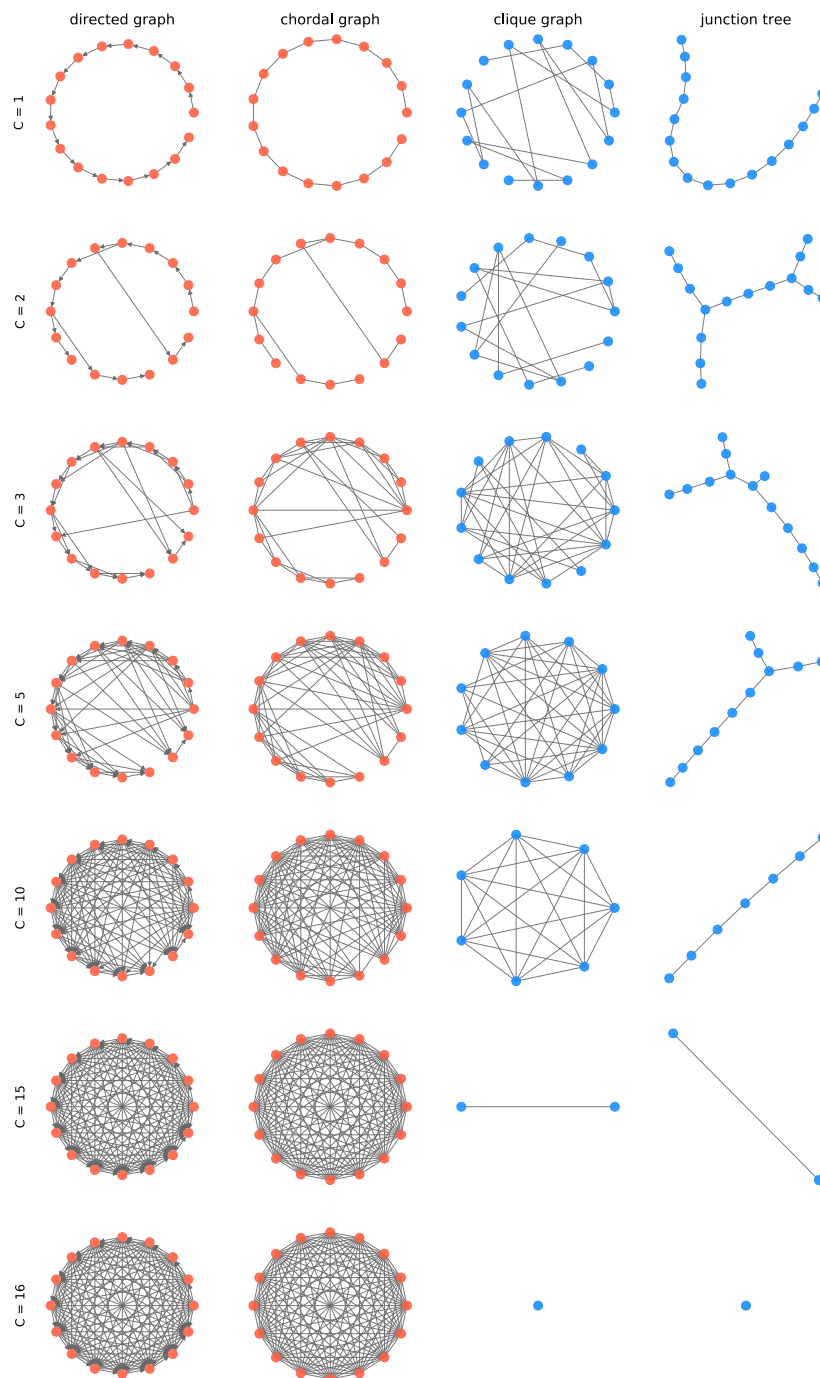


Figure 7.22. Different underlying directed graphs (1st column) between the experts with increasing degree of correlation (per rows) and the corresponding chordal graphs, clique graphs, and junctions trees (2st-4rd columns).



### 7.4.2.2 Factors of Clique Nodes

As discussed in Section 7.4.1, from the directed graph  $\mathcal{G} = (V, E)$  over the nodes  $V$ , we can compute the corresponding clique graph  $\mathcal{CG} = (Q, E_Q)$  over the cliques  $Q = \{\mathbf{q}_1, \dots, \mathbf{q}_K\}$ . Next, we discuss the assignment of the initial factors  $\phi_0$  over  $V$  to the factors  $\phi$  of the cliques in  $Q$ . Note that, this assignment is not unique, since the original nodes are present in several cliques. We define the assignment function  $ass(\mathbf{v}_j)$ , which assigns each node  $\mathbf{v}_j \in V$  to a clique  $\mathbf{q}_k$ , in which the node is contained  $\mathbf{v}_j \in \mathbf{q}_k$ . For the sake of uniqueness, we thereby choose the smallest index  $k = \arg \min_{i \in \{1, \dots, K\}} \mathbf{v}_j \in \mathbf{q}_i$ . Similarly, we defined the reverse assignment  $facts(\mathbf{q}_k) = \bigcup_{\mathbf{v}_j \in V} \mathbf{v}_j \mathbb{1}[ass(\mathbf{v}_j) = \mathbf{q}_k]$ , including all nodes  $\mathbf{v}_j$ , where an initial factor  $\phi_0$  is associated to clique  $\mathbf{q}_k$ . Then, the distribution over the cliques  $q(Q)$  can be formulated as

$$q(Q) = \prod_{\mathbf{q}_k \in Q} \phi(\mathbf{q}_k) = \prod_{\mathbf{q}_k \in Q} \prod_{\mathbf{v}_j \in facts(\mathbf{q}_k)} \phi_0(\mathbf{v}_j). \quad (7.13)$$

Note that, the distributions  $q(V)$  in (7.12) and (7.13)  $q(Q)$  are equal, they only constitute a different factorization of the joint distribution. In particular, the factorization (7.13) over the cliques allow efficient local posterior inference.

### 7.4.3 Belief Propagation

In this section we describe the *belief propagation (BP)* algorithm on the junction tree  $\mathcal{JT} = (Q, E_J)$ . We first select a root clique  $\mathbf{q}_0 \in Q$ , and propagate messages from the leaves in  $\mathcal{JT}$  towards the root among all edges  $(\mathbf{q}_i, \mathbf{q}_j) \in E_J$ . The messages are based on the *sum-product algorithm* [Koller and Friedman, 2009, Section 10.2], where the sums are replaced by integrals, since we consider continuous variables. In particular, the message from clique  $\mathbf{q}_i$  to  $\mathbf{q}_j$  can be computed as

$$\delta_{\mathbf{q}_i \rightarrow \mathbf{q}_j}(\mathbf{q}_j) = \int \phi(\mathbf{q}_i) \prod_{\mathbf{q}_k \in \text{neigh}(\mathbf{q}_i) \setminus \mathbf{q}_j} \delta_{\mathbf{q}_k \rightarrow \mathbf{q}_i}(\mathbf{q}_k) \, d[\mathbf{q}_i \setminus \mathbf{q}_j], \quad (7.14)$$

where first all incoming messages  $\delta_{\mathbf{q}_k \rightarrow \mathbf{q}_i}(\mathbf{q}_k)$  from all neighbors of  $\mathbf{q}_i$  in  $\text{neigh}(\mathbf{q}_i)$ , except  $\mathbf{q}_j$ , are multiplied together with the clique potential  $\phi(\mathbf{q}_i)$ . From this product, all variables contained in clique  $\mathbf{q}_i$  but not  $\mathbf{q}_j$ , are integrated out, so that the scope of the message  $\delta_{\mathbf{q}_i \rightarrow \mathbf{q}_j}(\mathbf{q}_j)$  consists of all common variables between clique  $\mathbf{q}_i$  and  $\mathbf{q}_j$ . This message passing process proceeds up the root  $\mathbf{q}_0$  of the tree. When the root clique has received all messages, it multiplies them with

its own initial potential to get the (unnormalized) belief potential  $\beta(\mathbf{q}_0)$ , which can be computed in general for clique  $\mathbf{q}_i$  as

$$\beta(\mathbf{q}_i) = \phi(\mathbf{q}_i) \prod_{\mathbf{q}_k \in \text{neigh}(\mathbf{q}_i)} \delta_{\mathbf{q}_k \rightarrow \mathbf{q}_i}(\mathbf{q}_k). \quad (7.15)$$

At this stage, the belief of the root clique is calibrated, however, the belief of all other cliques are not. In order to get a completely calibrated tree, the messages based on Equation (7.14) have to be sent backwards, i.e. from the root to all leaves. If all leaves have received their messages, the tree is fully calibrated. Note that, with this procedure, only all marginal *prior* beliefs of the cliques can be computed, since we did not condition on any observed variables.

#### 7.4.3.1 Dealing with Observed Variables

In this section, we briefly describe a modification of the above message passing algorithm that can deal with observed variables. Thereby, we split the computations in (7.14), in particular, we first compute the product over all involved factors

$$\beta^{-j}(\mathbf{q}_i) = \phi(\mathbf{q}_i) \prod_{\mathbf{q}_k \in \text{neigh}(\mathbf{q}_i) \setminus \mathbf{q}_j} \delta_{\mathbf{q}_k \rightarrow \mathbf{q}_i}(\mathbf{q}_k), \quad (7.16)$$

which is similar to the belief (7.15) without the message from clique  $\mathbf{q}_j$ . Instead of directly integrating out all variables from this product (7.16), we condition first on the observed variables  $\mathbf{v}_o \in \mathbf{q}_i$  in the clique  $\mathbf{q}_i$ , which is denoted as the following operation

$$\delta_{\mathbf{q}_i}(\mathbf{q}_i \setminus \mathbf{v}_o) = \Gamma \beta^{-}(\mathbf{q}_i) \mid \mathbf{v}_o. \quad (7.17)$$

Finally, all variables, which are not in the scope of clique  $\mathbf{q}_j$ , are integrated out from (7.17), that is,

$$\delta_{\mathbf{q}_i \rightarrow \mathbf{q}_j}(\mathbf{q}_j) = \int \delta_{\mathbf{q}_i}(\mathbf{q}_i \setminus \mathbf{v}_o) d[\mathbf{q}_i \setminus \{\mathbf{q}_j \cup \mathbf{v}_o\}]. \quad (7.18)$$

The (unnormalized) belief over each clique  $\mathbf{q}_i$  can again be computed by

$$\beta(\mathbf{q}_i) = \phi(\mathbf{q}_i) \prod_{\mathbf{q}_k \in \text{neigh}(\mathbf{q}_i)} \delta_{\mathbf{q}_k \rightarrow \mathbf{q}_i}(\mathbf{q}_k). \quad (7.19)$$

Note that, this belief correspond now to the marginal *posterior* distribution over clique  $\mathbf{q}_i$ .

### 7.4.3.2 Belief Propagation via Division

In the previous Section 7.4.3, we showed an approach for message passing in junction trees, based on the sum-product approach (7.14). Alternatively, we can use the *sum-product-divide* belief propagation approach [Koller and Friedman, 2009, Section 10.3], which maintains always the current beliefs  $\beta(\mathbf{q}_i)$  (7.19) and exploits the division operation, so that the belief  $\beta^{-j}(\mathbf{q}_i)$  in (7.16) without the message from  $\mathbf{q}_j$  can be computed as

$$\beta^{-j}(\mathbf{q}_i) = \frac{\beta(\mathbf{q}_i)}{\delta_{\mathbf{q}_j \rightarrow \mathbf{q}_i}}. \quad (7.20)$$

The conditioning (7.17) and marginalization (7.18) can be analogously computed as in the previous section. This approach is mathematically equivalent to that of the previous section, however, it is often more convenient for computing all marginal posterior distributions over all cliques, therefore, our implementation<sup>1</sup> is based on this approach.

### 7.4.3.3 Gaussian Belief Propagation (GaBP)

The belief propagation (BP) procedure relies on three operations, namely, the marginalization and the conditioning operation, as well as computing the product of two (unnormalized) densities. Thus, BP can be applied to all distributions, where these operations can be performed efficiently. Here, we focus on the multivariate Gaussians, since the initial potentials  $\phi_0$  are all Gaussians. In this case, these three operations can be efficiently computed when working with the *canonical* form of Gaussians, as described in Section 2.1.1.3. In particular, the marginalization is defined in (2.9), the conditioning in (2.10), and the product and division in (2.11).

### 7.4.4 Summary

The approach presented in this Section 7.4 allows to compute all marginal posterior distributions  $\mathcal{N}(\mathbf{a}_{\psi(j)} | \boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$ . In particular, by performing GaBP on the constructed junction tree based on the directed graph of the experts, all posterior marginal distributions of the cliques can be computed. From these cliques, it is easy to derive the marginal posterior distributions in the region of correlation  $\psi(j)$  of each expert, from which local predictions can be obtained and aggregated, as discussed in Section 7.2.4. This approach is an alternative method for

<sup>1</sup>This code is available on <https://github.com/manuelIDSIA/CPoE-BP>.

computing local marginal posterior distributions in an efficient and clean way by exploiting the structure of the induced junction tree. This approach establishes connections between GP approximation methods and local message passing algorithms, which opens up several new research directions for future work, as outlined in Section 8.2.

## 7.5 Summary of Contributions

### 7.5.1 Conclusion

In this chapter, we introduced a novel GP approximation algorithm "*Correlated Product of Experts*" (CPoE), where the degree of approximation can be adjusted by a locality and a sparsity parameter, so that the proposed method recovers independent PoEs, sparse global GP and full GP. We showed that our method consistently approximates full GP, in particular, we proved that increasing the correlations between the experts decreases monotonically the KL of the joint prior of full GP to them of our model. The presented algorithm has only a few hyperparameters, which allows an efficient deterministic and stochastic optimization. Further, our presented algorithm works with a general kernel, with several variables and also priors on the hyperparameters can be included. Moreover, the time and space complexity is linear in the number of experts and number of data samples, which makes it highly scalable. This is demonstrated with efficient implementations, so that a dataset with several ten thousands of samples can be processed in around a minute on a standard Laptop. In several experiments with synthetic and real world data, superior performance in a accuracy vs. time sense compared to state-of-the-art GP approximations methods is demonstrated for the deterministic and stochastic case, which makes our algorithm a competitive method for GPs approximations. Moreover, we present an alternative approach for inference in the same model based on local message passing or belief propagation algorithm. This novel approach for GP approximation establishes several connections to other probabilistic methods and algorithms.

### 7.5.2 Contributions

The content of this Chapter 7 is based on the paper "*Correlated Product of Experts for Sparse Gaussian Process Regression*" by Schürch et al. [2022], except the sections "*Generalizations of Bayesian Linear Models*" in 7.2.6 and "*CPoE via Gaussian Belief Propagation*" in 7.4 is unpublished material. In particular, the approach

presented in Section 7.4 constitutes an additional novel contribution and opens up several future work, as outlined in Section 8.2. The the other sections of this chapter are slightly adapted from the paper, where some sections are extended, and some shortened. We included several more explanations, in order to make some connections clearer, on the other hand, we did not include some details for the sake of brevity. In the following, we summarize the novel scientific contributions contained in this chapter:

- **General Dependency Structure:** We introduce a novel unifying GP approximation model with a general dependency structure among the latent function values, which unifies several existing GP approximation methods.
- **Combining Global and Local Approaches:** Our model combines the advantages from both global sparse GP models and local independent Product of Experts.
- **Recovering of Existing Methods:** The main idea of our model is to introduce some correlations between local sparse experts, where the degree of correlation and the sparsity can be adjusted by two parameters, so that independent PoE, sparse global GPs and full GPs can be recovered in the limiting cases.
- **Analytic Inference:** Our model allows analytic inference by exploiting the sparse posterior precision matrix induced by conditional independence assumptions among the latent function values. Thereby, the first inference approach is based on block-sparse Cholesky decomposition.
- **Efficient Hyperparameter Estimation:** For our model, we provide efficient hyperparameter estimation methods, including stochastic optimization approaches, which allows to train our model for a large number of training data points.
- **Superior Performance:** We compare our novel model with state-of-the-art methods for GP approximations and demonstrate, in several experiments for synthetic as well as real world data, superior performance in terms of accuracy vs. time.
- **Generalization of Bayesian Linear Models:** We explain, that several presented GP approximation models in this thesis can be seen as generalizations of the ordinary Bayesian linear model. This provides novel insight and justifications of our methods.

- **Connection to Gaussian Belief Propagation:** We present an alternative inference approach of the CPoE model based on local belief propagation between the experts. This opens up several connections to other probabilistic methods and algorithms as well as it initiates several directions for future work, as outlined in Section 8.2.

