The University Of Sheffield.

Sparse Machine Learning Methods for Autonomous Decision Making

Danil Kuzin

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

University of Sheffield
2018

ABSTRACT

Sparse regression methods are used for the reconstruction of compressed signals, that are usually sparse in some bases; or in feature selection problem, where only few features are meaningful. This thesis overviews the existing Bayesian methods for dealing with sparsity, improves them and provides new models for these problems. The novel models decrease complexity, allow to model structure and provide uncertainty distributions in such applications as medicine and computer vision.

The thesis starts with exploring Bayesian sparsity for the problem of compressive background subtraction. Sparsity naturally arises in this problem as foreground usually occupies only small part of the video frame. The use of Bayesian compressive sensing improves the solutions in independent and multi-task scenarios. It also raises an important problem of exploring the structure of the data, as foreground pixels are usually clustered in groups.

The problem of structure modelling in sparse problems is addressed with hierarchical Gaussian processes, that are the Bayesian way of imposing structure without specifying its exact patterns. Full Bayesian inference based on expectation propagation is provided for offline and online algorithms. The experiments demonstrate the applicability of these methods for the compressed background subtraction and brain activity localisation problems.

The majority of sparse Bayesian methods are computationally intensive. This thesis proposes a novel sparse regression method based on the Bayesian neural networks. It makes the prediction operation fast and additionally estimates the uncertainty of predictions, while requiring a longer training phase. The results are demonstrated in the active learning scenario, where the estimated uncertainty is used for experiment design.

Sparse methods are also used as part of other methods such as Gaussian processes that suffer from high computational complexity. The use of active sparse subsets of data improves the performance on large datasets. The thesis proposes a method of dealing with the complexity problem for online data updates using Bayesian filtering.

ACKNOWLEDGMENTS

I would like to thank my supervisor, Prof. Lyudmila Mihaylova for encouragement and guidance of my research. Her insights and knowledge were incredibly inspirational and important during my study. The advices and support kept me on the way to completing this thesis.

I am eternally grateful to my wife, Olga for her help with proofreading of papers and the thesis. Her support with long proofs of some of the methods was enormously helpful. Without the optimism and encouragement from Olga this work would not be possible.

I especially thank my parents, Andrey and Elena, for their financial support during the study. Their moral support was invaluably important for me as well during my life and allowed me to get to this point.

I also acknowledge the support of Seventh Framework Programme TRAcking in compleX sensor systems (TRAX) grant.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

OMP	orthogonal matching pursuit
ISTA	iterative shrinkage and thresholding algorithm
LISTA	learned iterative shrinkage and thresholding algorithm
ADMM	alternating direction method of multipliers
GP	Gaussian process
BCS	Bayesian compressive sensing
MTCS	multi-task Bayesian compressive sensing
EP	expectation propagation
EEG	electroencephalogram
DNN	deep neural network
BNN	Bayesian neural network
EnKF	ensemble Kalman filter
NMSE	normalised mean square error

LIST OF NOTATION

General notation

N	number of data points
n	index of data point
$\mathcal{D}$	dataset
K	size of observations
k	index of observation
$\mathbf{y}^{(n)} \in \mathbb{R}^K$	n-th observation
$\mathbf{Y} \in \mathbb{R}^{N \times K}$	set of observations
D	size of coefficients
d	index of coefficient
$\boldsymbol{\beta}^{(n)} \in \mathbb{R}^D$	n-th coefficient
$\mathbf{B} \in \mathbb{R}^{N \times D}$	set of coefficients
$\mathbf{X} \in \mathbb{R}^{K \times D}$	design matrix
$\boldsymbol{\varepsilon}^{(n)} \in \mathbb{R}^K$	n-th noise term
σ^2	variance of noise
$h_\lambda(\cdot)$	soft thresholding function
$\mathbf{I}$	identity matrix
$\delta_0(\cdot)$	delta function
$\Gamma(\cdot)$	gamma function
$\mathbb{E}[\cdot]$	expectation
$\mathrm{Var}[\cdot]$	variance
$\mathrm{Cov}[\cdot, \cdot]$	covariance
$\Phi(\cdot)$	standard Gaussian cumulative distribution function (cdf)
$\mathcal{N}(\cdot)$	Gaussian distribution
$\mathrm{Ber}(\cdot)$	Bernoulli distribution

$\mathrm{IG}(\cdot)$	inverse gamma distribution
$\mathcal{T}(\cdot)$	Student's t-distribution
$\mathrm{Gam}(\cdot)$	gamma distribution

Notation specific for Chapter 3

$\mathbf{b} \in \mathbb{R}^D$	background frame
$\mathbf{v}^{(n)} \in \mathbb{R}^D$	video frame
$\boldsymbol{\alpha}$	precision for prior of $\boldsymbol{\beta}^{(n)}$

Notation specific for Chapter 4

$g^{(n)}(\cdot)$	factor for distribution of $\boldsymbol{\beta}^{(n)}$
σ_β^2	variance of slab distribution
$\omega_d^{(n)}$	indicator for $\boldsymbol{\beta}^{(n)}$
$\boldsymbol{\Omega}$	set of indicators
$f_d^{(n)}(\cdot,\cdot)$	factor for distribution of $\omega_d^{(n)}$, $\beta_d^{(n)}$
$\gamma_d^{(n)}$	probability of spike
$\boldsymbol{\Gamma}$	set of spike probabilities
$h_d^{(n)}(\cdot,\cdot)$	factor for distribution of $\omega_d^{(n)}$, $\gamma_d^{(n)}$
$\boldsymbol{\mu}^{(n)}$	mean of the spatial GP
$\boldsymbol{\Sigma}_0$	covariance of the spatial GP
$r^{(n)}(\cdot,\cdot)$	factor for distribution of $\boldsymbol{\gamma}^{(n)}$, $\boldsymbol{\mu}^{(n)}$
$u^{(n)}(\cdot,\cdot)$	factor for distribution of $\boldsymbol{\mu}^{(n)}$, $\boldsymbol{\mu}^{(n-1)}$
$\mathbf{W}$	covariance of the temporal GP

Notation specific for Chapter 5

$\mathbf{W}$	LISTA bias matrix
w_{ij}	element of $\mathbf{W}$
$\mathbf{S}$	LISTA multiplication matrix
$s_{d'd''}$	element of $\mathbf{S}$

$\mathbf{c}_l$	non-sparse coefficient approximation after layer l
L	number of layers
$\widehat{\boldsymbol{\beta}}_l$	coefficient approximation after layer l
η	precision of prior for weights
$f(\cdot)$	output of the Bayesian LISTA neural network
Z	normalisation constant

Notation specific for Chapter 6

K	number of grid points
$\mathbf{g} \in \mathbb{R}^K$	mean of the GP
D	dimensionality of input
$\mathbf{x} \in \mathbb{R}^D$	domain of the target function
$\mathbf{X}_g \in \mathbb{R}^{K \times D}$	grid points
L_θ	number of parameters of the covariance function
$\boldsymbol{\theta} \in \mathbb{R}^{L_\theta}$	parameters of the covariance function
N	number of iterations
S	number of function observations
$\mathbf{y}^{(n)} \in \mathbb{R}^S$	function observations
$\mathbf{X}_{new}$	locations of function observation
σ_y^2	the variance of noise
$\boldsymbol{\eta} \in \mathbb{R}^L$	full vector of algorithm parameters
M	number of ensemble points

Chapter 1

INTRODUCTION

Nowadays, massive amounts of data are available for processing. Modern databases contain billions of data entries with thousands of features and, moreover, they are being updated online with more data. It is important to distinguish relevant features for each desired outcome to focus the analysis on them.

The statistical analysis that deals with this problem is usually called *variable selection* and it is often based on a set of methods called *sparse regression*. For example, in genetics, variable selection is used to discover genes responsible for diseases from the set of three billion base pairs in the human genome. In electroencephalogram source localisation, the electromagnetic field is measured at a brain cortex and the measurements are used to discover a particular area inside the brain being active at specific activities of a person, which is usually assumed to be small.

1.1 Different Forms of Sparsity

Sparsity appears in many forms and can be used to replicate nature of data or to improve computational complexity.

1.1.1 Compressive Sensing

In the signal processing field, signals are usually represented as linear combinations of basis functions. Real-life periodic signals are generally well approximated in the *Fourier basis* that is formed of sines and cosines, typically only a small amount of representation coefficients have large values and others are close to zero. Non-continuous signals or signals with a limited domain can be represented in the *wavelet basis*, that is a representation of a function by scaled and translated copies of a decaying oscillating waveform. The existence of such *sparse representations* of the real signals allows to reduce the number of samples required to

capture the signal. The area of *compressive sensing* researches ways of optimising sampling rates and it is, again, based on sparse regression methods.

1.1.2 Structural Sparsity

Further research of sparse regression problems leads to the following observation: sometimes genes responsible for deceases are grouped together in a small area of genome; an active brain area for a particular activity is usually small and localised. The knowledge about patterns of meaningful features allows to improve results of sparse regression, such approach is called *structured sparsity*.

1.1.3 Sparse Coding and Supervised Learning

Usually, the problem of sparse regression requires large computational resources for every new data point as the problems are solved independently. A different approach to the problem is based on supervised learning where an algorithm is built to solve sparse regression problems based on a dataset of input and output pairs. For example, a neural network can be trained for sparse regression in this context. It requires significant resources for initial training, but then allows to solve the sparse regression problem for new data efficiently.

1.1.4 Sparse Approximations of Computational Algorithms

The property of sparsity is also an important component of many algorithms. Sparse approximations increase the scalability of Gaussian process regression to larger amounts of data: they allow to select only a small subset of inputs that are most relevant. Sparse matrices of neural network weights may increase the performance of neural networks: they set some of the weights to zeros, thus removing redundant connections.

1.2 Outline and Key Contributions

Overall, this thesis develops machine learning methods for sparse regression, that allow faster, more accurate reconstruction of signals. It also develops sparse modifications of machine learning methods that improve computational requirements. This section provides an outline and key contributions of the thesis.

Background

The sparse regression problem is actively researched in statistics with the seminal works of Mitchell and Beauchamp (1988), Tibshirani (1996), Tipping (2001), Gregor and LeCun (2010). These significant developments are presented in Chapter 2 of the thesis.

Compressive Background Subtraction

In Chapter 3 the compressive sensing methods for object detection in video are described and sparse Bayesian models are applied for this problem. They improve the quality results compared to previously existed methods. The main contributions of this chapter are:

Bayesian compressive sensing for background subtraction. The compressive background subtraction problem is for the first time considered within the sparse Bayesian framework and its performance is compared with the existing frequentist methods.

Multitask Bayesian compressive sensing for background subtraction. The multitask Bayesian framework extends the model for the sequential data that has similar properties.

Structured Spike and Slab Models

Models for structured sparsity are described in details in Chapter 4, and new structured methods that can work without prior knowledge about the exact patterns of sparsity and with online updates of the data are presented there. The key contributions of the chapter are:

Spike and slab model with a hierarchical Gaussian process prior. The model has a flexible structure which is governed only by the covariance functions of the Gaussian processes. This allows to model different types of structures and does not require any specific knowledge about the structure such as determination of particular groups of coefficients with similar behaviour. If, however, there is an information about the structure, it can be easily incorporated into the covariance functions. The model is flexible as spatial and temporal dependencies are decoupled by different levels of the hierarchical Gaussian process prior. Therefore, the spatial and temporal structures are

modelled independently allowing to encode different assumptions for each type of the structure.

Offline inference algorithm. The developed Bayesian inference algorithm based on expectation propagation achieves full posterior inference and perform predictions.

Online inference algorithm. The novel online inference algorithm for streaming data based on Bayesian filtering improves computational time.

Real data applications. A thorough validation and evaluation of the proposed method over synthetic and real data is presented including electrical activity data for the EEG source localisation problem and video data for the compressive background subtraction problem.

Uncertainty Propagation in Sparse Bayesian Neural Networks

In Chapter 5 the sparse regression problem is considered in the context of supervised learning. A novel Bayesian neural network is proposed for sparse regression that combines efficient predictions from neural networks together with uncertainty estimates from Bayesian methods. The main contributions of this chapter are:

Uncertainty propagation for soft-thresholding. In the Bayesian neural network, uncertainty estimation of the predictions can be achieved as a result of sequential weight uncertainty propagation through the layers of a network which leads to complex resulting distributions due to nonlinearities involved. To make the posterior inference possible, for the first time a method for uncertainty propagation through the soft thresholding nonlinearity for a Bayesian neural network is proposed, that approximates the resulting distributions with the spike and slab family.

Bayesian LISTA. A posterior inference algorithm for weights and outputs of neural networks with the soft thresholding nonlinearity is developed. This allows to design a novel Bayesian LISTA neural network for sparse coding.

Ensemble Kalman Filters for Sparse Gaussian Processes

In Chapter 6 new methods for sparse approximation of Gaussian processes are presented that use the idea of sequential estimation to operate with online data and they reduce the computational requirements. The main contributions of this chapter can be summarised as:

Gaussian processes with the ensemble Kalman filter method. For the first time the ensemble Kalman filter for the problem of online Gaussian process regression and learning is proposed. The method treats the mean and hyperparameters of the Gaussian process as the state and parameters of the ensemble Kalman filter, respectively. This allows to reduce the computational complexity related to prediction, as the size of the invertible matrices is reduced according to the ensemble sizes. The online evaluation of the parameters and the state is performed on new upcoming samples of data. This procedure iteratively improves the accuracy of parameter estimates. The ensemble Kalman filter reduces the computational complexity required to obtain predictions with Gaussian processes preserving the accuracy level of these predictions.

Dual and joint versions. The dual and joint versions of the ensemble Kalman filter that differently approach the state-parameter relationship are presented in the chapter.

House prices experiments. The performance of the algorithms is compared using the synthetic dataset and real large dataset of the house prices.

Conclusions

Chapter 7 provides an overview of the main results of the thesis and directions for future work based on the methods proposed in the thesis.

1.3 Disseminated Results

The results from this research are disseminated in following papers:

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2015). "Compressive sensing approaches for autonomous object detection in video sequences". In: *Proceedings of the Sensor Data Fusion: Trends, Solutions, Applications Workshop (SDF)*. IEEE, pp. 1–6. DOI: 10.1109/SDF.2015.7347706

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2017). "Structured sparse modelling with hierarchical GP". in: *Proceedings of the 6th Signal Processing with Adaptive Sparse Structured Representations Workshop (SPARS)*. URL: http://spars2017.lx.it.pt/index_files/papers/SPARS2017_Paper_48.pdf
- Danil Kuzin, Le Yang, Olga Isupova, and Lyudmila Mihaylova (2018). "Ensemble Kalman filtering for online Gaussian process regression and learning". In: *Proceedings of the 21st International Conference on Information Fusion (FUSION)*. IEEE, pp. 39–46. DOI: 10.23919/ICIF.2018.8455785
- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2018a). "Spatio-temporal structured sparse regression with hierarchical Gaussian process priors". In: *IEEE Transactions on Signal Processing* vol. 66, issue 17, pp. 4598–4611. DOI: 10.1109/TSP.2018.2858207
- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2018b). "Uncertainty propagation in neural networks for sparse coding". In: *Proceedings of the Third Workshop on Bayesian Deep Learning (NeurIPS)*. URL: http://bayesiandeeplearning.org/2018/papers/47.pdf
- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2019). "Bayesian neural networks for sparse coding". Accepted at IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

Other co-authored works during my study at the University are:

- Olga Isupova, Danil Kuzin, and Lyudmila Mihaylova (2015). "Abnormal behaviour detection in video using topic modeling". In: *USES Conference Proceedings*. The University of Sheffield. DOI: 10.15445/02012015.18

- Olga Isupova, Lyudmila Mihaylova, Danil Kuzin, Garik Markarian, and Francois Septier (2015). "An expectation maximisation algorithm for behaviour analysis in video". In: *Proceedings of the 18th International Conference on Information Fusion (FUSION)*. IEEE, pp. 126–133. URL: https://ieeexplore.ieee.org/document/7266553/

- Olga Isupova, Danil Kuzin, and Lyudmila Mihaylova (2016). "Dynamic hierarchical Dirichlet process for abnormal behaviour detection in video". In: *Proceedings of the*

19th International Conference on Information Fusion (FUSION). IEEE, pp. 750–757. URL: https://ieeexplore.ieee.org/document/7527962/

- Olga Isupova, Danil Kuzin, and Lyudmila Mihaylova (2018). "Learning methods for dynamic topic modeling in automated behavior analysis". In: *IEEE Transactions on Neural Networks and Learning Systems* vol. 29, issue 9, pp. 3980–3993. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2017.2735364

Chapter 2

BACKGROUND

This chapter presents the sparse regression problem and important solutions for it that lay the foundation of the thesis. It also describes the compressive sensing problem that is used in most experiments throughout the thesis. In addition, it overviews a Gaussian process which is the significant part of the methodology in the thesis.

2.1 Sparse Regression

The main goal of sparse regression is to recover the *coefficient vectors* based on the *observed vectors* and the *design matrix* with the assumption of sparsity of the coefficient vectors. The prevalent form of relationship between coefficients and observations is linear, as this degree of freedom is enough for many real-world applications. It can be achieved with a linear regression approach.

Let the data be represented by N observations as $\mathcal{D} = \{\mathbf{y}^{(n)}, \boldsymbol{\beta}^{(n)}\}_{n=1}^{N}$. Every observation n consists of the coefficient vector $\boldsymbol{\beta}^{(n)} \in \mathbb{R}^D$ and the observed vector $\mathbf{y}^{(n)} \in \mathbb{R}^K$. The linear relationship is represented using the design matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$ of covariates as

$$\mathbf{y}^{(n)} = \mathbf{X}\boldsymbol{\beta}^{(n)} + \boldsymbol{\varepsilon}^{(n)}, \tag{2.1}$$

where $\boldsymbol{\varepsilon}^{(n)} \in \mathbb{R}^K$ is the noise that captures other factors not included in the linear model.

The sparsity assumption can be expressed with penalty constraints in the frequentist interpretation of statistical inference, or priors in the Bayesian interpretation.

2.1.1 Frequentist Interpretation

In the *frequentist statistics* the coefficients $\boldsymbol{\beta}^{(n)}$ are fixed and unknown, they are selected so that the resulting distribution on $\mathbf{y}^{(n)}$ matches the real distribution for training data.

The frequentist interpretation of sparsity is to solve a regularised regression problem. For example, l_p–norm penalty functions with $0 < p < 2$ encourage sparse solutions of the coefficient vector:

$$\boldsymbol{\beta}^{(n)} = \text{argmin}_{\boldsymbol{\beta}^{(n)}} \left[||\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\beta}^{(n)}||_2^2 + ||\boldsymbol{\beta}^{(n)}||_p^p \right], \tag{2.2}$$

where l_p norm is

$$||\boldsymbol{\beta}^{(n)}||_p = \left(\sum_{d=1}^{D} |\beta_d^{(n)}|^p \right)^{\frac{1}{p}}. \tag{2.3}$$

As p approaches zero, the penalty function becomes the function that counts the number of non-zero elements of the coefficient vector, which could be interpreted as the true sparsity penalty for many problems.

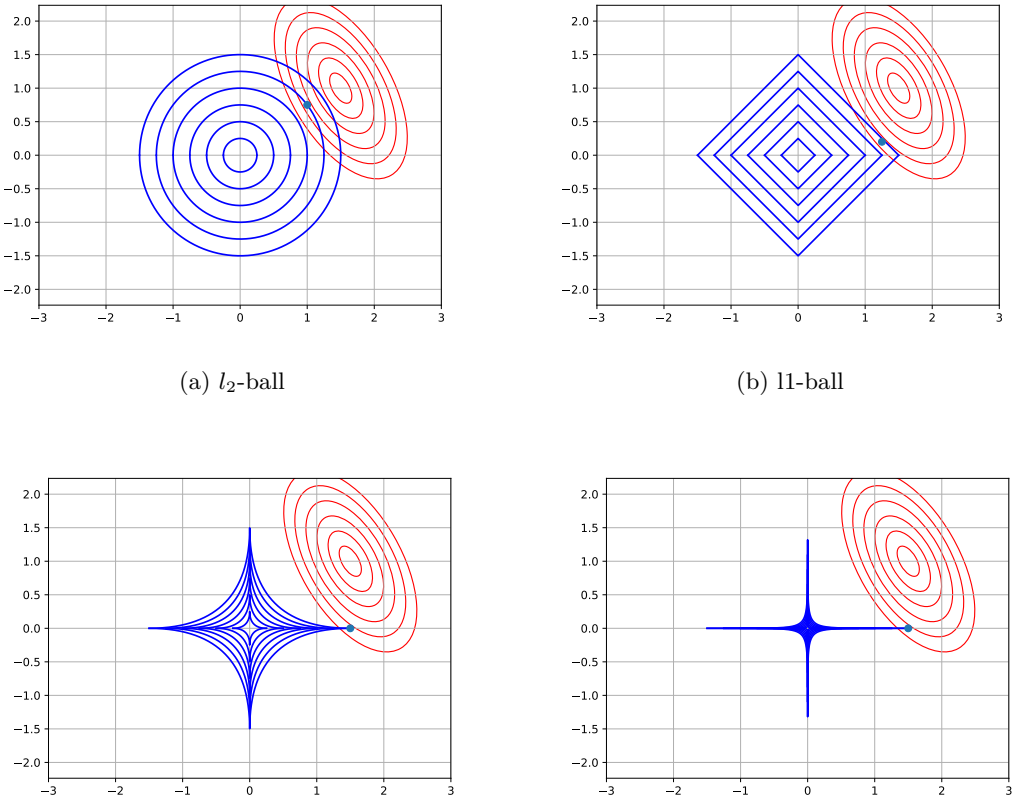
For the intuition, why l_p-penalty functions provide sparse solutions, Figure 2.1 demonstrates an example of two-dimensional contour lines for the error term $||\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\beta}^{(n)}||_2$ and the penalty term $||\boldsymbol{\beta}^{(n)}||_p$ for different p. For sparsity-inducing p, the sum of these terms is minimal when the solution is sparse.

Different choices of p in these penalty functions provide solutions with different properties, that are further discussed. While there exist a great amount of algorithms for different frequentist interpretations of sparsity (Bach et al. 2012a), this section presents only the examples that are used in the thesis.

Regularisation with Non-zero Counting Penalty

The non-zero counting penalty function ($p \to 0$) is the most obvious approach for sparsity, but the solution of such optimisation problem is NP-hard (Bach et al. 2012a). The globally optimal solution can be found only by comparing the target function for all possible values of coefficients, that is computationally intensive. Greedy algorithms, such as matching pursuits (Mallat and Zhang 1993), are used for this problem, however, they tend to find only locally optimal solutions.

Consider the problem (2.2) for one data point n. Orthogonal matching pursuit (OMP) is a greedy method that iteratively updates the set γ of the non-zero elements of $\boldsymbol{\beta}^{(n)}$. Initially, γ is empty; then, at every iteration l, OMP picks the next component of $\boldsymbol{\beta}^{(n)}$ with the index



(a) l_2-ball

(b) l1-ball

(c) $l_{1/2}$-ball

(d) $l_{1/8}$-ball

Figure 2.1: Example contour lines for different values of p. Red lines are the contour lines for the error term, with minimum at $\boldsymbol{\beta}^{(n)} = (1.5, 1.0)$. Blue lines are the contour lines for the penalty term, with minimum at $\boldsymbol{\beta}^{(n)} = (0, 0)$. The minimal sum of these terms is displayed as a point, one of which components moves towards zero as p approaches zero, therefore achieving sparsity.

d^* as the most correlated with the current residual and not included in the active set yet

$$d^* = \underset{d \notin \gamma}{\operatorname{argmin}} \left\{ \min_{\omega} ||\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\beta}_l^{(n)} - \omega \mathbf{x}_{:,d}||^2 \right\}, \tag{2.4}$$

where $\mathbf{x}_{:,d}$ denotes the d-th column of $\mathbf{X}$. The inner optimisation is solved as

$$\omega = \frac{\mathbf{x}_{:,d}^\top r_l}{||\mathbf{x}_{:,d}||_2^2}, \tag{2.5}$$

with $r_l = ||\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\beta}_l^{(n)}||_2$ being the current residual.

Regularisation with l_p-quasinorm, $0 < p < 1$

For $0 < p < 1$, the penalty term (2.3) is non-convex. While, in theory, these penalties may provide better results than the l_1 penalty and be more computationally efficient than the non-zero counting penalty, there are no optimisation methods developed for these penalties. The recent piece-wise quadratic error potentials method (Gorban et al. 2016) is an attempt to tackle this problem, however, it is not well-researched yet and is out of scope for this thesis.

Regularisation with l_1-norm

For $p = 1$, the problem (2.2) is called lasso (Tibshirani 1996). The penalty function is convex, but non-smooth. The convexity of the problem allows to use general efficient convex optimisation methods to find optimums.

Iterative shrinkage and thresholding algorithm (ISTA) (Daubechies et al. 2004) iteratively obtains the new approximation of the coefficient vector $[\widehat{\boldsymbol{\beta}}^{(n)}]_l$ at the iteration l as the linear transformation of the input $\mathbf{y}^{(n)}$ with the previous approximation $[\widehat{\boldsymbol{\beta}}^{(n)}]_{l-1}$ and then propagates the new approximation through the soft thresholding function $h_\lambda(\cdot)$

$$h_\lambda(v) = \operatorname{sgn}(v) \max(|v| - \lambda, 0), \tag{2.6}$$

where λ is a shrinkage parameter. The linear transformation is

$$\widehat{\boldsymbol{\beta}}_l^{(n)} = \mathbf{W}\mathbf{y}^{(n)} + \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}^{(n)}, \tag{2.7}$$

with weights $\mathbf{W} = \mathbf{X}^\top / E$, E — the upper bound on the largest eigenvalue of $\mathbf{X}^\top\mathbf{X}$ and $\mathbf{S} = \mathbf{I}_{D\times D} - \mathbf{W}\mathbf{X}$, $\mathbf{I}_{D\times D}$ is the identity matrix of size D.

Alternating direction method of multipliers (ADMM) is the general-purpose algorithm that can solve the convex l_1 optimisation problem using the augmented Lagrangian scheme (Boyd et al. 2011). It augments the problem with an additional variable $\mathbf{c}$, and weights γ, ρ, and $\mathbf{\Lambda}$

$$\mathcal{L} = 0.5||\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\beta}^{(n)}||_2^2 + \gamma||\mathbf{c}||_1 + 0.5\rho||\boldsymbol{\beta}^{(n)} - \mathbf{c}||_2^2 + \mathbf{\Lambda}(\boldsymbol{\beta}^{(n)} - \mathbf{c}), \tag{2.8}$$

and minimises the Lagrangian sequentially w.r.t. to the target $\boldsymbol{\beta}^{(n)}$ and augmented variable $\mathbf{c}$ thus solving two less complex optimisation problems: the least-square regression and independent one-dimensional l_1 minimisation that has soft thresholding as a solution.

Regularisation with l_p-norm, $1 < p < 2$

When $1 < p < 2$, the penalty term is convex and smooth. An example is the elastic net (Zou and Hastie 2005). Smooth penalty function allows to use more efficient gradient-based techniques, but solutions are less sparse then in the previous cases.

2.1.2 Bayesian Interpretation

In the Bayesian interpretation, the distribution of noise is assumed Gaussian, $\boldsymbol{\varepsilon}^{(n)} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, and distribution of coefficients $\boldsymbol{\beta}^{(n)}$ and noise variance σ are assumed unknown. A prior distribution of coefficients is used to update a posterior distribution of the coefficients based on the likelihood of the observed data. The posterior distribution is estimated using the Bayes rule

$$p(\boldsymbol{\beta}^{(n)}, \sigma|\mathbf{y}^{(n)}) = \frac{p(\mathbf{y}^{(n)}|\boldsymbol{\beta}^{(n)}, \sigma)p(\boldsymbol{\beta}^{(n)}, \sigma)}{\int p(\mathbf{y}^{(n)}|\boldsymbol{\beta}^{(n)}, \sigma)p(\boldsymbol{\beta}^{(n)}, \sigma)d\boldsymbol{\beta}^{(n)}d\sigma}. \tag{2.9}$$

The prior term $p(\boldsymbol{\beta}^{(n)}, \sigma)$ has been extensively studied in the literature to find an optimal representation of the prior knowledge. In this chapter the solutions leading to sparse posterior estimates of $\boldsymbol{\beta}^{(n)}$ are considered.

The main difficulty in the Bayesian approach is evaluation of the integral in (2.9). In order to make it tractable, *conjugacy* is used: given the *likelihood* $p(\mathbf{y}^{(n)}|\boldsymbol{\beta}^{(n)}, \sigma)$, the *prior* $p(\boldsymbol{\beta}^{(n)}, \sigma)$ is usually selected in the way that the *posterior* $p(\boldsymbol{\beta}^{(n)}, \sigma|\mathbf{y}^{(n)})$ lies in the same class of distributions as the prior. The examples are: the beta likelihood with the binomial prior, the multinomial likelihood with the Dirichlet prior, or the Gaussian likelihood with the Gaussian-inverse-Wishart prior. All these distributions belong to the *exponential family*.