# Chapter 8

## Conclusion and Future Work

### 8.1 Summary

This thesis provides novel contributions to scalable Gaussian processes. In particular, the properties of GPs with focus on regression methods were reviewed, the state-of-the-art methods for their approximations were analyzed, and their strengths and limitations were discussed. To make GPs scalable to large datasets, new methodologies and novel algorithms were proposed, theoretically supported and empirically evaluated by several experiments. In particular, the novel content can be summarized as follows.

- **Chapter 5 - Recursive Estimation for Sparse GPs:** A unifying algorithm for sparse GPs in the online and distributed setting was introduced.
- **Chapter 6 - Sequential Hyperparameters Learning for Sparse GPs:** Two novel algorithms for sequential hyperparameters learning were presented, enabling to scale GPs up to millions of training samples.
- **Chapter 7 - Correlated Product of Experts for Sparse GPs:** A new GP approximation method based on locally correlated experts was proposed, unifying several existing local and global GP approximations.

These proposed methods union high scalability with accurate performance, so that state-of-the-art performance in several tasks was achieved and a range of new applications are opened up for GPs. Our work can be extended into several directions, which range from practical improvements to the development of new methodologies and the application of our derived methods in new areas. In the following, we highlight some particularly interesting research direction for the future.

## 8.2 Future Work

### 8.2.1 Online Algorithm for Sparse GPs

In Chapter 6, we presented two novel approaches for sequential hyperparameter learning for a range of sparse GP models. In this section, an idea for an adapted and complete online algorithm for sparse GPs is presented. In our recursive model, we assumed global optimal hyperparameters  $\theta$ , even though we trained them with the aid of mini-batches sequentially over several epochs. Afterwards, these global optimal hyperparameters are fixed and used for inference at level I, that is, for analytically computing the posterior distribution of the inducing points. A different setting is *online* learning of the hyperparameters, as explained in Section 6.1. Thereby, we assume that the hyperparameters can vary for each mini-batch, and old mini-batches cannot be revisited again. In Section 5.2.11 we already outlined connections between our recursive model and dynamical state space systems. Thereby, the state of the varying system correspond to the global inducing points  $\mathbf{a}_j = f_{\theta_j}(\mathbf{A})$ , indexed by the current hyperparameters. This corresponds to the following model

$$\begin{aligned} \mathbf{y}_j &= \mathbf{f}_j + \varepsilon_j, \\ \mathbf{f}_j &= \mathbf{H}(\theta_j) \mathbf{a}_j + \gamma_j, \\ \mathbf{a}_j &= \mathbf{F}(\theta_j, \theta_{j-1}) \mathbf{a}_{j-1} + \mathbf{q}_j, \end{aligned}$$

where the projection matrix is similarly defined as  $\mathbf{H}(\theta_j) = \mathbf{K}_{XA}^{\theta_j} (\mathbf{K}_{AA}^{\theta_j})^{-1}$  and the novel transition matrix  $\mathbf{F}(\theta_j, \theta_{j-1}) = \mathbf{K}_{XA}^{\theta_j} (\mathbf{K}_{AA}^{\theta_{j-1}})^{-1}$  takes into account the varying hyperparameters. The optimal states  $\mathbf{a}_j$  can be obtained by the Kalman Filter, as discussed in Chapter 5, including the prediction step for the transition. Further, the optimal states can also be smoothed backwards via the Kalman-Smoother [Kalman et al., 1960]. A preliminary example is given in Figure 8.1, where we see the non-stationary behavior. We are convinced, that this constitutes a promising approach to probabilistic model flexible non-stationary functions.

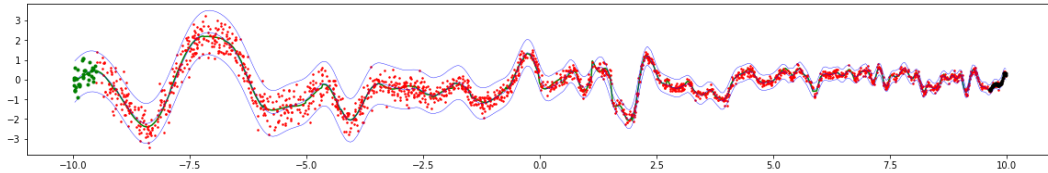


Figure 8.1. Online varying hyperparameters for sparse GPs.



### 8.2.2 Variational Approach for CPoE

In this section, we outline an approach to theoretically and thoroughly link our presented CPoE method to full GP in a *posterior* sense, instead of in a *prior* sense, as we showed in Section 7.2.5 and particularly in Proposition 7.8. Thereby, we have proven that the KL, between the true joint distribution  $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$  of full GP and our approximate joint distribution  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ , is decreasing for  $C \rightarrow J$ . This means, the approximation quality of the joint prior of our model is monotonically improving for increasing correlation. Although this demonstrates that our proposed method constitutes a well principled GP approximation in a *prior* sense, there is still a missing link between our model and full GP in a *posterior* sense. In Section 7.2.8, we presented a generalization of our model, so that for the maximal correlation, i.e.  $C = J$ , CPoE corresponds to a range of known sparse global GP methods. For instance, the variational approach by Titsias [2009] can be recovered, where a rigorous connection between the sparse GP model and full GP in the *posterior* sense exist. It would be interesting to directly derive a modified CPoE model via variational inference, so that the *posterior* of our model  $q_{c,\gamma}(\mathbf{a}, \mathbf{f}|\mathbf{y})$  would be thoroughly connected to full GP  $p(\mathbf{a}, \mathbf{f}|\mathbf{y})$ .

### 8.2.3 Efficient Drawing of Posterior Samples

As demonstrated in this thesis, there are several approaches for training scalable GPs, however, methods for accurately generating draws from the posterior distribution have received relatively little attention [Wilson et al., 2020]. Drawing a posterior sample of full GP involves the inversion of the covariance matrix  $\Sigma(\mathbf{X}_*) \in \mathbb{R}^{N_{test} \times N_{test}}$  in (3.16) of the posterior predictive distribution for test inputs  $\mathbf{X}_* \in \mathbb{R}^{N_{test} \times D}$ , which scales cubically in the number  $N_{test}$  of test locations. We outline here an idea to efficiently draw posterior samples based on the sparse and local decomposition in our CPoE method. In particular, we can exploit our sparse precision matrix in (7.9). This allows to draw a posterior sample, evaluated at all local inducing points, where the total number of these local inducing points can be in the order of the number of training points  $N$ . Therefore, if the evaluation points are known before training and  $N_{test} \approx N$ , the local inducing points can be set to the test points. Otherwise, this approach could be generalized to posterior samples evaluated at new test points. Based on the sparse posterior precision, local and correlated posterior samples can be obtained in a similar spirit to the local and correlated predictions in Chapter 7, which could then similarly aggregated based on probabilistic averaging techniques, resulting in an efficient procedure for drawing approximate GP posterior samples.

### 8.2.4 Large-Scale GP Implementations for Time-Series

In this section, we present an idea for a practical improvement of the implementations of our GP approximation method CPoE presented in Chapter 7. In particular, we plan to build a software library for times-series modeling with GPs based on the example in Section 7.3.1 and inspired by the work [Corani et al., 2021]. The main idea is to combine the automatic times-series learning framework [Corani et al., 2021] with our novel scalable GP approximation methods. In particular, this involves the flexible usage of composed kernels and includes different priors on the hyperparameters. As demonstrated in Section 7.3.1, our new GP approximation approach CPoE is ideal suited for this setting, so that automatic times-series estimation can be extended to huge datasets. From a practical point of view, we plan to reimplement<sup>1</sup> our algorithm CPoE based on *tensorflow* and *gpfLOW* [Abadi et al., 2015; Matthews et al., 2017], so that we can handle the different kernels and priors efficiently. This planned software package, based on our new proposed algorithms, would allow to efficiently exploit large time-series datasets with scalable GP algorithms.

### 8.2.5 Unifying GP Model for Regression and Classification

In Section 7.4, we presented an inference approach for the CPoE model by exploiting local belief propagation algorithms. In particular, from the local correlation structure, the corresponding junction tree (see Section 7.4.1) can be constructed, from which the local marginal distributions can be efficiently computed by passing messages among the edges of this junction tree. This is again illustrated in Figure 8.2, where the correlation structure among the local experts is depicted in the top row, and the corresponding junction tree in the bottom row. In a first step, we plan to thoroughly investigate this approach for the regression case (i.e. Gaussian likelihood), in particular the advantages over the Cholesky based approach presented in Section 7.2 in a sequential and distributed setting. In a second step, we want to generalize this approach to the non-Gaussian likelihood case (e.g. Bernoulli or Poisson distribution). The sparse and local approximate prior distribution is still assumed to be multivariate Gaussian and the structure of the junction tree is still the same. Unfortunately, the computations needed in the belief propagation algorithm cannot be performed analytically anymore. In this setting, the *expectation propagation (EP)* approach can be applied for approximate inference (see Section 4.2.3.2 and [Bishop, 2006, Section 10.7]). Thereby, the only difference is that each non-feasible local message

---

<sup>1</sup>This will be available on <https://github.com/manuelIDSIA/CPoE-tensorflow>.

distribution is projected by its mean and covariance to the mean and covariance of the closest Gaussian distribution in the KL-sense, as formulated in (4.47). This leads then to approximate inference for GPs with non-Gaussian likelihoods. An alternative and promising *exact* inference approach for non-Gaussian likelihoods in GPs is based on the *skew normal* or *skew Gaussian* distribution [Azzalini, 2013], which can be used for *skew GPs* [Benavoli et al., 2020]. However, *full* skew GPs are again not well suited for large datasets. Our presented local and sparse GP approach could be generalized to skew GPs, so that efficient and scalable GP inference would be possible for more general likelihoods.

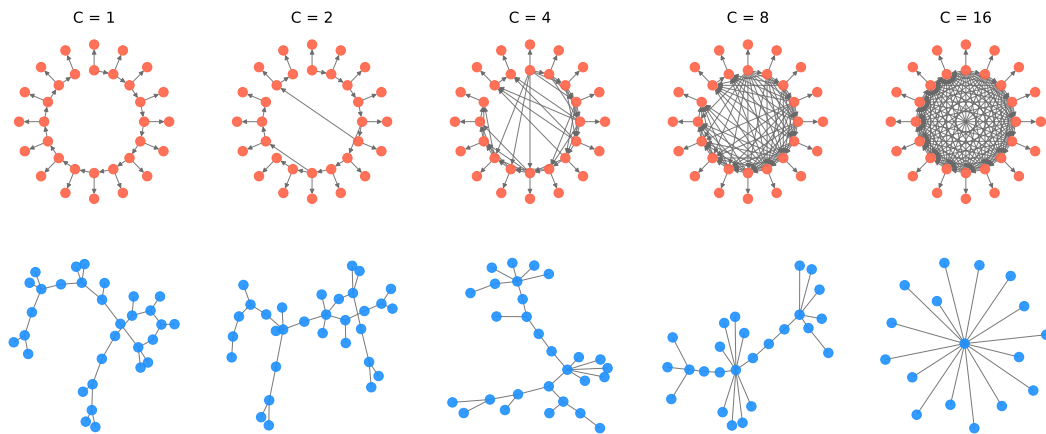


Figure 8.2. Correlated experts and corresponding junction trees for local inference.

### 8.2.6 Potential Application Areas for Large-Scale GPs

The novel GP approximation algorithms proposed in this thesis could have many potential application domains. Similar to full GPs, their scalable approximations can be applied in a wide range from social to natural science and engineering. While keeping the advantageous properties of full GPs, the new scalable approximations allow to exploit much larger datasets. Those algorithms are particularly well suited for applications, in which scalable, understandable, reliable and adaptive algorithms are of key importance, such as sequential design of experiments [Bect et al., 2012; Azzimonti et al., 2016], Bayesian optimization [Ginsbourger et al., 2010; Snoek et al., 2012], and reinforcement learning [Deisenroth et al., 2013; Grande et al., 2014]. As a subjective aim regarding future areas of applications, I would like to contribute in projects involving medical and environmental data because of my personal conviction of the importance about applied research in these areas.



# Bibliography

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL <https://www.tensorflow.org/>. Software available from [tensorflow.org](https://www.tensorflow.org/).

Anis Ben Abdesslem, Nikolaos Dervilis, David J Wagg, and Keith Worden. Automatic kernel selection for gaussian processes regression with approximate bayesian computation and sequential monte carlo. *Frontiers in Built Environment*, 3:52, 2017.

Ganesh Ajjanagadde, Anuran Makur, Jason Klusowski, Sheng Xu, et al. Lecture notes on information theory. 2017.

Adelchi Azzalini. *The skew-normal and related families*, volume 3. Cambridge University Press, 2013.

Dario Azzimonti, David Ginsbourger, Clément Chevalier, Julien Bect, and Yann Richet. Adaptive design of experiments for conservative estimation of excursion sets (siam uq 2016). In *2016 SIAM Conference on Uncertainty Quantification*, 2016.

Dario Azzimonti, Manuel Schürch, Alessio Benavoli, and Marco Zaffalon. Orthog-

- onally decoupled variational fourier features. *arXiv preprint arXiv:2007.06363*, 2020.
- Timothy D Barfoot. Fundamental linear algebra problem of gaussian inference. *arXiv preprint arXiv:2010.08022*, 2020.
- Julien Bect, David Ginsbourger, Ling Li, Victor Picheny, and Emmanuel Vazquez. Sequential design of computer experiments for the estimation of a probability of failure. *Statistics and Computing*, 22(3):773–793, 2012.
- Alessio Benavoli and Giorgio Corani. State space approximation of gaussian processes for time series forecasting.
- Alessio Benavoli and Marco Zaffalon. State space representation of non-stationary gaussian processes. 2016. URL <http://arxiv.org/abs/1601.01544>.
- Alessio Benavoli, Dario Azzimonti, and Dario Piga. Skew gaussian processes for classification. *arXiv preprint arXiv:2005.12987*, 2020.
- Christopher M Bishop. *Pattern recognition and machine learning*, volume 4. Springer, 2006.
- Steffen Börm and Jochen Garcke. Approximating gaussian processes with h2-matrices. In *European Conference on Machine Learning*, pages 42–53. Springer, 2007.
- Thang D Bui and Richard E Turner. Tree-structured gaussian process approximations. *Advances in Neural Information Processing Systems*, 27:2213–2221, 2014.
- Thang D Bui, Cuong Nguyen, and Richard E Turner. Streaming sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, pages 3301–3309, 2017a.
- Thang D Bui, Josiah Yan, and Richard E Turner. A unifying framework for sparse gaussian process approximation using power expectation propagation. *Journal of Machine Learning Research*, 18:1–72, 2017b.
- Sebastian Buschjäger, Thomas Liebig, and Katharina Morik. Gaussian model trees for traffic imputation. In *ICPRAM*, pages 243–254, 2019.

- Andrea Carron, Marco Todescato, Ruggero Carli, Luca Schenato, and Gianluigi Pillonetto. Machine learning meets kalman filtering. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 4594–4599. IEEE, 2016.
- Augustin Cauchy et al. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.
- Ching-An Cheng and Byron Boots. Incremental variational sparse gaussian process regression. In *Advances in Neural Information Processing Systems*, pages 4410–4418, 2016.
- Giorgio Corani, Alessio Benavoli, and Marco Zaffalon. Time series forecasting with gaussian processes needs priors. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 103–117. Springer, 2021.
- Lehel Csató and Manfred Opper. Sparse online gaussian processes. *Neural computation*, 14(3):641–668, 2002.
- John P Cunningham, Krishna V Shenoy, and Maneesh Sahani. Fast gaussian process methods for point process intensity estimation. In *Proceedings of the 25th international conference on Machine learning*, pages 192–199, 2008.
- Abhirup Datta, Sudipto Banerjee, Andrew O Finley, and Alan E Gelfand. Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.
- Marc Peter Deisenroth and Jun Wei Ng. Distributed gaussian processes. *arXiv preprint arXiv:1502.02843*, 2015.
- Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, 2013.
- Thomas G Dietterich. Steps toward robust artificial intelligence. *AI Magazine*, 38(3):3–24, 2017.
- John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

- Nicolas Durrande, Vincent Adam, Lucas Bordeaux, Stefanos Eleftheriadis, and James Hensman. Banded matrix operators for gaussian markov models in the automatic differentiation era. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2780–2789. PMLR, 2019.
- David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.
- Yanshuai Cao David J Fleet. Generalized product of experts for automatic and principled fusion of gaussian process predictions. *arXiv preprint arXiv:1410.7827*, 2014.
- Roger Fletcher. Practical methods of optimization, 1987. *John and Sons, Chichester*, 1987.
- Roger Fletcher. On the barzilai-borwein method. In *Optimization and control with applications*, pages 235–256. Springer, 2005.
- Elad Gilboa, Yunus Saatçi, and John P Cunningham. Scaling multidimensional inference for structured gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):424–436, 2013.
- David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. Kriging is well-suited to parallelize optimization. In *Computational intelligence in expensive optimization problems*, pages 131–162. Springer, 2010.
- Tilmann Gneiting. Compactly supported correlation functions. *Journal of Multivariate Analysis*, 83(2):493–508, 2002.
- GPY. GPY: A gaussian process framework in python. <http://github.com/SheffieldML/GPY>, since 2012.
- Robert B Gramacy and Daniel W Apley. Local gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, 24(2):561–578, 2015.
- Robert Grande, Thomas Walsh, and Jonathan How. Sample efficient reinforcement learning with gaussian processes. In *International Conference on Machine Learning*, pages 1332–1340. PMLR, 2014.
- Jouni Hartikainen and Simo Särkkä. Kalman filtering and smoothing solutions to temporal gaussian process regression models. In *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384. IEEE, 2010.



- James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. In *Conference for Uncertainty in Artificial Intelligence*, 2013.
- Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.
- Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.
- Rob Hyndman. *fpp2: Data for "Forecasting: Principles and Practice" (2nd Edition)*, 2020. URL <https://CRAN.R-project.org/package=fpp2>. R package version 2.4.
- Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.
- Simon J Julier and Jeffrey K Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, volume 4, pages 2369–2373. IEEE, 1997.
- Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.
- Lucas Kania, Manuel Schürch, Dario Azzimonti, and Alessio Benavoli. Sparse information filter for fast gaussian process regression. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2021.
- Matthias Katzfuss and Joseph Guinness. A general framework for vecchia approximations of gaussian processes. *Statistical Science*, 36(1):124–141, 2021.
- Hyoung-Moon Kim, Bani K Mallick, and Chris C Holmes. Analyzing nonstationary spatial data using piecewise gaussian processes. *Journal of the American Statistical Association*, 100(470):653–668, 2005.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

- Miguel Lázaro-Gredilla and Anibal Figueiras-Vidal. Inter-domain gaussian processes for sparse inference using inducing features. In *Advances in Neural Information Processing Systems*, pages 1087–1095, 2009.
- Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.
- Wangyan Li, Zidong Wang, Guoliang Wei, Lifeng Ma, Jun Hu, and Derui Ding. A survey on multisensor fusion and consensus filtering for sensor networks. *Discrete Dynamics in Nature and Society*, 2015, 2015.
- Haitao Liu, Jianfei Cai, Yi Wang, and Yew Soon Ong. Generalized robust bayesian committee machine for large-scale gaussian process regression. In *International Conference on Machine Learning*, pages 3131–3140. PMLR, 2018.
- Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *IEEE transactions on neural networks and learning systems*, 31(11):4405–4423, 2020.
- Dmitry M Malioutov, Jason K Johnson, and Alan S Willsky. Walk-sums and belief propagation in gaussian graphical models. *The Journal of Machine Learning Research*, 7:2031–2064, 2006.
- Songrit Maneewongvatana and David M Mount. On the efficiency of nearest neighbor searching with data clustered in lower dimensions. In *International Conference on Computational Science*, pages 842–851. Springer, 2001.
- Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.
- Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagr a, Zoubin Ghahramani, and James Hensman. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*, 18(40):1–6, apr 2017. URL <http://jmlr.org/papers/v18/16-537.html>.
- Arman Melkumyan and Fabio Tozeto Ramos. A sparse covariance function for exact gaussian process inference in large datasets. In *Twenty-first international joint conference on artificial intelligence*, 2009.
- Tom Minka et al. Divergence measures and message passing. Technical report, Citeseer, 2005.

- Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, Cambridge, 2012.
- Radford M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, CAN, 1995. AAINN02676.
- John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.
- Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.
- Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.
- Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.
- Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press, Cambridge, 2006.
- Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.
- Didier Rullière, Nicolas Durrande, François Bachoc, and Clément Chevalier. Nested kriging predictions for datasets with a large number of observations. *Statistics and Computing*, 28(4):849–867, 2018.
- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
- Hugh Salimbeni, Ching-An Cheng, Byron Boots, and Marc Deisenroth. Orthogonally decoupled variational gaussian processes. In *Advances in neural information processing systems*, pages 8711–8720, 2018.
- Simo Sarkka, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional bayesian filtering and smoothing: A look at gaussian process regression through kalman filtering. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

- Manuel Schürch, Dario Azzimonti, Alessio Benavoli, and Marco Zaffalon. Recursive estimation for sparse gaussian process regression. *Automatica*, 120: 109127, 2020.
- Manuel Schürch, Dario Azzimonti, Alessio Benavoli, and Marco Zaffalon. Correlated product of experts for sparse gaussian process regression. *arXiv preprint arXiv:2112.09519*, 2022.
- Matthias Seeger, Christopher Williams, and Neil Lawrence. Fast forward selection to speed up sparse gaussian process regression. In *Artificial Intelligence and Statistics 9*, number EPFL-CONF-161318, 2003.
- Yirong Shen, Matthias Seeger, and Andrew Ng. Fast gaussian process regression using kd-trees. *Advances in neural information processing systems*, 18, 2005.
- Jiaxin Shi, Michalis Titsias, and Andriy Mnih. Sparse orthogonal variational inference for gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 1932–1942. PMLR, 2020.
- Bernhard W Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–52, 1985.
- Alex J Smola and Peter L Bartlett. Sparse greedy gaussian process regression. In *Advances in neural information processing systems*, pages 619–625, 2001.
- Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.
- Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264, 2006.
- Edward Snelson and Zoubin Ghahramani. Local and global sparse gaussian process approximations. In *Artificial Intelligence and Statistics*, pages 524–531, 2007.
- Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.
- Kazuhiro Takahashi. Formation of sparse bus impedance matrix and its application to short circuit study. In *Proc. PICA Conference, June, 1973*, 1973.

- Tong Teng, Jie Chen, Yehong Zhang, and Bryan Kian Hsiang Low. Scalable variational bayesian kernel selection for sparse gaussian process regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5997–6004, 2020.
- Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.
- Marco Todescato, Andrea Carron, Ruggero Carli, Gianluigi Pillonetto, and Luca Schenato. Efficient spatio-temporal gaussian regression via kalman filtering. *arXiv preprint arXiv:1705.01485*, 2017.
- Volker Tresp. A bayesian committee machine. *Neural computation*, 12(11):2719–2741, 2000.
- Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact gaussian processes on a million data points. *Advances in Neural Information Processing Systems*, 32:14648–14659, 2019.
- Yair Weiss and William Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Advances in neural information processing systems*, 12, 1999.
- Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Advances in neural information processing systems*, 13, 2000.
- Andrew Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *International conference on machine learning*, pages 1775–1784. PMLR, 2015.
- Andrew Gordon Wilson, Christoph Dann, and Hannes Nickisch. Thoughts on massively scalable gaussian processes. *arXiv preprint arXiv:1511.01870*, 2015.
- James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. Efficiently sampling functions from gaussian process posteriors. In *International Conference on Machine Learning*, pages 10292–10302. PMLR, 2020.
- Stephen Wright, Jorge Nocedal, et al. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8): 1177–1193, 2012.

# Appendix A

## Orthogonally Decoupled Variational Fourier Features

In this section, we briefly present a new approach for an algorithm to train sparse GP methods exploiting two kind of different bases, as described in [Azzimonti, Schürch, Benavoli, and Zaffalon, 2020]. We briefly provide the necessary background and a summary of our approach. For more details we refer to the original paper and its supplementary material.

### Background

Sparse global inducing point GP approximation methods, as presented in Section 4.2, have long been the standard method to fit GPs to large datasets. In particular, Titsias [2009] introduced a variational method that keeps the original GP prior and approximates the posterior with variational inference, as discussed in Section 4.2.3.1. An alternative method for variational inference on sparse GPs was introduced by Cheng and Boots [2016], where the authors proposed a variational inference method based on a property of the reproducing kernel Hilbert space (RKHS) associated with the GP. This approach has been improved by Salimbeni et al. [2018], who proposed a powerful orthogonally decoupled basis that allows for an efficient natural gradient update rule. A parallel line of research studies inter-domain approaches (Rahimi and Recht [2008] and Lázaro-Gredilla and Figueiras-Vidal [2009]), which replace inducing points variables with more informative inducing features, for instance Fourier features.

## Summary

In the approach, as thoroughly described in [Azzimonti et al., 2020], we combine the flexibility of the orthogonally decoupled RKHS bases introduced in Salimbeni et al. [2018] and the explanatory power of inter-domain approaches to propose a new method for training sparse Gaussian processes. In particular, we build a variational distribution parametrized in the mean by an inducing point basis and in the covariance by a variational Fourier features basis. Since the basis in the mean does not require a matrix inversion we can use a large number of inducing points for the basis in the mean and obtain an approximation close to the true posterior mean. On the other hand, by using a variational Fourier features basis in the covariance, we exploit the higher informative power of such features to obtain better covariance estimates. The orthogonal structure of the RKHS basis guarantees that the range of the two basis does not overlap. In several experiments, we demonstrated that our method is competitive with the state-of-the-art on synthetic as well as real datasets. For more details we refer to [Azzimonti, Schürch, Benavoli, and Zaffalon, 2020].



# Appendix B

## Appendix of Chapter 5

### B.1 Proofs

**Proof B.1 (Proof of Proposition 5.4)** *The lower bound to the log marginal likelihood  $\log q(\mathbf{y}_j | \mathbf{y}_{1:j-1}) \geq \mathcal{L}(q_j(\mathbf{f}_j, \mathbf{a}))$  in (5.20) can be formulated accordingly to (4.35) for  $\mathbf{z} = [\mathbf{f}_j, \mathbf{a}]^T$ , leading to*

$$\mathcal{L}(q_j(\mathbf{f}_j, \mathbf{a})) = \int q_j(\mathbf{f}_j, \mathbf{a}) \log \frac{p(\mathbf{f}_j, \mathbf{a}, \mathbf{y}_j | \mathbf{y}_{1:j-1})}{q_j(\mathbf{f}_j, \mathbf{a})} d\mathbf{f}_j d\mathbf{a}. \quad (\text{B.1})$$

Thereby, the true joint sequential distribution is given as

$$p(\mathbf{f}_j, \mathbf{a}, \mathbf{y}_j | \mathbf{y}_{1:j-1}) = p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}) p(\mathbf{a} | \mathbf{y}_{1:j-1}), \quad (\text{B.2})$$

where we assume by recursion that the true distribution correspond to the previous optimal distribution  $p(\mathbf{a} | \mathbf{y}_{1:j-1}) = q_{j-1}^*(\mathbf{a}) = \mathcal{N}(\mathbf{a} | \boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_{j-1})$ . Plugging the structure of the variational distribution (5.19) and (B.2) into (B.1) leads to

$$\begin{aligned} \mathcal{L}(q_j(\mathbf{a})) &= \int p(\mathbf{f}_j | \mathbf{a}) q_j(\mathbf{a}) \log \frac{p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}) q_{j-1}^*(\mathbf{a})}{p(\mathbf{f}_j | \mathbf{a}) q_j(\mathbf{a})} d\mathbf{f}_j d\mathbf{a} \\ &= \int p(\mathbf{f}_j | \mathbf{a}) q_j(\mathbf{a}) \log \frac{p(\mathbf{y}_j | \mathbf{f}_j) q_{j-1}^*(\mathbf{a})}{q_j(\mathbf{a})} d\mathbf{f}_j d\mathbf{a}, \end{aligned}$$

where  $p(\mathbf{f}_j | \mathbf{a})$  cancels out. It can be rearranged to

$$\int q_j(\mathbf{u}) \left\{ \int p(\mathbf{f}_j | \mathbf{a}) \log p(\mathbf{y}_j | \mathbf{f}_j) d\mathbf{f}_j + \log \frac{q_{j-1}(\mathbf{a})}{q_j(\mathbf{a})} \right\} d\mathbf{a},$$

in which the integral involving  $\mathbf{f}_j$  is computed as

$$\begin{aligned}
& \int p(\mathbf{f}_j|\mathbf{a}) \log p(\mathbf{y}_j|\mathbf{f}_j) d\mathbf{f}_j \\
&= \mathbb{E}_{\mathbf{f}_j|\mathbf{a}} \left[ -\frac{B}{2} \log(2\pi\sigma_n^2) - \frac{1}{2} [\mathbf{y}_j - \mathbf{f}_j]^T \frac{1}{\sigma_n^2} [\mathbf{y}_j - \mathbf{f}_j] \right] \\
&= -\frac{B}{2} \log(2\pi\sigma_n^2) - \frac{1}{2\sigma_n^2} \left( \mathbf{y}_j^T \mathbf{y}_j + \mathbb{E}_{\mathbf{f}_j|\mathbf{a}} [\mathbf{f}_j^T \mathbf{f}_j - 2\mathbf{y}_j^T \mathbf{f}_j] \right).
\end{aligned} \tag{B.3}$$

Using the true training conditional  $p(\mathbf{f}_j|\mathbf{a}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}, \mathbf{K}_{X_jX_j} - \mathbf{Q}_{X_jX_j})$  for computing the expectation by the formula  $\mathbb{E}[\mathbf{x}^T \mathbf{A} \mathbf{x}] = \text{Tr}[\mathbf{A}\Sigma] + \boldsymbol{\mu}^T \mathbf{A} \boldsymbol{\mu}$  with  $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \Sigma)$ , yields

$$\begin{aligned}
& -\frac{K}{2} \log(2\pi\sigma_n^2) - \frac{1}{2\sigma_n^2} \left( \mathbf{y}_j^T \mathbf{y}_j + \text{tr}[\mathbf{K}_{X_jX_j} - \mathbf{Q}_{X_jX_j}] + \mathbf{a}^T \mathbf{H}_j^T \mathbf{H}_j \mathbf{a} - 2\mathbf{y}_j^T \mathbf{H}_j \mathbf{a} \right) \\
&= \log[\mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \sigma_n^2 \mathbb{I})] - \frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{X_jX_j} - \mathbf{Q}_{X_jX_j}]
\end{aligned}$$

for the integral in (B.3). Substitute this expression back, the sequential lower bound becomes

$$\int q_j(\mathbf{a}) \log \frac{\mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \sigma_n^2 \mathbb{I}) q_{j-1}(\mathbf{a})}{q_j(\mathbf{a})} d\mathbf{a} - \frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{X_jX_j} - \mathbf{Q}_{X_jX_j}].$$

We can now maximize this bound with respect to  $q_j(\mathbf{a})$  by taking functional derivatives with respect to  $q_j(\mathbf{a})$  yielding the optimal distribution

$$q_j^*(\mathbf{a}) = \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j\mathbf{a}, \sigma_n^2 \mathbb{I}) q_{j-1}(\mathbf{a}) = \mathcal{N}\left(\mathbf{u} \mid \frac{1}{\sigma_n^2} \boldsymbol{\Sigma}_j \mathbf{H}_j^T \mathbf{y}_j + \boldsymbol{\Sigma}_{j-1}^{-1} \boldsymbol{\mu}_{j-1}, \boldsymbol{\Sigma}_j\right),$$

where  $\boldsymbol{\Sigma}_j = \left(\boldsymbol{\Sigma}_{j-1}^{-1} + \frac{1}{\sigma_n^2} \mathbf{H}_j^T \mathbf{H}_j\right)^{-1}$  with the corresponding optimal collapsed bound

$$\mathcal{L}_{\text{REC}}(q_j^*(\mathbf{a})) = \log \mathcal{N}(\mathbf{y}_j|\mathbf{H}_j \boldsymbol{\mu}_{j-1}, \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \mathbf{H}_j^T + \sigma_n^2 \mathbb{I}) - \frac{1}{2\sigma_n^2} \text{tr}[\mathbf{K}_{X_jX_j} - \mathbf{Q}_{X_jX_j}],$$

which match the results in Proposition 5.4 and completes the proof.