In most cases the integral in (2.9) is intractable. This leads to a great amount of inference techniques proposed in the literature. A general overview of inference methods is given by Bishop (2006), Koller and Friedman (2009), and Murphy (2012).

The Bayesian models for sparse regression can be classified into the models with *weak sparsity* prior and *strong sparsity* prior. The weak sparsity prior is a unimodal distribution of the coefficient vector with a sharp peak at zero. The strong sparsity prior is a mixture of latent binary variables that capture whether a coefficient is zero or not.

Strong Sparsity

The two important strong sparsity prior models are the *spike and slab* model and the *Bernoulli–Gaussian* model. These models put the discrete probability of being zero on each coefficient.

In the first spike and slab formulation (Mitchell and Beauchamp 1988) each component of $\boldsymbol{\beta}^{(n)}$ is selected from a mixture of a spike, that is the delta-function in zero, and a slab, that is the uniform distribution with width $2a$:

$$\beta_d^{(n)}|\omega \sim \begin{cases} \{0\}, & \text{with the probability } \omega; \\ \mathrm{U}[-a;a] \setminus \{0\}, & \text{with the probability } 1-\omega. \end{cases} \tag{2.10}$$

Sparsity naturally comes from the prior formulation as the coefficients are exactly zeros with the predefined probability ω. This sparsity representation has been further developed by utilising different distributions for slabs.

The spike and slab model can be represented in terms of mixtures of Gaussian distributions (George and McCulloch 1993)

$$\beta_d^{(n)}|\omega_d, \tau_d, c_d \sim \omega_d \mathcal{N}(\beta_d^{(n)}; 0, \tau_d^2) + (1-\omega_d)\mathcal{N}(\beta_d^{(n)}; 0, c_d^2\tau_d^2). \tag{2.11}$$

In this model the spike is a narrow Gaussian distribution with the variance τ_d, while the slab is a wide Gaussian distribution, where the variance is $c_d\tau_d$.

Following the Bayesian approach, hidden variables $z_d^{(n)}$ that are indicators of spikes are added to the model (Polson and Scott 2010), they have the Bernoulli distribution. The

inverse gamma distributions are placed over the model variance σ_β^2 (Murphy 2012)

$$\beta_d^{(n)}|z_d^{(n)}, \sigma_\beta^2 \sim (1 - z_d^{(n)})\mathcal{N}(\beta_d^{(n)}; 0, \sigma_\beta^2) + z_d^{(n)}\delta_0, \tag{2.12a}$$

$$z_d^{(n)}|\omega \sim \text{Ber}(z_d^{(n)}; \omega), \tag{2.12b}$$

$$\sigma^2|a_\sigma, b_\sigma \sim \text{IG}(\sigma^2; a_\sigma, b_\sigma). \tag{2.12c}$$

The graphical model for such spike and slab formulation can be found in Figure 2.2a.

The Bernoulli-Gaussian model is also widely considered (Soussen et al. 2011; Zhou et al. 2009). Coefficients are presented as a pair-wise product of

$$\boldsymbol{\beta}^{(n)} = \mathbf{z}^{(n)} \circ \widehat{\boldsymbol{\beta}}^{(n)}, \tag{2.13a}$$

$$z_d^{(n)}|\omega \sim \text{Ber}(z_d^{(n)}; \omega), \tag{2.13b}$$

$$\widehat{\beta}_d^{(n)}|\sigma_\beta^2 \sim \mathcal{N}(\widehat{\beta}_d^{(n)}; 0, \sigma_\beta^2), \tag{2.13c}$$

$$\sigma^2|a_\sigma, b_\sigma \sim \text{IG}(\sigma^2; a_\sigma, b_\sigma). \tag{2.13d}$$

The model is different from the spike and slab formulation in the sense of a hierarchy between coefficients and latent variables. This can be seen from graphical model in Figure 2.2b.

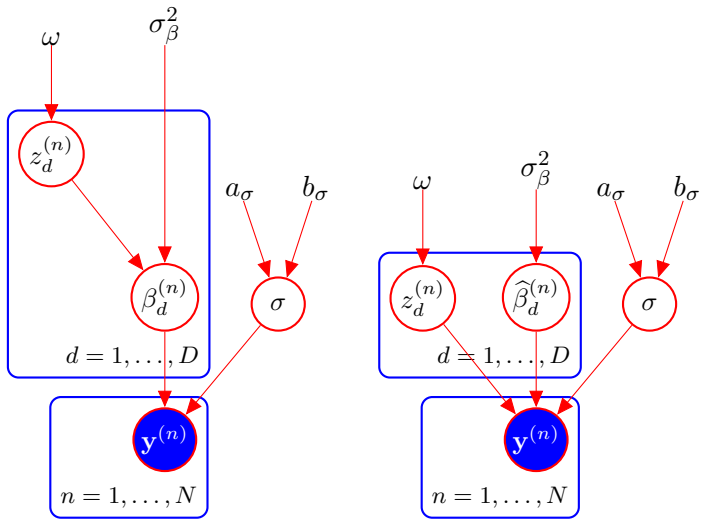
Inference

The most popular inference schemes for these models are *sampling methods* (Chipman et al. 2001). In addition, the message passing algorithms can be used: *approximate message passing* (Donoho et al. 2009) and *expectation propagation* (Hernández-Lobato, Hernández-Lobato, et al. 2015). The advantage of the Bernoulli–Gaussian formulation is that it allows to implement the mean-field variational inference (Titsias and Lázaro-Gredilla 2011).

Weak Sparsity

The weak sparsity prior models are the other class of priors for sparse Bayesian regression.

Many exponential family distributions can be represented as a ratio of two independent variables, where the denominator has the standard Gaussian distribution (Andrews and Mallows 1974). These models, that are called *scale mixtures of Gaussians*, allow to create symmetrical unimodal distributions that are peaked in zero and have heavy tails.



(a) Spike-and-Slab (b) Bernoulli–Gaussian

Figure 2.2: Graphical models for the sparse regression problem. In the spike and slab model (a) latent variables $\{z_d^{(n)}, \beta_d^{(n)}\}$ are organised in hierarchy and the indicators of spikes $z_d^{(n)}$ can be integrated out. In the Bernoulli-Gaussian model (b) the product $z_d^{(n)}\widehat{\beta}_d^{(n)}$ always exists.

Scale mixtures of Gaussians can be represented as a hierarchical model:

$$\boldsymbol{\beta}^{(n)}|\boldsymbol{\mu}, \boldsymbol{\Sigma} \sim \mathcal{N}(\boldsymbol{\beta}^{(n)}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{2.14}$$

$$\boldsymbol{\mu}, \boldsymbol{\Sigma}|\tau \sim \psi(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \upsilon). \tag{2.15}$$

where ψ is the mixing distribution parameterised by υ, which may vary in different models. Marginalisation of the parameters of the Gaussian distribution leads to

$$p(\boldsymbol{\beta}^{(n)}|\upsilon) = \int \mathcal{N}(\boldsymbol{\beta}^{(n)}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\psi(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \upsilon)d\boldsymbol{\mu}d\boldsymbol{\Sigma}. \tag{2.16}$$

For sparse priors, $\boldsymbol{\mu}$ is set to a zero vector to ensure that the distributions have a peak exactly at this point.

Global–local mixtures of Gaussians To represent sparsity, the following factorisation of

scale mixtures of Gaussians can be used (Polson and Scott 2010):

$$\beta_d^{(n)}|\tau, \lambda_d \sim \mathcal{N}(\beta_d^{(n)}; 0, \tau\lambda_d), \tag{2.17}$$

$$\lambda_d \sim \pi(\lambda_d), \tag{2.18}$$

$$\tau|\sigma \sim \phi(\tau; \sigma), \tag{2.19}$$

where π and ϕ are priors for the local variance λ and global variance τ, respectively. Many of the existing sparse priors have this representation. They are summarised in Table 2.1

Posterior for $\beta_d^{(n)}$	**Mixing density π ($\tau \equiv 1$)**	**Reference**
Laplace	Exponential	West (1987)
Student's t	Inverse Gamma	Tipping (2001)
Normal/Jeffreys	Jeffreys	Figueiredo (2003)
Horseshoe	Inverse Beta	Carvalho et al. (2010)
Generalised double Pareto	Exponential-Gamma	Armagan et al. (2013)
Dirichlet Laplace	Exponential-Dirchlet-Gamma	Bhattacharya et al. (2015)

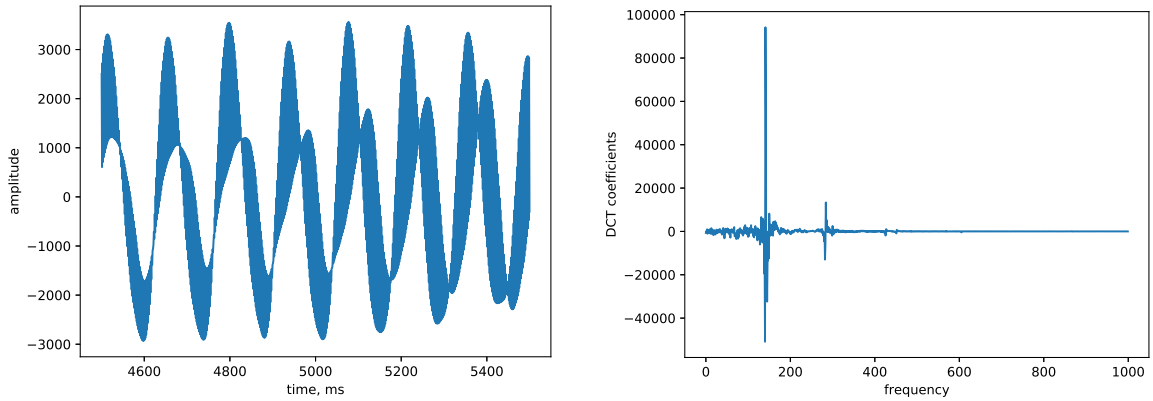
Table 2.1: Weak sparsity priors represented as the scale mixture of Gaussians

Laplace and Student's t priors are very popular in the literature and different approaches to inference have been proposed: Gibbs sampling (Hans 2009; Park and Casella 2008), expectation propagation (Seeger 2008), variational inference (Armagan 2009), double-loop algorithm (Seeger and Nickisch 2011) and expectation maximisation (Tipping and Faul 2003).

The Horseshoe density has the infinite spike at zero and heavy tails (Carvalho et al. 2010). This leads to better sparse recovery (Bhattacharya et al. 2015; Polson and Scott 2010).

2.2 Compressive Sensing

Signal acquisition and compression are important areas in signal processing. The compressed information can be used to reconstruct the original signal and for its analysis. The *compressive sensing* framework integrates the acquisition and compression steps together. This allows



(a) Part of the signal

(b) First coefficients of DCT transform

Figure 2.3: Example signal and its DCT transform. It can be noted that after this transform, the resulting coefficients are sparse, as there are only few dominating frequencies.

to reduce the number of measurements required for the ideal reconstruction of the signal. Compression is achieved utilising sparse representations of the signals in preselected basis.

2.2.1 Signal Representation

In signal processing signals are usually represented in bases. If the vectors $\{\boldsymbol{\psi}_d\}_{d=1}^{D}$ form an orthonormal basis of $\mathbb{R}^D$ then any signal $\boldsymbol{\theta}^{(n)} \in \mathbb{R}^D$ from the set of signals $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(1)} \dots \boldsymbol{\theta}^{(N)}\}$ can be represented in the form

$$\boldsymbol{\theta}^{(n)} = \sum_{d=1}^{D} \beta_d^{(n)} \boldsymbol{\psi}_d, \tag{2.20}$$

where basis coefficients $\beta_d^{(n)} = \langle \boldsymbol{\theta}^{(n)}, \boldsymbol{\psi}_d \rangle = \sum_{l=1}^{D} \theta_l^{(n)} \psi_{ld}$ are projections of the signal onto the basis vectors. Consider matrix $\boldsymbol{\Psi} := [\boldsymbol{\psi}_1 \dots \boldsymbol{\psi}_D]$. The representation of the signal can be written as

$$\boldsymbol{\theta}^{(n)} = \boldsymbol{\Psi} \boldsymbol{\beta}^{(n)}. \tag{2.21}$$

Often, signals can be represented in the Fourier-related basis, that is a sum of sinusoidal functions with scaled amplitudes. It can be achieved with, for example, a discrete cosine transform (DCT). Figure 2.3 demonstrates the transform of the sample signal with DCT.

For signals with discontinuities, Fourier coefficients become oscillating and, therefore, $\boldsymbol{\beta}^{(n)}$ is dense. This is called the *Gibbs phenomenon* (Mallat 2008). Wavelets (Daubechies 1992;

Mallat 2008) are another important example of bases that better approximate non-smooth or localised signals. They usually achieve sparse representation in image compression.

2.2.2 Transform Coding

The information about sparse representation is used for compressing signals. One of the examples of the algorithms for compression is called *transform coding* (Baraniuk, Cevher, et al. 2010). It is used in MPEG and JPEG formats for media compression. The algorithm consists of the following steps:

1. acquire the full signal $\boldsymbol{\theta}^{(n)}$;
2. compute set of basis coefficients $\boldsymbol{\beta}^{(n)} = \boldsymbol{\Psi}^{-1}\boldsymbol{\theta}^{(n)}$;
3. locate S largest coefficients and discard others, where S is the sparsity of the signal;
4. encode locations and values of the largest coefficients.

Then the original signal can be reconstructed from this information.

2.2.3 Compressive Sensing

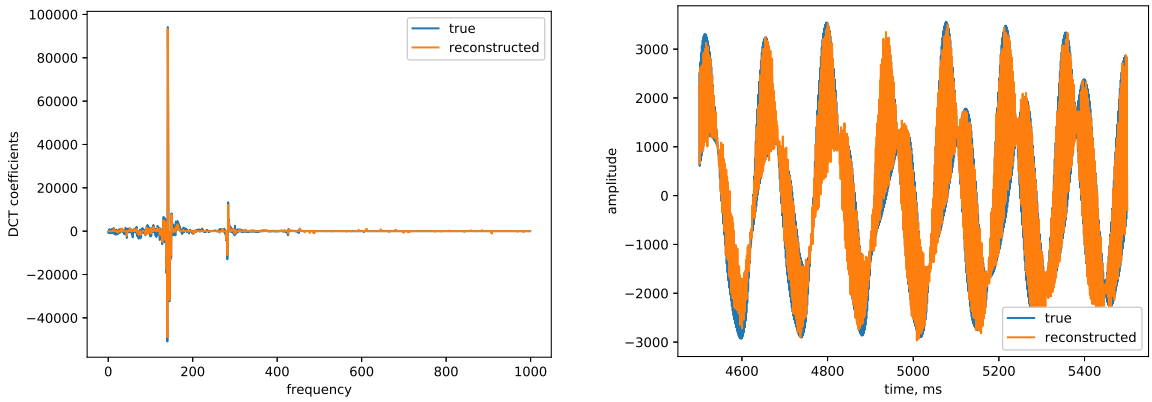
Compressive sensing (Candes, Romberg, et al. 2006; Donoho 2006) integrates the acquisition and compression steps and allows to reconstruct the original signal from less measurements and without the information of coefficient locations compared to transform coding. This includes two components: random projections and information that the signal is sparse in some basis.

Assume that the signal $\boldsymbol{\theta}^{(n)}$ is not observed directly and only the random projections $\mathbf{y}^{(n)}$ of the signal are aquired

$$\mathbf{y}^{(n)} = \mathbf{A}\boldsymbol{\theta}^{(n)}. \tag{2.22}$$

Here matrix $\mathbf{A}$ is the random projections matrix that produces a linear transformation of the signal, which is less in dimensionality that the original signal. The signal $\boldsymbol{\theta}^{(n)}$ can't be restored directly from the measurements, therefore the additional assumption of sparsity is used (2.21). The resulting problem is

$$\mathbf{y}^{(n)} = \mathbf{A}\boldsymbol{\Psi}\boldsymbol{\beta}^{(n)}, \tag{2.23}$$



(a) DCT coefficients

(b) Signal

Figure 2.4: Reconstruction after compressive sensing. Matrix $\mathbf{A}$ is used to generate random projections with size 10% of the signal. Matrix $\mathbf{\Psi}$ is the inverse DCT operator.

where $\boldsymbol{\beta}^{(n)}$ is a sparse vector. This prior information is used to regularise the problem and find a unique solution $\boldsymbol{\beta}^{(n)}$ and, therefore, restore the signal $\boldsymbol{\theta}^{(n)}$. The reconstruction results for the previously considered sample signal are presented in Figure 2.4.

Usually, the components of the matrix $\mathbf{A}$ are sampled from the independent Gaussian distributions, but in general it can be any matrix, which is close to orthonormal[1].

2.3 Gaussian Processes

Gaussian process (GP) is one of the Bayesian approaches for placing a prior distribution over the space of functions. This is a broad area that has different applications: latent variable models (Damianou et al. 2016) that are used for non-linear dimensionality reduction, Bayesian optimization (Brochu et al. 2010) that is used for black-box optimisation of unknown functions, and spatio-temporal modelling (Sarkka et al. 2013).

2.3.1 Definition

In probability theory, random variables are sometimes represented in collections. A collection of random variables indexed by set $\mathcal{T}$ is called stochastic process $\xi(\mathbf{t})$, $\mathbf{t} \in \mathcal{T}$. Index sets can

[1]The measure of how close the matrix is to orthonormal is derived with restricted isometry property (Candes and Tao 2005).

represent time, space, or more general concepts.

Gaussian process $f(\mathbf{t})$ is a stochastic process, such that for every finite subset of indices $\mathbf{T} = \{\mathbf{t}^{(1)}, \ldots, \mathbf{t}^{(N)}\} \in \mathcal{T}$ values $\mathbf{f} = \{f(\mathbf{t}^{(1)}), \ldots, f(\mathbf{t}^{(N)})\}$ have a multivariate Gaussian distribution with a mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{2.24}$$

The mean and covariance for subsets of indices are generalised into the mean and covariance functions of the GP

- *Mean function*

$$m(\mathbf{t}) = \mathbb{E}f(\mathbf{t}). \tag{2.25}$$

- *Covariance function*

$$k(\mathbf{t}^{(n')}, \mathbf{t}^{(n'')}) = \mathrm{cov}(f(\mathbf{t}^{(n')}), f(\mathbf{t}^{(n'')})). \tag{2.26}$$

With the mean and covariance functions, the mean and covariance matrix for the subset of indices are

$$\boldsymbol{\mu} = \begin{bmatrix} m(\mathbf{t}^{(1)}) \\ \ldots \\ m(\mathbf{t}^{(N)}) \end{bmatrix}, \ \boldsymbol{\Sigma} = \begin{bmatrix} k(\mathbf{t}^{(1)}, \mathbf{t}^{(1)}) & \ldots & k(\mathbf{t}^{(1)}, \mathbf{t}^{(N)}) \\ \ldots & \ldots & \ldots \\ k(\mathbf{t}^{(N)}, \mathbf{t}^{(1)}) & \ldots & k(\mathbf{t}^{(N)}, \mathbf{t}^{(N)}) \end{bmatrix}. \tag{2.27}$$

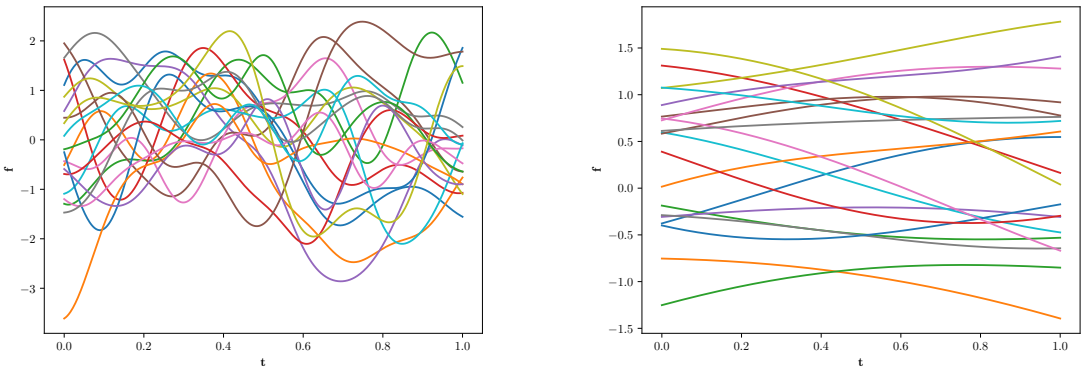
Mean and covariance functions completely define a GP. Different covariance function families characterise smoothness and stationarity of a GP.

An example of an infinitely smooth covariance function is the squared exponential function

$$k(\mathbf{t}^{(n')}, \mathbf{t}^{(n'')}) = \sigma^2 \exp\left\{ \sum_k \frac{(t_k^{(n')} - t_k^{(n'')})^2}{l_k} \right\}, \tag{2.28}$$

where σ^2 is the variance parameter, $\mathbf{l}$ is the vector of lengthscale parameters. The example of samples from a GP with the squared exponential function is demonstrated in Figure 2.5[2]

[2]Examples of GPs in this section are created with the GPy (2012) software



(a) lengthscale 0.1

(b) lengthscale 1

Figure 2.5: Samples from the GP with the squared exponential covariance function. This covariance function provides very smooth samples with different length-scales.

2.3.2 Regression

One of the basic machine learning problems solved with GPs is regression: predict unknown function values at the test points $\mathbf{t}^*$ based on known function values at the training data points $\{\mathbf{t}^{(n)}, y^{(n)}\}_{n=1}^N$. Usually observations are corrupted with noise

$$y^{(n)} = f(\mathbf{t}^{(n)}) + \varepsilon^{(n)},\ \varepsilon^{(n)} \sim \mathcal{N}(0, \sigma^2). \tag{2.29}$$

Denote $\mathbf{y} = [y^{(1)}, \ldots, y^{(N)}]$. In case of the Gaussian noise, $p(f(\mathbf{t}^*)|\mathbf{y})$ is a conditional Gaussian distribution and it is possible to analytically compute predictions

$$p\left(f(\mathbf{t}^*)|\mathbf{y}\right) = \int p\left(f(\mathbf{t}^*)|\mathbf{f}\right) p\left(\mathbf{f}|\mathbf{y}\right) d\mathbf{f}. \tag{2.30}$$

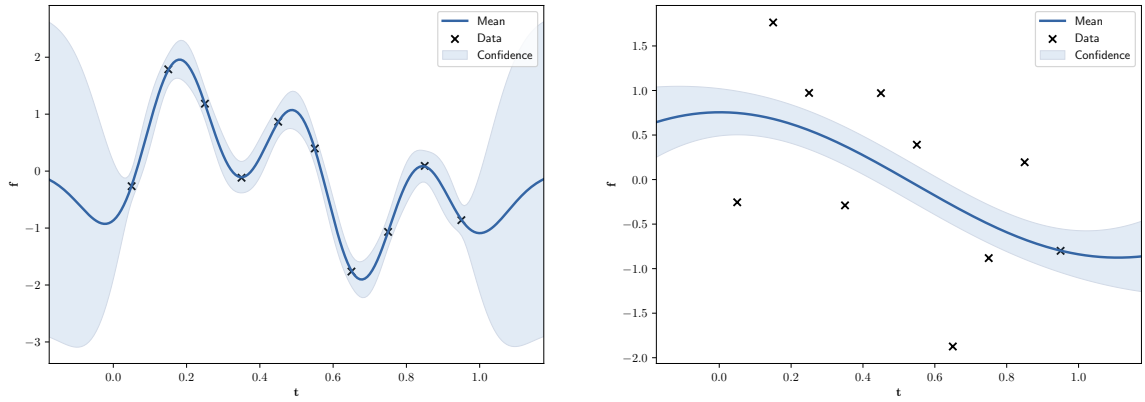
In this equation, all distributions are Gaussian, therefore predictions are also Gaussian

$$f(\mathbf{t}^*|\mathbf{y}) \sim \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*). \tag{2.31}$$

The parameters of this distribution are computed based on the properties of Gaussian distributions. Denote $\mathbf{T} = [\mathbf{t}^{(1)}, \ldots, \mathbf{t}^{(N)}]$, then

$$\boldsymbol{\mu}^* = k(\mathbf{t}^*, \mathbf{T})[k(\mathbf{T}, \mathbf{T}) + \sigma^2\mathbf{I}]^{-1}\mathbf{y}, \tag{2.32a}$$

$$\boldsymbol{\Sigma}^* = k(\mathbf{t}^*, \mathbf{t}^*) - k(\mathbf{t}^*, \mathbf{T})[k(\mathbf{T}, \mathbf{T}) + \sigma^2\mathbf{I}]^{-1}k(\mathbf{T}, \mathbf{t}^*). \tag{2.32b}$$



(a) lengthscale 0.1

(b) lengthscale 1

Figure 2.6: Predictions of Gaussian process regression. For every point, its predicted mean and variance can be computed.

Parameters of the covariance function can be optimised by maximising the likelihood of the observations $p(\mathbf{y}|\mathbf{t}^{(1)}, \ldots, \mathbf{t}^{(N)})$, however, this optimisation may converge to local extrema. An example of GP regression is provided in Figure 2.6.

2.3.3 Classification

Another basic machine learning problem is two-class classification: map test data into one of two classes $y \in \{0, +1\}$, based on known classes for train data $\{\mathbf{t}^{(n)}, y^{(n)}\}_{n=1}^{N}$. The relationship between observations and class labels is non-linear, it can be represented with, for example, the logistic function

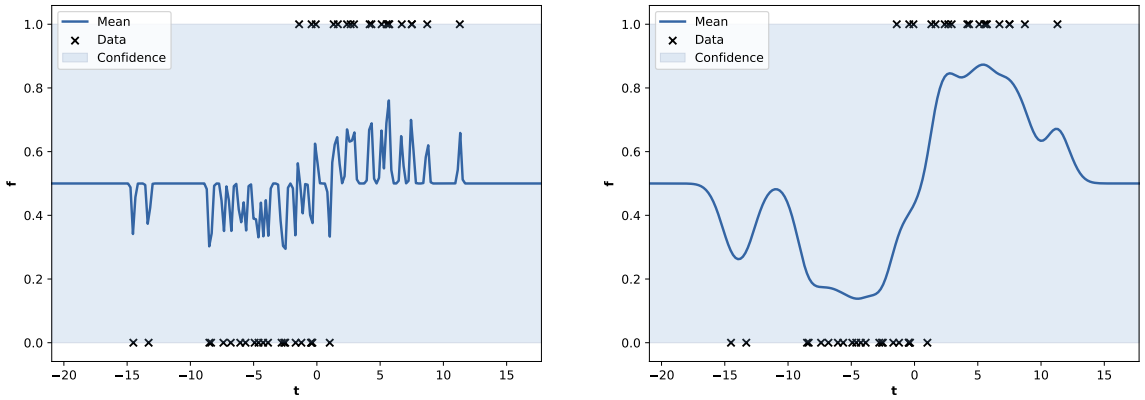
$$y^{(n)} = \frac{1}{1 + \exp(-f(\mathbf{t}^{(n)}) + \varepsilon^{(n)})}, \tag{2.33}$$

that outputs values in the interval $(0, 1)$.

The likelihood function is not Gaussian, and this integral cannot be analytically computed

$$p\left(f(\mathbf{t}^*)|\mathbf{y}\right) = \int p\left(f(\mathbf{t}^*)|\mathbf{f}\right) p\left(\mathbf{f}|\mathbf{y}\right) d\mathbf{f}. \tag{2.34}$$

The same problem holds for all types of non-linear functions that can be used for classification. Different approaches were proposed for this problem, such as numerical integration,



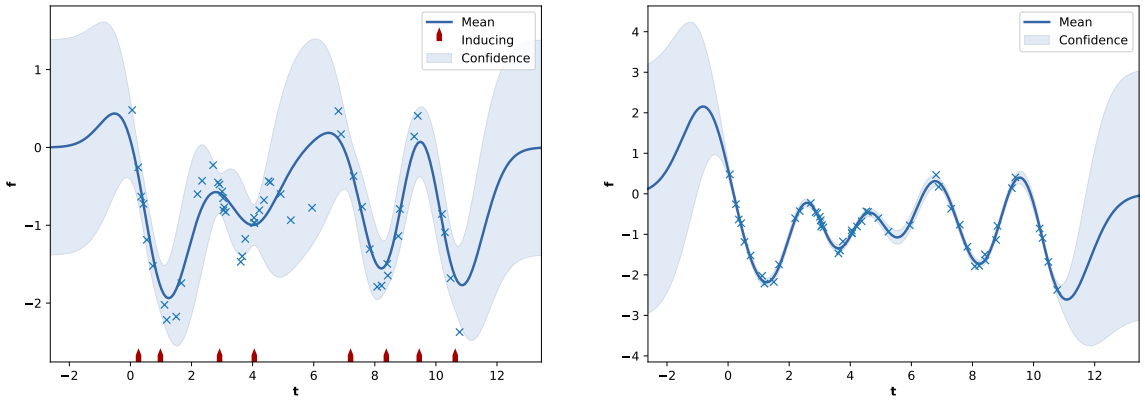
(a) lengthscale 0.1

(b) lengthscale 1

Figure 2.7: GP classification. The class label $\{0, 1\}$ is predicted based on the distribution of the latent function at the corresponding point.

sampling methods (Markov chain Monte Carlo), approximate Bayesian inference (expectation propagation, variational inference, Laplace approximation).