## B.2 Details for Recursive Gradient Propagation

We show here the computation for the recursive gradient propagation from Section 6.2 for the PEP model. For other models,  $a_j$  and  $\bar{\mathbf{V}}$  from the Table 5.2 could

be used correspondingly. We will use the following notation:  $\text{diag}[\mathbf{A}] = \mathbf{d}$ ,  $d_i = a_{ii}$ ,  $\text{Diag}[\mathbf{d}] = \mathbf{A}$ ,  $a_{ii} = d_i$ ,  $a_{ij} = 0$ ,  $\mathbf{A} \odot \mathbf{B} = \mathbf{C}$ ,  $c_{ij} = a_{ij}b_{ij}$ ,  $\mathbf{A} \div \mathbf{B} = \mathbf{C}$ ,  $c_{ij} = \frac{a_{ij}}{b_{ij}}$ ,  $\mathbf{A}^{\odot 2} = \mathbf{C}$ ,  $c_{ij} = a_{ij}^2$ ,  $\dot{\mathbf{A}} = \frac{\partial \mathbf{A}(\theta)}{\partial \theta}$ ,  $\forall \theta \in \Theta$ ,  $\text{sum}[\mathbf{A}] = \sum_{i,j} a_{ij}$ ,  $1_{[z]} = 1$  if  $z = \text{true}$ , 0 other.

### Initialization

$$\begin{aligned} \boldsymbol{\eta}_0 &= \mathbf{0}; & \dot{\boldsymbol{\eta}}_0 &= \mathbf{0}; \\ \boldsymbol{\Lambda}_0 &= \mathbf{K}_{AA}^{-1}; & \dot{\boldsymbol{\Lambda}}_0 &= -\mathbf{K}_{AA}^{-1} \dot{\mathbf{K}}_{AA} \mathbf{K}_{AA}^{-1}; \\ \psi_0 &= -\frac{N}{2} \log 2\pi; & \dot{\psi}_0 &= 0; \\ \boldsymbol{\Sigma}_0 &= \mathbf{K}_{AA}; & \log \text{Det}_0 &= \log |\boldsymbol{\Lambda}_0|; \end{aligned}$$

### Natural Mean and Precision Updates

$$\begin{aligned} \mathbf{H}_j &= \mathbf{K}_{X_j A} \mathbf{K}_{AA}^{-1}; \\ \mathbf{d}_j &= \text{diag} \left[ \mathbf{K}_{X_j X_j} - \mathbf{K}_{X_j A} \mathbf{K}_{AA}^{-1} \mathbf{K}_{A X_j} \right]; \\ \mathbf{v}_j &= \alpha \mathbf{d}_j + \sigma_n^2 \mathbf{1}; \quad \mathbf{V}_j^{-1} = \text{Diag} \left[ \mathbf{1} \div \mathbf{v}_j \right]; \\ a_j &= \frac{1 - \alpha}{\alpha} \left( \sum_{i=1}^B \log([\mathbf{v}_j]_i) - B \log \sigma_n^2 \right); \\ \mathbf{r}_j &= \mathbf{y}_j - \mathbf{H}_j \boldsymbol{\Sigma}_{j-1} \boldsymbol{\eta}_{j-1}; \\ \boldsymbol{\eta}_j &= \boldsymbol{\eta}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j; \\ \boldsymbol{\Lambda}_j &= \boldsymbol{\Lambda}_{j-1} + \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j; \\ \boldsymbol{\Sigma}_j, \log |\boldsymbol{\Lambda}_j| &= \boldsymbol{\Lambda}_j^{-1}, \log |\boldsymbol{\Lambda}_j|; \\ \mathbf{S}_j^{-1} &= \mathbf{V}_j^{-1} - \mathbf{V}_j^{-1} \mathbf{H}_j \boldsymbol{\Sigma}_j \mathbf{H}_j^T \mathbf{V}_j^{-1}; \\ \psi_j &= \psi_{j-1} - \frac{1}{2} \left( \log |\boldsymbol{\Lambda}_j| - \log |\boldsymbol{\Lambda}_{j-1}| - \log |\mathbf{V}_j^{-1}| + \mathbf{r}_j^T \mathbf{S}_j^{-1} \mathbf{r}_j + a_j \right) \end{aligned}$$

### Intermediate Derivatives

$$\begin{aligned} \dot{L}_{d\mathbf{H}_j} &= 2 \left( \mathbf{V}_j^{-1} \mathbf{H}_j \boldsymbol{\Sigma}_j - \mathbf{S}_j^{-1} \mathbf{r}_j (\boldsymbol{\Sigma}_{j-1} \boldsymbol{\eta}_{j-1} + \boldsymbol{\Sigma}_j \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{r}_j)^T \right) \\ \dot{L}_{d\mathbf{v}_j} &= - \left( \text{diag} \left[ \mathbf{H}_j \boldsymbol{\Sigma}_j \mathbf{H}_j^T \right] - \frac{1}{\alpha} \mathbf{v}_j + (\mathbf{r}_j - \mathbf{H}_j \boldsymbol{\Sigma}_j \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{r}_j)^{\odot 2} \right) \div \mathbf{v}_j^{\odot 2} \\ \dot{L}_{d\mathbf{K}_{X_j A}} &= \dot{L}_{d\mathbf{H}_j} \mathbf{K}_{AA}^{-1} - 2\alpha \text{Diag} \left[ \dot{L}_{d\mathbf{v}_j} \right] \mathbf{H}_j \\ \dot{L}_{d\mathbf{K}_{AA}} &= -\mathbf{H}_j^T \left( \dot{L}_{d\mathbf{H}_j} \mathbf{K}_{AA}^{-1} - \alpha \text{Diag} \left[ \dot{L}_{d\mathbf{v}_j} \right] \mathbf{H}_j \right) \end{aligned}$$

$$\begin{aligned}
\dot{L}_{dj_{X_j X_j}} &= \alpha \dot{L}_{dv_j} \\
\dot{L}_{d\Lambda_j} &= \Sigma_j - \Sigma_{j-1} + 2\Sigma_{j-1} \mathbf{H}_j^T \mathbf{S}_j^{-1} \mathbf{r}_j \boldsymbol{\eta}_{j-1}^T \Sigma_{j-1} + \Sigma_j \mathbf{H}_j^T \mathbf{V}_j^{-1} \mathbf{r}_j \mathbf{r}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j \Sigma_j \\
\dot{L}_{d\eta_j} &= -2\Sigma_{j-1} \mathbf{H}_j^T \mathbf{S}_j^{-1} \mathbf{r}_j \\
\dot{L}_{d_{d_n}} &= 2\sigma_n^2 \text{sum}[\dot{L}_{dv_j}] - 2B \frac{1-\alpha}{\alpha}
\end{aligned}$$

**Derivative Updates, loop over  $\theta_i \in \theta$ :**

$$\begin{aligned}
\dot{\psi}_j &= \dot{\psi}_{j-1} - \frac{1}{2} \left( \text{sum}[\dot{L}_{d\eta_j} \odot \dot{\eta}_{j-1}] + \text{sum}[\dot{L}_{d\Lambda_j} \odot \dot{\Lambda}_{j-1}] + \text{sum}[\dot{L}_{dK_{AA}} \odot \dot{K}_{AA}] \right. \\
&\quad \left. + \text{sum}[\dot{L}_{dK_{X_j A}} \odot \dot{K}_{X_j A}] + \text{sum}[\dot{L}_{dj_{X_j X_j}} \odot \dot{j}_{X_j X_j}] + 1_{[\theta_j = \sigma_n]} \dot{L}_{d_{d_n}} \right) \\
\dot{H}_j &= \dot{K}_{X_j A} K_{AA}^{-1} - K_{X_j A} K_{AA}^{-1} \dot{K}_{AA} K_{AA}^{-1}; \\
\dot{d}_j &= \text{diag}[\dot{K}_{X_j X_j} - \dot{K}_{X_j A} K_{AA}^{-1} K_{AA} K_{AX_j} + K_{X_j A} K_{AA}^{-1} \dot{K}_{AA} K_{AA}^{-1} K_{AX_j} - K_{X_j A} K_{AA}^{-1} \dot{K}_{AX_j}]; \\
\dot{V}_j^{-1} &= -1_{[\theta_i \neq \sigma_n]} \alpha \mathbf{V}_j^{-1} \text{Diag}[\dot{d}_j] \mathbf{V}_j^{-1} - 1_{[\theta_i = \sigma_n]} 2\sigma_n^2 \dot{V}_j^{-1} \dot{V}_j^{-1}; \\
\dot{\eta}_j &= \dot{\eta}_{j-1} + \dot{H}_j^T \mathbf{V}_j^{-1} \mathbf{y}_j + \mathbf{H}_j^T \dot{V}_j^{-1} \mathbf{y}_j; \\
\dot{\Lambda}_j &= \dot{\Lambda}_{j-1} + \dot{H}_j^T \mathbf{V}_j^{-1} \mathbf{H}_j + \mathbf{H}_j^T \dot{V}_j^{-1} \mathbf{H}_j + \mathbf{H}_j^T \mathbf{V}_j^{-1} \dot{H}_j
\end{aligned}$$

For the noise  $\sigma_n$ , all jernel derivatives are zero, therefore the calculations simplify significantly.

# Appendix C

## Appendix of Chapter 7

### C.1 Proofs

**Proof C.1 (Proof of Proposition 7.1; Joint Distribution)** . *The matrices in the conditional distributions (7.5) and (7.6) in Proposition 7.1 can be obtained via Gaussian conditioning (2.3) from the assumed joint densities*

$$\begin{aligned} p(\mathbf{f}_j^i, \mathbf{a}_{\psi(j)}) &= \mathcal{N}(\mathbf{0}, \mathbf{K}_{[X_j^i; A_{\psi(j)}][X_j^i; A_{\psi(j)}]}), \\ p(\mathbf{a}_j, \mathbf{a}_{\pi(j)}) &= \mathcal{N}(\mathbf{0}, \mathbf{K}_{[A_j; A_{\pi(j)}][A_j; A_{\pi(j)}]}), \end{aligned}$$

resulting in

$$\begin{aligned} \mathbf{H}_j &= \mathbf{K}_{X_j A_{\psi(j)}} \mathbf{K}_{A_{\psi(j)} A_{\psi(j)}}^{-1}, \\ \bar{\mathbf{V}}_j &= \text{Diag}[\mathbf{K}_{X_j X_j} - \mathbf{K}_{X_j A_{\psi(j)}} \mathbf{K}_{A_{\psi(j)} A_{\psi(j)}}^{-1} \mathbf{K}_{A_{\psi(j)} X_j}], \\ \mathbf{F}_j &= \mathbf{K}_{A_j A_{\pi(j)}} \mathbf{K}_{A_{\pi(j)} A_{\pi(j)}}^{-1}, \\ \mathbf{Q}_j &= \mathbf{K}_{A_j A_j} - \mathbf{K}_{A_j A_{\pi(j)}} \mathbf{K}_{A_{\pi(j)} A_{\pi(j)}}^{-1} \mathbf{K}_{A_{\pi(j)} A_j}, \end{aligned}$$

with  $\mathbf{F}_1 = \mathbf{0}$  and  $\mathbf{Q}_1 = \mathbf{K}_{A_1 A_1}$ .

**Proof C.2 (Proof used in Definition 7.4; Joint Distribution II)** *In the case  $\gamma = 1$ , thus  $\mathbf{a}_j = \mathbf{f}_j$  and  $\mathbf{a} = \mathbf{f}$ , the joint distribution can be written as  $q(\mathbf{f}, \mathbf{a}, \mathbf{y}) = q(\mathbf{f}, \mathbf{f}, \mathbf{y}) = q(\mathbf{f}, \mathbf{y})$ , which is equivalent to*

$$\begin{aligned} \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\psi(j)}) p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) \frac{p(\mathbf{f}_j \mathbf{f}_{\psi(j)})}{p(\mathbf{f}_{\psi(j)})} p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) \end{aligned}$$

since

$$\frac{p(\mathbf{f}_j, \mathbf{f}_{\psi(j)})}{p(\mathbf{f}_{\psi(j)})} = \frac{p(\mathbf{f}_j, \mathbf{f}_j, \mathbf{f}_{\psi(j)\setminus j})}{p(\mathbf{f}_j, \mathbf{f}_{\psi(j)\setminus j})} = 1.$$

**Proof C.3 (Proof of Proposition 7.7; Equality to Full GP ) Full GP:** For  $\gamma = 1$ , the joint distribution of our model is formulated in Definition 7.4 and Proof C.2. For  $C = J$ , we have

$$q_J(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\pi_j(j)}),$$

where the predecessor set  $\pi_j(j)$  correspond to  $\{1, \dots, j-1\}$  and thus the conditional variables  $\mathbf{f}_{\pi_j(j)} = \mathbf{f}_{1:j-1}$ . The posterior  $q_J(\mathbf{f} | \mathbf{y})$  is proportional to the joint distribution  $q_J(\mathbf{f}, \mathbf{y})$  (see Proof C.12), thus we have

$$q_J(\mathbf{f} | \mathbf{y}) \propto \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{1:j-1}),$$

which is equal to the posterior distribution of full GP (7.1). Also the hyperparameter optimization is the same, since the marginal likelihood  $q_J(\mathbf{y})$  can be derived from the joint  $q_J(\mathbf{f}, \mathbf{y})$  (see Proof C.13). Further, in the prediction step, for  $C = J$  we have  $J_2 = C - J + 1 = 1$  predictive expert which is based on the full region  $\psi(J) = \{1, \dots, J\}$ . Therefore we conclude that the two models in considerations are the same.

**Sparse global GP:** Similarly, for  $C = J$  but  $\gamma < 1$ , we have

$$\begin{aligned} q_J(\mathbf{f}, \mathbf{y}) &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{1:j}) p(\mathbf{a}_j | \mathbf{a}_{1:j-1}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}) p(\mathbf{a}) = p(\mathbf{y} | \mathbf{f}) p(\mathbf{f} | \mathbf{a}) p(\mathbf{a}) \end{aligned}$$

so that the posterior correspond to that of sparse GP in (7.2). The prediction step simplifies also to 1 predictive expert based on the full region. Moreover, the marginal likelihood is the same for  $C = J$  and could be adapted as shown in Section 7.2.8.

**Independent local GP:** For  $C = \gamma = 1$  we have

$$q_1(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_\emptyset) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j),$$

which is equal to (7.3). Prediction and hyperparameter estimation similar as above.

**Proof C.4 (Proof of Proposition 7.2; Prior Approximation)** Here we prove the first part for the prior over  $\mathbf{a}$ , the second part is proven in Proof C.5. Using Proposition 7.10 (with Proof C.15), the prior  $q(\mathbf{a})$  can be equivalently written as

$$\prod_{j=1}^J \mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{S}_{(j)}^{-1}) \propto -\frac{1}{2} \mathbf{a}_{\pi^+(j)}^T \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j \mathbf{a}_{\pi^+(j)},$$

with  $\mathbf{S}_{(j)} = \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j \in \mathbb{R}^{LC \times LC}$  and  $\tilde{\mathbf{F}}_j = [-\mathbf{F}_j \quad \mathbb{I}] \in \mathbb{R}^{L \times LC}$ . This  $LC$ -dimensional Gaussian for  $\mathbf{a}_{\pi^+(j)}$  can be augmented to a  $M$ -dimensional Gaussian for  $\mathbf{a}$  proportional to

$$-\frac{1}{2} \mathbf{a}^T \tilde{\mathbf{F}}_j^T \tilde{\mathbf{Q}}_j^{-1} \tilde{\mathbf{F}}_j \mathbf{a} \propto \mathcal{N}(\mathbf{a} | \mathbf{0}, (\tilde{\mathbf{F}}_j^T \tilde{\mathbf{Q}}_j^{-1} \tilde{\mathbf{F}}_j)^{-1}),$$

where  $\tilde{\mathbf{Q}}_j^{-1} \in \mathbb{R}^{M \times M}$  a zero matrix except  $\mathbf{Q}_j^{-1} \in \mathbb{R}^{L \times L}$  at the entries  $[\pi^+(j), \pi^+(j)]$ . Further, the matrix  $\tilde{\mathbf{F}}_j \in \mathbb{R}^{M \times M}$  has one sparse row at  $j$ , that is,

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & -\mathbf{F}_j^1 & 0 & -\mathbf{F}_j^i & \cdots & -\mathbf{F}_j^{I_j} & \mathbb{I} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where  $\mathbf{F}_j^i \in \mathbb{R}^{L \times L}$  is the  $i$ th part of  $\mathbf{F}_j \in \mathbb{R}^{L \times L(C-1)}$ , which corresponds to the contribution of the  $i$ th predecessor  $\pi^i(j)$ .