An example of GP classification is presented in Figure 2.7.



(a) Sparse predictions with randomly selected inducing points

(b) Full GP prediction

Figure 2.8: Inducing points for Gaussian process regression. A subset of training data was used for predictions. The locations of inducing points can be optimised for better predictions.

2.3.4 Scalability

All operations with GPs require inverting a matrix of size $N \times N$, where N is the number of training data points. This has $\mathcal{O}(N^3)$ computational complexity and requires $\mathcal{O}(N^2)$ storage, which limits the application of GPs to large datasets.

Currently, there is an ongoing work to reduce the computational and memory complexities, which is mostly based on the idea of sparse approximations: choose a small active subset of data points, called inducing points, that give similar prediction results to the whole training dataset.

An example of inducing points for GPs is presented in Figure 2.8

2.4 Summary

This chapter provides on overview of relevant sparse methods. First, the sparse linear regression problem is introduced with the frequentist and Bayesian approaches. Then, the overview of sparse representation in signal processing is described. Finally, sparsity in Gaussian processes is presented.

Chapter 3

COMPRESSIVE BACKGROUND SUBTRACTION

Sparse models are actively applied for image and video processing (Mairal, Bach, and Ponce 2014). One of the essential problems in video processing is background subtraction, that is detection of changes in sequential video frames. This is important for object localisation and classification, which can be used, for example, for gesture recognition or traffic monitoring. Sparsity is natural for the background subtraction problem, as the foreground objects occupy the small regions on a frame. Background subtraction hence represents a natural application area for sparse modelling. The idea to apply compressive sensing for background subtraction is originally proposed by Cevher et al. (2008), where the sparse regression problem is formulated as the optimisation problem with l_1-optimisation.

The Bayesian approach for compressive sensing (Ji, Xue, et al. 2008) provides two desirable properties for the solution: first, it naturally provides the uncertainty estimation of the predictions from the posterior distribution; second, it allows to use adaptive approach for design matrix selection, thus improving efficiency of compression. In this chapter the sparse Bayesian models are considered for compressed background subtraction. As it is shown in the experiments section, they also achieve better computational time.

The chapter is organised as following. In Section 3.1 the sparse model of background subtraction is explained. The Bayesian compressive sensing approaches for this problem is presented in Section 3.2. The experimental results are presented in Section 3.3. Section 3.4 summarises the chapter.

The materials of this chapter were published as

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2015). "Compressive sensing approaches for autonomous object detection in video sequences". In: *Proceedings of the*

(a) Background frame (b) Frame with object (c) Foreground mask

Figure 3.1: Example of background subtraction problem: extract foreground car silhouette from the image. For static camera foreground is sparse, as it occupies only small part of the image.

Sensor Data Fusion: Trends, Solutions, Applications Workshop (SDF). IEEE, pp. 1–6. DOI: 10.1109/SDF.2015.7347706

3.1 Background Subtraction

In a typical background subtraction application the data consists of the sequential frames $\mathbf{V}^{(n)} \in \mathbb{R}^{D_1 \times D_2}$, $n \in \{1, \ldots, N\}$ from the camera. Assume that the camera is static and it is possible to acquire a frame $\mathbf{B} \in \mathbb{R}^{D_1 \times D_2}$ from the camera that is referenced as the background. The problem is to estimate the mask of the foreground objects in the camera frames. The example of camera frames is presented in Figure 3.1.

To preprocess the video, the camera frames are converted to greyscale and flattened: the resulting background frame is vector $\mathbf{b} \in \mathbb{R}^D$, the video frames are vectors $\mathbf{v}^{(n)} \in \mathbb{R}^D$, where $D = D_1 D_2$.

Usually the foreground objects take only a part of the image, therefore the majority of the foreground mask $\boldsymbol{\beta}^{(n)} = \mathbf{v}^{(n)} - \mathbf{b}$ values are close to zero. This leads to the application of sparse regression and compressive sensing theory to this problem. They reduce the number of measurements that need to be taken (Candès and Wakin 2008) and also the results may be denoised (Mairal, Bach, and Ponce 2014). The values of the foreground mask are estimated

based on the set of the compressed measurements $\mathbf{y}^{(n)} \in \mathbb{R}^K$

$$\mathbf{y}^{(n)} = \mathbf{X}\boldsymbol{\beta}^{(n)}, \tag{3.1}$$

where the design matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$ consists of i.i.d. Gaussian variables, according to Baraniuk, Davenport, et al. (2008).

Since $\boldsymbol{\beta}^{(n)} = \mathbf{v}^{(n)} - \mathbf{b}$, the estimates of the coefficients $\mathbf{y}^{(n)}$ can be done on the acquisition step as

$$\mathbf{y}^{(n)} = \mathbf{X}\mathbf{v}^{(n)} - \mathbf{X}\mathbf{b}. \tag{3.2}$$

The vectors $\mathbf{X}\mathbf{b}$ and $\mathbf{X}\mathbf{v}^{(n)}$ are the linear combinations of the pixels of the video frames, and a single pixel camera (Duarte et al. 2008) may be used for simultaneous capturing and compression of the video.

In this chapter the Bayesian weak sparsity models for sparse regression are used for the background subtraction problem and their performance is compared with OMP (Section 2.1.1).

3.2 Bayesian Compressive Sensing

Model

In Bayesian compressive sensing (BCS), the system (3.1) is reformulated as a linear regression model (Ji, Xue, et al. 2008)

$$\mathbf{y}^{(n)} = \mathbf{X}\boldsymbol{\beta}^{(n)} + \boldsymbol{\varepsilon}^{(n)}, \tag{3.3}$$

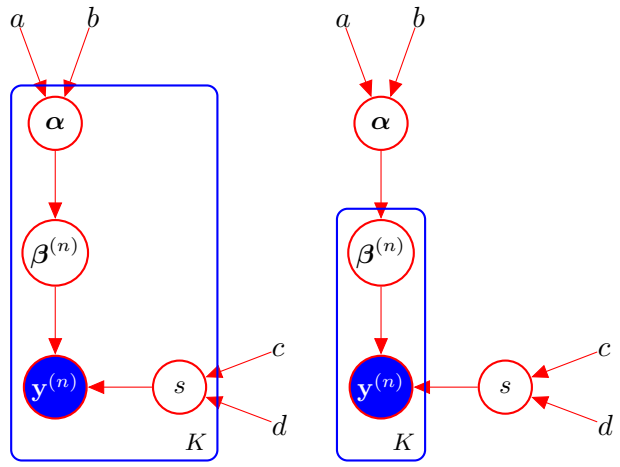
where $\boldsymbol{\varepsilon}^{(n)}$ is a vector which elements are the independent noise from the Gaussian distribution $\varepsilon_d^{(n)} \sim \mathcal{N}(0, \sigma^2)$ with the variance σ^2. Therefore, the likelihood can be expressed as

$$p(\mathbf{y}^{(n)}|\boldsymbol{\beta}^{(n)}, \sigma^2) = \prod_{k=1}^{K} \mathcal{N}(y_k^{(n)}; \mathbf{x}_{k,:}\boldsymbol{\beta}^{(n)}, \sigma^2\mathbf{I}), \tag{3.4}$$

where $\mathbf{x}_{k,:}$ is the k-th row of the matrix $\mathbf{X}$.

To implement the full Bayesian approach, the prior distributions are imposed on all parameters

$$p(\boldsymbol{\beta}^{(n)}|\boldsymbol{\alpha}) = \prod_{d=1}^{D} \mathcal{N}(\beta_d^{(n)}; 0, \alpha_d^{-1}), \tag{3.5}$$



(a) Bayesian compressive sensing (b) Multitask Bayesian compressive sensing

Figure 3.2: Graphical models for Bayesian compressive sensing. Multitask model shares the hyperparameters for several signals of similar structure, thus reducing required number of measurements.

where $\boldsymbol{\alpha}$ is a prior parameter vector;

$$p(\boldsymbol{\alpha}) = \prod_{d=1}^{D} \Gamma(\alpha_d; a, b), \tag{3.6}$$

$$p(\sigma^2) = \mathrm{IG}(\sigma^2; c, d). \tag{3.7}$$

The graphical model is displayed in Figure 3.2a.

According to the Bayes rule the posterior distribution can be written as follows

$$p(\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{y}^{(n)}) = \frac{p(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{y}^{(n)})}, \tag{3.8}$$

where $p(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2)$ is the likelihood term, $p(\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2)$ is the prior term, $p(\mathbf{y}^{(n)})$ is the evidence term. The latter can be expressed as

$$p(\mathbf{y}^{(n)}) = \int\limits_{\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2} p(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2)\, p(\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2)\, d\boldsymbol{\beta}^{(n)}\, d\boldsymbol{\alpha}\, d\sigma^2. \tag{3.9}$$

This integral is intractable, therefore it should be approximated.

Inference

In Bayesian compressive sensing (Ji, Xue, et al. 2008), the decomposition of the posterior probability into the product of the tractable and intractable probabilities is used and the intractable part is approximated with the delta-function in its mode

$$p(\boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{y}^{(n)}) = p(\boldsymbol{\beta}^{(n)} | \mathbf{y}^{(n)}, \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{y}^{(n)}). \tag{3.10}$$

The Bayes rule for the first term of (3.10) is

$$p(\boldsymbol{\beta}^{(n)} | \mathbf{y}^{(n)}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}, \sigma^2) p(\boldsymbol{\beta}^{(n)} | \boldsymbol{\alpha})}{p(\mathbf{y}^{(n)} | \boldsymbol{\alpha}, \sigma^2)}. \tag{3.11}$$

These are all the Gaussians, so the probability $p(\boldsymbol{\beta}^{(n)} | \boldsymbol{\alpha}, \sigma^2, \mathbf{y}^{(n)})$ can be calculated based on the properties of Gaussians. It is the Gaussian distribution $\mathcal{N}(\boldsymbol{\beta}^{(n)}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with parameters

$$\boldsymbol{\Sigma} = (\sigma^{-2} \mathbf{X}^\top \mathbf{X} + \mathbf{A})^{-1}, \tag{3.12}$$

$$\boldsymbol{\mu} = \sigma^{-2} \boldsymbol{\Sigma} \mathbf{X}^\top \mathbf{y}^{(n)}, \tag{3.13}$$

where $\mathbf{A} = \mathrm{diag}(\alpha_1, \ldots, \alpha_D)$.

The second term of the posterior probability (3.10) can be expressed as

$$p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{y}^{(n)}) = \frac{p(\mathbf{y}^{(n)} | \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}) p(\sigma^2)}{p(\mathbf{y}^{(n)})}. \tag{3.14}$$

The denominator here is not tractable. The most probable values of $\boldsymbol{\alpha}, \sigma^2$ are used. To achieve this, the term $p(\mathbf{y}^{(n)} | \boldsymbol{\alpha}, \sigma^2)$ needs to be maximised

$$p(\mathbf{y}^{(n)} | \boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}, \sigma^2) p(\boldsymbol{\beta}^{(n)} | \boldsymbol{\alpha}) d\boldsymbol{\beta}^{(n)}. \tag{3.15}$$

Maximisation of (3.15) w.r.t. $\boldsymbol{\alpha}$ and σ^2 gives the following iterative process

$$\alpha_d^{\mathrm{new}} = \frac{\gamma_d}{\mu_d^2}, \tag{3.16}$$

$$(\sigma^2)^{\mathrm{new}} = \frac{\|\mathbf{y}^{(n)} - \mathbf{X}\boldsymbol{\mu}\|_2^2}{\sigma^{-2} - \Sigma_{dd}\gamma_d}, \tag{3.17}$$

where $\gamma_d = 1 - \alpha_d \Sigma_{dd}$, Σ_{dd} is the diagonal element of the matrix $\boldsymbol{\Sigma}$ (3.12), $\boldsymbol{\mu}$ is the mean vector (3.13). Then the steps (3.16) and (3.17) iterate with the steps (3.12) and (3.13) until convergence.

Note that the marginal distribution on $\boldsymbol{\beta}$ is

$$p(\beta_d^{(n)}) = \frac{b^a \Gamma\left(a + \frac{1}{2}\right)}{(2\pi)^{\frac{1}{2}} \Gamma(a)} \left(b + \frac{(\beta_d^{(n)})^2}{2} \right)^{-\left(a+\frac{1}{2}\right)}. \tag{3.18}$$

This is the Student's t-distribution, that has the most probable area concentrated around zero. Thereby, it leads to the sparse vector $\boldsymbol{\beta}^{(n)}$.

3.2.1 Multitask Bayesian Compressive Sensing (MTCS)

In Ji, Dunson, et al. (2009) the Bayesian method to process several signals that have a similar sparse structure is proposed. The multitask setting reduces the number of measurements that should be taken comparing to processing all the signals independently. The hyperparameter $\boldsymbol{\alpha}$ is considered to be shared by all the tasks. The graphical model is displayed in Figure 3.2b.

3.2.2 Design matrix selection

The uncertainty estimation, that is achieved with the Bayesian approach allows to adaptively modify the matrix $\mathbf{X}$ with the goal of reducing uncertainty of $\boldsymbol{\beta}^{(n)}$. Such approach is usually called active learning. The common approach for the adaptive design is the minimisation of the entropy of target variables (Settles 2009). It is shown by Ji, Xue, et al. (2008), that the minimisation of the differential entropy of $\boldsymbol{\beta}^{(n)}$ can be achieved by choosing the rows of $\mathbf{X}$ such that they maximise the variance of the expected measurement $\mathbf{y}^{(n+1)}$.

3.2.3 Complexity

At every iteration the most computationally intensive step is (3.12), that involves matrix inversion. It's complexity is $\mathcal{O}(D^3)$. For OMP, computational complexity is $\mathcal{O}(KD)$ (Tropp and Gilbert 2007).

Though the complexity is high for all methods, compressive background subtraction can be used in scenarios with limited resources. One of these scenarios is the usage of sigle pixel cameras (Takhar et al. 2006), that use a single optical sensor to sample and compress in one measurement process. Another scenario is embedded systems where compression is performed on the device with limited resoures and random projections, and reconstruction can be achieved relatively cheap on powerful hardware.

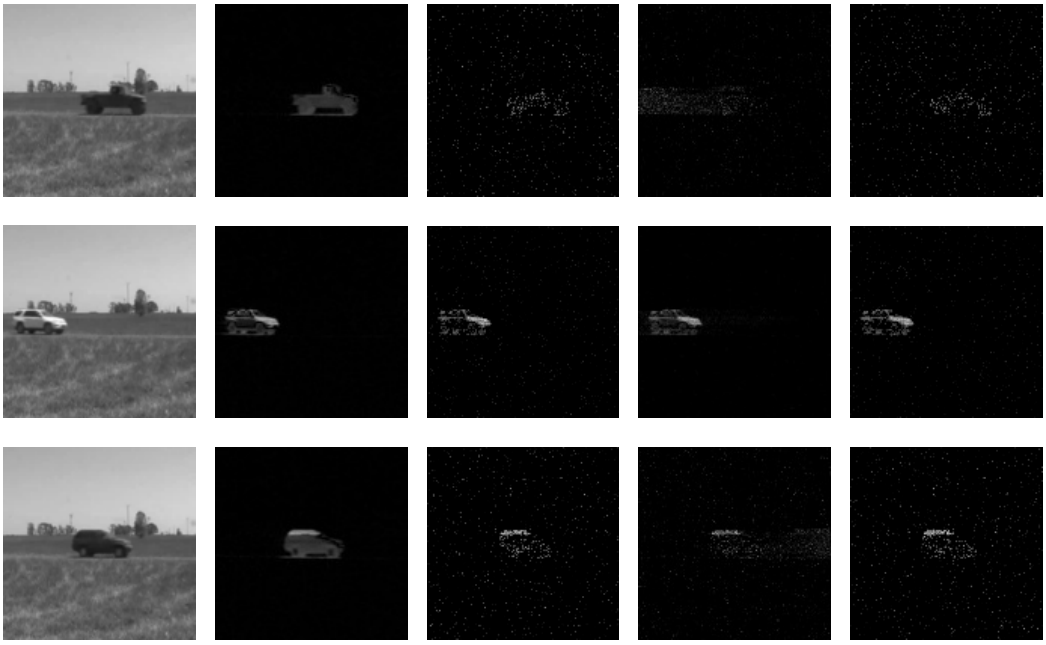


Figure 3.3: Comparison of foreground reconstruction based on 2000 measurements by the algorithms. The three rows correspond to the three sample frames. From left to right columns: the input uncompressed frame, uncompressed background subtraction, compressed background subtraction with Bayesian compressive sensing, compressed background subtraction with multi-task Bayesian compressive sensing, compressed background subtraction with orthogonal matching pursuit

3.3 Experiments

For the background subtraction problem the Convoy dataset (Warnell et al. 2015) is used, which consists of 260 greyscale frames and the background frame. The frames are scaled to the less resolution of 128×128 to avoid memory problems. For the multitask algorithm the batches of 40 frames are run together, while for the Bayesian compressive sensing and OMP algorithms all the frames are processed independently. There are two sets of the experiments: one with $K = 2000$ measurements and the other with $K = 5000$ measurements. For both sets of the experiments all three methods are run for 10 times with 10 different design matrices $\mathbf{X}$ shared among the methods. For the quantitative comparison the median values of quality measures among these runs are presented.

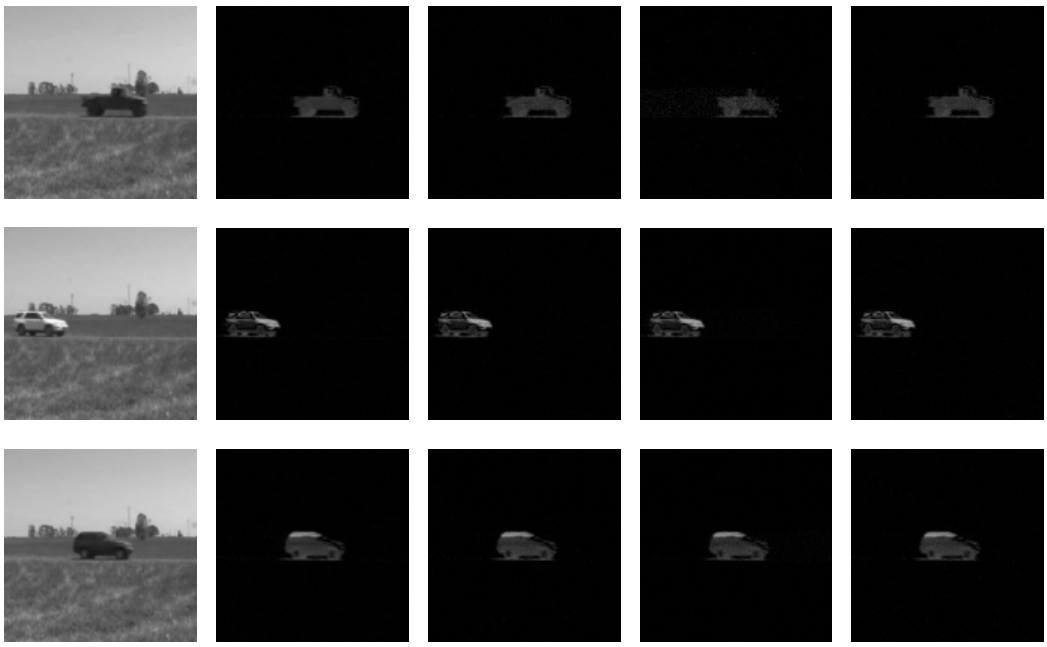


Figure 3.4: Comparison of foreground reconstruction based on 5000 measurements by the algorithms. The three rows correspond to the three sample frames. From left to right columns: the input uncompressed frame, uncompressed background subtraction, compressed background subtraction with Bayesian compressive sensing, compressed background subtraction with multi-task Bayesian compressive sensing, compressed background subtraction with orthogonal matching pursuit

The qualitative comparison of the models with the same design matrix $\mathbf{X}$ is displayed in Figures 3.3 - 3.4. The three demonstrative frames are presented. One can notice that with the same design matrix the models demonstrate similar results. The figures show that 2000 measurements can be used for object region detection, while 5000 measurements which is only about 30% of the input resolution are enough even to distinguish parts of the objects like doors and windows of the cars.

For the quantitative comparison of the results the following measures are used:

Reconstruction error. $\frac{\|\boldsymbol{\beta}^{(n)} - \widehat{\boldsymbol{\beta}}^{(n)}\|_2}{\|\boldsymbol{\beta}^{(n)}\|_2}$, where $\boldsymbol{\beta}^{(n)}$ is the signal ground truth, $\widehat{\boldsymbol{\beta}}^{(n)}$ is the signal, reconstructed by the algorithm;

Background subtraction quality measure (BS quality). $\frac{|S(\boldsymbol{\beta}^{(n)}) \cap S(\widehat{\boldsymbol{\beta}}^{(n)})|}{|S(\boldsymbol{\beta}^{(n)}) \cup S(\widehat{\boldsymbol{\beta}}^{(n)})|}$, where $S(\boldsymbol{\beta}^{(n)})$ is the ground truth foreground pixels, $S(\widehat{\boldsymbol{\beta}}^{(n)})$ is the algorithm detected foreground pixels, $|\cdot|$ is the cardinality of the set;

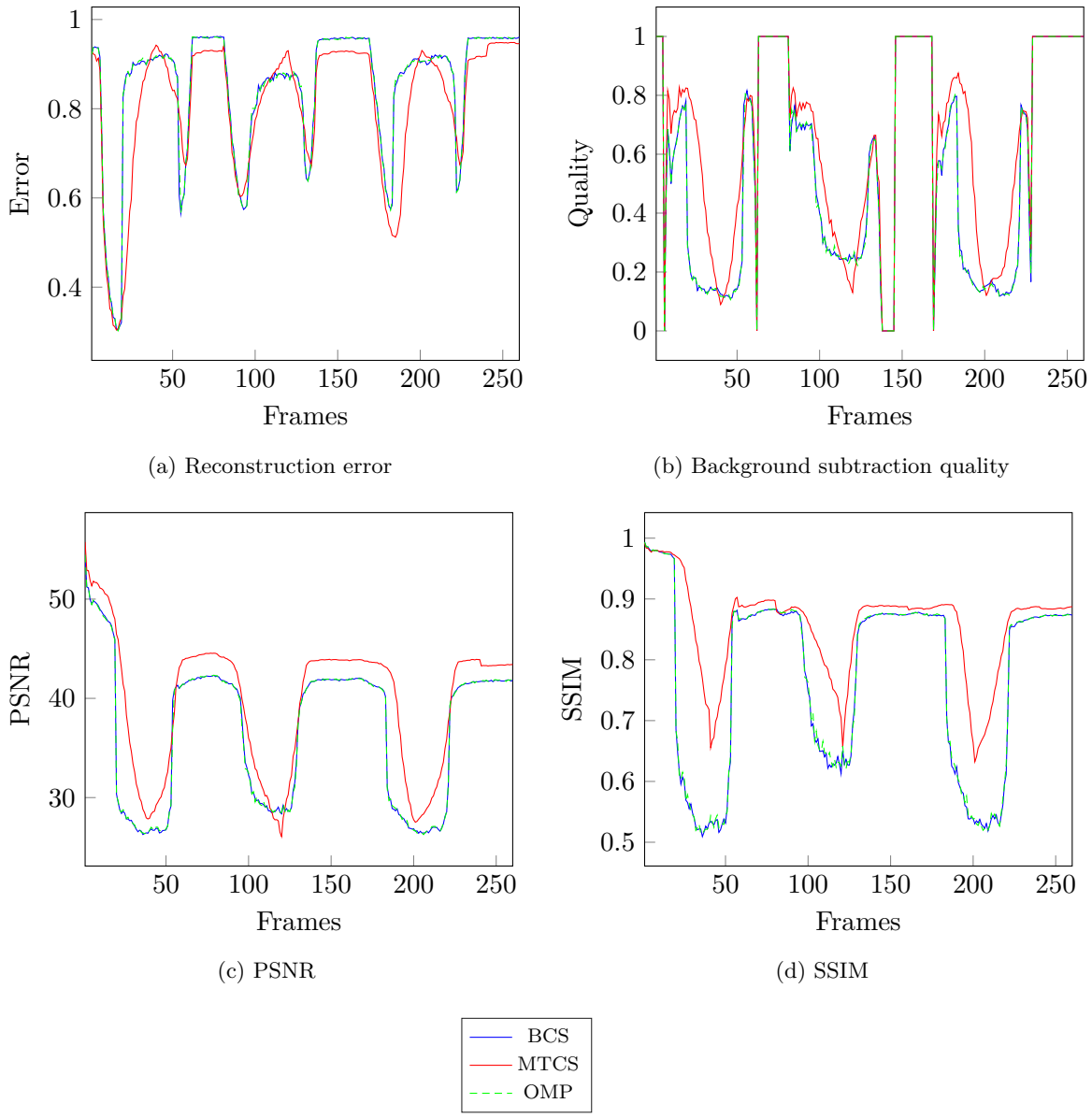
Peak signal-to-noise ratio (PSNR). $10\log_{10}\left(\frac{\text{peakval}^2}{\text{MSE}}\right)$, where peakval is the maximum possible pixel value, that is 255 in our case. MSE is the mean square error between $\boldsymbol{\beta}^{(n)}$ and $\widehat{\boldsymbol{\beta}}^{(n)}$;

Structural similarity index (SSIM). $\frac{(2\mu_{\boldsymbol{\beta}^{(n)}}\mu_{\widehat{\boldsymbol{\beta}}^{(n)}} + C_1)(2\sigma_{\boldsymbol{\beta}^{(n)}\widehat{\boldsymbol{\beta}}^{(n)}} + C_2)}{(\mu^2_{\boldsymbol{\beta}^{(n)}} + \mu^2_{\widehat{\boldsymbol{\beta}}^{(n)}} + C_1)(\sigma^2_{\boldsymbol{\beta}^{(n)}} + \sigma^2_{\widehat{\boldsymbol{\beta}}^{(n)}} + C_2)}$, where $\mu_{\boldsymbol{\beta}^{(n)}}$, $\mu_{\widehat{\boldsymbol{\beta}}^{(n)}}$, $\sigma_{\boldsymbol{\beta}^{(n)}}$, $\sigma_{\widehat{\boldsymbol{\beta}}^{(n)}}$, $\sigma_{\boldsymbol{\beta}^{(n)}\widehat{\boldsymbol{\beta}}^{(n)}}$ are the local means, standard deviations, and cross-covariance for the images $\boldsymbol{\beta}^{(n)}$ and $\widehat{\boldsymbol{\beta}}^{(n)}$ respectively, and C_1, C_2 are the regularisation constants.

The difference between the uncompressed current frame $\mathbf{v}^{(n)}$ and the uncompressed background frame $\mathbf{b}$ is used as the ground truth signal $\boldsymbol{\beta}^{(n)}$ for every frame (the second columns in Figures 3.3 - 3.4), since this is the signal which is compressed by (3.1).

The results are presented in Figures 3.5 - 3.6. All the quality measures – reconstruction error, BS quality, PSNR and SSIM – are calculated for every frame. The mean values among the frames for each measure and computational time can be found in Tables 3.1a – 3.1b. The computational time is provided for a batch of 40 frames (BCS and OMP process each frame independently with 4 parallel workers, multitask BCS processes all 40 frames together). Implementation is made on the laptop with i7-4702HQ CPU with 2.20GHz, 16 GB RAM using MATLAB 2015a.

Multitask Bayesian compressive sensing demonstrates the best results according to almost each measure. Bayesian compressive sensing and OMP show the competitive results but Bayesian compressive sensing works faster. It is worth to note that multitask Bayesian compressive sensing has the biggest variance among the runs with the different design matrices, while the variances of the Bayesian compressive sensing and OMP runs for the same matrices are quite small.