By using the property in (2.11), the original product  $q(\mathbf{a})$  is then

$$\begin{aligned} \prod_{j=1}^J \mathcal{N}(\mathbf{a} | \mathbf{0}, (\tilde{\mathbf{F}}_j^T \tilde{\mathbf{Q}}_j^{-1} \tilde{\mathbf{F}}_j)^{-1}) &= \mathcal{N}\left(\mathbf{a} | \mathbf{0}, \left(\sum_{j=1}^J \tilde{\mathbf{F}}_j^T \tilde{\mathbf{Q}}_j^{-1} \tilde{\mathbf{F}}_j\right)^{-1}\right) \\ &= \mathcal{N}(\mathbf{a} | \mathbf{0}, (\mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F})^{-1}) = \mathcal{N}(\mathbf{a} | \mathbf{0}, \mathbf{S}^{-1}) \end{aligned}$$

with  $\mathbf{Q}^{-1} = \text{Diag}[\mathbf{Q}_1^{-1}, \dots, \mathbf{Q}_J^{-1}]$  and  $\mathbf{F}$  corresponds then to the matrix depicted in Fig. 7.6. Note that,  $\mathbf{S}$  is positive definite since  $\mathbf{Q}^{-1}$  positive definite because each  $\mathbf{Q}_j^{-1}$  is positive definite, which concludes the proof.

**Proof C.5 ((Sub)proof of Proposition 7.2 (Projection Approximation) )** The projection  $q(\mathbf{f}|\mathbf{a}) = q_C(\mathbf{f}|\mathbf{a})$  is

$$q_C(\mathbf{f}|\mathbf{a}) = \prod_{j=1}^J p(\mathbf{f}_j|\mathbf{a}_{\psi(j)}) = \prod_{j=1}^J \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j),$$

where  $\mathbf{H}_j \in \mathbb{R}^{B \times LC}$  and  $\bar{\mathbf{V}}_j \in \mathbb{R}^{B \times B}$ . The log of this density in  $\mathbf{f}_j \in \mathbb{R}^B$  is proportional to

$$\propto -\frac{1}{2}(\mathbf{f}_j - \mathbf{H}_j\mathbf{a}_{\psi(j)})^T \bar{\mathbf{V}}_j^{-1} (\mathbf{f}_j - \mathbf{H}_j\mathbf{a}_{\psi(j)}),$$

which can be equivalently written as

$$-\frac{1}{2}(\mathbb{I}_j\mathbf{f} - \bar{\mathbf{H}}_j\mathbf{a})^T \bar{\bar{\mathbf{V}}}_j^{-1} (\mathbb{I}_j\mathbf{f} - \bar{\mathbf{H}}_j\mathbf{a}),$$

with  $\bar{\bar{\mathbf{V}}}_j \in \mathbb{R}^{M \times M}$  with  $\bar{\mathbf{V}}_j$  at  $[\psi(j), \psi(j)]$ . Further,  $\bar{\mathbf{H}}_j \in \mathbb{R}^{BJ \times M}$  the following matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \mathbf{H}_j^1 & 0 & \mathbf{H}_j^i & \cdots & \mathbf{H}_j^{C-1} & \mathbf{H}_j^C & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where  $j$ th row not empty with  $\mathbf{H}_j^i \in \mathbb{R}^{B \times L}$  the  $i$ th entry in  $\mathbf{H}_j$ , which corresponds to  $\psi^i(j)$ . Further,  $\mathbb{I}_j \in \mathbb{R}^{BJ \times BJ}$  a zero matrix with  $\mathbb{I} \in \mathbb{R}^{B \times B}$  at  $[j, j]$ . For the original product of the projections

$$q_C(\mathbf{f}|\mathbf{a}) = \prod_{j=1}^J \mathcal{N}(\mathbf{0}|\mathbb{I}_j^T\mathbf{f} - \bar{\mathbf{H}}_j\mathbf{a}, \bar{\bar{\mathbf{V}}}_j),$$

using the product rule of Gaussians in (2.11), we obtain

$$\begin{aligned} & \mathcal{N}\left(\mathbf{0}|\bar{\mathbf{V}}\sum_j^J \bar{\bar{\mathbf{V}}}_j^{-1} (\mathbb{I}_j\mathbf{f} - \bar{\mathbf{H}}_j\mathbf{a}), \left(\sum_j^J \bar{\bar{\mathbf{V}}}_j^{-1}\right)^{-1}\right) \\ &= \mathcal{N}\left(\mathbf{0}|\bar{\mathbf{V}}\bar{\mathbf{V}}^{-1}\sum_j^J (\mathbb{I}_j\mathbf{f} - \bar{\mathbf{H}}_j\mathbf{a}), \bar{\mathbf{V}}\right) = \mathcal{N}\left(\mathbf{0}|\sum_j^J (\mathbb{I}_j)\mathbf{f} - \sum_j^J (\bar{\mathbf{H}}_j)\mathbf{a}, \bar{\mathbf{V}}\right) \\ &= \mathcal{N}(\mathbf{0}|\mathbb{I}\mathbf{f} - \mathbf{H}\mathbf{a}, \bar{\mathbf{V}}) = \mathcal{N}(\mathbf{f}|\mathbf{H}\mathbf{a}, \bar{\mathbf{V}}). \end{aligned}$$

This concludes the statement, since  $\bar{\mathbf{V}}$  positive definite.



**Proof C.6 (Proof of Proposition 7.8; Decreasing Prior KL)** *We first show the decomposition*

$$\mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] = \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f}|\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{y}|\mathbf{f}].$$

*Starting with Equation (7.10), we get*

$$\begin{aligned} & \mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] \\ &= \mathbb{E}_{p(\mathbf{f}, \mathbf{a}, \mathbf{y})} \left[ \log \frac{q_{C+T}(\mathbf{f}, \mathbf{a}, \mathbf{y})}{q_C(\mathbf{f}, \mathbf{a}, \mathbf{y})} \right] \\ &= \mathbb{E}_{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{a})p(\mathbf{a})} \left[ \log \frac{q_{C+T}(\mathbf{y}|\mathbf{f})q_{C+T}(\mathbf{f}|\mathbf{a})q_{C+T}(\mathbf{a})}{q_C(\mathbf{y}|\mathbf{f})q_C(\mathbf{f}|\mathbf{a})q_C(\mathbf{a})} \right] \\ &= \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{a})p(\mathbf{a}) \log \frac{q_{C+T}(\mathbf{y}|\mathbf{f})q_{C+T}(\mathbf{f}|\mathbf{a})q_{C+T}(\mathbf{a})}{q_C(\mathbf{y}|\mathbf{f})q_C(\mathbf{f}|\mathbf{a})q_C(\mathbf{a})} d\mathbf{a} d\mathbf{f} d\mathbf{y} \\ &= \int p(\mathbf{a}) \left( \int p(\mathbf{f}|\mathbf{a}) \left[ \int p(\mathbf{y}|\mathbf{f}) \log \frac{q_{C+T}(\mathbf{y}|\mathbf{f})}{q_C(\mathbf{y}|\mathbf{f})} d\mathbf{y} \dots \right. \right. \\ & \quad \left. \left. + \log \frac{q_{C+T}(\mathbf{f}|\mathbf{a})}{q_C(\mathbf{f}|\mathbf{a})} \right] d\mathbf{f} + \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right) d\mathbf{a} \\ &= \int p(\mathbf{a}) \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} d\mathbf{a} + \int p(\mathbf{a}) \int p(\mathbf{f}|\mathbf{a}) \log \frac{q_{C+T}(\mathbf{f}|\mathbf{a})}{q_C(\mathbf{f}|\mathbf{a})} d\mathbf{f} \\ & \quad + \int p(\mathbf{f}) \int p(\mathbf{y}|\mathbf{f}) \log \frac{q_{C+T}(\mathbf{y}|\mathbf{f})}{q_C(\mathbf{y}|\mathbf{f})} d\mathbf{y} d\mathbf{f} \\ &= \mathbb{E}_{p(\mathbf{a})} \left[ \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right] + \mathbb{E}_{p(\mathbf{a})} \left[ \mathbb{E}_{p(\mathbf{f}|\mathbf{a})} \left[ \log \frac{q_{C+T}(\mathbf{f}|\mathbf{a})}{q_C(\mathbf{f}|\mathbf{a})} \right] \right] \\ & \quad + \mathbb{E}_{p(\mathbf{f})} \left[ \mathbb{E}_{p(\mathbf{y}|\mathbf{f})} \left[ \log \frac{q_{C+T}(\mathbf{y}|\mathbf{f})}{q_C(\mathbf{y}|\mathbf{f})} \right] \right] \\ &= \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f}|\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{y}|\mathbf{f}], \end{aligned}$$

*where we used Equation (7.10). We also immediately see that*

$$\mathbb{D}_{(C,T)}[\mathbf{y}|\mathbf{f}] = 0,$$

*since  $q_C(\mathbf{y}|\mathbf{f}) = q_{C+T}(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$  is exact. The proofs for  $\mathbb{D}_{(C,T)}[\mathbf{a}] \geq 0$  and  $\mathbb{D}_{(C,T)}[\mathbf{f}|\mathbf{a}] \geq 0$  are given in Proof C.7, C.8 and C.8, respectively.*

**Proof C.7 (Proof of Subproof I of Proof C.6)** *We prove*

$$\mathbb{D}_{(C,T)}[\mathbf{a}] = \mathbb{E}_{p(\mathbf{a})} \left[ \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right] \geq 0.$$

We abbreviate  $q_1(\mathbf{a}) = q_C(\mathbf{a})$  and  $q_2(\mathbf{a}) = q_{C+T}(\mathbf{a})$ . The difference  $\mathbb{D}_{(C,T)}[\mathbf{a}]$  is

$$\begin{aligned} & \int p(\mathbf{a}) \log \frac{q_2(\mathbf{a})}{q_1(\mathbf{a})} d\mathbf{a} = \int p(\mathbf{a}) \log \frac{\prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{\prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a} \\ & = \int p(\mathbf{a}) \sum_{j=1}^J \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a} = \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a}. \end{aligned}$$

We recall property (iii) in Definition 7.2, thus we have  $\pi_2(j) = \pi_1(j) \cup \phi(j)$ , where  $\phi(j)$  is the additional predecessor of expert  $j$  in the model 2 compared to model 1. In the following, we abbreviate  $\pi_1(j) = \pi(j)$  yielding

$$\begin{aligned} & \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}, \mathbf{a}_{\phi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ & = \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}, \mathbf{a}_{\phi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ & = \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ & = \sum_{j=1}^J \int p(\tilde{\mathbf{a}}_j) \log \frac{p(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\tilde{\mathbf{a}}_j \\ & = \sum_{j=1}^J I(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)}) \geq 0, \end{aligned}$$

where  $\tilde{\mathbf{a}}_j = \mathbf{a}_j \cup \mathbf{a}_{\phi(j)} \cup \mathbf{a}_{\pi(j)}$  and  $I(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})$  the conditional mutual information, which is always positive [Ajjanagadde et al., 2017, p. 30] and therefore concludes the first part of the proof.

**Proof C.8 (Subproof II of Proof C.6)** We prove

$$\mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] = \mathbb{E}_{p(\mathbf{a})} \left[ \mathbb{E}_{p(\mathbf{f} | \mathbf{a})} \left[ \log \frac{q_{C+T}(\mathbf{f} | \mathbf{a})}{q_C(\mathbf{f} | \mathbf{a})} \right] \right] \geq 0.$$

We abbreviate  $q_1(\mathbf{f} | \mathbf{a}) = q_C(\mathbf{f} | \mathbf{a})$  and  $q_2(\mathbf{f} | \mathbf{a}) = q_{C+T}(\mathbf{f} | \mathbf{a})$ . The difference  $\mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}]$  is

$$\int p(\mathbf{a}) \int p(\mathbf{f} | \mathbf{a}) \log \frac{q_2(\mathbf{f} | \mathbf{a})}{q_1(\mathbf{f} | \mathbf{a})} d\mathbf{f} d\mathbf{a} = \int p(\mathbf{a}, \mathbf{f}) \log \frac{\prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{\prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a}$$

$$= \int p(\mathbf{a}, \mathbf{f}) \sum_{j=1}^J \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a} = \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a}.$$

We recall the definition of  $\psi_c(j)$  in Definition 7.2, namely,  $\psi_c(j) = \pi_c(j) \cup \{j, \dots, C\}$  if  $j < C$  and  $\psi_c(j) = \pi_c(j) \cup j$  otherwise. Further, we have  $\pi_2(j) = \pi_1(j) \cup \phi(j)$ , where  $\phi(j)$  is the additional predecessor of expert  $j$  in the model 2 compared to model 1. Therefore, we have  $\psi_2(j) = \psi_1(j) \cup \phi(j)$  for all  $j$ .

[Proof: If  $j < C$ , we have  $\pi_1(j) = \pi_2(j)$  since  $\phi(j)$  empty. Therefore, we have  $\psi_1(j) = \pi_1(j) \cup \{j, \dots, C\} = \pi_2(j) \cup \{j, \dots, C\} = \psi_2(j)$  for all  $j = 1, \dots, C-1$ . If  $j \geq C$ , we have  $\psi_1(j) = \pi_1(j) \cup j$  and  $\psi_2(j) = \pi_2(j) \cup j = \pi_1(j) \cup \phi(j) \cup j = \psi_1(j) \cup \phi(j)$  for all  $j = C, \dots, J$ . ]

We abbreviate  $\psi_1(j) = \psi(j)$  and substitute  $\psi_2(j) = \psi(j) \cup \phi(j)$  yielding

$$\begin{aligned} & \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}, \mathbf{a}_{\phi(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)})} d\mathbf{f} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}, \mathbf{a}_{\phi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\tilde{\mathbf{a}}_j, \mathbf{f}_j) \log \frac{p(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})} d\tilde{\mathbf{a}}_j \\ &= \sum_{j=1}^J I(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)}) \geq 0, \end{aligned}$$

where  $\tilde{\mathbf{a}}_j = \mathbf{a}_{\phi(j)} \cup \mathbf{a}_{\psi(j)}$  and  $I(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\psi(j)})$  the conditional mutual information which is always positive [Ajjanagadde et al., 2017, p. 30] and therefore concludes the first part of the proof.

Moreover, the difference in the joint prior is

$$\begin{aligned} \mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] &= \mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}] = \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] \\ &= \frac{1}{2} \log \frac{|\tilde{\mathbf{V}}_C| |\mathbf{S}_C^{-1}|}{|\tilde{\mathbf{V}}_{C+T}| |\mathbf{S}_{C+T}^{-1}|} = \frac{1}{2} \log \frac{|\tilde{\mathbf{V}}_C| |\mathbf{Q}_{C+T}|}{|\tilde{\mathbf{V}}_{C+T}| |\mathbf{Q}_C|} \geq 0. \end{aligned}$$

**Proof C.9 (Subproof III of Proof C.6; Prior KL)** For the second part, we use (7.10), where the difference in KL of 2 Gaussians with zero mean and same base distribution is formulated. In our case we have

$$\frac{1}{2} \left( \text{tr}((\mathbf{S}_C - \mathbf{S}_{C+T})\mathbf{K}_{AA}) + \log \frac{|\mathbf{S}_{C+T}|}{|\mathbf{S}_C|} \right),$$

where the trace is 0 by Proposition 7.12 and thus  $\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{S}_{C+T}|}{|\mathbf{S}_C|}$ . Since  $\mathbf{S}_C = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}$  and  $|\mathbf{F}| = 1$ , we have  $\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C+T}|}$ , which concludes the proof.

**Proof C.10 (Proof of Proposition 7.14; Decreasing Prior Entropy)** For the third part of the statement, the entropy  $H$  of  $q_C(\mathbf{a})$  is

$$H[q_C(\mathbf{a})] = \frac{1}{2} (-\log |\mathbf{S}_C| + JL(1 + \log 2\pi)) = \frac{1}{2} (\log |\mathbf{Q}_C| + JL(1 + \log 2\pi)),$$

where we used Eq. (2.7) and  $|\mathbf{F}| = 1$ . The second part follows from Proposition 7.7. Using Proof C.9, which states

$$\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C+T}|} \geq 0,$$

it follows

$$\log |\mathbf{Q}_C| \geq \log |\mathbf{Q}_{C+T}|$$

for any  $T \in \{1, \dots, C-1\}$  and therefore

$$H[q_{C+T}(\mathbf{a})] \leq H[q_C(\mathbf{a})],$$

which concludes the proof.

**Proof C.11 ((Sub)Proof of Proposition 7.3; Marginalized Joint Distribution)** From the joint distribution in Definition 7.4 over all variables, the latent function values  $\mathbf{f}$  can be integrated out resulting in

$$q(\mathbf{a}, \mathbf{y}) = \int q(\mathbf{f}, \mathbf{a}, \mathbf{y}) d\mathbf{f} = \int p(\mathbf{y}|\mathbf{f}) q(\mathbf{f}|\mathbf{a}) d\mathbf{f} q(\mathbf{a}),$$

where the integral can be computed via (2.2) yielding

$$q(\mathbf{a}, \mathbf{y}) = \int p(\mathbf{y}|\mathbf{f}) q(\mathbf{f}|\mathbf{a}) d\mathbf{f} = \int \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I}) \mathcal{N}(\mathbf{f}|\mathbf{H}\mathbf{f}, \bar{\mathbf{V}}) d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V})$$

with  $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$ . Thus

$$q(\mathbf{a}, \mathbf{y}) = q(\mathbf{y}|\mathbf{a}) q(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V}) \mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1}),$$

which concludes the proof.

**Proof C.12 (Proof of Proposition 7.3; Posterior Approximation)** *The posterior approximation is*

$$q(\mathbf{a}|\mathbf{y}) = \frac{q(\mathbf{a}, \mathbf{y})}{q(\mathbf{y})} \propto q(\mathbf{a}, \mathbf{y}) = q(\mathbf{y}|\mathbf{a})q_C(\mathbf{a}),$$

where the first equality comes from the definition of conditional probabilities, the proportionality because the marginal likelihood  $q(\mathbf{y})$  is independent of  $\mathbf{a}$  and the last equality exploits Proof C.11. Since

$$q(\mathbf{y}|\mathbf{a})q_C(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V})\mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1}),$$

the desired posterior distribution can be analytically computed via (2.6) yielding

$$q(\mathbf{a}|\mathbf{y}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

with  $\boldsymbol{\Sigma} = (\mathbf{H}^T \mathbf{V}^{-1} \mathbf{H} + \mathbf{S})^{-1}$ ,  $\boldsymbol{\mu} = \boldsymbol{\Sigma} \boldsymbol{\eta}$  and  $\boldsymbol{\eta} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{y}$ .

**Proof C.13 (Proof of Proposition 7.4; Marginal Likelihood)** *The marginal likelihood  $q(\mathbf{y})$  is obtained by integrating (2.2) over the joint distribution  $q(\mathbf{y}, \mathbf{a})$ , leading to*

$$q(\mathbf{y}) = \int q(\mathbf{y}, \mathbf{a}) d\mathbf{a} = \int q(\mathbf{y}|\mathbf{a})q(\mathbf{a}) d\mathbf{a} = \int \mathcal{N}(\mathbf{H}\mathbf{a}, \mathbf{V})\mathcal{N}(\mathbf{0}, \mathbf{S}^{-1}) d\mathbf{a} = \mathcal{N}(\mathbf{0}, \mathbf{P}),$$

where  $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}$ .

**Proof C.14 (Proof of Proposition 7.12; Exact Diagonal of Prior)** *Using Proposition 7.11, the trace can be written as*

$$\begin{aligned} \text{tr}(\mathbf{S}\mathbf{K}_{AA}) &= \text{tr}\left(\left(\sum_{j=1}^J \overline{\mathbf{K}}_{A_{\pi^+(j)}A_{\pi^+(j)}}^{-1} - \overline{\mathbf{K}}_{A_{\pi(j)}A_{\pi(j)}}^{-1}\right)\mathbf{K}_{AA}\right) \\ &= \sum_{j=1}^J \text{tr}\left(\overline{\mathbf{K}}_{A_{\pi^+(j)}A_{\pi^+(j)}}^{-1}\mathbf{K}_{AA}\right) - \text{tr}\left(\overline{\mathbf{K}}_{A_{\pi(j)}A_{\pi(j)}}^{-1}\mathbf{K}_{AA}\right). \end{aligned}$$

By construction of the matrices  $\overline{\mathbf{K}}_{A_\phi, A_\phi}^{-1}$ , they contain the matrix  $\mathbf{K}_{A_\phi, A_\phi}^{-1}$  at the entries  $[\phi, \phi]$ . Therefore, the resulting product, when multiplying with  $\mathbf{K}_{AA}$ , is a matrix with identity  $\mathbb{I}_T$  at the position  $[\phi, \phi]$  with  $T = |\phi|$  and 0 at the diagonal where not  $\phi$ . The quantity above is then

$$\sum_{j=1}^J L|\pi^+(j)| - L|\pi(j)| = \sum_{j=1}^J L(\min(j, C) - \min(j-1, C-1)) = JL.$$

**Proof C.15 (Proof of Proposition 7.10; Prior Approximation II)** *The prior approximation is*

$$q(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \mathcal{N}(\mathbf{a}_j | \mathbf{F}_j \mathbf{a}_{\pi(j)}, \mathbf{Q}_j),$$

for which the quadratic term inside the exponential of the individual Gaussian can be written as

$$-\frac{1}{2}(\mathbf{a}_j - \mathbf{F}_j \mathbf{a}_{\pi(j)})^T \mathbf{Q}_j^{-1} (\mathbf{a}_j - \mathbf{F}_j \mathbf{a}_{\pi(j)}) - \frac{1}{2} \begin{bmatrix} \mathbf{a}_{\pi(j)}^T & \mathbf{a}_j^T \end{bmatrix} \begin{bmatrix} -\mathbf{F}_j^T \\ \mathbb{I} \end{bmatrix} \mathbf{Q}_j^{-1} \begin{bmatrix} -\mathbf{F}_j & \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbf{a}_{\pi(j)} \\ \mathbf{a}_j \end{bmatrix}$$

which corresponds to a Gaussian

$$\mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{S}_{(j)}^{-1})$$

with  $\mathbf{S}_{(j)} = \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j \in \mathbb{R}^{LC \times LC}$  and  $\tilde{\mathbf{F}}_j = \begin{bmatrix} -\mathbf{F}_j & \mathbb{I} \end{bmatrix} \in \mathbb{R}^{L \times LC}$ , which proves the first part. We can augment this Gaussian for  $\mathbf{a}_{\pi^+(j)} \in \mathbb{R}^{LC}$  to

$$-\frac{1}{2} \mathbf{a}^T \bar{\mathbf{S}}_{(j)}^{-1} \mathbf{a} \propto \mathcal{N}(\mathbf{a} | \mathbf{0}, \bar{\mathbf{S}}_{(j)}^{-1})$$

over  $\mathbf{a} \in \mathbb{R}^M$ , where  $\bar{\mathbf{S}}_{(j)} \in \mathbb{R}^{M \times M}$  is the augmented matrix consisting of  $\mathbf{S}_{(j)}$  at the entries  $[\pi^+(j), \pi^+(j)]$  and 0 otherwise. Using (2.11), the original product  $q(\mathbf{a})$  is then

$$\prod_{j=1}^J \mathcal{N}(\mathbf{a} | \mathbf{0}, \bar{\mathbf{S}}_{(j)}^{-1}) = \mathcal{N}\left(\mathbf{a} | \mathbf{0}, \left(\sum_{j=1}^J \bar{\mathbf{S}}_{(j)}\right)^{-1}\right)$$

and thus  $\mathbf{S} = \sum_{j=1}^J \bar{\mathbf{S}}_{(j)}$  positive definite, which concludes the proof.

**Proof C.16 (Proof of Proposition 7.11; Prior Approximation III)** *The prior  $q(\mathbf{a})$  can be written as*

$$\begin{aligned} q(\mathbf{a}) &= \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \frac{p(\mathbf{a}_j, \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_{\pi(j)})} = \prod_{j=1}^J \frac{p(\mathbf{a}_{\pi^+(j)})}{p(\mathbf{a}_{\pi(j)})} \\ &= \prod_{j=1}^J \frac{\mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{K}_{\mathbf{A}_{\pi^+(j)} \mathbf{A}_{\pi^+(j)}})}{\mathcal{N}(\mathbf{a}_{\pi(j)} | \mathbf{0}, \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}})}. \end{aligned}$$

Similarly to the Proof C.15, we can augment the  $CL$ -dimensional and the  $(C-1)L$ -dimensional Gaussian in the nominator and denominator, respectively, to a

$M$ -dimensional Gaussian with covariance  $\overline{\mathbf{K}}_{A_\phi A_\phi}^{-1}$  consisting of  $\mathbf{K}_{A_\phi A_\phi}^{-1}$  at the entries  $[\phi, \phi]$  and 0 otherwise. This gives with (2.11)

$$\prod_{j=1}^J \frac{\mathcal{N}(\mathbf{a}|\mathbf{0}, \overline{\mathbf{K}}_{A_{\pi^+(j)} A_{\pi^+(j)}})}{\mathcal{N}(\mathbf{a}|\mathbf{0}, \overline{\mathbf{K}}_{A_{\pi(j)} A_{\pi(j)}})} = \mathcal{N}\left(\mathbf{a}|\mathbf{0}, \left(\sum_{j=1}^J \overline{\mathbf{K}}_{A_{\pi^+(j)} A_{\pi^+(j)}}^{-1} - \overline{\mathbf{K}}_{A_{\pi(j)} A_{\pi(j)}}^{-1}\right)^{-1}\right),$$

. Since

$$\mathbf{S} = \sum_{j=1}^J \overline{\mathbf{K}}_{A_{\pi^+(j)} A_{\pi^+(j)}}^{-1} - \overline{\mathbf{K}}_{A_{\pi(j)} A_{\pi(j)}}^{-1} \text{ positive definite, this concludes the proof.}$$

**Proof C.17 (Proof of Proposition 7.5; Prediction Aggregation)** The predictive posterior distribution is defined as

$$p(f_*|\mathbf{y}) = \prod_{j=C}^J p(f_{*j}|\mathbf{y})^{\beta_{*j}}.$$

Since the local predictions  $p(f_{*j}|\mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$  are all univariate Gaussians, we obtain via the product rule of Gaussians in (2.11) directly

$$m_* = v_{*j} \sum_{j=C}^J \beta_{*j} \frac{m_{*j}}{v_{*j}} \quad \text{and} \quad \frac{1}{v_*} = \sum_{j=C}^J \frac{\beta_{*j}}{v_{*j}}.$$

Using the usual likelihood  $p(y_*|f_*) = \mathcal{N}(f_*, \sigma_n^2)$  yields with (2.2) the final noisy prediction  $p(y_*|\mathbf{y}) = \int p(y_*|f_*) p(f_*|\mathbf{y}) df_* = \mathcal{N}(m_*, v_* + \sigma_n^2)$ .

**Proof C.18 (Proof of Proposition 7.6; Local Predictions)** The predictive conditional  $p(f_{*j}|\mathbf{a}_{\psi(j)})$  can be again derived via (2.3) from the assumed joint

$$p(f_{*j}, \mathbf{a}_{\psi(j)}) = \mathcal{N}\left(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_*, \mathbf{A}_{\psi(j)}][\mathbf{x}_*, \mathbf{A}_{\psi(j)}]}\right)$$

leading to  $\mathcal{N}(\mathbf{h}_* \mathbf{a}_{\psi(j)}, v_*)$  with

$$\mathbf{h}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1}$$

and

$$v_* = \mathbf{K}_{\mathbf{x}_* \mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_* \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{x}_*}.$$

Moreover, the local posteriors  $q(\mathbf{a}_{\psi(j)}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$  are obtained from the corresponding entries  $\psi(j)$  of the mean  $\boldsymbol{\mu}$  and covariance  $\boldsymbol{\Sigma}$  (via partial inversion

7.2.7.2) in Prop, 7.3. Finally, the local predictions  $p(f_{*j}|\mathbf{y})$  in Proposition (7.6) can then be computed with Gaussian integration (2.2) yielding

$$q(f_{*j}|\mathbf{y}) = \int p(f_{*j}|\mathbf{a}_{\psi(j)})p(\mathbf{a}_{\psi(j)}|\mathbf{y})d\mathbf{a}_{\psi(j)},$$

which correspond to the desired quantities

$$\mathcal{N}(m_{*j}, v_{*j}) = \mathcal{N}(\mathbf{h}_* \boldsymbol{\mu}_{\psi(j)}, \mathbf{h}_*^T \boldsymbol{\Sigma}_{\psi(j)} \mathbf{h}_* + v_*).$$

**Proof C.19 (Proof for Figure 7.13; Joint Prior Covariance)** For the joint prior

$$q_C(\mathbf{a}, \mathbf{f}, \mathbf{y}) = \mathcal{N}([\mathbf{a}; \mathbf{f}; \mathbf{y}] \mid \mathbf{0}, \mathbf{W}_\gamma)$$

with covariance

$$\mathbf{W}_\gamma = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{af} & \boldsymbol{\Sigma}_{ay} \\ \boldsymbol{\Sigma}_{fa} & \boldsymbol{\Sigma}_{ff} & \boldsymbol{\Sigma}_{fy} \\ \boldsymbol{\Sigma}_{ya} & \boldsymbol{\Sigma}_{yf} & \boldsymbol{\Sigma}_{yy} \end{bmatrix}$$

corresponding to Fig. 7.13, we show that we recover the marginal and conditional distributions  $q_C(\mathbf{a})$ ,  $q_C(\mathbf{f}|\mathbf{a})$  and  $p(\mathbf{y}|\mathbf{f})$ . For  $q_C(\mathbf{a})$ , the marginalization corresponds to selecting the corresponding mean and covariance, i.e.  $\mathcal{N}(\mathbf{a}|\mathbf{0}, \boldsymbol{\Sigma}_{aa}) = \mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1})$ . For  $q_C(\mathbf{f}|\mathbf{a})$ , we use Eq. (2.3) yielding

$$\begin{aligned} & \mathcal{N}(\mathbf{f} \mid \boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} \mathbf{a}, \boldsymbol{\Sigma}_{ff} - \boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} \boldsymbol{\Sigma}_{af}) \\ & = \mathcal{N}(\mathbf{f} \mid \mathbf{H}\mathbf{a}, (\mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \bar{\mathbf{V}}) - \mathbf{H}(\mathbf{S}^{-1}\mathbf{H}^T)) = \mathcal{N}(\mathbf{f} \mid \mathbf{H}\mathbf{a}, \bar{\mathbf{V}}) \end{aligned}$$

since  $\boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} = (\mathbf{H}\mathbf{S}^{-1})\mathbf{S} = \mathbf{H}$ . Similarly for  $p(\mathbf{y}|\mathbf{f})$ , with Eq. (2.3) we get

$$\begin{aligned} & \mathcal{N}(\mathbf{y} \mid \boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} \mathbf{f}, \boldsymbol{\Sigma}_{yy} - \boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} \boldsymbol{\Sigma}_{fy}) \\ & = \mathcal{N}(\mathbf{y} \mid \mathbb{I}\mathbf{f}, (\mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}) - \mathbb{I}(\mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \bar{\mathbf{V}})) = \mathcal{N}(\mathbf{y} \mid \mathbf{f}, \sigma_n^2 \mathbb{I}) \end{aligned}$$

since  $\boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} = \mathbb{I}$ .

## C.2 Additional Material

### Derivatives of LML

The log marginal likelihood in Section 7.2.7.3 is proportional to

$$-\frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} + \frac{1}{2} \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} - \frac{1}{2} \log |\boldsymbol{\Sigma}^{-1}| - \frac{1}{2} \log |\mathbf{V}| - \frac{1}{2} \log |\mathbf{Q}|.$$



In the following, we provide the partial derivative with respect to  $\theta$  for each additive term.

$$\frac{\partial}{\partial \theta} \left[ -\frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} \right] = \frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \frac{\partial \mathbf{V}}{\partial \theta} \mathbf{V}^{-1} \mathbf{y}$$

$$\frac{\partial}{\partial \theta} \left[ \frac{1}{2} \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \right] = \frac{\partial \boldsymbol{\eta}^T}{\partial \theta} \boldsymbol{\mu} - \frac{1}{2} \boldsymbol{\mu}^T \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \boldsymbol{\mu}$$

$$\frac{\partial}{\partial \theta} \left[ -\frac{1}{2} \log |\boldsymbol{\Sigma}^{-1}| \right] = -\frac{1}{2} \text{tr} \left\{ \boldsymbol{\Sigma} \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$$

In the last expression, the whole posterior covariance is needed, however, it turns out that only the entries, which are non-zero in the precision, are needed. The right term in the last expression equals  $\text{sum} \left\{ \boldsymbol{\Sigma} \odot \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$ , where  $\odot$  denotes the pointwise multiplication. Therefore it is enough to only compute  $\text{sum} \left\{ \bar{\boldsymbol{\Sigma}} \odot \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$ , where  $\bar{\boldsymbol{\Sigma}}$  is the partial inversion (for more details 7.2.7.2). which is sparse as well and already computed for the local predictions in Proposition 7.6. The derivatives of the log-determinants can be computed as follows:

$$\frac{\partial}{\partial \theta} \left[ -\frac{1}{2} \log |\mathbf{V}| \right] = -\frac{1}{2} \text{sum} \left\{ \mathbf{V}^{-1} \odot \frac{\partial \mathbf{V}}{\partial \theta} \right\},$$

$$\frac{\partial}{\partial \theta} \left[ -\frac{1}{2} \log |\mathbf{Q}| \right] = -\frac{1}{2} \text{sum} \left\{ \mathbf{Q}^{-1} \odot \frac{\partial \mathbf{Q}}{\partial \theta} \right\}$$

The derivatives  $\frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta}$ ,  $\frac{\partial \mathbf{V}}{\partial \theta}$  and  $\frac{\partial \mathbf{Q}}{\partial \theta}$  can be computed via chain rule of derivatives.