(a) Reconstruction error

(b) Background subtraction quality

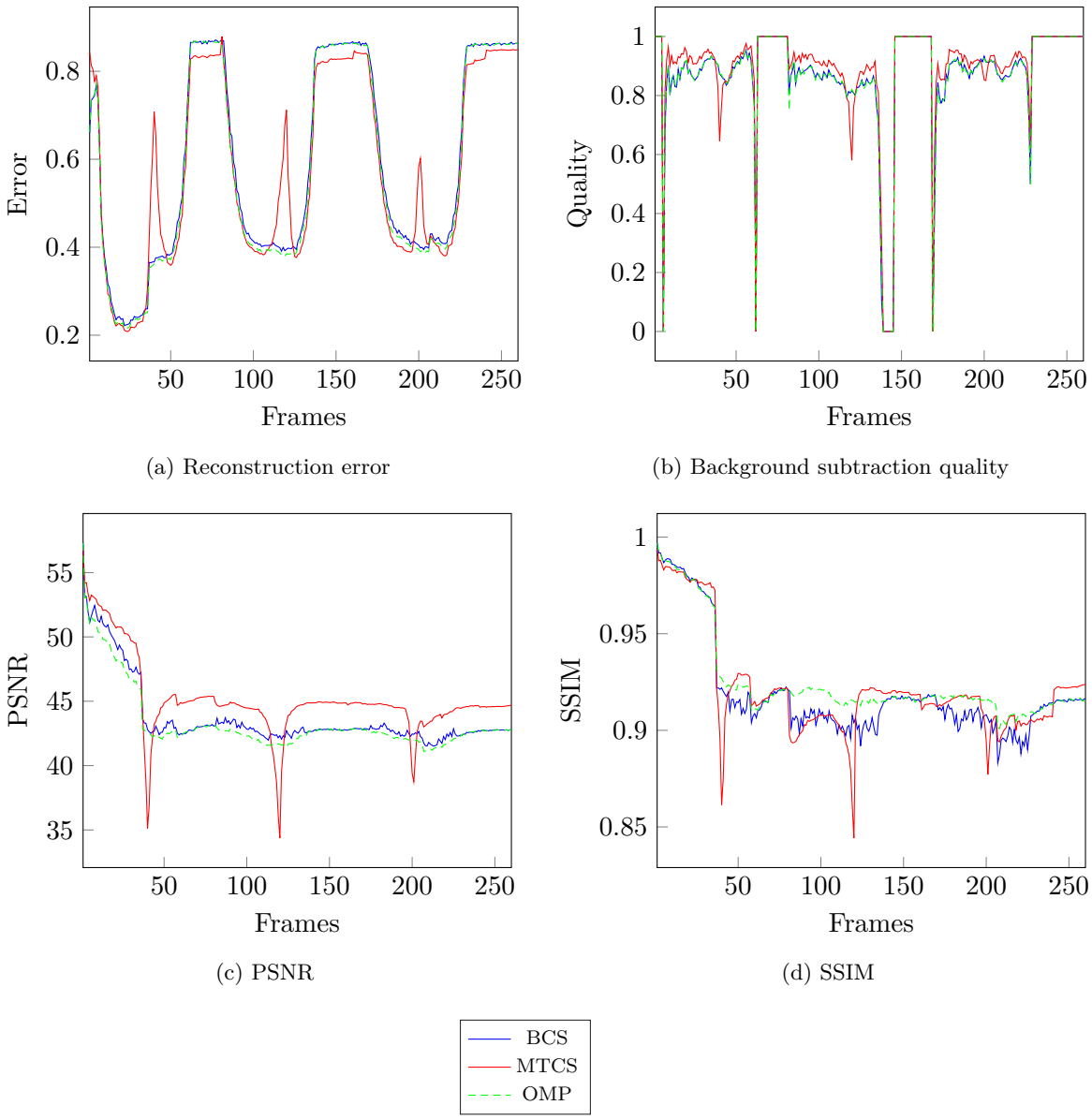
(c) PSNR

(d) SSIM

Figure 3.5: Quantitative method comparison on the frame level for the set of the experiments with 2000 measurements

3.4 Summary

This chapter presents two Bayesian compressive sensing algorithms in the application of background subtraction. These are the applications of the Bayesian compressive sensing and of the multitask Bayesian compressive sensing algorithms. The results presented in



(a) Reconstruction error

(b) Background subtraction quality

(c) PSNR

(d) SSIM

Figure 3.6: Quantitative method comparison on the frame level for the set of the experiments with 5000 measurements

Figures 3.3 – 3.4 demonstrate the satisfactory reconstruction quality of the original image based on only 5000 measurements (that is $\approx 30\%$ of the original image size).

The conventional Bayesian compressive sensing method demonstrates the similar results to the greedy algorithm OMP but BCS is more effective in terms of the computational time.

Table 3.1: Mean quality measures

(a) Method comparison based on 2000 measurements

Algorithm	Reconstruction error	BS quality	PSNR	SSIM	Time (hours)
BCS	0.8037	0.3518	34.2007	0.7198	**0.23**
Multitask BCS	**0.7608**	**0.4820**	**37.542**	**0.8384**	0.67
OMP	0.8028	0.3510	34.1705	0.7204	0.51

(b) Method comparison based on 5000 measurements

Algorithm	Reconstruction error	BS quality	PSNR	SSIM	Time (hours)
BCS	0.4713	0.8119	43.8251	0.9186	**0.9**
Multitask BCS	0.4702	**0.8421**	**45.0028**	0.9212	8.5
OMP	**0.4578**	0.8109	43.2720	**0.9266**	4.8

If the computational time is not critical the extension of the Bayesian method designed for a multitask problem can improve the performance in terms of the different measures. Therefore, other extensions of the Bayesian method to include the prior information need further research.

In this chapter the components of the foreground intensities are assumed independent. For most cases the objects are grouped into several clusters, therefore more sophisticated sparsity models can be introduced to reflect the structure of the foreground. Chapter 4 presents the hierarchical sparse Bayesian model that is capable of modelling structured data.

Chapter 4

STRUCTURED SPIKE AND SLAB MODELS

In Chapter 3 the weak sparse Bayesian models with the independent prior for sparse coefficients are considered, but often the independence assumption is not valid (Bach et al. 2012b), as non-zero elements tend to appear in groups, and thus an unknown structure of the latent variables may exist. For example, wavelet coefficients of images are usually organised in trees (Mallat 2008), chromosomes have a spatial structure along a genome (Hastie et al. 2015), video from single-pixel cameras has a temporal structure (Yang et al. 2014). In these cases, it is useful to introduce additional hierarchical or group penalties that promote such structures in recovered signals. In this chapter the structured formulation of the spike and slab model is presented, that accounts for the group structure of sparse coefficients. This is achieved with a hierarchical Gaussian process prior on the latent variables. Such hierarchical prior allows to model spatial structural dependencies for signal components that can evolve in time.

The chapter is organised as follows. Section 4.1 discussed the existing work on group sparsity and Section 4.2 provides an overview of existing spike and slab models. The proposed model is presented in Section 4.3 and Section 4.4 presents the inference algorithm for the model. Section 4.5 presents the online version of the proposed algorithm. Section 4.6 presents the numerical experiments. Section 4.7 summarises the chapter.

The materials of this chapter were published as

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2017). "Structured sparse modelling with hierarchical GP". in: *Proceedings of the 6th Signal Processing with Adaptive Sparse Structured Representations Workshop (SPARS)*. URL: `http://spars2017.lx.it.pt/index_files/papers/SPARS2017_Paper_48.pdf`

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2018a). "Spatio-temporal

structured sparse regression with hierarchical Gaussian process priors". In: *IEEE Transactions on Signal Processing* vol. 66, issue 17, pp. 4598–4611. DOI: 10.1109/TSP.2018.2858207

4.1 Group Sparsity

Different spatial structure assumptions for sparse models have been extensively studied in the literature. The group lasso (Sprechmann et al. 2011; Yuan and Lin 2006) extends the classical lasso method for group sparsity such that coefficients form groups and all coefficients in a group are either non-zero or zero together, but groups are required to be defined in advance. In contrast to group lasso, structural dependencies in the proposed model are defined by the parameters of covariance functions of the Gaussian processes and the actual groups are inferred from the data.

Group weak sparsity models include smooth relevance vector machines (Schmolck and Everson 2007), spatio-temporal coupling of the parameters for the scale mixture of Gaussians representation (Van Gerven et al. 2010; Wu, Park, et al. 2014), row and element sparsity (Chen et al. 2016), block sparsity (Zhang and Rao 2011).

For spike and slab priors a spatio-temporal structure can be modelled with a one-level Gaussian processes prior (Andersen et al. 2017), where the prior is imposed on all locations of non-zero components together. The covariance matrix is represented as the Kronecker product of the temporal and spatial matrices.

In contrast to the one-level GP, the proposed model introduces an additional level of a GP prior for temporal dependencies. Therefore, the temporal and spatial structures are decoupled. The proposed model is thus more flexible. Broadly speaking, the top-level GP can encode the slow change of groups of spikes positions in time while the low-level GP allows to model the local changes of each group. The one-level GP prior model also requires significantly more memory to store the covariance function for modelling both spatial and temporal structural dependencies as it is built as a Kronecker product of spatial and temporal covariance matrices. The resulting size of the covariance matrix scales quadratically with spatio-temporal dimensionality, which makes it infeasible even for average size problems, whereas in the proposed model the total size of two covariance matrices scales linearly.

More importantly, in the proposed model structural dependencies are considered at every

timestamp whereas in Andersen et al. (2017) the GP prior is imposed on the whole batch of data. This consideration of every timestamp is promising in terms of incremental inference — all latent variables should be inferred for the new time moment in the same manner as for the batch inference. Meanwhile it is unclear how to apply the one-level GP model for the incremental data without re-processing the previous data.

GPs are widely used to model complex structures and dynamics in data not only in sparse problems. In Deisenroth and Mohamed (2012) GP is used as a prior for nonlinear state transition and observation functions for state-space Bayesian filtering. Hierarchical GP models are proposed to model structures in Lawrence and Moore (2007).

4.2 Spike and Slab Models

This section presents a roadmap of models that are used in the formulation of the proposed spatio-temporal structured sparse model. It starts from the basic spike and slab model, as in Section 2.1.2, and continues with its extension for structured data.

The generative model for the spatio-temporal regression problem can be formulated in the following way:

- The data is collected for the sequence of the N discrete timestamps. Indexes are denoted by $n \in [1, \dots, N]$.

- At each timestamp n the unknown signal of size D is denoted by $\boldsymbol{\beta}^{(n)} = [\beta_1^{(n)}, \dots, \beta_D^{(n)}]^\top$. Signals at all timestamps are concatenated into a matrix $\mathbf{B} = [\boldsymbol{\beta}^{(1)}, \dots, \boldsymbol{\beta}^{(N)}]$.

- The observations of size K are denoted by $\mathbf{y}^{(n)} = [y_1^{(n)}, \dots, y_K^{(n)}]^\top$. They are obtained with the design matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$. Observations at all timestamps are concatenated into matrix $\mathbf{Y} = [\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(N)}]$.

- An independent Gaussian noise with the variance σ^2 is added to the observations.

The probabilistic model can be then expressed as

$$p\left(\mathbf{y}^{(n)} | \boldsymbol{\beta}^{(n)}\right) = \mathcal{N}\left(\mathbf{y}^{(n)}; \mathbf{X}\boldsymbol{\beta}^{(n)}, \sigma^2 \mathbf{I}\right) \quad \forall n. \tag{4.1}$$

It is assumed that the dimensionality K of observations $\mathbf{y}^{(n)}$ is less than the dimensionality D of signals $\boldsymbol{\beta}^{(n)}$, therefore the problem of recovery of signal $\boldsymbol{\beta}^{(n)}$ from observations $\mathbf{y}^{(n)}$ is

underdetermined and it can have an infinite number of solutions. Sparsity-inducing priors allow to specify additional constraints that lead to a unique optimal solution.

4.2.1 Factor Graphs

For Bayesian models, factor graphs are used to visualise complex distributions (Wainwright and Jordan 2008) in the form of undirected graphical models. They are also important for the approximate inference method described in Section 4.4.

The joint probability density function $p(\cdot)$ of latent variables ζ_i can be factorised as a product of factors ψ_C that are functions of a corresponding set of latent variables $\boldsymbol{\zeta}_C$

$$p\left(\zeta_1, ..., \zeta_m\right) = \frac{1}{Z} \prod_C \psi_C\left(\boldsymbol{\zeta}_C\right), \tag{4.2}$$

where Z is a normalisation constant. This factorisation can be represented as a bipartite graph with variable vertices corresponding to ζ_i, factor vertices corresponding to ψ_C and edges connecting corresponding vertices.

The distribution of latent variables $\boldsymbol{\beta}^{(n)}$ in (4.1) can be represented as a factor

$$g^{(n)}\left(\boldsymbol{\beta}^{(n)}\right) = \mathcal{N}\left(\mathbf{y}^{(n)}; \mathbf{X}\boldsymbol{\beta}^{(n)}, \sigma^2\mathbf{I}\right). \tag{4.3}$$

The factor graphs are used in this chapter to visualise different spike and slab models. In Figure 4.1a, Figure 4.2a, Figure 4.4 circles represent variable vertices and small squares represent factor vertices.

4.2.2 Spike and Slab Model

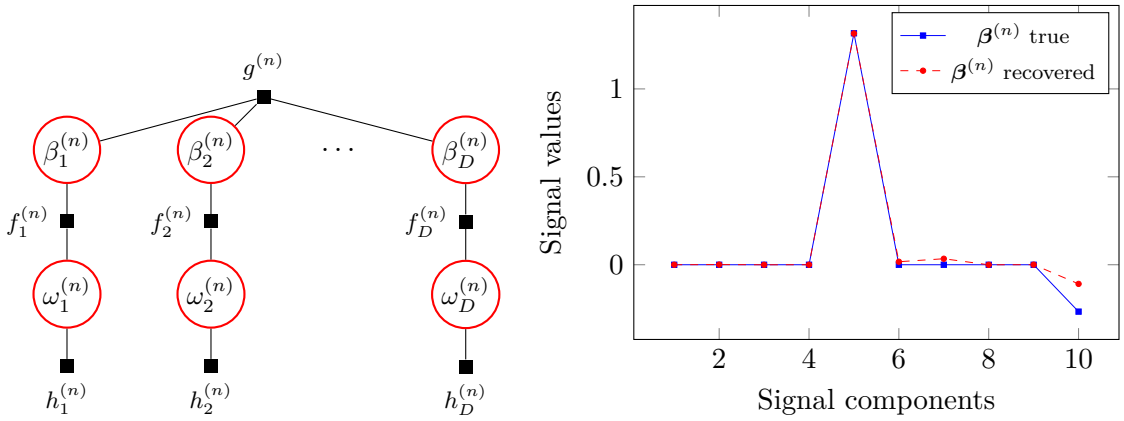
Sparsity can be induced with the spike and slab model (George and McCulloch 1993), where additional latent variables $\boldsymbol{\Omega} = \{\omega_d^{(n)}\}_{n=1:N,\, d=1:D}$ indicate if signal components $\beta_d^{(n)}$ are zeros. This is represented as a mixture of a spike and a slab

$$p\left(\beta_d^{(n)} | \omega_d^{(n)}\right) = \omega_d^{(n)} \delta_0\left(\beta_d^{(n)}\right) + \left(1 - \omega_d^{(n)}\right) \mathcal{N}\left(\beta_d^{(n)}; 0, \sigma_\beta^2\right). \tag{4.4}$$

The conditional distributions $p\left(\beta_d^{(n)} | \omega_d^{(n)}\right)$ are further denoted by factors $f_d^{(n)}\left(\omega_d^{(n)}, \beta_d^{(n)}\right)$.

In this model $\{\omega_d^{(n)}\}_{d=1:D}$ are considered conditionally independent given $\boldsymbol{\beta}^{(n)}$. The prior is imposed on the indicators

$$p\left(\omega_d^{(n)}\right) = \mathrm{Ber}\left(\omega_d^{(n)}; z\right). \tag{4.5}$$



(a) Factor graph

(b) Example data and recovery results

Figure 4.1: Spike and slab model for one time moment (different time moments are independent). All signal components are conditionally independent given data, therefore structural assumptions cannot be modelled.

The prior distributions $p\left(\omega_d^{(n)}\right)$ are further denoted by $h^{\text{ind}}{}_d^{(n)}\left(\omega_d^{(n)}\right)$. The problem (4.1) – (4.5) can be solved independently for each n.

The model can be represented as a factor graph (Figure 4.1a) with a product of factors (4.1), (4.4), (4.5) for all n and d.

The posterior $p(\mathbf{B}, \boldsymbol{\Omega})$ of latent variables $\mathbf{B}$ and $\boldsymbol{\Omega}$ is

$$p = \prod_{n=1}^{N} \left[g^{(n)}\left(\boldsymbol{\beta}^{(n)}\right) \prod_{d=1}^{D} \left[f_d^{(n)}\left(\omega_d^{(n)}, \beta_d^{(n)}\right) h^{\text{ind}}{}_d^{(n)}\left(\omega_d^{(n)}\right) \right] \right]. \tag{4.6}$$

Figure 4.1b demonstrates an example of $\boldsymbol{\beta}^{(n)}$ generated by this model and recovery results with z=0.8, K=5, D=10, N=1.

4.2.3 Spike and Slab Model with a Spatial Structure

A spatial structure can be implemented by adding interdependencies for the locations of spikes in $\beta_d^{(n)}$ (Andersen et al. 2017; Wu, Zhang, et al. 2015; Zhao, Gao, et al. 2016). This is achieved by modelling the probabilities of spikes with the additional latent variables $\boldsymbol{\Gamma} = [\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(N)}] = \{\gamma_d^{(n)}\}_{n=1:N,\, d=1:D}$ that are samples from a Gaussian process. The properties of the structure are defined through the covariance function of the GP, which in

this chapter is assumed to be squared exponential:

$$p\left(\boldsymbol{\gamma}^{(n)}\right) = \mathcal{N}\left(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}_0\right), \; \boldsymbol{\Sigma}_0(i,j) = \alpha_\Sigma \exp\left(-\frac{(i-j)^2}{2\ell_\Sigma^2}\right), \tag{4.7}$$

where $\boldsymbol{\mu}^{(n)}$ is the mean vector and $\boldsymbol{\Sigma}_0$ is the covariance matrix with the hyperparameters α_Σ and ℓ_Σ^2. However, the model is not limited to squared exponential covariance functions and others can be used in practice.

The conditional independence assumption for $\omega_d^{(n)}$ from (4.5) is replaced by

$$p\left(\omega_d^{(n)}|\gamma_d^{(n)}\right) = \text{Ber}\left(\omega_d^{(n)}; \Phi\left(\gamma_d^{(n)}\right)\right), \tag{4.8}$$

$$p\left(\boldsymbol{\gamma}^{(n)}\right) = \mathcal{N}\left(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}_0\right), \tag{4.9}$$

where $\Phi(\cdot)$ is the standard Gaussian cumulative distribution function (cdf). Scaling is required to normalise probabilities to the $[0, 1]$ interval and it is convenient to use $\Phi(\cdot)$ for this purpose in the derivations with GPs (Rasmussen and Williams 2006). The conditional distributions $p\left(\omega_d^{(n)}|\gamma_d^{(n)}\right)$ are denoted by factors $h_d^{(n)}\left(\omega_d^{(n)}, \gamma_d^{(n)}\right)$. The prior distributions $p\left(\boldsymbol{\gamma}^{(n)}\right)$ are denoted by $r^{(n)}\left(\gamma^{(n)}\right)$.

In this model $\{\boldsymbol{\gamma}^{(n)}\}_{n=1:N}$ are independent and therefore the problem can be solved separately for each timestamp. Using the introduced factors (4.1), (4.4) and (4.8) – (4.9), a factor graph can be built as in Figure 4.2a. The posterior $p(\mathbf{B}, \boldsymbol{\Omega}, \boldsymbol{\Gamma})$ of the latent variables is given by

$$p = \prod_{n=1}^{N}\left[g^{(n)}\left(\boldsymbol{\beta}^{(n)}\right)\prod_{d=1}^{D}\left[f_d^{(n)}\left(\omega_d^{(n)}, \beta_d^{(n)}\right)h_d^{(n)}\left(\omega_d^{(n)}\gamma_d^{(n)}\right)\right]r^{(n)}\left(\boldsymbol{\gamma}^{(n)}\right)\right]. \tag{4.10}$$

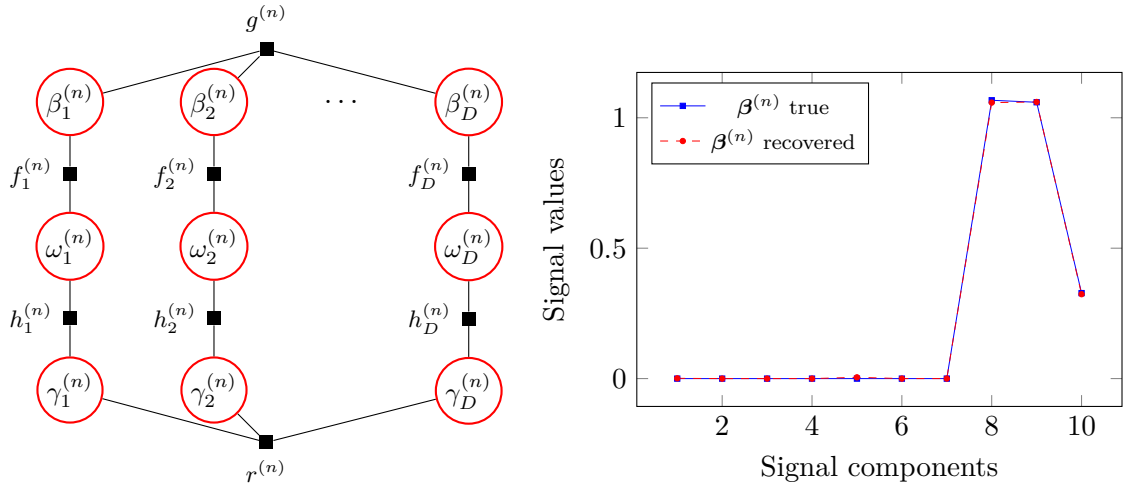
Figure 4.2b demonstrates the recovery results for the data with $\mu_d^{(n)}$=0.8, $\forall d$=$[1, \ldots, D]$, K=5, D=10, N=1.

4.2.4 Gaussian Processes Dynamics System

Gaussian processes dynamics system models allow inference in time series using probability distributions over transition and measurement dynamics, e.g. in Deisenroth and Mohamed (2012)

$$\boldsymbol{\mu}^{(n)} \sim \mathcal{N}\left(\boldsymbol{\mu}^{(n)}|\boldsymbol{\mu}^{(n-1)}, \boldsymbol{\Sigma}\right), \tag{4.11}$$

$$\boldsymbol{\gamma}^{(n)} \sim \mathcal{N}\left(\boldsymbol{\gamma}^{(n)}|\boldsymbol{\mu}^{(n)}, \mathbf{V}\right), \tag{4.12}$$



(a) Factor graph

(b) Example data and recovery results

Figure 4.2: Spike and slab model with a spatial structure for one time moment. The locations of spikes have a GP distribution, therefore encouraging a structure in space, but they are independent in time.

where $\mathbf{M} = [\boldsymbol{\mu}^{(1)}, \ldots, \boldsymbol{\mu}^{(N)}]$ are the latent states and $\boldsymbol{\Gamma} = [\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(N)}]$ are the observation vectors.

A factor graph (Figure 4.3a) for the model is expressed by factors

$$u^{(n)} = \mathcal{N}\left(\boldsymbol{\mu}^{(n)} | \boldsymbol{\mu}^{(n-1)}, \boldsymbol{\Sigma}\right), \tag{4.13}$$

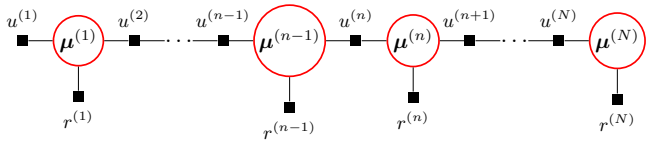
$$r^{(n)} = \mathcal{N}\left(\boldsymbol{\gamma}^{(n)} | \boldsymbol{\mu}^{(n)}, \mathbf{V}\right), \tag{4.14}$$

where $u^{(n)}$ models connections between latent states at current and previous time moments; $r^{(n)}$ identifies dependencies between observations and latent states at each time moment.

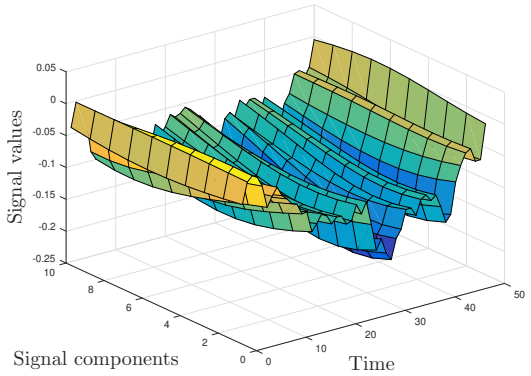
The posterior of hidden variables $p(\mathbf{M}, \boldsymbol{\Gamma})$ is

$$p = \prod_{n=1}^{N} u^{(n)} r^{(n)}. \tag{4.15}$$

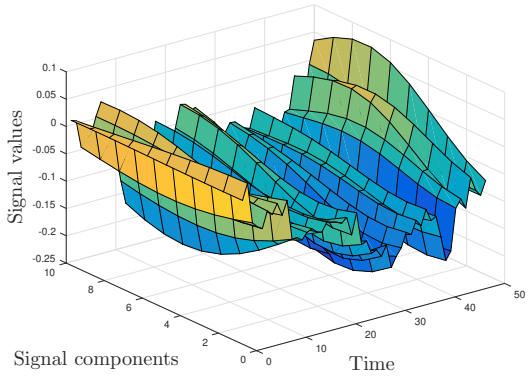
The example data and its recovery under the Gaussian process dynamic system model are presented in Figures 4.3b and 4.3c.



(a) Factor graph



(b) Example data



(c) Recovery results

Figure 4.3: Gaussian process dynamic system model

4.3 The Proposed Spatio-temporal Structured Spike and Slab Model

In this chapter a spatio-temporal latent structure of the positions of non-zero signal components is considered for the underdetermined recovery problem (4.1). The following assumptions are introduced:

1. $\boldsymbol{\beta}^{(n)}$ is sparse at each timestamp n;

2. non-zero elements in $\boldsymbol{\beta}^{(n)}$ are clustered in groups for each timestamp n;

3. these groups can move and evolve in time.

This recovery problem is addressed with the hierarchical Bayesian approach. As in the Section 4.2.2, the first assumption can be implemented in the model using the spike and slab prior

$$\mathbf{y}^{(n)} \sim \mathcal{N}\left(\mathbf{y}^{(n)}; \mathbf{X}\boldsymbol{\beta}^{(n)}, \sigma^2\mathbf{I}\right), \tag{4.16}$$

$$\beta_d^{(n)} \sim \omega_d^{(n)}\delta_0\left(\beta_d^{(n)}\right) + \left(1-\omega_d^{(n)}\right)\mathcal{N}\left(\beta_d^{(n)}; 0, \sigma_\beta^2\right). \tag{4.17}$$

Similarly to Section 4.2.3, the second model assumption can be implemented by adding spatial dependencies for the positions of spikes in $\beta_d^{(n)}$. This is achieved by modelling the probabilities of spikes $\boldsymbol{\Omega}$ with the scaled GP on $\boldsymbol{\Gamma}$

$$\omega_d^{(n)} \sim \text{Ber}\left(\omega_d^{(n)}; \Phi\left(\gamma_d^{(n)}\right)\right), \tag{4.18}$$

$$\boldsymbol{\gamma}^{(n)} \sim \mathcal{N}\left(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}_0\right), \; \boldsymbol{\Sigma}_0(i,j) = \alpha_\Sigma \exp\left(-\frac{(i-j)^2}{2\ell_\Sigma^2}\right). \tag{4.19}$$

GPs specify a prior over an unknown structure. This is particularly useful as it allows to avoid a specification of any structural patterns — the only parameter for structural modelling is the GP covariance function.

The third condition is addressed with the dynamic hierarchical GP prior. The mean $\mathbf{M} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N]$ for the spatial GP evolves over time according to the top-level temporal GP

$$\boldsymbol{\mu}^{(n)} \sim \mathcal{N}\left(\boldsymbol{\mu}^{(n)}; \boldsymbol{\mu}^{(n-1)}, \mathbf{W}\right), \; \mathbf{W}(i,j) = \alpha_W \exp\left(-\frac{(i-j)^2}{2\ell_W^2}\right), \tag{4.20}$$

where $\mathbf{W}$ is the squared exponential covariance matrix of the temporal GP with the hyper-parameters α_W and ℓ_W^2.

This allows to implicitly specify the prior over the evolution function of the structure. The rate of the evolution is controlled with the top-level GP covariance function.