### C.3 Tables

Here we provide more results for the experiments in Section 7.3 and the datasets in Table 7.5a. In the following, we report different average quantities for several test points  $\mathbf{x}_*$ ,  $\mathbf{y}_*$ , corresponding to the predictive distributions  $p(\mathbf{y}_* | \mathbf{y}) = \mathcal{N}(m_*, v_*)$ . The considered quantities are *Kullback-Leibler-(KL)-divergence (KL)* to full GP, *Continuous Ranked Probability Score (CRPS)* and *95%-coverage (COV)*, *root mean squared error (RMSE)*, *absolut error (ABSE)*, *negative log probability (NLP)*, *root mean squared error to full GP (ERR)* and *log marginal likelihood (LML)*.

We use the KL to compare the closeness of predictive distributions of different GP approximation models to the one of full GP  $\mathcal{N}(m, \nu)$ . Since both are univariate Gaussians, the  $KL(\mathcal{N}(m, \nu) \parallel \mathcal{N}(m_*, \nu_*))$  can be computed as following  $\frac{1}{2} \left( \log \frac{\nu_*}{\nu} + \frac{\nu}{\nu_*} + \frac{(m-m_*)^2}{\nu_*} - 1 \right)$ . The CRPS can be used to assess the respective accuracy of two probabilistic forecasting models. In particular, it is a measure between the forecast CDF  $F_*$  of  $\mathcal{N}(m_*, \nu_*)$  and the empirical CDF of the observation  $y_*$  and is defined as  $CRPS(F_*, y_*) \int (F(z) - 1_{z \geq y_*})^2 dz$ . The 95%-confidence interval can be computed as  $c_{1,2} = m_* \pm 1.96\sqrt{\nu_*}$ . The 95%-coverage is then defined as  $COV = 1_{c_1 \leq y_* \leq c_2}$ . The negative log probability is  $-p(y_* | y) = \frac{1}{2} \log(2\pi\nu_*) + \frac{(y_* - m_*)^2}{2\nu_*}$ . For all quantities except LML (large values are better) and COV (should be close to 0.95), small values mean better predictions.

	time	LML	KL	ERR	CRPS	RMSE	ABSE	NLP	COV
fullGP	7.3 ± 0.6	-314.2 ± 5.1	0.0 ± 0.0	0.0 ± 0.0	0.162 ± 0.004	0.311 ± 0.011	0.218 ± 0.005	0.47 ± 0.12	0.92 ± 0.01
SGP(25)	6.4 ± 0.6	-595.4 ± 10.7	440.3 ± 19.6	0.314 ± 0.008	0.234 ± 0.005	0.422 ± 0.01	0.324 ± 0.005	1.11 ± 0.04	0.96 ± 0.01
SGP(50)	14.5 ± 2.6	-539.6 ± 10.2	405.0 ± 31.3	0.291 ± 0.012	0.222 ± 0.004	0.402 ± 0.008	0.308 ± 0.005	1.01 ± 0.03	0.95 ± 0.01
SGP(100)	36.4 ± 2.9	-494.6 ± 7.8	352.9 ± 29.5	0.264 ± 0.011	0.211 ± 0.004	0.384 ± 0.007	0.292 ± 0.006	0.92 ± 0.03	0.95 ± 0.01
minVar	1.5 ± 0.1	-389.8 ± 2.9	122.2 ± 13.1	0.156 ± 0.012	0.175 ± 0.004	0.335 ± 0.011	0.236 ± 0.005	0.61 ± 0.09	0.92 ± 0.01
GPoE	1.4 ± 0.1	-389.8 ± 2.9	174.4 ± 9.4	0.166 ± 0.01	0.186 ± 0.004	0.342 ± 0.01	0.255 ± 0.007	0.68 ± 0.05	0.96 ± 0.01
BCM	1.4 ± 0.1	-389.8 ± 2.9	338.1 ± 32.7	0.185 ± 0.012	0.195 ± 0.005	0.354 ± 0.01	0.265 ± 0.007	1.16 ± 0.12	0.82 ± 0.01
RBCM	1.4 ± 0.1	-389.8 ± 2.9	427.9 ± 35.0	0.166 ± 0.013	0.187 ± 0.005	0.342 ± 0.011	0.249 ± 0.006	1.43 ± 0.21	0.79 ± 0.01
GRBCM	1.7 ± 0.1	-465.0 ± 3.1	224.6 ± 30.3	0.202 ± 0.011	0.19 ± 0.004	0.352 ± 0.01	0.262 ± 0.006	0.71 ± 0.05	0.92 ± 0.01
CPoE(1)	1.5 ± 0.0	-397.0 ± 2.8	111.1 ± 12.5	0.146 ± 0.011	0.175 ± 0.004	0.333 ± 0.011	0.237 ± 0.006	<b>0.59 ± 0.09</b>	0.93 ± 0.01
CPoE(2)	2.1 ± 0.1	-345.1 ± 5.6	89.6 ± 14.3	0.124 ± 0.013	0.172 ± 0.004	0.326 ± 0.011	0.232 ± 0.006	0.6 ± 0.1	0.91 ± 0.01
CPoE(3)	2.5 ± 0.1	-337.0 ± 5.5	82.2 ± 14.3	0.116 ± 0.013	<b>0.17 ± 0.004</b>	<b>0.323 ± 0.01</b>	0.231 ± 0.005	<b>0.59 ± 0.1</b>	0.91 ± 0.01
CPoE(4)	2.8 ± 0.1	-339.4 ± 5.0	<b>79.5 ± 13.9</b>	<b>0.111 ± 0.012</b>	0.171 ± 0.004	0.324 ± 0.011	0.232 ± 0.005	0.6 ± 0.1	0.91 ± 0.01

Table C.1. Results for dataset *concrete*.

	time	LML	KL	ERR	CRPS	RMSE	ABSE	NLP	COV
fullGP	25.5 ± 1.1	-994.2 ± 1.1	0.0 ± 0.0	0.0 ± 0.0	0.283 ± 0.002	0.511 ± 0.004	0.39 ± 0.005	1.49 ± 0.02	0.94 ± 0.0
SGP(25)	7.5 ± 0.7	-1082.8 ± 0.9	93.49 ± 3.86	0.232 ± 0.005	0.316 ± 0.003	0.561 ± 0.004	0.445 ± 0.005	1.68 ± 0.02	0.94 ± 0.0
SGP(50)	9.7 ± 1.4	-1042.7 ± 5.2	41.4 ± 5.59	0.146 ± 0.012	0.299 ± 0.003	0.537 ± 0.003	0.416 ± 0.006	1.59 ± 0.01	0.94 ± 0.0
SGP(100)	14.4 ± 0.8	-1009.6 ± 1.2	9.86 ± 1.73	0.069 ± 0.006	0.285 ± 0.002	0.514 ± 0.004	0.395 ± 0.005	1.51 ± 0.02	0.94 ± 0.0
minVar	2.0 ± 0.2	-1025.8 ± 1.1	19.39 ± 1.78	0.101 ± 0.005	<b>0.282 ± 0.002</b>	<b>0.508 ± 0.005</b>	<b>0.39 ± 0.003</b>	<b>1.48 ± 0.02</b>	0.93 ± 0.0
GPoE	1.9 ± 0.1	-1025.8 ± 1.1	54.22 ± 1.64	0.162 ± 0.003	0.301 ± 0.002	0.535 ± 0.004	0.424 ± 0.006	1.6 ± 0.01	0.96 ± 0.0
BCM	1.9 ± 0.1	-1025.8 ± 1.1	257.61 ± 8.81	0.209 ± 0.005	0.313 ± 0.003	0.555 ± 0.006	0.422 ± 0.004	2.02 ± 0.04	0.82 ± 0.0
RBCM	1.9 ± 0.1	-1025.8 ± 1.1	38.35 ± 1.56	0.132 ± 0.003	0.295 ± 0.003	0.528 ± 0.005	0.408 ± 0.005	1.56 ± 0.02	0.92 ± 0.0
GRBCM	2.3 ± 0.2	-1048.9 ± 1.7	69.12 ± 6.48	0.196 ± 0.01	0.307 ± 0.004	0.551 ± 0.007	0.431 ± 0.006	1.64 ± 0.02	0.94 ± 0.0
CPoE(1)	2.1 ± 0.1	-1025.8 ± 1.1	12.18 ± 0.92	0.079 ± 0.003	0.284 ± 0.002	0.51 ± 0.004	0.393 ± 0.003	1.49 ± 0.02	0.94 ± 0.0
CPoE(2)	2.8 ± 0.1	-1010.1 ± 1.5	8.44 ± 0.66	0.066 ± 0.003	0.285 ± 0.002	0.512 ± 0.004	0.394 ± 0.004	1.5 ± 0.02	0.93 ± 0.0
CPoE(3)	3.1 ± 0.1	-1007.0 ± 1.5	7.83 ± 0.58	0.064 ± 0.002	0.285 ± 0.002	0.513 ± 0.004	0.394 ± 0.004	1.5 ± 0.02	0.93 ± 0.0
CPoE(4)	3.3 ± 0.1	-1004.8 ± 1.5	<b>7.59 ± 0.63</b>	<b>0.062 ± 0.003</b>	0.285 ± 0.002	0.513 ± 0.004	0.393 ± 0.004	1.5 ± 0.02	0.93 ± 0.0

Table C.2. Results for dataset *mg*.

	time	LML	KL	ERR	CRPS	RMSE	ABSE	NLP	COV
fullGP	114.8 ± 4.3	-2113.6 ± 5.6	0.0 ± 0.0	0.0 ± 0.0	0.255 ± 0.005	0.471 ± 0.01	0.348 ± 0.007	1.3 ± 0.04	0.95 ± 0.0
SGP(50)	34.8 ± 4.8	-2319.6 ± 7.4	137.62 ± 7.41	0.259 ± 0.009	0.288 ± 0.005	0.531 ± 0.012	0.395 ± 0.007	1.57 ± 0.04	0.95 ± 0.0
SGP(100)	46.6 ± 6.1	-2242.4 ± 7.5	108.14 ± 6.24	0.229 ± 0.008	0.279 ± 0.005	0.514 ± 0.012	0.382 ± 0.007	1.5 ± 0.04	0.95 ± 0.0
SGP(150)	56.6 ± 6.8	-2205.9 ± 6.6	90.94 ± 6.01	0.21 ± 0.009	0.275 ± 0.005	0.508 ± 0.012	0.376 ± 0.007	1.47 ± 0.04	0.94 ± 0.0
minVar	7.2 ± 0.2	-2312.6 ± 6.8	63.58 ± 2.93	0.19 ± 0.01	0.272 ± 0.006	0.508 ± 0.016	0.374 ± 0.008	1.41 ± 0.04	0.95 ± 0.0
GPoE	7.2 ± 0.2	-2312.6 ± 6.8	98.01 ± 3.06	0.2 ± 0.013	0.279 ± 0.006	0.515 ± 0.02	0.378 ± 0.008	1.49 ± 0.03	0.97 ± 0.0
BCM	7.2 ± 0.2	-2312.6 ± 6.8	222.78 ± 4.12	0.2 ± 0.008	0.28 ± 0.007	0.511 ± 0.016	0.38 ± 0.008	1.75 ± 0.1	0.87 ± 0.01
RBCM	7.2 ± 0.2	-2312.6 ± 6.8	635.61 ± 21.61	0.194 ± 0.011	0.285 ± 0.007	0.513 ± 0.018	0.378 ± 0.008	2.54 ± 0.18	0.77 ± 0.01
GRBCM	6.5 ± 0.2	-2397.3 ± 6.2	105.64 ± 5.13	0.24 ± 0.008	0.284 ± 0.005	0.525 ± 0.012	0.391 ± 0.007	1.5 ± 0.04	0.95 ± 0.01
CPoE(1)	7.8 ± 0.2	-2316.1 ± 6.8	62.99 ± 2.94	0.186 ± 0.011	0.272 ± 0.006	0.507 ± 0.018	0.372 ± 0.008	1.41 ± 0.04	0.96 ± 0.0
CPoE(2)	10.6 ± 0.2	-2164.9 ± 6.7	36.45 ± 3.02	0.142 ± 0.011	0.264 ± 0.005	0.491 ± 0.015	<b>0.361 ± 0.008</b>	<b>1.36 ± 0.04</b>	0.95 ± 0.0
CPoE(3)	12.9 ± 0.2	-2165.9 ± 6.7	36.27 ± 2.99	0.141 ± 0.01	<b>0.263 ± 0.005</b>	0.49 ± 0.014	<b>0.361 ± 0.008</b>	<b>1.36 ± 0.04</b>	0.95 ± 0.0
CPoE(4)	14.9 ± 0.2	-2166.2 ± 6.7	<b>36.03 ± 3.0</b>	<b>0.14 ± 0.01</b>	<b>0.263 ± 0.005</b>	<b>0.489 ± 0.014</b>	<b>0.361 ± 0.008</b>	<b>1.36 ± 0.04</b>	0.95 ± 0.0

Table C.3. Results for dataset *space*.

	time	LML	KL	ERR	CRPS	RMSE	ABSE	NLP	COV
fullGP	237.9 ± 12.2	-3722.3 ± 7.4	0.0 ± 0.0	0.0 ± 0.0	0.34 ± 0.005	0.635 ± 0.012	0.459 ± 0.006	1.92 ± 0.04	0.94 ± 0.0
SGP(20)	21.9 ± 2.5	-3785.3 ± 5.8	27.8 ± 4.1	0.15 ± 0.01	0.343 ± 0.004	0.635 ± 0.011	0.463 ± 0.005	1.93 ± 0.03	0.95 ± 0.0
SGP(50)	26.4 ± 3.6	-3758.7 ± 7.6	22.4 ± 3.9	0.14 ± 0.01	0.342 ± 0.004	0.633 ± 0.011	0.461 ± 0.006	1.93 ± 0.03	0.94 ± 0.0
SGP(100)	58.9 ± 7.0	-3746.9 ± 7.4	15.6 ± 3.5	0.11 ± 0.01	<b>0.34 ± 0.005</b>	<b>0.631 ± 0.012</b>	<b>0.457 ± 0.006</b>	1.92 ± 0.04	0.94 ± 0.0
minVar	6.4 ± 0.4	-3847.3 ± 7.2	25.1 ± 1.5	0.15 ± 0.0	0.346 ± 0.005	0.647 ± 0.013	0.466 ± 0.006	1.94 ± 0.04	0.94 ± 0.0
GPoE	6.3 ± 0.4	-3847.3 ± 7.2	50.3 ± 1.0	0.19 ± 0.0	0.353 ± 0.004	0.652 ± 0.011	0.478 ± 0.006	1.99 ± 0.02	0.96 ± 0.0
BCM	6.3 ± 0.3	-3847.3 ± 7.2	1838.2 ± 46.8	0.16 ± 0.0	0.373 ± 0.006	0.642 ± 0.011	0.473 ± 0.006	5.33 ± 0.24	0.67 ± 0.01
RBCM	6.3 ± 0.3	-3847.3 ± 7.2	1147.4 ± 64.8	0.12 ± 0.0	0.362 ± 0.006	0.638 ± 0.012	0.466 ± 0.006	4.01 ± 0.21	0.73 ± 0.01
GRBCM	7.6 ± 0.4	-3864.0 ± 7.6	36.4 ± 1.9	0.18 ± 0.0	0.353 ± 0.004	0.661 ± 0.011	0.477 ± 0.005	1.98 ± 0.03	0.94 ± 0.0
CPoE(1)	6.4 ± 0.4	-3848.6 ± 7.3	16.8 ± 0.6	0.12 ± 0.0	0.342 ± 0.004	0.638 ± 0.012	0.463 ± 0.005	1.92 ± 0.03	0.95 ± 0.0
CPoE(2)	7.5 ± 0.3	-3737.3 ± 7.0	8.1 ± 0.5	0.08 ± 0.0	0.341 ± 0.005	0.636 ± 0.012	0.463 ± 0.006	1.92 ± 0.04	0.94 ± 0.0
CPoE(3)	9.3 ± 0.5	-3736.5 ± 7.2	6.2 ± 0.6	0.07 ± 0.0	0.341 ± 0.005	0.636 ± 0.012	0.461 ± 0.006	1.92 ± 0.04	0.94 ± 0.0
CPoE(4)	10.4 ± 0.3	-3733.7 ± 7.0	<b>4.7 ± 0.5</b>	<b>0.06 ± 0.0</b>	<b>0.34 ± 0.005</b>	0.635 ± 0.012	0.46 ± 0.006	<b>1.91 ± 0.04</b>	0.94 ± 0.0