According to these assumptions, the model can be expressed as a factor graph (Figure 4.4) with factors

$$g^{(n)}\left(\boldsymbol{\beta}^{(n)}\right) = \mathcal{N}\left(\mathbf{y}^{(n)}; \mathbf{X}\boldsymbol{\beta}^{(n)}, \sigma^2\mathbf{I}\right), \tag{4.21a}$$

$$f_d^{(n)}\left(\beta_d^{(n)}, \omega_d^{(n)}\right) = \omega_d^{(n)}\delta_0\left(\beta_d^{(n)}\right) + \left(1-\omega_d^{(n)}\right)\mathcal{N}\left(\beta_d^{(n)}; 0, \sigma_\beta^2\right), \tag{4.21b}$$

$$h_d^{(n)}\left(\omega_d^{(n)}, \gamma_d^{(n)}\right) = \text{Ber}\left(\omega_d^{(n)}; \Phi\left(\gamma_d^{(n)}\right)\right), \tag{4.21c}$$

$$r^{(n)}\left(\boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)}\right) = \mathcal{N}\left(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}_0\right), \tag{4.21d}$$

$$u^{(n)}\left(\boldsymbol{\mu}^{(n)}, \boldsymbol{\mu}^{(n-1)}\right) = \mathcal{N}\left(\boldsymbol{\mu}^{(n)}; \boldsymbol{\mu}^{(n-1)}, \mathbf{W}\right). \tag{4.21e}$$

The full posterior distribution $p(\mathbf{B}, \boldsymbol{\Omega}, \boldsymbol{\Gamma}, \mathbf{M})$ is then

$$p = \prod_{n=1}^{N}\left[g^{(n)}\left(\boldsymbol{\beta}^{(n)}\right)\prod_{d=1}^{D}\left[f_d^{(n)}\left(\beta_d^{(n)}, \omega_d^{(n)}\right)h_d^{(n)}\left(\omega_d^{(n)}, \gamma_d^{(n)}\right)\right]r^{(n)}\left(\boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)}\right)\right] \times \prod_{n=2}^{N} u^{(n)}\left(\boldsymbol{\mu}^{(n)}, \boldsymbol{\mu}^{(n-1)}\right). \tag{4.22}$$

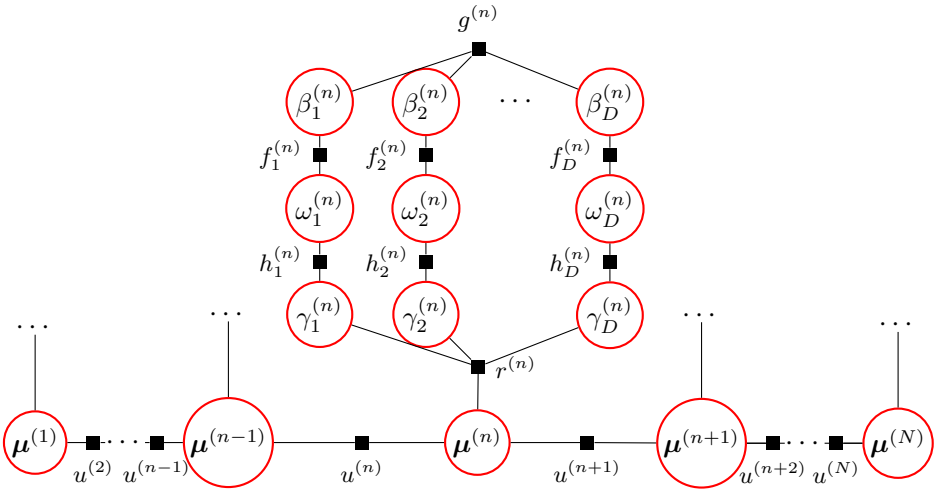


Figure 4.4: Proposed spike and slab model with a spatio-temporal structure. The locations of spikes have a GP distribution in space with parameters that are controlled by a top-level GP and they evolve in time, therefore promoting temporal dependence.

4.4 Expectation Propagation for the Hierarchical Spike and Slab Model

The exact posterior for the proposed hierarchical spike and slab model is intractable, therefore approximate inference methods should be used. In this chapter expectation propagation (EP) (Minka 2001b) is employed. EP is shown to be the most effective Bayesian inference method for sparse modelling (Hernández-Lobato, Hernández-Lobato, et al. 2015).

4.4.1 Expectation Propagation

EP is a deterministic inference method that approximates the posterior distribution using the factor decomposition (4.2), where each factor is approximated with distributions $\widetilde{\psi}_C(\cdot)$ from the exponential family:

$$\widetilde{p}(\zeta_1, ..., \zeta_m) = \frac{1}{\widetilde{Z}} \prod_C \widetilde{\psi}_C(\zeta_C), \tag{4.23}$$

where $\widetilde{p}$ is an approximating distribution and $\widetilde{Z}$ is a normalisation constant. Approximating factorised distribution is determined by minimisation of the Kullback-Leibler (KL) divergence with the true distribution. The KL-divergence is a common measure of similarity between distributions.

Direct approximation is intractable due to intractability of the true posterior. Minimisation of the KL divergence between individual factors ψ_C and $\widetilde{\psi}_C$ may not provide good approximation for the resulted product. In EP, approximation of each factor is performed in the context of other factors to improve a result for the final product. Iteratively one of the factors is chosen for refinement. The chosen factor $\widetilde{\psi}_C$ is refined to minimise the KL-divergence between the product $q \propto \widetilde{\psi}_C \prod_{C' \neq C} \widetilde{\psi}_{C'}$ and $\psi_C \prod_{C' \neq C} \widetilde{\psi}_{C'}$, where the approximating factor is replaced with a factor from the true posterior.

Factor refinement consists of five steps which are summarised below.

1. *Compute a cavity distribution* $q^{\setminus C} \propto \dfrac{q}{\widetilde{\psi}_C}$: the joint distribution without the factor $\widetilde{\psi}_C$

2. *Compute a tilted distribution* $\psi_C q^{\setminus C}$: the product of the cavity distribution and the true factor

3. *Refine the approximation* q: $q^* = \text{argmin KL}\left(\psi_C q^{\setminus C} || q\right)$ by minimising the KL-divergence between the tilted distribution $\psi_C q^{\setminus C}$ and the approximating distribution q. This is equivalent to matching the moments of the distributions (Minka 2001b).

4. *Compute an updated factor* $\widetilde{\psi}_C^{\text{new}} \propto \dfrac{q^*}{q^{\setminus C}}$ using the refined approximation and cavity distribution.

5. *Update the current joint posterior* $q^{\text{new}} \propto \widetilde{\psi}_C^{\text{new}} \prod_{C' \neq C} \widetilde{\psi}_{C'}$ with the newly updated factor $\widetilde{\psi}_C^{\text{new}}$.

The expectation propagation algorithm for this chapter is based on the following product and quotient rules for Gaussian and Bernoulli distributions.

Product of Gaussians

A product of two Gaussian distributions is an unnormalised Gaussian distribution

$$\mathcal{N}(\mathbf{x}; \mathbf{m}_1, \mathbf{\Sigma}_1)\mathcal{N}(\mathbf{x}; \mathbf{m}_2, \mathbf{\Sigma}_2) \propto \mathcal{N}(\mathbf{x}; \mathbf{m}, \mathbf{\Sigma}), \tag{4.24}$$

where

$$\mathbf{\Sigma}^{-1} = \mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1},\ \mathbf{\Sigma}^{-1}\mathbf{m} = \mathbf{\Sigma}_1^{-1}\mathbf{m}_1 + \mathbf{\Sigma}_2^{-1}\mathbf{m}_2. \tag{4.25}$$

Quotient of Gaussians

A quotient of two Gaussian distributions is an unnormalised Gaussian distribution[1]

$$\frac{\mathcal{N}(\mathbf{x};\mathbf{m}_1,\mathbf{\Sigma}_1)}{\mathcal{N}(\mathbf{x};\mathbf{m}_2,\mathbf{\Sigma}_2)} \propto \mathcal{N}(\mathbf{x};\mathbf{m},\mathbf{\Sigma}), \tag{4.26}$$

where

$$\mathbf{\Sigma}^{-1} = \mathbf{\Sigma}_1^{-1} - \mathbf{\Sigma}_2^{-1},\ \mathbf{\Sigma}^{-1}\mathbf{m} = \mathbf{\Sigma}_1^{-1}\mathbf{m}_1 - \mathbf{\Sigma}_2^{-1}\mathbf{m}_2. \tag{4.27}$$

Product of Bernoulli

A product of two Bernoulli distributions is an unnormalised Bernoulli distribution

$$\mathrm{Ber}(x;\Phi(z_1))\mathrm{Ber}(x;\Phi(z_2)) \propto \mathrm{Ber}(x;\Phi(t(z_1,z_2))), \tag{4.28}$$

where

$$t(z_1,z_2) = \Phi^{-1}\left(\left[\frac{(1-\Phi(z_1))(1-\Phi(z_2))}{\Phi(z_1)\Phi(z_2)} + 1\right]^{-1}\right). \tag{4.29}$$

Quotient of Bernoulli

A quotient of two Bernoulli distributions is an unnormalised Bernoulli distribution

$$\frac{\mathrm{Ber}(x;\Phi(z_1))}{\mathrm{Ber}(x;\Phi(z_2))} \propto \mathrm{Ber}(x;\Phi(d(z_1,z_2))), \tag{4.30}$$

where

$$d(z_1,z_2) = \Phi^{-1}\left(\left[\frac{(1-\Phi(z_1))\Phi(z_2)}{(1-\Phi(z_2))\Phi(z_1)} + 1\right]^{-1}\right). \tag{4.31}$$

4.4.2 Approximating Factors

Here the key components of the EP inference algorithm for the proposed model are provided. The true posterior p (4.22) is approximated with the distribution q

$$q = \prod_n q_{g^{(n)}} q_{f^{(n)}} q_{h^{(n)}} q_{r^{(n)}} q_{u^{(n)}}, \tag{4.32}$$

where each factor q_a, $a \in \{g^{(n)}, f^{(n)}, h^{(n)}, r^{(n)}, u^{(n)}\}$, is from the exponential family and all latent variables are separated in the factors.

[1] Although quotient can lose positive semidefiniteness, it will still be referred as a Gaussian distribution

Below the factors q_a of the approximating posterior q are introduced. Gaussian and Bernoulli distributions are used in the factors, which parameters are updated during the iterations of the EP algorithm.

The factors $g^{(n)} = \mathcal{N}(\mathbf{y}^{(n)}; \mathbf{X}\boldsymbol{\beta}^{(n)}, \sigma^2\mathbf{I})$ from (4.21a) can be viewed as the distributions of $\boldsymbol{\beta}^{(n)}$ with fixed observed variables $\mathbf{y}^{(n)}$: $q_{g^{(n)}} = \mathcal{N}(\boldsymbol{\beta}^{(n)}; \mathbf{m}_{g^{(n)}}, \mathbf{V}_{g^{(n)}})$, where $\mathbf{m}_{g^{(n)}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}^{(n)}$, $\mathbf{V}_{g^{(n)}} = \sigma^2(\mathbf{X}^\top\mathbf{X})^{-1}$.

In the EP inference algorithm, each of the introduced approximating factors $q_{f^{(n)}}$, $q_{h^{(n)}}$, $q_{r^{(n)}}$, $q_{u^{(n)}}$ is iteratively updated according to the factor refinement procedure as in Section 4.4.1. Note that the factors $q_{g^{(n)}}$ are not updated, as the corresponding factors $g^{(n)}$ from the true posterior distribution are already from the exponential family.

The factors $f^{(n)} = \prod_{d=1}^{D} f_d^{(n)}$ from (4.21b) are approximated with the products of Gaussian and Bernoulli distributions

$$q_{f^{(n)}} = \mathcal{N}(\boldsymbol{\beta}^{(n)}; \mathbf{m}_{f^{(n)}}, \mathbf{V}_{f^{(n)}}) \prod_{d=1}^{D} \mathrm{Ber}(\omega_d^{(n)}; \Phi(z_{f_d^{(n)}})), \tag{4.33}$$

where the components of $\boldsymbol{\beta}^{(n)}$ are independent. Therefore, the covariance matrices $\mathbf{V}_{f^{(n)}}$ are diagonal. Distribution parameters $\mathbf{m}_{f^{(n)}}$, $\mathbf{V}_{f^{(n)}}$, $z_{f_d^{(n)}}$ are updated during the EP iterations.

The approximation for factors $h^{(n)} = \prod_{d=1}^{D} h_d^{(n)}$ from (4.21c) is similar to $f^{(n)}$. They are approximated with the products of Gaussian and Bernoulli distributions

$$q_{h^{(n)}} = \mathcal{N}(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\nu}_{h^{(n)}}, \mathbf{S}^{(n)}) \prod_{d=1}^{D} \mathrm{Ber}(\omega_d^{(n)}; \Phi(z_{h_d^{(n)}})), \tag{4.34}$$

where the components of $\boldsymbol{\gamma}^{(n)}$ are independent. Single covariance matrix $\mathbf{S}_h$ is used for all time moments. Distribution parameters $\boldsymbol{\nu}_{h^{(n)}}$, $\mathbf{S}_h$, $z_{h_d^{(n)}}$ are updated during EP iterations.

The approximation for factors $r^{(n)} = \mathcal{N}(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}_0)$ and $u^{(n)} = \mathcal{N}(\boldsymbol{\mu}^{(n)}; \boldsymbol{\mu}^{(n-1)}, \mathbf{W})$ from (4.21d) and (4.21e) is intended to separate the latent variables and it is represented as products of Gaussian distributions

$$q_{r^{(n)}} = \mathcal{N}(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\nu}_{r^{(n)}}, \mathbf{S}_r)\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}_{r^{(n)}}, \mathbf{D}_r), \tag{4.35}$$

$$q_{u^{(n)}} = \mathcal{N}(\boldsymbol{\mu}^{(n-1)}; \mathbf{e}_{u^{(n)}\leftarrow}, \mathbf{D}_{u\leftarrow})\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}_{u^{(n)}\rightarrow}, \mathbf{D}_{u\rightarrow}). \tag{4.36}$$

Distribution parameters $\mathbf{e}_{r^{(n)}}$, $\mathbf{D}_r$, $\boldsymbol{\nu}_{r^{(n)}}$, $\mathbf{S}_r$, $\mathbf{e}_{u^{(n)}\leftarrow}$, $\mathbf{D}_{u\leftarrow}$, $\mathbf{e}_{u^{(n)}\rightarrow}$, $\mathbf{D}_{u\rightarrow}$ are updated during EP iterations.

4.4.3 Full Posterior Approximation

The posterior approximation q given by (4.32) thus contains the products of Gaussian and Bernoulli distributions that are equal to unnormalised Gaussian and Bernoulli distributions, respectively. This can be conveniently expressed in terms of the natural parameters and q can be represented in terms of distributions of the latent variables.

For $\boldsymbol{\beta}^{(n)}$ in the posterior distribution q the Gaussian product property leads to the Gaussian distribution $\mathcal{N}(\boldsymbol{\beta}^{(n)}; \mathbf{m}^{(n)}, \mathbf{V}^{(n)})$ with natural parameters

$$\mathbf{V}^{(n)^{-1}} = \mathbf{V}_{g^{(n)}}^{-1} + \mathbf{V}_{f^{(n)}}^{-1},\ \mathbf{V}^{(n)^{-1}}\mathbf{m}^{(n)} = \mathbf{V}_{g^{(n)}}^{-1}\mathbf{m}_{g^{(n)}} + \mathbf{V}_{f^{(n)}}^{-1}\mathbf{m}_{f^{(n)}}. \tag{4.37}$$

Similarly, $\boldsymbol{\gamma}^{(n)}$ in q is distributed as $\mathcal{N}(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\nu}^{(n)}, \mathbf{S})$, where natural parameters are

$$\mathbf{S}^{-1} = \mathbf{S}_h^{-1} + \mathbf{S}_r^{-1},\ \mathbf{S}^{-1}\boldsymbol{\nu}^{(n)} = \mathbf{S}_h^{-1}\boldsymbol{\nu}_{h^{(n)}} + \mathbf{S}_r^{-1}\boldsymbol{\nu}_{r^{(n)}}. \tag{4.38}$$

The top GP latent variables $\boldsymbol{\mu}^{(n)}$ have the Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}^{(n)}, \mathbf{D})$ with natural parameters

$$\mathbf{D}^{-1} = \mathbf{D}_r^{-1} + \mathbf{D}_{u\rightarrow}^{-1}\mathbb{1}_{n>1} + \mathbf{D}_{u\leftarrow}^{-1}\mathbb{1}_{n<N}, \tag{4.39a}$$

$$\mathbf{D}^{-1}\mathbf{e}^{(n)} = \mathbf{D}_r^{-1}\mathbf{e}_{r^{(n)}} + \mathbf{D}_{u\rightarrow}^{-1}\mathbf{e}_{u^{(n)}\rightarrow}\mathbb{1}_{n>1} + \mathbf{D}_{u\leftarrow}^{-1}\mathbf{e}_{u^{(n+1)}\leftarrow}\mathbb{1}_{n<N}, \tag{4.39b}$$

where $\mathbb{1}$ is the indicator function.

The distributions for $\boldsymbol{\omega}^{(n)}$ are $\prod_{d=1}^{D}\mathrm{Ber}(\omega_d^{(n)}; \Phi(z_d^{(n)}))$ with parameters

$$z_d^{(n)} = \Phi^{-1}\left(\left[\frac{(1-\Phi(z_{f_d^{(n)}}))(1-\Phi(z_{h_d^{(n)}}))}{\Phi(z_{f_d^{(n)}})\Phi(z_{h_d^{(n)}})} + 1\right]^{-1}\right). \tag{4.40}$$

The full approximating posterior q is then

$$\begin{aligned} q = &\prod_{n=1}^{N}\mathcal{N}(\boldsymbol{\beta}^{(n)}; \mathbf{m}^{(n)}, \mathbf{V}^{(n)})\prod_{t=1}^{T}\prod_{d=1}^{D}\mathrm{Ber}(\omega_d^{(n)}; \Phi(z_d^{(n)})) \\ &\times \prod_{n=1}^{N}\mathcal{N}(\boldsymbol{\gamma}^{(n)}; \boldsymbol{\nu}^{(n)}, \mathbf{S})\prod_{n=1}^{N}\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}^{(n)}, \mathbf{D}). \end{aligned} \tag{4.41}$$

In the derivation of updates for the factors $f_d^{(n)}$, $h_d^{(n)}$, $r^{(n)}$ the superscript (n) is omitted, as they are conditionally independent for different time stamps.

EP Update for Factor f_d

Cavity distribution The unnormalised cavity distribution $q^{\backslash f_d}(\beta_d, \omega_d) = \frac{q(\beta_d,\omega_d)}{q_{f_d}(\beta_d,\omega_d)}$ can be computed as

$$\begin{aligned} q^{\backslash f_d} &= \frac{\mathcal{N}(\beta_d; \mathbf{m}(d), \mathbf{V}(d,d))\text{Ber}(\omega_d; \Phi(z_d))}{\mathcal{N}(\beta_d; \mathbf{m}_f(d), \mathbf{V}_f(d,d))\text{Ber}(\omega_d; \Phi(z_{f_d}))} \\ &\propto \mathcal{N}(\beta_d; {m_d}^{\backslash f}, {v_d}^{\backslash f})\text{Ber}(\omega_d; \Phi({z_d}^{\backslash f})), \end{aligned}$$

where

$$\begin{aligned} ({v_d}^{\backslash f})^{-1} &= \mathbf{V}^{-1}(d,d) - \mathbf{V}_f^{-1}(d,d), \\ ({v_d}^{\backslash f})^{-1}{m_d}^{\backslash f} &= \mathbf{V}^{-1}(d,d)\mathbf{m}(d) - \mathbf{V}_f^{-1}(d,d)\mathbf{m}_f(d,d), \\ {z_d}^{\backslash f} &= z_{h_d}. \end{aligned}$$

Moments matching The moments of the tilted distribution $q^{\backslash f_d} f_d$ are

$$\begin{aligned} Z_d &= \Phi({z_d}^{\backslash f})\mathcal{N}(0; {m_d}^{\backslash f}, {v_d}^{\backslash f}) + (1 - \Phi({z_d}^{\backslash f}))\mathcal{N}(0; {m_d}^{\backslash f}, {v_d}^{\backslash f} + \sigma_\beta^2), \\ \mathbb{E}\beta_d &= \frac{1 - \Phi({z_d}^{\backslash f})}{Z_d}\mathcal{N}(0; {m_d}^{\backslash f}, {v_d}^{\backslash f})\frac{{m_d}^{\backslash f}\sigma_\beta^2}{{v_d}^{\backslash f} + \sigma_\beta^2}, \\ \mathbb{E}{\beta_d}^2 &= \frac{1 - \Phi({z_d}^{\backslash f})}{Z_d}\mathcal{N}(0; {m_d}^{\backslash f}, {v_d}^{\backslash f})\left(\frac{({m_d}^{\backslash f})^2\sigma_\beta^4}{({v_d}^{\backslash f} + \sigma_\beta^2)^2} + \frac{{v_d}^{\backslash f}\sigma_\beta^2}{{v_d}^{\backslash f} + \sigma_\beta^2}\right), \\ \mathbb{E}\omega_d &= \frac{\Phi({z_d}^{\backslash f})}{Z_d}\mathcal{N}(0; {m_d}^{\backslash f}, {v_d}^{\backslash f}). \end{aligned}$$

The new approximation $q^*(\beta_d, \omega_d)$ is

$$q^* = \mathcal{N}(\beta_d; {m_d}^{q^*}, {v_d}^{q^*})\text{Ber}(\omega_d; \Phi({z_d}^{q^*})),$$

where

$${m_d}^{q^*} = \mathbb{E}\beta_d,\ {v_d}^{q^*} = \mathbb{E}{\beta_d}^2 - (\mathbb{E}\beta_d)^2,\ {z_d}^{q^*} = \Phi^{-1}(\mathbb{E}\omega_d).$$

Factor update The new factor approximation $q_{f_d}^{\text{new}}(\beta_d, \omega_d) = \frac{q^*(\beta_d,\omega_d)}{q^{\backslash f_d}(\beta_d,\omega_d)}$ can be computed as

$$\begin{aligned} q_{f_d}^{\text{new}} &= \frac{\mathcal{N}\left(\beta_d; {m_d}^{q^*}, {v_d}^{q^*}\right)\text{Ber}\left(\omega_d; \Phi\left({z_d}^{q^*}\right)\right)}{\mathcal{N}\left(\beta_d; {m_d}^{\backslash f}, {v_d}^{\backslash f}\right)\text{Ber}\left(\omega_d; \Phi\left({z_d}^{\backslash f}\right)\right)} \\ &\propto \mathcal{N}\left(\beta_d; \mathbf{m}_f^{\text{new}}(d), \mathbf{V}_f^{\text{new}}(d,d)\right)\text{Ber}\left(\omega_d; \Phi\left(z_{f_d}^{\text{new}}\right)\right), \end{aligned}$$

where

$$\begin{aligned}
\left(\mathbf{V}_f^{\text{new}}\right)^{-1}(d,d) &= \left({v_d}^{q^*}\right)^{-1} - \left({v_d}^{\backslash f}\right)^{-1}, \\
\left(\mathbf{V}_{f_d}^{\text{new}}\right)^{-1}(d,d)\mathbf{m}_{f_d}^{\text{new}}(d) &= \left({v_d}^{q^*}\right)^{-1} {m_d}^{q^*} - \left({v_d}^{\backslash f}\right)^{-1} m_{f_d}^{\backslash f}, \\
z_{f_d}^{\text{new}} &= d\left({z_d}^{q^*}, {z_d}^{\backslash f}\right).
\end{aligned}$$

EP Update for Factor h_d

Cavity distribution The unnormalised cavity distribution $q^{\backslash h_d}(\gamma_d, \omega_d) = \frac{q(\gamma_d,\omega_d)}{q_{h_d}(\gamma_d,\omega_d)}$ can be computed as

$$\begin{aligned}
q^{\backslash h_d} &= \frac{\mathcal{N}(\gamma_d; \boldsymbol{\nu}(d), \mathbf{S}(d,d))\text{Ber}(\omega_d; \Phi(z_d))}{\mathcal{N}(\gamma_d; \boldsymbol{\nu}_h(d), \mathbf{S}_h(d,d))\text{Ber}(\omega_d; \Phi(z_{h_d}))} \\
&\propto \mathcal{N}(\gamma_d; {\nu_d}^{\backslash h}, {s_d}^{\backslash h})\text{Ber}(\omega_d; \Phi({z_d}^{\backslash h})),
\end{aligned}$$

where

$$\begin{aligned}
({s_d}^{\backslash h})^{-1} &= \mathbf{S}^{-1}(d,d) - \mathbf{S}_h^{-1}(d,d) \\
({s_d}^{\backslash h})^{-1}{\nu_d}^{\backslash h} &= \mathbf{S}^{-1}(d,d)\boldsymbol{\mu}(d) - \mathbf{S}_h^{-1}(d,d)\boldsymbol{\nu}_h(d,d) \\
{z_d}^{\backslash h} &= z_{f_d}.
\end{aligned}$$

Moments matching The moments of the tilted distribution $q^{\backslash h_d} h_d$ are

$$\begin{aligned}
Z_d &= \Phi({z_d}^{\backslash h})\Phi(a) + (1 - \Phi({z_d}^{\backslash h}))(1 - \Phi(a)), \\
\mathbb{E}\gamma_d &= \frac{1}{Z_d}(\Phi({z_d}^{\backslash h})K + (1 - \Phi({z_d}^{\backslash h}))({\nu_d}^{\backslash h} - K)), \\
\mathbb{E}{\gamma_d}^2 &= \frac{1}{Z_d}\Bigg[(2\Phi({z_d}^{\backslash h}) - 1)\Bigg(({\nu_d}^{\backslash h})^2\Phi(a) + {s_d}^{\backslash h}\Phi(a) \\
&\quad + \frac{2{\nu_d}^{\backslash h}{s_d}^{\backslash h}\mathcal{N}(a;0,1)}{\sqrt{1 + {s_d}^{\backslash h}}} - \frac{({s_d}^{\backslash h})^2 a\mathcal{N}(a;0,1)}{1 + {s_d}^{\backslash h}}\Bigg) \\
&\quad + (1 - \Phi({z_d}^{\backslash h}))(({s_d}^{\backslash h} + ({\nu_d}^{\backslash h})^2)\Bigg], \\
\mathbb{E}\omega_d &= \frac{\Phi({z_d}^{\backslash h})\Phi(a)}{Z_d},
\end{aligned}$$

where

$$a = \frac{{\nu_d}^{\backslash h}}{\sqrt{1 + {s_d}^{\backslash h}}}, \quad K = {s_d}^{\backslash h}\frac{\mathcal{N}(a;0,1)}{\sqrt{1 + {s_d}^{\backslash h}}} + {\nu_d}^{\backslash h}\Phi(a).$$

The new approximation $q^*(\gamma_d, \omega_d)$ is

$$q^* = \mathcal{N}(\gamma_d; {\nu_d}^{q^*}, {s_d}^{q^*})\text{Ber}(\omega_d; \Phi({z_d}^{q^*})),$$

where

$${\nu_d}^{q^*} = \mathbb{E}\gamma_d,\ {s_d}^{q^*} = \mathbb{E}{\gamma_d}^2 - (\mathbb{E}\gamma_d)^2,\ {z_d}^{q^*} = \Phi^{-1}(\mathbb{E}\omega_d).$$

Factor update The new factor approximation $q_{h_d}^{\text{new}}(\gamma_d, \omega_d) = \dfrac{q^*(\gamma_d, \omega_d)}{q^{\setminus h_d}(\gamma_d, \omega_d)}$ can be computed as

$$\begin{aligned} q_{h_d}^{\text{new}} &= \frac{\mathcal{N}\left(\gamma_d; {\nu_d}^{q^*}, {s_d}^{q^*}\right)\text{Ber}\left(\omega_d; \Phi\left({z_d}^{q^*}\right)\right)}{\mathcal{N}\left(\gamma_d; {\nu_d}^{\setminus h}, {s_d}^{\setminus h}\right)\text{Ber}\left(\omega_d; \Phi\left({z_d}^{\setminus h}\right)\right)} \\ &\propto \mathcal{N}\left(\gamma_d; \boldsymbol{\nu}_h^{\text{new}}(d), \mathbf{S}_h^{\text{new}}(d,d)\right)\text{Ber}\left(\omega_d; \Phi\left(z_{h_d}^{\text{new}}\right)\right), \end{aligned}$$

where

$$\begin{aligned} (\mathbf{S}_h^{\text{new}})^{-1}(d,d) &= \left(s_d^{q^*}\right)^{-1} - \left({s_d}^{\setminus h}\right)^{-1}, \\ (\mathbf{S}_h^{\text{new}})^{-1}(d,d)\boldsymbol{\nu}_h^{\text{new}}(d) &= \left(s_d^{q^*}\right)^{-1}\nu_d^{q^*} - \left({s_d}^{\setminus h}\right)^{-1}\nu_d^{\setminus h}, \\ z_{h_d}^{\text{new}} &= d\left(z_d^{q^*}, {z_d}^{\setminus h}\right). \end{aligned}$$