Table C.4. Results for dataset *abalone*.

	time	LML	KL	ERR	CRPS	RMSE	ABSE	NLP	COV
fullGP	161.5 ± 3.6	-1232.1 ± 7.4	0.0 ± 0.0	0.0 ± 0.0	0.148 ± 0.001	0.267 ± 0.001	0.207 ± 0.001	0.17 ± 0.01	0.94 ± 0.0
SGP(100)	42.2 ± 6.5	-4033.6 ± 27.1	603.7 ± 9.4	0.4 ± 0.01	0.265 ± 0.003	0.476 ± 0.005	0.369 ± 0.004	1.35 ± 0.02	0.96 ± 0.0
SGP(200)	49.8 ± 3.3	-3141.3 ± 17.8	408.4 ± 3.7	0.29 ± 0.0	0.218 ± 0.001	0.392 ± 0.001	0.303 ± 0.001	0.96 ± 0.0	0.96 ± 0.0
SGP(300)	54.8 ± 2.2	-2732.8 ± 13.5	323.1 ± 5.0	0.25 ± 0.0	0.201 ± 0.001	0.363 ± 0.001	0.281 ± 0.001	0.8 ± 0.01	0.96 ± 0.0
minVar	9.3 ± 0.2	-2820.5 ± 9.0	211.0 ± 2.3	0.2 ± 0.0	0.183 ± 0.001	0.333 ± 0.001	0.256 ± 0.001	0.59 ± 0.01	0.94 ± 0.0
GPoE	9.4 ± 0.1	-2820.5 ± 9.0	342.3 ± 2.6	0.23 ± 0.0	0.202 ± 0.001	0.354 ± 0.002	0.278 ± 0.002	0.84 ± 0.01	0.99 ± 0.0
BCM	9.4 ± 0.1	-2820.5 ± 9.0	1629.2 ± 24.7	0.25 ± 0.0	0.218 ± 0.002	0.367 ± 0.002	0.278 ± 0.002	3.45 ± 0.07	0.64 ± 0.0
RBCM	9.4 ± 0.2	-2820.5 ± 9.0	939.3 ± 17.4	0.2 ± 0.0	0.193 ± 0.001	0.331 ± 0.002	0.253 ± 0.001	2.06 ± 0.05	0.71 ± 0.0
GRBCM	11.9 ± 0.2	-2981.3 ± 9.6	129.8 ± 3.0	0.14 ± 0.0	0.168 ± 0.001	0.303 ± 0.001	0.235 ± 0.001	0.43 ± 0.01	0.94 ± 0.0
CPoE(1)	9.2 ± 0.1	-2822.7 ± 8.9	152.4 ± 1.7	0.15 ± 0.0	0.17 ± 0.001	0.307 ± 0.001	0.237 ± 0.001	0.46 ± 0.0	0.97 ± 0.0
CPoE(2)	12.9 ± 0.1	-1811.2 ± 11.1	79.9 ± 1.3	0.11 ± 0.0	0.161 ± 0.001	0.29 ± 0.001	0.225 ± 0.001	0.33 ± 0.01	0.95 ± 0.0
CPoE(3)	19.8 ± 0.3	-1466.0 ± 9.9	46.9 ± 1.0	0.09 ± 0.0	0.155 ± 0.001	0.279 ± 0.001	0.217 ± 0.001	0.26 ± 0.01	0.95 ± 0.0
CPoE(4)	27.8 ± 0.2	-1363.8 ± 9.2	<b>32.8 ± 1.0</b>	<b>0.07 ± 0.0</b>	<b>0.153 ± 0.001</b>	<b>0.276 ± 0.001</b>	<b>0.215 ± 0.001</b>	<b>0.24 ± 0.01</b>	0.94 ± 0.0

Table C.5. Results for dataset *kin*.

	time	LML	CRPS	RMSE	ABSE	NLP	COV
SGP(250)	77.7 ± 0.4	-4163.9 ± 23.7	0.207 ± 0.002	0.366 ± 0.004	0.282 ± 0.002	0.93 ± 0.01	0.98 ± 0.0
SGP(500)	112.1 ± 1.2	-3242.2 ± 12.6	0.183 ± 0.001	0.324 ± 0.002	0.252 ± 0.001	0.67 ± 0.01	0.98 ± 0.0
SGP(1000)	244.1 ± 2.9	-2534.7 ± 9.0	0.166 ± 0.001	0.294 ± 0.002	0.23 ± 0.001	0.46 ± 0.01	0.98 ± 0.0
minVar	14.4 ± 0.5	-3388.8 ± 7.9	0.173 ± 0.002	0.314 ± 0.004	0.242 ± 0.002	0.48 ± 0.02	0.94 ± 0.0
GPoE	14.4 ± 0.5	-3388.8 ± 7.9	0.193 ± 0.001	0.34 ± 0.003	0.267 ± 0.002	0.76 ± 0.01	0.99 ± 0.0
BCM	14.4 ± 0.5	-3388.8 ± 7.9	0.21 ± 0.001	0.35 ± 0.003	0.266 ± 0.002	3.6 ± 0.1	0.63 ± 0.0
RBCM	14.4 ± 0.5	-3388.8 ± 7.9	0.188 ± 0.001	0.318 ± 0.003	0.244 ± 0.002	2.39 ± 0.09	0.69 ± 0.0
GRBCM	16.5 ± 0.4	-3388.8 ± 7.9	0.164 ± 0.001	0.294 ± 0.003	0.229 ± 0.002	0.37 ± 0.02	0.94 ± 0.0
CPoE(1)	13.8 ± 0.2	-3393.9 ± 8.0	0.163 ± 0.001	0.292 ± 0.003	0.226 ± 0.002	0.38 ± 0.01	0.97 ± 0.0
CPoE(2)	18.9 ± 0.3	-2076.6 ± 12.9	0.155 ± 0.001	0.278 ± 0.002	0.217 ± 0.001	0.27 ± 0.01	0.95 ± 0.0
CPoE(3)	31.7 ± 0.6	-1655.2 ± 8.7	<b>0.151 ± 0.001</b>	<b>0.27 ± 0.002</b>	<b>0.211 ± 0.001</b>	<b>0.21 ± 0.01</b>	0.95 ± 0.0

Table C.6. Results for dataset *kin2* for the stochastic versions.

	time	LML	CRPS	RMSE	ABSE	NLP	COV
SGP(250)	70.9 ± 3.7	-3905.6 ± 23.3	0.207 ± 0.002	0.373 ± 0.004	0.287 ± 0.002	0.85 ± 0.02	0.96 ± 0.00
SGP(500)	86.1 ± 1.8	-2968.6 ± 11.7	0.181 ± 0.001	0.325 ± 0.003	0.252 ± 0.001	0.57 ± 0.01	0.96 ± 0.00
SGP(1000)	143.6 ± 3.6	-2277.2 ± 8.6	0.162 ± 0.001	0.292 ± 0.002	0.225 ± 0.001	0.36 ± 0.01	0.96 ± 0.00
minVar	13.8 ± 0.2	-3384.5 ± 7.8	0.173 ± 0.002	0.314 ± 0.004	0.241 ± 0.002	0.48 ± 0.02	0.94 ± 0.00
GPoE	13.8 ± 0.2	-3384.5 ± 7.8	0.193 ± 0.001	0.34 ± 0.002	0.267 ± 0.002	0.75 ± 0.01	0.99 ± 0.00
BCM	13.8 ± 0.2	-3384.5 ± 7.8	0.209 ± 0.001	0.35 ± 0.003	0.266 ± 0.002	3.63 ± 0.07	0.63 ± 0.00
RBCM	13.8 ± 0.2	-3384.5 ± 7.8	0.187 ± 0.001	0.317 ± 0.003	0.243 ± 0.001	2.38 ± 0.06	0.69 ± 0.00
GRBCM	18.8 ± 0.4	-3608.7 ± 8.4	0.164 ± 0.001	0.294 ± 0.002	0.229 ± 0.002	0.38 ± 0.02	0.94 ± 0.00
CPoE(1)	16.2 ± 0.8	-3389.8 ± 8.0	0.162 ± 0.001	0.292 ± 0.003	0.225 ± 0.002	0.37 ± 0.01	0.97 ± 0.00
CPoE(2)	21.5 ± 0.7	-2071.4 ± 13.0	0.155 ± 0.001	0.278 ± 0.002	0.217 ± 0.001	0.26 ± 0.01	0.95 ± 0.00
CPoE(3)	34.3 ± 0.9	-1650.7 ± 8.3	<b>0.15 ± 0.001</b>	<b>0.27 ± 0.002</b>	<b>0.211 ± 0.001</b>	<b>0.21 ± 0.01</b>	0.94 ± 0.00

Table C.7. Results for dataset *kin2* for the deterministic batch version.

	time	LML	CRPS	RMSE	ABSE	NLP	COV
SGP(250)	248.6 ± 0.6	-15182.0 ± 35.7	0.254 ± 0.003	0.48 ± 0.009	0.335 ± 0.004	1.42 ± 0.03	0.95 ± 0.00
SGP(500)	346.9 ± 3.4	-15074.6 ± 37.2	0.253 ± 0.003	0.478 ± 0.009	0.333 ± 0.004	1.41 ± 0.03	0.95 ± 0.00
SGP(1000)	727.6 ± 3.5	-14961.2 ± 31.4	0.252 ± 0.003	<b>0.476 ± 0.009</b>	0.332 ± 0.004	1.4 ± 0.03	0.95 ± 0.00
minVar	28.2 ± 1.0	-15387.4 ± 17.5	0.257 ± 0.003	0.491 ± 0.009	0.337 ± 0.005	1.42 ± 0.04	0.94 ± 0.00
GPoE	28.3 ± 1.0	-15387.4 ± 17.5	0.289 ± 0.003	0.534 ± 0.009	0.371 ± 0.004	1.64 ± 0.02	0.96 ± 0.00
BCM	28.5 ± 0.9	-15387.4 ± 17.5	0.321 ± 0.004	0.536 ± 0.01	0.373 ± 0.004	20.72 ± 1.0	0.45 ± 0.00
RBCM	28.5 ± 0.9	-15387.4 ± 17.5	0.303 ± 0.005	0.515 ± 0.01	0.358 ± 0.004	15.98 ± 0.9	0.51 ± 0.01
GRBCM	33.5 ± 1.2	-15387.4 ± 17.5	0.262 ± 0.003	0.499 ± 0.009	0.346 ± 0.004	1.44 ± 0.03	0.94 ± 0.00
CPoE(1)	24.5 ± 0.1	-15404.2 ± 17.8	0.259 ± 0.004	0.492 ± 0.01	0.335 ± 0.005	1.43 ± 0.04	0.95 ± 0.00
CPoE(2)	33.4 ± 0.2	-13645.5 ± 19.8	0.251 ± 0.003	0.479 ± 0.009	0.328 ± 0.004	1.36 ± 0.04	0.94 ± 0.00
CPoE(3)	52.0 ± 0.5	-13483.2 ± 15.6	<b>0.249 ± 0.004</b>	<b>0.476 ± 0.01</b>	<b>0.324 ± 0.004</b>	<b>1.34 ± 0.04</b>	0.94 ± 0.00

Table C.8. Results for dataset *cadata*.

	time	LML	CRPS	RMSE	ABSE	NLP	COV
SGP(250)	473.4 ± 1.0	9370.3 ± 60.7	0.0746 ± 0.0005	0.1407 ± 0.0008	0.097 ± 0.001	-0.39 ± 0.01	0.95 ± 0.00
SGP(500)	730.1 ± 1.1	12112.0 ± 68.2	0.0695 ± 0.0003	0.1304 ± 0.001	0.09 ± 0.001	-0.49 ± 0.01	0.95 ± 0.00
SGP(1000)	1718.5 ± 1.8	16034.0 ± 91.6	0.0628 ± 0.0003	0.1172 ± 0.0009	0.081 ± 0.0	-0.64 ± 0.01	0.96 ± 0.00
minVar	71.3 ± 23.1	27128.2 ± 20.2	0.0516 ± 0.0008	0.1024 ± 0.0034	<b>0.067 ± 0.001</b>	<b>-1.88 ± 0.04</b>	0.93 ± 0.00
GPoE	71.4 ± 23.2	27128.2 ± 20.2	0.0862 ± 0.0004	0.1322 ± 0.0013	0.096 ± 0.001	-0.57 ± 0.01	1.0 ± 0.00
BCM	71.5 ± 23.2	27128.2 ± 20.2	0.095 ± 0.001	0.1544 ± 0.001	0.115 ± 0.001	7.86 ± 0.3	0.48 ± 0.01
RBCM	71.6 ± 23.2	27128.2 ± 20.2	0.0726 ± 0.0009	0.1196 ± 0.0013	0.086 ± 0.001	11.45 ± 0.47	0.5 ± 0.01
GRBCM	84.6 ± 23.0	27128.2 ± 20.2	0.06 ± 0.0007	0.1102 ± 0.001	0.079 ± 0.001	-0.52 ± 0.08	0.79 ± 0.01
CPoE(1)	45.4 ± 0.2	-41213.2 ± 883.2	0.0516 ± 0.0005	0.0998 ± 0.0019	0.067 ± 0.001	-1.86 ± 0.02	0.96 ± 0.00
CPoE(2)	67.3 ± 0.4	-37867.5 ± 911.8	0.0509 ± 0.0006	0.0977 ± 0.0015	0.067 ± 0.001	-1.8 ± 0.02	0.93 ± 0.00
CPoE(3)	134.3 ± 1.2	-37204.6 ± 949.1	<b>0.0507 ± 0.0005</b>	<b>0.0975 ± 0.0011</b>	<b>0.067 ± 0.001</b>	-1.78 ± 0.02	0.92 ± 0.00

Table C.9. Results for dataset *sarcos*.

	time	LML	CRPS	RMSE	ABSE	NLP	COV
SGP(250)	443.2 ± 2.1	-53395.2 ± 80.2	0.334 ± 0.004	0.59 ± 0.008	0.475 ± 0.007	1.77 ± 0.02	0.96 ± 0.0
SGP(500)	632.9 ± 2.7	-52988.7 ± 58.9	0.329 ± 0.005	0.582 ± 0.008	0.467 ± 0.007	1.75 ± 0.02	0.96 ± 0.0
SGP(1000)	1362.5 ± 4.8	-52592.1 ± 46.9	0.325 ± 0.005	<b>0.575 ± 0.008</b>	0.459 ± 0.007	1.74 ± 0.02	0.96 ± 0.0
minVar	45.8 ± 1.0	-39976.0 ± 22.6	0.294 ± 0.003	0.607 ± 0.006	0.387 ± 0.003	1.4 ± 0.03	0.93 ± 0.0
GPoE	45.6 ± 0.8	-39976.0 ± 22.6	0.302 ± 0.003	0.6 ± 0.006	0.409 ± 0.005	1.43 ± 0.02	0.97 ± 0.0
BCM	45.7 ± 0.9	-39976.0 ± 22.6	0.316 ± 0.005	0.615 ± 0.009	0.416 ± 0.007	2.47 ± 0.1	0.82 ± 0.01
RBCM	45.7 ± 0.9	-39976.0 ± 22.6	0.312 ± 0.004	0.647 ± 0.008	0.425 ± 0.006	1.61 ± 0.05	0.91 ± 0.01
GRBCM	59.4 ± 1.1	-39976.0 ± 22.6	0.31 ± 0.004	0.642 ± 0.008	0.421 ± 0.005	1.5 ± 0.04	0.92 ± 0.01
<b>CPoE(1)</b>	45.1 ± 0.3	-40075.2 ± 22.1	0.289 ± 0.003	0.596 ± 0.006	0.38 ± 0.004	<b>1.35 ± 0.03</b>	0.94 ± 0.0
<b>CPoE(2)</b>	70.3 ± 0.6	-39571.2 ± 65.5	0.287 ± 0.004	0.589 ± 0.007	0.38 ± 0.005	1.36 ± 0.03	0.93 ± 0.0
<b>CPoE(3)</b>	123.8 ± 1.4	-39439.5 ± 98.8	<b>0.282 ± 0.004</b>	<b>0.575 ± 0.008</b>	<b>0.372 ± 0.006</b>	1.37 ± 0.04	0.92 ± 0.01

Table C.10. Results for dataset *casp*.