EP Update for Factor r

Cavity distribution The unnormalised cavity distribution $q^{\setminus r}(\boldsymbol{\gamma}, \boldsymbol{\mu}) = \frac{q(\boldsymbol{\gamma}, \boldsymbol{\mu})}{q_r(\boldsymbol{\gamma}, \boldsymbol{\mu})}$ can be computed as

$$\begin{aligned} q^{\setminus r} &= \frac{\mathcal{N}(\boldsymbol{\gamma}; \boldsymbol{\nu}, \mathbf{S})\mathcal{N}(\boldsymbol{\mu}; \mathbf{e}, \mathbf{D})}{\mathcal{N}(\boldsymbol{\gamma}; \boldsymbol{\nu}_r, \mathbf{S}_r)\mathcal{N}(\boldsymbol{\mu}; \mathbf{e}_r, \mathbf{D}_r)} \\ &\propto \mathcal{N}(\boldsymbol{\gamma}; \boldsymbol{\nu}^{\setminus r}, \mathbf{S}^{\setminus r})\mathcal{N}(\boldsymbol{\mu}; \mathbf{e}^{\setminus r}, \mathbf{D}^{\setminus r}), \end{aligned}$$

where

$$\begin{aligned} (\mathbf{S}^{\setminus r})^{-1} &= (\mathbf{S})^{-1} - (\mathbf{S}_r)^{-1} \\ (\mathbf{S}^{\setminus r})^{-1}\boldsymbol{\nu}^{\setminus r} &= (\mathbf{S})^{-1}\boldsymbol{\nu} - (\mathbf{S}_r)^{-1}\boldsymbol{\nu}_r \\ (\mathbf{D}^{\setminus r})^{-1} &= (\mathbf{D})^{-1} - (\mathbf{D}_r)^{-1} \\ (\mathbf{D}^{\setminus r})^{-1}\mathbf{e}^{\setminus r} &= (\mathbf{D})^{-1}\mathbf{e} - (\mathbf{D}_r)^{-1}\mathbf{e}_r. \end{aligned}$$

Find the update for the factor q_r^{new} For the factor q_r parameters of the Gaussian distributions found during the moment matching step are cancelled out during the factor update

step and the resulting formulae are

$$q_r^{\text{new}}(\boldsymbol{\gamma}, \boldsymbol{\mu}) \propto \mathcal{N}\left(\boldsymbol{\gamma}; \boldsymbol{\nu}^{\text{new}}, \mathbf{S}^{\text{new}}\right) \mathcal{N}\left(\boldsymbol{\mu}; \mathbf{e}^{\text{new}}, \mathbf{D}^{\text{new}}\right),$$

where

$$\begin{aligned}
(\mathbf{S}^{\text{new}})^{-1} &= (\mathbf{D}^{\setminus r} + \boldsymbol{\Sigma}_0)^{-1} \\
(\mathbf{S}^{\text{new}})^{-1}\boldsymbol{\nu}^{\text{new}} &= (\mathbf{I} - (\mathbf{S}^{\text{new}})^{-1}\boldsymbol{\Sigma}_0)(\mathbf{D}^{\setminus r})^{-1}\mathbf{e}^{\setminus r} \\
(\mathbf{D}^{\text{new}})^{-1} &= (\mathbf{S}^{\setminus r} + \boldsymbol{\Sigma}_0)^{-1} \\
(\mathbf{D}^{\text{new}})^{-1}\mathbf{e}^{\text{new}} &= (\mathbf{I} - (\mathbf{D}^{\text{new}})^{-1}\boldsymbol{\Sigma}_0)(\mathbf{S}^{\setminus r})^{-1}\boldsymbol{\nu}^{\setminus r}.
\end{aligned}$$

EP Update for Factor $u^{(n)}$

Cavity distribution The unnormalised cavity distribution $q^{\setminus u^{(n)}}(\boldsymbol{\mu}^{(n-1)}, \boldsymbol{\mu}^{(n)}) = \dfrac{q(\boldsymbol{\mu}^{(n-1)}, \boldsymbol{\mu}^{(n)})}{q_{u^{(n)}}(\boldsymbol{\mu}^{(n-1)}, \boldsymbol{\mu}^{(n)})}$ can be computed as

$$\begin{aligned}
q^{\setminus u^{(n)}} &= \frac{\mathcal{N}(\boldsymbol{\mu}^{(n-1)}; \mathbf{e}^{(n-1)}, \mathbf{D})\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}^{(n)}, \mathbf{D})}{\mathcal{N}(\boldsymbol{\mu}^{(n-1)}; \mathbf{e}_{u^{(n)}\leftarrow}, \mathbf{D}_{u\leftarrow})\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}_{u^{(n)}\rightarrow}, \mathbf{D}_{u\rightarrow})} \\
&\propto \mathcal{N}(\boldsymbol{\mu}^{(n-1)}; \mathbf{e}^{(n-1)^{\setminus u}}, \mathbf{D}^{(n-1)^{\setminus u}})\mathcal{N}(\boldsymbol{\mu}^{(n)}; \mathbf{e}^{(n)^{\setminus u}}, \mathbf{D}^{(n)^{\setminus u}}),
\end{aligned}$$

where

$$\begin{aligned}
(\mathbf{D}^{(n-1)^{\setminus u}})^{-1} &= \mathbf{D}^{-1} - (\mathbf{D}_{u\leftarrow})^{-1} \\
(\mathbf{D}^{(n-1)^{\setminus u}})^{-1}\mathbf{e}^{(n-1)^{\setminus u}} &= \mathbf{D}^{-1}\mathbf{e}^{(n-1)} - (\mathbf{D}_{u\leftarrow})^{-1}\mathbf{e}_{u^{(n)}\leftarrow} \\
(\mathbf{D}^{(n)^{\setminus u}})^{-1} &= \mathbf{D}^{-1} - (\mathbf{D}_{u\rightarrow})^{-1} \\
(\mathbf{D}^{(n)^{\setminus u}})^{-1}\mathbf{e}^{(n)^{\setminus u}} &= \mathbf{D}^{-1}\mathbf{e}^{(n)} - (\mathbf{D}_{u\rightarrow})^{-1}\mathbf{e}_{u^{(n)}\rightarrow}.
\end{aligned}$$

Find the update for the factor $q_{u^{(n)}}^{new}$ For the factor $q_{u^{(n)}}$ parameters of the Gaussian distributions found during the moment matching step are cancelled out during the factor update step and the resulting formulae are

$$q_{u^{(n)}}^{\text{new}}(\boldsymbol{\mu}^{(n-1)}, \boldsymbol{\mu}^{(n)}) \propto \mathcal{N}\left(\boldsymbol{\mu}^{(n)}; \mathbf{e}_{u^{(n)}\rightarrow}^{\text{new}}, \mathbf{D}_{u\rightarrow}^{\text{new}}\right) \mathcal{N}\left(\boldsymbol{\mu}^{(n-1)}; \mathbf{e}_{u^{(n)}\leftarrow}^{\text{new}}, \mathbf{D}_{u\leftarrow}^{\text{new}}\right),$$

where

$$
\begin{aligned}
(\mathbf{D}^{\text{new}}_{u\rightarrow})^{-1} &= (\mathbf{D}^{(n-1)\setminus u} + \mathbf{W})^{-1} \\
(\mathbf{D}^{\text{new}}_{u\rightarrow})^{-1}\mathbf{e}^{\text{new}}_{u^{(n)}\rightarrow} &= (\mathbf{I} - (\mathbf{D}^{\text{new}}_{u\rightarrow})^{-1}\mathbf{W})(\mathbf{D}^{(n-1)\setminus u})^{-1}\mathbf{e}^{(n-1)\setminus u} \\
(\mathbf{D}^{\text{new}}_{u\leftarrow})^{-1} &= (\mathbf{D}^{(n)\setminus u} + \mathbf{W})^{-1} \\
(\mathbf{D}^{\text{new}}_{u\leftarrow})^{-1}\mathbf{e}^{\text{new}}_{u^{(n)}\leftarrow} &= (\mathbf{I} - (\mathbf{D}^{\text{new}}_{u\leftarrow})^{-1}\mathbf{W})(\mathbf{D}^{(n)\setminus u})^{-1}\mathbf{e}^{(n)\setminus u}.
\end{aligned}
$$

4.5 Online Inference with Bayesian Filtering

In this section the problem (4.1) is considered for streaming data, i.e. when new data becomes available at every timestamp. The conventional batch inference can be infeasible for large or streaming data. The developed online Bayesian filtering algorithm for the model presented in Section 4.3 allows to iteratively update the approximation of $\boldsymbol{\beta}^{(n)}$ based on new samples of data.

Bayesian filtering consist of two steps that are iterated for each new sample of data:

- *prediction*, where an estimate of a hidden system state at the next time step is predicted based on the observations available at the current time moment;

- *update*, where this estimate is updated once an observation at the next time moment is obtained.

In the proposed model the hidden state is represented by the latent variables $\boldsymbol{\beta}^{(n)}$, $\boldsymbol{\omega}^{(n)}$, $\boldsymbol{\gamma}^{(n)}$ and $\boldsymbol{\mu}^{(n)}$ that should be inferred based on observations $\mathbf{y}^{(n)}$.

4.5.1 Prediction

At the prediction step for the timestamp $n+1$ the current estimate of the posterior distribution of the latent variables $p\left(\boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)} | \mathbf{y}^{(1)\ldots(n)}\right)$ is available. It is based on all observations $\mathbf{y}^{(1)\ldots(n)} = [\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}]$ up to the timestamp n. The initial estimate of this posterior can be obtained by the offline inference algorithm from Section 4.4 applied to the initial N_{init} timestamps.

Marginalisation of the latent variables for the current timestamp n allows to obtain predictions for the latent variables for the next timestamp $n+1$

$$\begin{aligned} & p\left(\boldsymbol{\beta}^{(n+1)},\ \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \mathbf{y}^{(1)\ldots(n)}\right) \\ & \quad = \int p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)}\right) \\ & \qquad \times p\left(\boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)} | \mathbf{y}^{(1)\ldots(n)}\right) d\boldsymbol{\beta}^{(n)} d\boldsymbol{\omega}^{(n)} d\boldsymbol{\gamma}^{(n)} d\boldsymbol{\mu}^{(n)}. \end{aligned} \tag{4.42}$$

The first term in the integral (4.42) is factorised according to the generative model (4.21)

$$\begin{aligned} & p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)}\right) \\ & = p\left(\boldsymbol{\beta}^{(n+1)} | \boldsymbol{\omega}^{(n+1)}\right) p\left(\boldsymbol{\omega}^{(n+1)} | \boldsymbol{\gamma}^{(n+1)}\right) p\left(\boldsymbol{\gamma}^{(n+1)} | \boldsymbol{\mu}^{(n+1)}\right) p\left(\boldsymbol{\mu}^{(n+1)} | \boldsymbol{\mu}^{(n)}\right) \end{aligned} \tag{4.43}$$

Therefore, the terms related to variables $\boldsymbol{\beta}^{(n+1)}$, $\boldsymbol{\omega}^{(n+1)}$ and $\boldsymbol{\gamma}^{(n+1)}$ are independent from the integral variables in (4.42) and the integral can be rewritten as

$$\begin{aligned} & \int p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)}\right) \\ & \qquad \times p\left(\boldsymbol{\beta}^{(n)}, \boldsymbol{\omega}^{(n)}, \boldsymbol{\gamma}^{(n)}, \boldsymbol{\mu}^{(n)} | \mathbf{y}^{(1)\ldots(n)}\right) d\boldsymbol{\beta}^{(n)} d\boldsymbol{\omega}^{(n)} d\boldsymbol{\gamma}^{(n)} d\boldsymbol{\mu}^{(n)} \\ & = p(\boldsymbol{\beta}^{(n+1)} | \boldsymbol{\omega}^{(n+1)}) p\left(\boldsymbol{\omega}^{(n+1)} | \boldsymbol{\gamma}^{(n+1)}\right) p\left(\boldsymbol{\gamma}^{(n+1)} | \boldsymbol{\mu}^{(n+1)}\right) \\ & \qquad \times \int p\left(\boldsymbol{\mu}^{(n+1)} | \boldsymbol{\mu}^{(n)}\right) p\left(\boldsymbol{\mu}^{(n)} | \mathbf{y}^{(1)\ldots(n)}\right) d\boldsymbol{\mu}^{(n)}, \end{aligned} \tag{4.44}$$

where $\boldsymbol{\beta}^{(n)}$, $\boldsymbol{\omega}^{(n)}$ and $\boldsymbol{\gamma}^{(n)}$ are marginalised out.

The initial estimate of the posterior $p\left(\boldsymbol{\mu}^{(N_{\text{init}})} | \mathbf{y}^{(1):(N_{\text{init}})}\right)$ obtained from the offline EP algorithm is a Gaussian distribution:

$$p\left(\boldsymbol{\mu}^{(N_{\text{init}})} | \mathbf{y}^{(1):(N_{\text{init}})}\right) = \mathcal{N}\left(\boldsymbol{\mu}^{(N_{\text{init}})}; \mathbf{e}^{(1):(N_{\text{init}})}, \mathbf{D}^{(1):(N_{\text{init}})}\right), \tag{4.45}$$

where $\mathbf{e}^{(1):(N_{\text{init}})}$ and $\mathbf{D}^{(1):(N_{\text{init}})}$ are the mean and the covariance matrix of the estimate of the posterior for $\boldsymbol{\mu}^{(N_{\text{init}})}$ obtained based on observations $\mathbf{y}^{(1):(N_{\text{init}})}$.

According to the generative model, the first term of the integral in (4.44) is also Gaussian (see (4.21e)), therefore the integral is also a Gaussian distribution on $\boldsymbol{\mu}^{(n+1)}$ for $n = N_{\text{init}}$:

$$\int p\left(\boldsymbol{\mu}^{(n+1)} | \boldsymbol{\mu}^{(n)}\right) p\left(\boldsymbol{\mu}^{(n)} | \mathbf{y}^{(1)\ldots(n)}\right) d\boldsymbol{\mu}^{(n)} = \mathcal{N}\left(\boldsymbol{\mu}^{(n+1)}; \mathbf{e}^{(1):(n)}, \mathbf{D}_{\text{predict}}^{(1):(n)}\right) \stackrel{\text{def}}{=} \hat{p}(\boldsymbol{\mu}^{(n+1)}), \tag{4.46}$$

where $\mathbf{D}_{\text{predict}}^{(1):(n)} = \mathbf{W} + \mathbf{D}^{(1):(n)}$ is the covariance of the predicted distribution.

Substitution of (4.44) and (4.46) back into (4.42) provides the predicted distribution:

$$\begin{aligned} & p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \mathbf{y}^{(1)\ldots(n)}\right) \\ =& p\left(\boldsymbol{\beta}^{(n+1)} | \boldsymbol{\omega}^{(n+1)}\right) p\left(\boldsymbol{\omega}^{(n+1)} | \boldsymbol{\gamma}^{(n+1)}\right) p\left(\boldsymbol{\gamma}^{(n+1)} | \boldsymbol{\mu}^{(n+1)}\right) \hat{p}\left(\boldsymbol{\mu}^{(n+1)}\right). \end{aligned} \tag{4.47}$$

4.5.2 Update

At the update step the predicted distribution (4.47) of the latent variables for the next timestamp is corrected with the new data $\mathbf{y}^{(n+1)}$

$$\begin{aligned} & p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \mathbf{y}^{(1)\ldots(n+1)}\right) \\ =& \frac{1}{Z} p\left(\mathbf{y}^{(n+1)} | \boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)}\right) p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \mathbf{y}^{(1)\ldots(n)}\right) \\ =& \frac{1}{Z} p\left(\mathbf{y}^{(n+1)} | \boldsymbol{\beta}^{(n+1)}\right) p\left(\boldsymbol{\beta}^{(n+1)} | \boldsymbol{\omega}^{(n+1)}\right) p\left(\boldsymbol{\omega}^{(n+1)} | \boldsymbol{\gamma}^{(n+1)}\right) p\left(\boldsymbol{\gamma}^{(n+1)} | \boldsymbol{\mu}^{(n+1)}\right) \hat{p}\left(\boldsymbol{\mu}^{(n+1)}\right), \end{aligned} \tag{4.48}$$

where Z is the normalisation constant.

Since components of the vectors $\boldsymbol{\beta}^{(n+1)}$ and $\boldsymbol{\omega}^{(n+1)}$ are conditionally independent, the terms $p\left(\boldsymbol{\beta}^{(n+1)} | \boldsymbol{\omega}^{(n+1)}\right)$ and $p\left(\boldsymbol{\omega}^{(n+1)} | \boldsymbol{\gamma}^{(n+1)}\right)$ are further factorised:

$$\begin{aligned} & p\left(\boldsymbol{\beta}^{(n+1)}, \boldsymbol{\omega}^{(n+1)}, \boldsymbol{\gamma}^{(n+1)}, \boldsymbol{\mu}^{(n+1)} | \mathbf{y}^{(1)\ldots(n+1)}\right) \\ =& \frac{1}{Z} p\left(\mathbf{y}^{(n+1)} | \boldsymbol{\beta}^{(n+1)}\right) \left[\prod_{d=1}^{D} p(\beta_d^{(n+1)} | \omega_d^{(n+1)}) p(\omega_d^{(n+1)} | \gamma_d^{(n+1)})\right] \\ & \times p\left(\boldsymbol{\gamma}^{(n+1)} | \boldsymbol{\mu}^{(n+1)}\right) \hat{p}\left(\boldsymbol{\mu}^{(n+1)}\right). \end{aligned} \tag{4.49}$$

The resulting formula for update (4.49) is the same as the posterior distribution (4.22) with the only exception in the term related to $\boldsymbol{\mu}^{(n+1)}$. The approximation of this posterior is proposed in Section 4.4. The algorithm is only required to be adjusted for the new factor $\hat{p}(\boldsymbol{\mu}^{(n+1)})$.

The factor $\hat{p}(\boldsymbol{\mu}^{(n+1)})$ is a Gaussian distribution, i.e. it is from the exponential family already and it only depends on a single latent variable, therefore this factor should not be updated in the EP iterations. The information from this factor will be passed through the general approximating distribution q to the other factors.

In the EP algorithm used for inference of the updated distribution (4.49) the distribution for $\boldsymbol{\mu}^{(n)}$ is approximated with the Gaussian distribution for any n. Therefore, the identity (4.46) is true for any n and the whole procedure can be applied for all timestamps.

4.5.3 Minibatch Filtering

The developed Bayesian filtering procedure can be easily extended to the case of inferring minibatches for timestamps $[n+1:n+M]$, where M is the size of a minibatch:

$$p(\boldsymbol{\beta}^{(n+1)\ldots(n+M)}, \boldsymbol{\omega}^{(n+1)\ldots(n+M)}, \boldsymbol{\gamma}^{(n+1)\ldots(n+M)}, \boldsymbol{\mu}^{(n+1)\ldots(n+M)} | \mathbf{y}^{(1)\ldots(n+M)}), \tag{4.50}$$

rather than for the next timestamp $n+1$ only as in (4.49).

Indeed, due to conditional independence marginalisation (4.42) also comes down to the integral (4.46) similar to (4.44). And the update step can also be performed by the EP algorithm with the only difference that it should be applied for M timestamps rather than one.

4.5.4 Implementation Details

There are no theoretical guarantees of EP convergence. However, it can be achieved using *damping* (Minka and Lafferty 2002): during step 4 of the factor refinement procedure in Section 4.4.1 the factor is updated as $q_a^{\text{damp}} = (q_a^{\text{new}})^{\eta}(q_a^{\text{old}})^{1-\eta}$, where q_a^{old} is the value of the factor from the previous iteration, q_a^{new} is the updated value of the factor, $\eta \in (0, 1]$ is the damping coefficient. It is exponentially decreased as $\eta = \eta^{\text{old}}\xi$ after each iteration, where $\xi \in (0, 1]$ is the decay parameter that governs the speed of exponential decrease and η^{old} is the value of the damping coefficient from the previous iteration.

It is also known that during the EP updates negative variances can appear (Hernández-Lobato, Hernández-Lobato, et al. 2015). In this case, negative variances are replaced with a large value representing $+\infty$.

4.6 Experiments

This section presents validation and evaluation results for the proposed algorithms. The performance of these two-level GP algorithms is compared with:

- the spatio-temporal spike and slab model with a one-level GP prior and its modification with common precision approximation (Andersen et al. 2017);

- the popular alternating direction method of multipliers (ADMM) method (Boyd et al.

2011), which is a convex optimisation method used here for the lasso problem (Tibshirani 1996);

- the spatio-temporal sparse Bayesian learning (STSBL) algorithm (Zhang, Jung, et al. 2014).

For quantitative comparison, the following measures are used:

NMSE (normalised mean square error). For a batch of data $\{\boldsymbol{\beta}^{(n)}\}_{n=1}^{N}$ and estimates $\{\widehat{\boldsymbol{\beta}}^{(n)}\}_{n=1}^{N}$, NMSE is computed as

$$\text{NMSE} = \frac{1}{N}\sum_{n=1}^{N}\sqrt{\frac{\sum_{d=1}^{D}\left(\widehat{\beta}_d^{(n)} - \beta_d^{(n)}\right)^2}{\sum_{d=1}^{D}\left(\beta_d^{(n)}\right)^2}}. \tag{4.51}$$

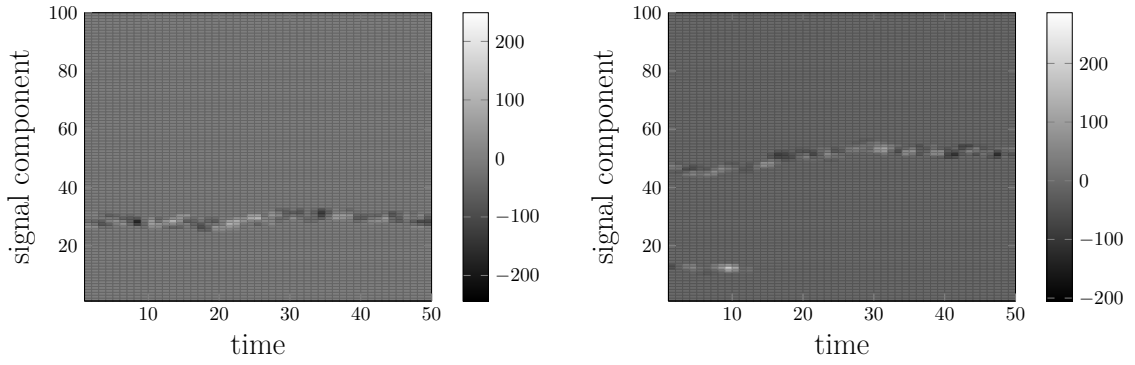
F measure. In sparse coding it is also important to obtain the correct locations of spikes (i.e zeros) and slabs (i.e. non-zeros) in the estimates. The problem is therefore viewed as a skewed two-class classification problem where the number of spikes is higher than the number of slabs. F-measure (Murphy 2012) is used to evaluate the accuracy of such problems. It is defined as the harmonic mean of precision and recall

$$\text{F-measure} = 2\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}, \tag{4.52}$$

where precision is the fraction of estimated slab locations that are correct, recall is the fraction of true slab locations among all predicted slab locations.

The NMSE shows the normalised error of signal reconstruction, with 0 corresponding to an ideal match. The F-measure shows how well slab locations are restored. An F-measure equal to 1 means that the true and estimated signals coincide, whilst 0 corresponds to lack of similarity between them. Arguably, for the sparse regression problem, the NMSE is less meaningful than the F-measure (Xin et al. 2016).

Both two-level and one-level GP algorithms are iterated until convergence, which is measured by difference in the estimate of the signal $\widehat{\mathbf{B}}$ at the current and previous iterations.



(a) Data (b) Data

Figure 4.5: Examples of the true signal $\mathbf{B}$ for the synthetic data. In each example two groups of slabs generated at $n = 1$ evolve in time until $n = 50$.

4.6.1 Synthetic Data

In this experiment the algorithm performance is studied on synthetic data with known true values of the signal $\mathbf{B}$ and slab locations $\mathbf{\Omega}$. The synthetic data represents the signals that have slowly evolving in time groups of non-zero elements. To create a spatio-temporal structure of slabs at the first timestamp $n = 1$ two groups of slab locations are generated with Poisson-distributed sizes for the signal $\boldsymbol{\beta}^{(n)}$ of dimensionality $D = 100$. Then, from $n = 2$ to $N = 50$, these groups randomly evolve: each border of each group can go up, down, or stay at the same location with such probabilities that in average the sparsity level remains 95%. In such way locations of the slab groups are generated. The values of non-zero elements of the signal are then drawn from the distribution $\mathcal{N}(0, 10^4)$. This procedure is repeated 10 times to generate 10 data samples. The examples of generated $\mathbf{B}$ are shown in Figure 4.5.

The elements of the design matrix $\mathbf{X}$ are generated as i.i.d. samples from the standard Gaussian. For each of the data samples, observations $\mathbf{Y} = \mathbf{XB}$ of different length K are generated. The value K/N is referred as an undersampling ratio. It changes from 10% to 55%.

The algorithms are evaluated in terms of average F-measure, NMSE and time[2] (Figure 4.6)

[2]Time is evaluated with 4.2GHz Intel Core i7 CPU and 16GB RAM.

on this data. On the interval between 10% and 20% of the undersampling ratio both inference methods for the two-level GP model and full EP inference for the one-level GP model show competitive results in terms of the accuracy metrics while outperforming the other methods. On the interval between 20% and 30% of the undersampling ratio the inference methods for the one- and two-level GP models are already able to perfectly reconstruct the sparse signal while both ADMM and STSBL show less accurate results. STSBL achieves the perfect reconstruction starting from the undersampling ratio 30% and ADMM achieves these results starting from the undersampling ratio 50%.

In the proposed EP algorithm for the two-level GP model (Section 4.3), the complexity of each iteration is $\mathcal{O}(D^3N)$, as matrices of size $D \times N$ are inverted for each timestamp to compute cavity distributions for the factors u and r. In the proposed online inference algorithm (Section 4.5), first the offline version is trained on size N_{init}. Then, when new data of size M is available, the previous results are used as prior and the complexity of update is $\mathcal{O}(D^3M)$, while in the offline version it is $\mathcal{O}(D^3(N_{\text{init}} + M))$.

On average, the proposed two-level GP algorithm requires similar to the full one-level GP algorithm number of iterations for convergence: approximately 30 iterations on the interval between 10% and 20% of the undersampling ratio, 15 iterations on the interval between 20% and 30%, and less than 10 iterations for the higher undersampling ratios. The approximate inference algorithm for the one-level GP model takes slightly more iterations to converge.

In the one-level GP algorithm (Andersen et al. 2017) the complexity of one iteration is $\mathcal{O}(D^3N^3)$. This is related to inversion of the full spatio-temporal covariance matrix. It is addressed with low rank and common precision approximations (Andersen et al. 2017), which reduce both the computational complexity and the quality of the results. The L-rank approximation, where L is a parameter of the algorithm, reduces the computational complexity to $\mathcal{O}(D^2LN)$ and the common precision approximation reduces it to $\mathcal{O}(D^2N + N^2D)$.

In terms of the computational time the full EP inference for the one-level GP model is the slowest method. The approximated inference for the one-level GP model significantly improve its performance in terms of the computational time while also cause loss in accuracy. The ADMM method shows similar results to the approximated one-level GP model in terms of the computational time, but has even bigger loss in terms of both accuracy measures.

The STSBL takes slightly more time for the lower values of the undersampling ratio, which helps it to achieve better results than the ADMM method in terms of the accuracy measures. The proposed offline and online inference methods for the two-level GP method demonstrate a satisfactory trade-off between computational time and accuracy. They obtain competitive results in terms of accuracy measures as the full EP inference for the one-level GP model while require significantly less computational time. In terms of computational time the proposed method demonstrates competitive results with the STSBL method.

The proposed online inference method for the two-level GP model allows to save computational time while preserving the accuracy of the recovered signal. Note that the developed inference methods for the two-level GP model outperform competitors in the lowest undersampling ratio interval, i.e. they require less measurements to get the same quality as other algorithms.

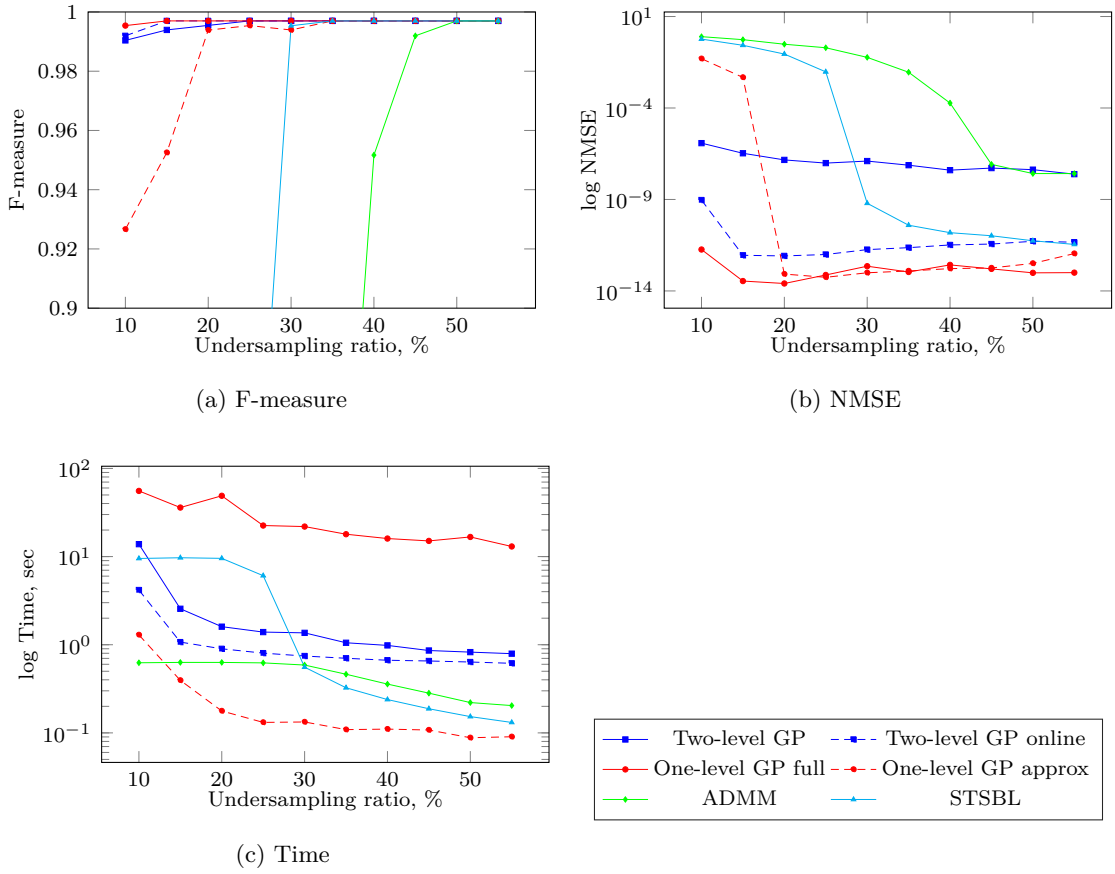
4.6.2 Real Data: Moving Object Detection in Video

The considered methods for sparse regression are compared on the problem of object detection in video sequences. The Convoy dataset (Warnell et al. 2015) is used where a background frame is subtracted from each video frame (Section 3.3). As moving objects take only part of a frame the considered signal of the subtracted video frames is sparse. Moreover, objects are represented as clusters of pixels, which evolve in time. Therefore, the background subtraction application fully satisfies the proposed spatio-temporal structured model assumptions.

The frames with subtracted background are resized to 32×32 pixels and reshaped as vectors $\boldsymbol{\beta}^{(n)} \in \mathbb{R}^D$, $D = 1024$. The number of frames in the dataset is $N = 260$. The sparse observations are obtained as $\mathbf{Y} = \mathbf{XB}$, where $\mathbf{X} \in \mathbb{R}^{K \times D}$ is the matrix with i.i.d. Gaussian elements. 10 different random design matrices $\mathbf{X}$ are used to generate 10 data samples. The number of observations K is chosen such that the undersampling ratio K/N changes from 10% to 55%.

For this problem the full EP inference for the one-level GP model is infeasible due to its memory requirements, therefore only the common precision approximated inference for the one-level GP model is considered.

The average F-measure and NMSE obtained by all the algorithms on the Convoy data are presented in Figure 4.7. The proposed algorithm shows the best results for the undersampling



(a) F-measure

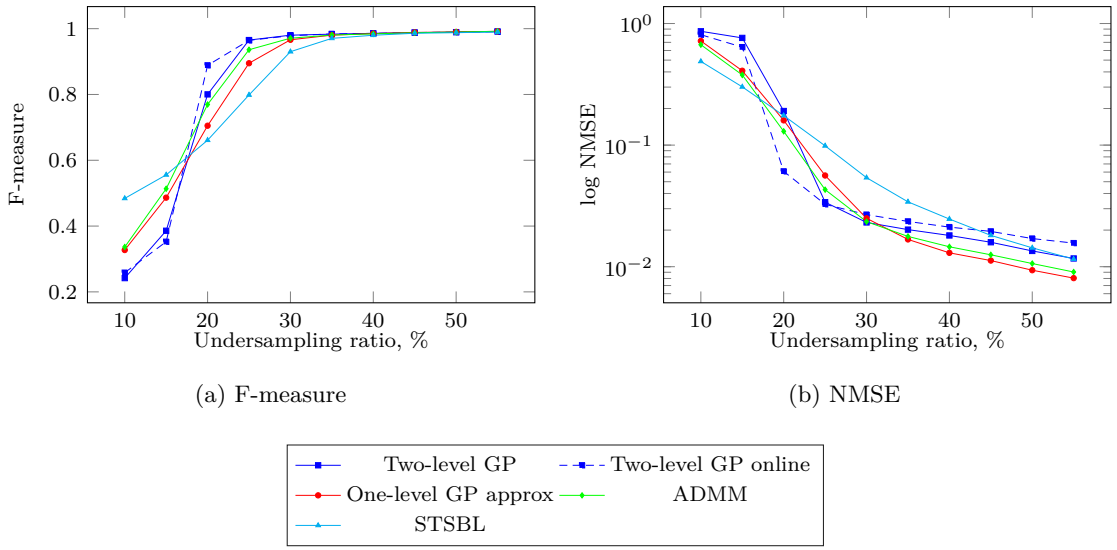
(b) NMSE

(c) Time

Figure 4.6: Performance of the algorithms on the synthetic data. Note that the NMSE plots have logarithmic scale of y-axis. As the convergence criteria is $\frac{||\widehat{\mathbf{B}}^{\text{new}} - \widehat{\mathbf{B}}^{\text{old}}||_\infty}{||\widehat{\mathbf{B}}^{\text{old}}||_\infty} < 10^{-3}$, values below 10^{-3} are less significant. The proposed algorithms referred as two-level GP and two-level GP online outperform others in the $10 - 20\%$ interval, where the number of observations is the lowest.

ratio $20 - 30\%$. For larger values of the undersampling ratio all the algorithms provide close almost ideal results of reconstruction.

Figure 4.8 presents the reconstructed sample frame from the Convoy data. For all the algorithms, the reconstruction results are provided for the undersampling ratio 10%, where the proposed algorithms slightly underperform the competitors in terms of the quality metrics, for the undersampling ratio 20%, where the proposed algorithm outperforms the competitors both in terms of NMSE and the F-measure, and for the undersampling ratio



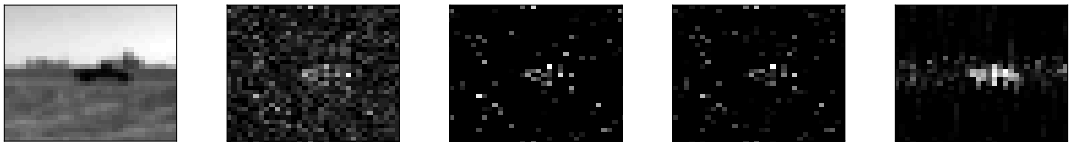
(a) F-measure

(b) NMSE

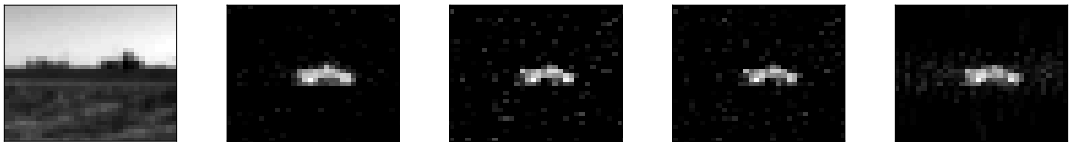
Figure 4.7: Performance of the algorithms on the Convoy data. The proposed algorithms referred as two-level GP and two-level GP online outperform the others in the $20-30\%$ interval. On the interval $10-15\%$ all methods cannot reconstruct the true signal. The NMSE plot shows that the proposed algorithms underperform the competitors for the values higher than 30%, but the visual difference in performance becomes insignificant that is demonstrated in Figure 4.8.

40%, where the proposed algorithms show a little higher NMSE. It is clearly seen that for the undersampling ratio 10% the difference in the quality metrics is insignificant since none of the methods is able to reconstruct the signal. The STSBL represents an exceptional example but still the frame reconstructed by this method contains considerable amount of noise. For the undersampling ratio 20% the proposed method provides the clear reconstructed frame in contrast to the reconstructed frames by all the competitors that are more noisy. Meanwhile, for the undersampling ratio 40% the difference between reconstruction results by all four algorithms is not remarkable.

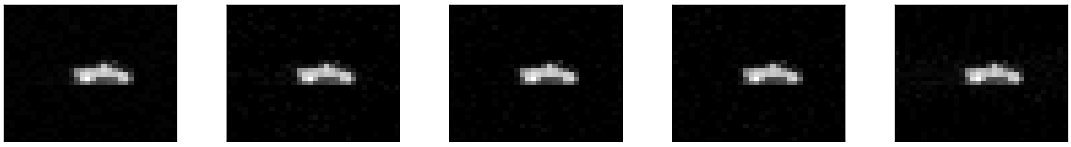
Note that similar to the synthetic data experiment the proposed algorithms obtain the best results for the lowest undersampling ratio values where the reconstruction is reasonable, i.e. they require a less number of observations.



(a) Original frame (b) Two-level GP 10% (c) One-level GP 10% (d) ADMM 10% (e) STSBL 10%



(f) Background frame (g) Two-level GP 20% (h) One-level GP 20% (i) ADMM 20% (j) STSBL 20%



(k) Reference object detection (l) Two-level GP 40% (m) One-level GP 40% (n) ADMM 40% (o) STSBL 40%

Figure 4.8: Sample frame with reconstruction results from sparse observations for the Convoy data. (a), (f): the original and static background non-compressed frames; (k): object detection results based on non-compressed frame difference (static background frame is subtracted from the original frame); (b), (g), (l): reconstruction of compressed object detection results based on the proposed online two-level GP method; (c), (h), (m): reconstruction of the compressed object detection results based on the one-level GP method; (d), (i), (n): reconstruction of the compressed object detection results based on the ADMM method; (e), (j), (o): reconstruction of the compressed object detection results based on the STSBL method. (b), (c), (d), and (e) show the results for the undersampling rate 10%, where all the algorithms fail to reconstruct the true signal. (g), (h), (i), and (j) show the reconstruction for the undersampling rate 20%, where the difference in performance between the algorithms is visible. While for the undersampling rate 40% ((l), (m), (n), and (o)) reconstruction results are indistinguishable in quality.

4.6.3 Real Data: EEG Source Localisation

The third experiment is devoted to the EEG source localisation problem.

Electrical activity inside the brain creates electromagnetic field at the head surface. The goal of the non-invasive EEG source localisation is to find 3D locations of dipoles such that their electromagnetic field coincides with the field measured by electrodes on the human head cortex. This is called electromagnetic source imaging. It is divided into two problems: forward problem, which is the evaluation of the potentials and magnetic fileds for activity dipoles, and inverse problem, that is the localisation of sources based on measurements.

The electromagnetic source imaging is important for localisation of active areas in human-brain interfaces and treatment of neurological disorders (Arvaneh et al. 2011; Baillet, Mosher, et al. 2001; Jatoi et al. 2014; Xu et al. 2018). This problem is ill-posed in the sense that there exist an infinite number of possible active areas inside the brain that could produce the same field on the head cortex. To regularise the problem, activity source locations are assumed to be spatially grouped and temporally evolve, similar to Baillet and Garnero (1997). Similar idea applies to the MEG source localisation (Solin et al. 2016).

The electromagnetic field on head surface can be evaluated with integral equations (Geselowitz 1967). The boundary element method (Akalin-Acar and Gençer 2004) is a popular approach for numerically solving these equations. It allows to compute the potentials at the discretised head surface with the the lead field matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$, that appears in the approximate solution of integral equations

$$\mathbf{y}^{(n)} = \mathbf{X}\boldsymbol{\beta}^{(n)} + \boldsymbol{\varepsilon}^{(n)}, \quad \forall n \in [1, \dots, N], \tag{4.53}$$

where $\mathbf{y}^{(n)} \in \mathbb{R}^K$ is the vector containing observations of potential differences taken from K electrodes placed on a human head cortex, $\boldsymbol{\beta}^{(n)} \in \mathbb{R}^D$ is the current density of dipole activation on the grid voxels inside the brain.

In this experiment, observations are taken from $K = 69$ electrodes, or *channels*, corresponding to the grid of 272 potential dipole activations. As the 3D locations of dipoles are used, the dimensionality of the grid voxels $\boldsymbol{\beta}^{(n)}$ is $D = 3 \times 272$, and the vector is flattened as

$$\boldsymbol{\beta}^{(n)} = \left[\beta_{1x}^{(n)}, \beta_{1y}^{(n)}, \beta_{1z}^{(n)}, \beta_{2x}^{(n)}, \beta_{2y}^{(n)}, \beta_{2z}^{(n)}, \dots, \beta_{\frac{D}{3}z}^{(n)}\right]^\top. \tag{4.54}$$

For each grid voxel d inside the brain with location coordinates $loc(d) = (\beta_d^{(n)}, \beta_d^{(n)}, \beta_d^{(n)})$ the corresponding dipole moments $(\beta_{dx}^{(n)}, \beta_{dy}^{(n)}, \beta_{dz}^{(n)})$ along the 3D axis are considered.

The following covariance function is used, as promotes close values for collinear dipole moments corresponding to close grid positions:

$$\mathbf{K}(i,j) = \alpha_{\mathbf{K}} \exp\left(-\frac{d(i,j)^2}{2\ell_{\mathbf{K}}^2}\right), \quad \mathbf{K} \in \{\boldsymbol{\Sigma}_0, \mathbf{W}\}, \tag{4.55}$$

where the distance is computed as

$$d(i,j) = \begin{cases} \infty, \text{if axis for dipole moments } i,\ j \text{ are different} \\ ||loc(i) - loc(j)||_2^2, \text{otherwise.} \end{cases} \tag{4.56}$$

Hyperparameters are selected so that the sampled potential differences have the similar behaviour as the provided data.

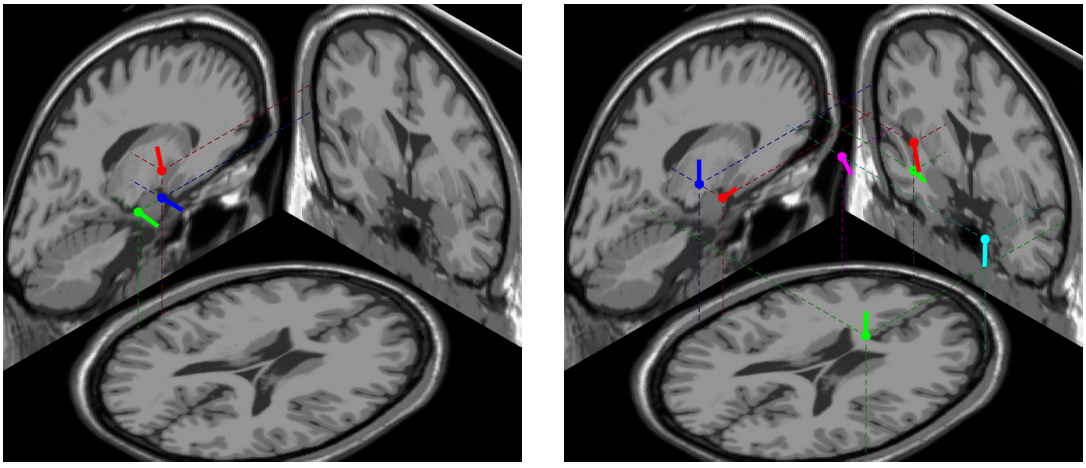
The data and lead field matrix for the experiments is processed with EEGLAB (Delorme and Makeig 2004). The data provided in EEGLAB is used for the source localisation problem with annotated events.

Figure 4.9 presents located dipoles by the proposed method for the fourth event at two given time moments. The first time moment is taken right after the event happened and there is no response to it in the brain activity yet. The second time moment is chosen when the response is detected. Figure 4.10 shows the comparison of measured and restored potential differences by the proposed algorithm.

The true density $\mathbf{B}$ is unknown for the EEG source localisation problem, therefore, NMSE between the observations $\mathbf{y}^{(n)}$ and reconstructed $\mathbf{X}\widehat{\boldsymbol{\beta}}^{(n)}$ is used for the quantitative comparison in this experiment. The obtained results for all the algorithms around the time of the brain response are presented in Figure 4.11. The proposed two-level GP algorithm shows the best results among the competitors. Note that in this experiment the undersampling ratio is approximately 8%, which confirms that the proposed method is able to provide better results for lower values of the undersampling ratio.

4.6.4 Parameters Selection

For the proposed algorithms and for the one-level GP the parameters η and ξ are grid optimised to make the comparison fair. The prior shape hyperparameters ℓ_Σ, ℓ_W, α_Σ, α_W and variances σ_x^2 and σ^2 are specified so that sampled data has the same form as training data. ADMM and STSBL use the default values of parameters. The selected



(a) Located dipole moments 1 ms after the event (b) Located dipole moments 170 ms after the event

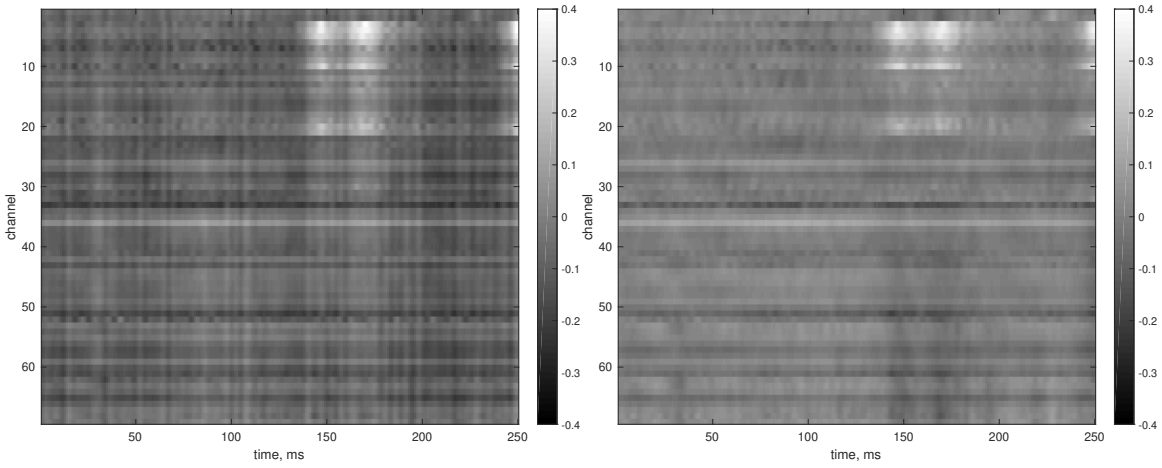
Figure 4.9: Located dipoles by the proposed two-level GP method for the EEG source localisation problem. There is no brain response immediately after the event and (a) demonstrates reconstructed brain active area that remains active during the whole period and it is not related to the event. While (b) shows the reconstructed active area when the brain response to the event is detected.

hyperparameter values for the proposed algorithms for all datasets are presented in Table 4.1 for the reproducibility of the experiments.

4.7 Conclusions

This chapter proposes a new hierarchical Gaussian process model of spatio-temporal structure representation with complex temporal evolution in sparse Bayesian inference methods. This is achieved using the flexible hierarchical GP prior for the spike and slab model, where spatial and temporal structural dependencies are encoded by different levels of the prior. Offline and online methods are developed for posterior inference for this model.

The introduced model can be applied to different areas such as compressive sensing and EEG source localisation. The results show the superiority of the proposed method in comparison with the non-hierarchical GP method, the alternating direction method of multipliers and the spatio-temporal sparse Bayesian learning method. The developed algorithms demonstrate better performance both in terms of signal value reconstruction and



(a) Measured EEG (b) Reconstructed EEG, estimated as $\mathbf{X}\hat{\mathbf{B}}$

Figure 4.10: Reconstruction by the proposed two-level GP method of the EEG signal. As the true active dipole areas are not known, reconstruction quality is measured between the true observations and the simulated observations from the reconstructed dipoles. Reconstructed EEG signal has lower magnitude, potentially because noise has been taken into account, but it has a similar shape to the orinial signal.

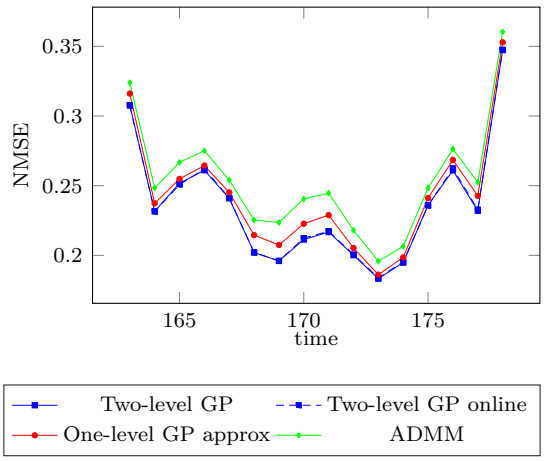


Figure 4.11: Results for NMSE between $\mathbf{y}^{(n)}$ and $\mathbf{X}\widehat{\boldsymbol{\beta}}^{(n)}$ during the brain response time. The proposed algorithm, referred as two-level GP has the lowest NMSE among the others.

localisation of non-zero signal components: within the low amount of measurements range it achieves around 15% improvement in terms of slab localisation quality.

Table 4.1: Two-level GP hyperparameters

Parameter	Synthetic	Convoy	EEG
σ_β^2	10^4	160	$4 * 10^5$
σ^2	10^{-4}	4	10^{-3}
η	0.999	0.99	0.9
ξ	0.9999	0.999	0.8
ℓ_W	15	15	22.17
ℓ_Σ	10	10	0.2217
α_W	10	10	10^{-2}
α_Σ	10	10	0.05

In this chapter and Chapter 3 weak and strong Bayesian models for sparsity have been considered, which can be viewed as Bayesian versions of the penalised sparse regression problem. Another approach is to achieve sparsity with neural networks, which leads to the potential Bayesian neural networks for sparsity. This concept is presented in the Chapter 5.

Chapter 5

UNCERTAINTY PROPAGATION IN SPARSE BAYESIAN NEURAL NETWORKS

In previous chapters, several new properties of weak and strong sparse Bayesian models are presented. In this chapter, a novel Bayesian approach based on reformulation of iterative frequentist solutions is proposed. It uses deep neural networks (DNNs) to deal with the sparsity problem: first, the models are trained on a large sample of training data, then they can make fast predictions for new data. However, common neural network models loose the properties of Bayesian models, such as uncertainty estimation for parameter learning and predictions. In this chapter, the Bayesian neural network (BNNs) is proposed for the sparsity problem, which maintains the advantages of both approaches: uncertainty estimation and fast predictions.

The rest of the chapter is organised as following: first, the introduction for the Bayesian neural networks is given in Section 5.1. The review of neural networks for sparse coding is given in Section 5.2 and a novel Bayesian neural network is presented in Section 5.3. Then, uncertainty propagation is described in Section 5.4 and probabilistic backpropagation in Section 5.5. After that, the experimental results of the algorithm are shown in Section 5.6 and the summary is presented in Section 5.7.

The materials of this chapter were published as

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2018b). "Uncertainty propagation in neural networks for sparse coding". In: *Proceedings of the Third Workshop on Bayesian Deep Learning (NeurIPS)*. URL: `http://bayesiandeeplearning.org/2018/papers/47.pdf`

- Danil Kuzin, Olga Isupova, and Lyudmila Mihaylova (2019). "Bayesian neural networks for sparse coding". Accepted at IEEE International Conference on Acoustics,

Speech and Signal Processing (ICASSP)

5.1 Bayesian Neural Networks

Consider the nonlinear regression problem

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\varepsilon}, \tag{5.1}$$

where the exact mapping $f(\cdot)$ is unknown, and a set of training data samples $\mathcal{D}$ is available to restore $f(\cdot)$.

Nowadays, a common approach to model the nonlinear problems is neural networks (LeCun, Bengio, et al. 2015). They approximate $f(\cdot)$ as a series of layers. Sequentially, an input to the network is transformed with linear and simple non-linear layers to obtain the approximation of $f(\cdot)$. Some of the layers have parameters $\boldsymbol{\Theta}$, that can be learned by optimisation of the marginal likelihood $p(\mathbf{x}|\boldsymbol{\Theta})$ on the training data. Usually, modern neural networks have large number of parameters that can lead to overfitting during the training procedure and overconfidence in estimates.

The Bayesian approach to neural networks attempts to solve the above problems. The prior distributions $p(\boldsymbol{\Theta})$ can be imposed on the parameters and, then, based on the training data, the posterior distribution can be computed for weights and predictions. Due to large data volume and high dimensionality of parameter space, most of the approximate Bayesian inference methods become infeasible. Below, current ideas that extend Bayesian methods for neural networks are described.

5.1.1 Sampling Methods

The Bayesian approach for neural networks was initially considered by Neal (1994), with the Markov chain Monte Carlo methods used for inference. In further works, new sampling methods were proposed for neural networks, such as Langevin dynamics (Ahn et al. 2012; Welling and Teh 2011), No-U-Turn sampler (Hoffman and Gelman 2014), sampling with variational initialization (Hoffman 2017).

In Bayesian inference, sampling methods can possibly achieve the highest quality, but they require high computational resources.

5.1.2 Variational Inference

Variational inference is usually computationally cheaper than sampling methods, but it introduces bias related to variational approximation. Originally, variational inference for neural networks is considered by Graves (2011). The way of reducing variance in gradient estimator with reparametrisation trick is proposed by Kingma and Welling (2014) and Rezende et al. (2014). The dropout element originally proposed for the regularisation of neural networks (Srivastava et al. 2014), can be viewed as a way to introduce uncertainty for the network and interpreted with variational inference (Wang and Manning 2013). Combined with log-uniform prior for the weights it leads to the Bayesian formulation of the network (Kingma, Salimans, et al. 2015).

5.1.3 Expectation Propagation

The idea of gradient backpropagation for frequentist neural networks was extended into stochastic backpropagation (Hernández-Lobato and Adams 2015; Rezende et al. 2014). It infers marginal posterior distributions, by propagating the simple approximated distributions through the network. The details of it are presented in Section 5.5.

5.2 Neural Networks for Sparse Coding

Consider the sparse linear regression problem (2.1). In Section 2.1.1, the ISTA algorithm is described, that iteratively updates the estimate of the coefficient vector $\widehat{\boldsymbol{\beta}}$ with linear and soft-thresholding functions. Its parameters are the matrices $\mathbf{W}$, $\mathbf{S}$.

The learned ISTA (LISTA), (Gregor and LeCun 2010) algorithm learns the values of matrices $\mathbf{W}$, $\mathbf{S}$ based on set of pairs $\{\mathbf{Y}, \mathbf{B}\} = \{\mathbf{y}^{(n)}, \boldsymbol{\beta}^{(n)}\}_{n=1}^{N}$, where N is the number of these pairs. To achieve this, the ISTA algorithm is limited with the fixed amount of iterations, L and interpreted as a recurrent neural network. Overall, Algorithm 1 describes the scheme of predicting a coefficient vector $\boldsymbol{\beta}^{(n)}$ for an observation $\mathbf{y}^{(n)}$.

Matrices $\mathbf{W}$, $\mathbf{S}$ are the parameters that are initialised as in ISTA and then updated with the backpropagation algorithm. Vectors $\mathbf{c}_l$, $\mathbf{b}$ are intermediate vectors that describe forward propagation.

Algorithm 1 LISTA forward propagation

Require: observation $\mathbf{y}$, current weights $\mathbf{W}, \mathbf{S}$, number of layers L

1: *Initialisation.* Dense layer $\mathbf{b} \leftarrow \mathbf{W}\mathbf{y}$
2: *Initialisation.* Soft-thresholding nonlinearity $\widehat{\boldsymbol{\beta}}_0 \leftarrow h_\lambda(\mathbf{b})$
3: **for** $l = 1$ **to** L **do**
4: Dense layer $\mathbf{c}_l \leftarrow \mathbf{b} + \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$
5: Soft-thresholding nonlinearity $\widehat{\boldsymbol{\beta}}_l \leftarrow h_\lambda(\mathbf{c}_l)$
6: **end for**
7: **return** $\widehat{\boldsymbol{\beta}} \leftarrow \widehat{\boldsymbol{\beta}}_L$

5.3 Bayesian Neural Network for Sparse Coding

This section presents the proposed Bayesian neural network for sparse coding, that is based on the LISTA network. To formulate the Bayesian version of LISTA, the prior distributions are imposed on the unknown weights

$$p(\mathbf{W}) = \prod_{d=1}^{D}\prod_{k=1}^{K}\mathcal{N}(w_{dk}|0,\eta^{-1}), \tag{5.2a}$$

$$p(\mathbf{S}) = \prod_{d'=1}^{D}\prod_{d''=1}^{D}\mathcal{N}(s_{d'd''}|0,\eta^{-1}), \tag{5.2b}$$

where w_{dk} is a component of the matrix $\mathbf{W}$, $s_{d'd''}$ is a component of the matrix $\mathbf{S}$, η is the precision of the Gaussian distribution. To introduce the uncertainty of observations, assume that the observations have the Gaussian distribution with the precision γ, centred at the output of the Bayesian LISTA neural network $f(\mathbf{y}^{(n)}; \mathbf{W}, \mathbf{S}, \lambda)$. The likelihood of $\mathbf{B}$ is defined as

$$p(\mathbf{B}|\mathbf{Y}, \mathbf{W}, \mathbf{S}, \gamma, \lambda) = \prod_{n=1}^{N}\prod_{d=1}^{D}\mathcal{N}(\beta_d^{(n)}; [f(\mathbf{y}^{(\mathbf{n})}; \mathbf{W}, \mathbf{S}, \lambda)]_d, \gamma^{-1}), \tag{5.3}$$

where $[\cdot]_d$ denotes the d-th component of a vector. The prior of the introduced Gaussian precisions are set to the gamma distribution with parameters $a^{\cdot}$ and $b^{\cdot}$:

$$p(\gamma) = \mathrm{Gam}\left(\gamma; a^\gamma, b^\gamma\right), \tag{5.4a}$$

$$p(\eta) = \mathrm{Gam}\left(\eta; a^\eta, b^\eta\right). \tag{5.4b}$$

The posterior distribution of unknown parameters is then

$$p(\mathbf{W}, \mathbf{S}, \gamma, \eta | \mathbf{B}, \mathbf{Y}, \lambda) = \frac{p(\mathbf{B}|\mathbf{Y}, \mathbf{W}, \mathbf{S}, \gamma, \lambda) p(\mathbf{W}|\eta) p(\mathbf{S}|\eta) p(\eta) p(\gamma)}{p(\mathbf{B}|\mathbf{Y}, \lambda)}. \tag{5.5}$$

The shrinkage parameter λ is a hyperparameter of the model.

5.4 Uncertainty Propagation through Soft-Thresholding

For every observation $\mathbf{y}^{(n)}$ at every layer of the Bayesian LISTA, the current approximation $\widehat{\boldsymbol{\beta}}_{l-1}$ is assumed to have the spike and slab distribution with following parameters: $\boldsymbol{\omega}$ is a probability of a spike, $\mathbf{m}$ is a mean of a slab Gaussian distribution, and $\mathbf{v}$ is a variance of the slab distribution

$$[\widehat{\boldsymbol{\beta}}_{l-1}]_d \sim \omega_d \delta_0 \left([\widehat{\boldsymbol{\beta}}_{l-1}]_d \right) + (1 - \omega_d) \mathcal{N}(m_d, v_d), \tag{5.6}$$

where ω_d, m_d, and v_d are the components of vectors $\boldsymbol{\omega}$, $\mathbf{m}$, and $\mathbf{v}$, respectively.

In this section, it is shown that the output $\widehat{\boldsymbol{\beta}}_l$ of the next layer can be approximated with a spike and slab distribution and, therefore, it maintains the same family of distributions. This leads to the proposed probabilistic backpropagation algorithm that is presented in Section 5.5. Further in this chapter the superscript (n) is ommited as the uncertainty propagation and the backpropagtion are performed sequentially and independently for each pair from the training dataset.

5.4.1 Initialisation Dense Layer

The first step in the Bayesian LISTA is initialisation of dense layer (step 1), the Bayesian version of Algorithm 1. The matrix $\mathbf{W}$ consists of Gaussian-distributed components and $\mathbf{y}$ is a deterministic vector.

Lemma 1 (Product of Gaussian matrix and deterministic vector)**.** *Let $\mathbf{W} \in \mathbb{R}^{D \times K}$ be a matrix of independent Gaussian-distributed random variables: $w_{dk} \sim \mathcal{N}(m^w_{dk}, v^w_{dk})$, and $\mathbf{y} \in \mathbb{R}^K$ be a deterministic vector. Then their product $\mathbf{W}\mathbf{y}$ is a vector $\mathbf{b} \in \mathbb{R}^D$ of random variables $b_d \sim \mathcal{N}(m^b_d, w^b_d)$, where*

$$m^b_d = \sum_{k=1}^{K} y_k m^w_{dk}, \tag{5.7a}$$

$$w^b_d = \sum_{k=1}^{K} y^2_k v^w_{dk}. \tag{5.7b}$$

Proof. The statement follows from the property that the family of normal distributions is closed under linear transformations. □

According to Lemma 1, **b** in (step 1) is a vector of Gaussian-distributed components.

5.4.2 Soft-Thresholding Nonlinearity

At the initialisation soft-thresholding step 2 of the Bayesian LISTA forward propagation, the Gaussian vector **b** is taken as an input of the soft-thresholding function.

When a Gaussian-distributed random variable $x \sim \mathcal{N}(x; m, v)$ is propagated through the soft-thresholding function: $x^* = h_\lambda(x)$ the probability mass of a resulting random variable x^* is split into two parts. The values of x from the interval $[-\lambda, \lambda]$ are converted to 0 by the soft-thresholding operator. Therefore, the probability mass of the original distribution that lies in $[-\lambda, \lambda]$ is squeezed into the probability of x^* being zero. The values of x from outside of the $[-\lambda, \lambda]$ interval are shifted towards 0. Therefore, the distribution of $x^* \neq 0$ represents the tails of the original Gaussian distribution.

Lemma 2 (Propagation of a Gaussian distribution through soft-thresholding)**.** *The distribution of x^* can be parametrised by the probability of being zero, ω^*, the mean m^* and the variance v^* of the truncated Gaussian distribution.*

Proof. The probability ω^* of a zero equals to the probability mass of the original distribution from the interval $[-\lambda, \lambda]$

$$\omega^* = \mathrm{P}(x^* = 0) = \mathrm{P}(x \in [-\lambda, \lambda]) = \Phi\left(\frac{\lambda - m}{\sqrt{v}}\right) - \Phi\left(\frac{-\lambda - m}{\sqrt{v}}\right). \tag{5.8}$$

where $\Phi(\cdot)$ is the standard Gaussian cumulative distribution function.

The soft-thresholding function shifts elements that are greater than λ or less than $-\lambda$ towards 0. Let $\psi(\cdot)$ denote the density of the soft-thresholded distribution on $x^* \neq 0$, $\phi(\cdot)$ denote the density of the original Gaussian distribution on x. Then the first moment of $x^* \neq 0$ is

$$m^* = \int_{-\infty}^{+\infty} x\psi(x)dx = \int_{-\infty}^{0} x\phi(x - \lambda)dx + \int_{0}^{+\infty} x\phi(x + \lambda)dx, \tag{5.9}$$

where

$$\int_{-\infty}^{0} x\phi(x-\lambda)dx = -\frac{\sqrt{v}}{\sqrt{2\pi}}e^{\frac{-(\lambda+m)^2}{2v}} + (\lambda+m)\Phi\left(-\frac{\lambda+m}{\sqrt{v}}\right)$$
$$\int_{0}^{+\infty} x\phi(x+\lambda)dx = \frac{\sqrt{v}}{\sqrt{2\pi}}e^{\frac{-(m-\lambda)^2}{2v}} + (m-\lambda)\left(1-\Phi\left(-\frac{\lambda-m}{\sqrt{v}}\right)\right).$$

The second moment of $x^* \neq 0$ is given as

$$s = \int_{-\infty}^{+\infty} x^2\psi(x)dx = \int_{-\infty}^{0} x^2\phi(x-\lambda)dx + \int_{0}^{+\infty} x^2\phi(x+\lambda)dx, \tag{5.10}$$

where

$$\int_{-\infty}^{0} x^2\phi(x-\lambda)dx = -\frac{\sqrt{v}}{\sqrt{2\pi}}(\lambda+m)e^{\frac{-(\lambda+m)^2}{2v}} + (\sigma^2+(\lambda+m)^2)\Phi\left(-\frac{\lambda+m}{\sqrt{v}}\right)$$
$$\int_{0}^{+\infty} x^2\phi(x+\lambda)dx = \frac{\sqrt{v}}{\sqrt{2\pi}}(m-\lambda)e^{\frac{-(m-\lambda)^2}{2v}} + (\sigma^2+(m-\lambda)^2)\left(1-\Phi\left(\frac{\lambda-m}{\sqrt{v}}\right)\right).$$

The resulting variance is then

$$v^* = s - (m^*)^2 \tag{5.11}$$

□

Based on the results of Lemma 2, the distribution of $\widehat{\boldsymbol{\beta}}_0$ from step 2 is approximated with a spike and slab distribution with parameters derived in the Lemma: spike probability ω is equal to ω^*, slab mean m and variance v are set to the corresponding parameters of the truncated Gaussian m^* and v^*.

5.4.3 Main Layers

The distributions of the inputs at the step 4 of the Bayesian LISTA for each l are: the vector $\mathbf{b}$ and matrix $\mathbf{S}$ that consist of Gaussian-distributed components and $\widehat{\boldsymbol{\beta}}_{l-1}$, which is a vector of the spike and slab random variables.

In order to determine the distribution of the output $\mathbf{c}_l$, the Lemmas about the spike and slab distribution are formulated.

Lemma 3 (Moments of a spike and slab distribution)**.** *Let a random variable ξ have a spike and slab distribution with probability of spike ω, slab mean m and slab variance v. Then its moments are*

$$\mathbb{E}\xi = (1-\omega)m \tag{5.12a}$$

$$\operatorname{Var}\xi = (1-\omega)(v+\omega m^2). \tag{5.12b}$$

Proof.

$$\begin{aligned}
\mathbb{E}\xi &= \int x\big(\omega\delta_0(x) + (1-\omega)\mathcal{N}(x; m, v)\big)dx \\
&= \omega\int x\delta_0(x)dx + (1-\omega)\int x\mathcal{N}(x; m, v)dx = (1-\omega)m \\
\mathbb{E}\xi^2 &= \int x^2\big(\omega\delta_0(x) + (1-\omega)\mathcal{N}(x; m, v)\big)dx = \\
&= \omega\int x^2\delta_0(x)dx + (1-\omega)\int x^2\mathcal{N}(x; m, v)dx = (1-\omega)(v+m^2) \\
\operatorname{Var}\xi &= \mathbb{E}\xi^2 - (\mathbb{E}\xi)^2 = (1-\omega)(v+\omega m^2).
\end{aligned}$$

□

Lemma 4 (Product of a Gaussian matrix and a spike and slab vector)**.** *Let* $\mathbf{S} \in \mathbb{R}^{D\times D}$ *be a matrix of independent Gaussian-distributed random variables:* $s_{d'd''} \sim \mathcal{N}(m^s_{d'd''}, v^s_{d'd''})$*, and* $\widehat{\boldsymbol{\beta}} \in \mathbb{R}^D$ *be a vector with spike-and-slab-distributed variables:* $\widehat{\beta}_d \sim \omega_d\delta_0 + (1-\omega_d)\mathcal{N}(m_d, v_d)$*. The components of the matrix* $\mathbf{S}$ *and the vector* $\widehat{\boldsymbol{\beta}}$ *are mutually independent. Let* $\mathbf{e} \in \mathbb{R}^D$ *denote the product* $\mathbf{S}\widehat{\boldsymbol{\beta}}$*. Then the marginal mean and variance of elements* e_d *of the vector* $\mathbf{e}$ *are:*

$$\mathbb{E}e_d = \sum_{d'=1}^{D} m^s_{dd'}(1-\omega_{d'})m_{d'}, \tag{5.13a}$$

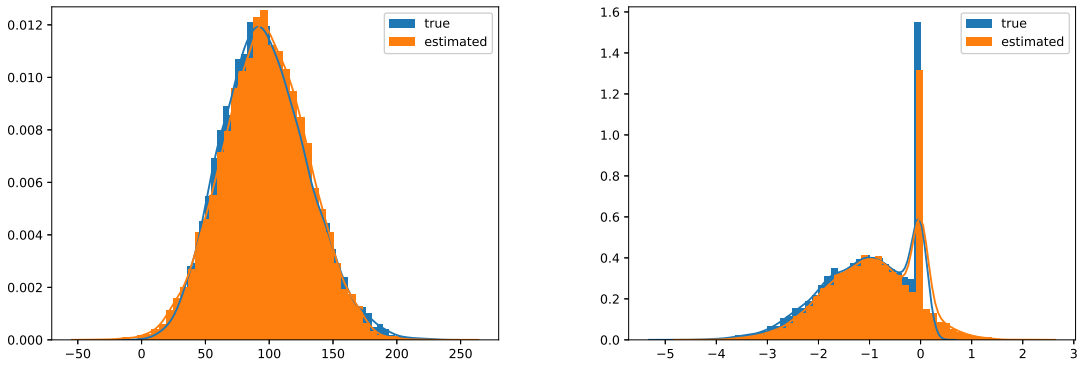
$$\operatorname{Var}e_d = \sum_{d'=1}^{D}[(m^s_{dd'})^2(1-\omega_{d'})^2 v_{d'} + (1-\omega_{d'})^2(m_{d'})^2 v^s_{dd'} + v^s_{dd'}(1-\omega_{d'})^2 v_{d'}]. \tag{5.13b}$$

Proof.

$$\begin{aligned}
\mathbb{E}e_d &= \sum_{d'=1}^{D}\mathbb{E}[s_{dd'}\widehat{\beta}_{d'}] = \sum_{d'=1}^{D} m^s_{dd'}\mathbb{E}\widehat{\beta}_{d'} \\
\operatorname{Var}e_d &= \sum_{d'=1}^{D}\operatorname{Var}[s_{dd'}\widehat{\beta}_{d'}] = \sum_{d'=1}^{D}[(m^s_{d'd''})^2\operatorname{Var}\widehat{\beta}_{d'} + (\mathbb{E}\widehat{\beta}_{d'})^2 v^s_{dd'} + \operatorname{Var}\widehat{\beta}_{d'} v^s_{dd'}].
\end{aligned}$$

where $\mathbb{E}\widehat{\beta}_{d'}$, $\operatorname{Var}\widehat{\beta}_{d'}$ are computed according to Lemma 3. □

Let $\mathbf{e}_l = \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$ at the step (4) of the Bayesian LISTA. The mutual independence of $\mathbf{S}$ and $\widehat{\boldsymbol{\beta}}_{l-1}$ is assumed. Then, according to the central limit theorem, $[\mathbf{e}_l]_d$ can be approximated as a Gaussian-distributed variable when D is sufficiently large. The parameters of the approximating Gaussian distribution are set to the marginal mean and variance given in Lemma 4. The quality of this approximation is discussed in Section 5.4.5.



(a) Approximation of the product of Gaussians

(b) Approximation of the propagation through soft-thresholding

Figure 5.1: Approximations of Bayesian LISTA

The output $\mathbf{c}_l$ at the step 4 is then represented as a sum of two Gaussian-distributed vectors: $\mathbf{b}$ and $\mathbf{e}_l$

Lemma 5 (Sum of Gaussian vectors)**.** *If* $\mathbf{b} \in \mathbb{R}^D$ *and* $\mathbf{e} \in \mathbb{R}^D$ *are both vectors of independent Gaussian-distributed random variables:* $b_d \sim \mathcal{N}(m_d^b, v_d^b)$, $e_d \sim \mathcal{N}(m_d^e, v_d^e)$ *then their sum* $\mathbf{c} = \mathbf{b} + \mathbf{e}$ *is a vector of independent Gaussian-distributed random variables* $c_d \sim \mathcal{N}(m_d^c, v_d^c)$ *with*

$$m_d^c = m_d^b + m_d^e, \tag{5.14a}$$

$$v_d^c = v_d^b + v_d^e. \tag{5.14b}$$

Proof. Based on properties of Gaussian distributions. □

Therefore, $\mathbf{c}_l$ is a vector of Gaussian-distributed components, which parameters can be found according to Lemma 5.

Then $\widehat{\boldsymbol{\beta}}_l$ at the step 5 of the Bayesian LISTA is the result of soft-thresholding of a Gaussian distribution, which is approximated with the spike and slab distribution, similar to the step 2.

5.4.4 Bayesian LISTA Forward Propagation

All steps of the Bayesian LISTA are formulated above. They provide distributions for outputs for all elements of the network. The main result of this section describes how the proposed uncertainty propagation works.

1. $\mathbf{b} = \mathbf{W}\mathbf{y}$ is the Gaussian distribution with parameters computed according to Lemma 1;

2. $\widehat{\boldsymbol{\beta}}_0 = h_\lambda(\mathbf{b})$ is approximated with the spike and slab distribution, which parameters are computed according to Lemma 2.

3. $\mathbf{e}_l = \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$ is approximated with the Gaussian distribution, which parameters are computed according to Lemma 4;

4. $\mathbf{c}_l = \mathbf{b} + \mathbf{e}_l$ is the Gaussian distribution with parameters computed according to Lemma 5;

5. $\widehat{\boldsymbol{\beta}}_l = h_\lambda(\mathbf{c}_l)$ is approximated with the spike and slab distribution, which parameters are computed according to Lemma 2.

5.4.5 Approximation Quality

In forward propagation of uncertainty two approximations are used. First, in Lemma 4 a Gaussian matrix is multiplied by a spike and slab vector and their product is approximated with the Gaussian distribution. Second, in Lemma 2 the result of soft-thresholding of a Gaussian vector is approximated with the spike and slab distribution.

Figure 5.1a demonstrates the comparison of the sampled distribution and approximating distribution for Lemma 4. For sampled distribution, 10000 values were sampled from the Gaussian matrix and the spike and slab vector and their product is computed, then one of the dimensionalities is plotted. The approximating distribution is computed according to Lemma 4.

Figure 5.1b demonstrates the comparison of the sampled distribution and approximating distribution for Lemma 2. For sampled distribution, 10000 values are sampled from the

Gaussian vector and propagated through soft-thresholding, then one of the dimensionalities is plotted. The approximating distribution is computed according to Lemma 2.

5.5 Backpropagation

The exact posterior (5.5) is approximated with a factorised distribution

$$q(\mathbf{W}, \mathbf{S}, \gamma, \eta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \mathcal{N}(w_{dk} | m_{dk}^{w}, v_{dk}^{w}) \prod_{d'=1}^{D} \prod_{d''=1}^{D} \mathcal{N}(s_{d'd''} | m_{d'd''}^{s}, v_{d'd''}^{s}) \times \mathrm{Gam}(\gamma; a^{\gamma}, b^{\gamma}) \mathrm{Gam}(\eta; a^{\eta}, b^{\eta}). \tag{5.15}$$

For Bayesian inference, the probabilistic backpropagation algorithm (Hernández-Lobato and Adams 2015) is expanded for computing parameter updates. It is based on assumed density filtering (ADF) and expectation propagation (EP) and allows to update parameters of the distributions based on the derivative of the logarithm of a normalisation constant. ADF iteratively incorporates factors from the true posterior p (5.5) into the factorised approximating distribution q (5.15), whereas in EP factors in the q are iteratively replaced by factors from p.

When a factor from p is incorporated into q, q as a function of Gaussian-distributed weights $\mathbf{W}$ and $\mathbf{S}$ has the following form:

$$q(a) = Z^{-1} f(a) \mathcal{N}(a; m, v), \tag{5.16}$$

where Z is a normalisation constant, $f(a)$ is an arbitrary function, $a \in \{w_{dk}, s_{d'd''}; \forall d, k, d', d''\}$.

According to Minka (2001a), new parameters of the Gaussian distribution for a are computed as

$$m^{\text{new}} = m + v \frac{\partial \log Z}{\partial m}, \tag{5.17}$$

$$v^{\text{new}} = v - v^2 \left[\left(\frac{\partial \log Z}{\partial m} \right)^2 - 2 \frac{\partial \log Z}{\partial v} \right]. \tag{5.18}$$

Therefore, to find new values of $\mathbf{W}$, $\mathbf{S}$, it is required to compute derivatives of the logarithm of Z when the factor of the true posterior p is incorporated in q.

5.5.1 Likelihood

The ADF approach is used to iteratively incorporate the likelihood factors (5.3) of the true posterior into the approximating distribution q. The normalisation constant of the

approximating distribution q with the likelihood term for the current data point can be computed as follows:

$$Z = \int \prod_{d=1}^{D} \left[\mathcal{N}(\beta_d | f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda), \gamma^{-1})\right] q(\mathbf{W}, \mathbf{S}, \gamma, \eta) \mathrm{d}\mathbf{W} \mathrm{d}\mathbf{S} \mathrm{d}\gamma \mathrm{d}\eta \tag{5.19}$$

Let $\mathbf{W}$, $\mathbf{S}$ be sampled from q. The output from the network $\widehat{\boldsymbol{\beta}} = f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda)$ is approximated with the spike and slab distribution with parameters $\boldsymbol{\omega}^{\widehat{\boldsymbol{\beta}}}$, $\mathbf{m}^{\widehat{\boldsymbol{\beta}}}$, and $\mathbf{v}^{\widehat{\boldsymbol{\beta}}}$. Then the normalisation constant is

$$\begin{aligned} Z \approx \int \mathrm{Gam}\left(\gamma; \alpha^{\gamma}, \beta^{\gamma}\right) \prod_{d=1}^{D} \Big[& \mathcal{N}\left(\beta_d; [\widehat{\boldsymbol{\beta}}]_d, \gamma^{-1}\right) \\ & \times \left(\omega_d^{\widehat{\boldsymbol{\beta}}} \delta_0\left([\widehat{\boldsymbol{\beta}}]_d\right) + \left(1 - \omega_d^{\widehat{\boldsymbol{\beta}}}\right) \mathcal{N}\left([\widehat{\boldsymbol{\beta}}]_d; m_d^{\widehat{\boldsymbol{\beta}}}, v_d^{\widehat{\boldsymbol{\beta}}}\right)\right)\Big] \mathrm{d}\widehat{\boldsymbol{\beta}} \mathrm{d}\gamma \end{aligned} \tag{5.20}$$

As it is discussed in section 2.1.2, the mixture of Gaussian and inverse-gamma distribution is a Student's t-distribution. The parameters can be computer as

$$\int \mathrm{Gam}\left(\gamma; \alpha^{\gamma}, \beta^{\gamma}\right) \mathcal{N}\left(\beta_d; [\widehat{\boldsymbol{\beta}}]_d, \gamma^{-1}\right) d\gamma = \mathcal{T}\left(\beta_d; [\widehat{\boldsymbol{\beta}}]_d, \beta^{\gamma}/\alpha^{\gamma}, 2\alpha^{\gamma}\right). \tag{5.21}$$

The Student's t-distribution density can be parametrised in different ways, here the following parametrisation is used

$$\mathcal{T}(x; \mu, \beta, \nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)\sqrt{\pi\nu\beta}} \left(1 + \frac{(x-\mu)^2}{\nu\beta}\right)^{-\frac{\nu+1}{2}}, \tag{5.22}$$

where $\Gamma(\cdot)$ denotes the gamma function. Using this property, the normalisation constant is

$$\begin{aligned} Z \approx \prod_{d=1}^{D} \Big[& \omega_d^{\widehat{\boldsymbol{\beta}}} \int \mathcal{N}\left(\beta_d; 0, \gamma^{-1}\right) \mathrm{Gam}\left(\gamma; \alpha^{\gamma}, \beta^{\gamma}\right) d\gamma \\ & + \left(1 - \omega_d^{\widehat{\boldsymbol{\beta}}}\right) \int \mathcal{T}\left(\beta_d; [\widehat{\boldsymbol{\beta}}]_d, \beta^{\gamma}/\alpha^{\gamma}, 2\alpha^{\gamma}\right) \mathcal{N}\left([\widehat{\boldsymbol{\beta}}]_d; m_d^{\widehat{\boldsymbol{\beta}}}, v_d^{\widehat{\boldsymbol{\beta}}}\right) \mathrm{d}[\widehat{\boldsymbol{\beta}}]_d \Big] \end{aligned} \tag{5.23}$$

The Student's t-distribution density can be approximated with Gaussian distribution with the same mean and variance. The quality of such approximation is discussed by Hernández-Lobato and Adams (2015)

$$Z \approx \prod_{d=1}^{D} \left[\omega_d^{\widehat{\boldsymbol{\beta}}} \mathcal{T}\left(\beta_d; 0, \beta^{\gamma}/\alpha^{\gamma}, 2\alpha^{\gamma}\right) + \left(1 - \omega_d^{\widehat{\boldsymbol{\beta}}}\right) \mathcal{N}\left(\beta_d; m_d^{\widehat{\boldsymbol{\beta}}}, \beta^{\gamma}/(\alpha^{\gamma} - 1) + v_d^{\widehat{\boldsymbol{\beta}}}\right)\right]. \tag{5.24}$$

Parameters of the approximating posterior distribution are then updated with the derivatives of this normalisation constant using equations (5.17-5.18). The derivatives can be computed using the automatic differentiation frameworks, such as TensorFlow (Abadi et al. 2016).

5.5.2 Prior

Prior factors from p (5.2), (5.4) are incorporated into q with the EP algorithm (Hernández-Lobato and Adams 2015), i.e. they replace the corresponding approximating factors from q, and then q is updated to minimise its KL divergence with q that has the true factor incorporated.

5.5.3 Hyperparameter Optimisation

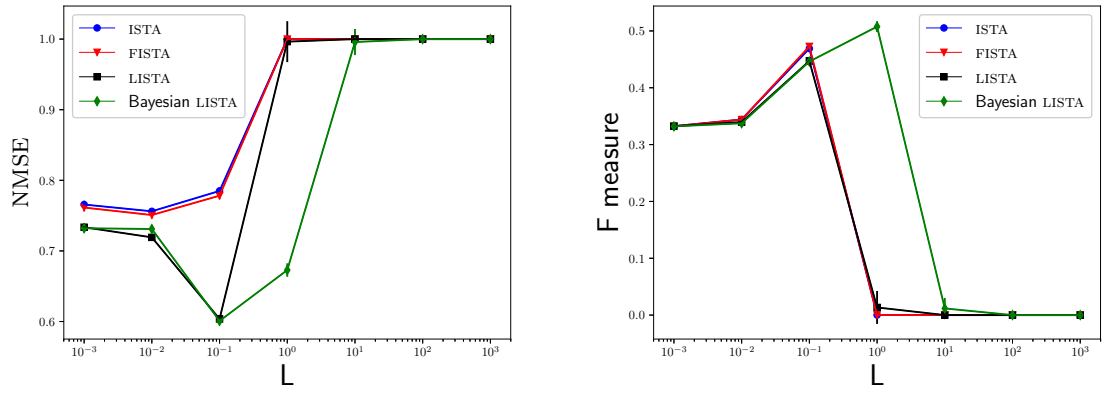
The only hyperparameter in the proposed Bayesian LISTA is the shrinkage parameter λ. It can be optimised using the Type II maximum likelihood procedure. The Type II likelihood, i.e. the evidence $p(\mathbf{B}|\mathbf{Y}, \lambda)$, of the Bayesian LISTA is equal to the normalisation constant Z (5.19) computed for the whole training dataset $\mathbf{B}$. Given the approximation (5.24), the optimal hyperparameter λ can be found by a gradient-based optimiser.

5.6 Experiments

The proposed Bayesian LISTA is evaluated in the context of the sparse coding problem with an undercomplete dictionary, where the number of measurements K is much smaller than the dimensionality of the vector $\boldsymbol{\beta}$.

The proposed Bayesian LISTA is compared with the classical LISTA (Gregor and LeCun 2010) in terms of the predictive accuracy. As baselines, ISTA (Daubechies et al. 2004) and Fast ISTA (FISTA) (Beck and Teboulle 2009) are also used. FISTA adds the momentum to ISTA and improves its convergence speed. The number of iterations in these algorithms and the number of layers in Bayesian and classical LISTA networks are set to L. To measure the performance of the algorithms the NMSE and F-measure are used (defined in Section 4.6).

The performance on small datasets is demonstrated to highlight that the proposed algorithm can infer accurate predictions when the dataset size is not sufficient for LISTA to learn.



(a) (b)

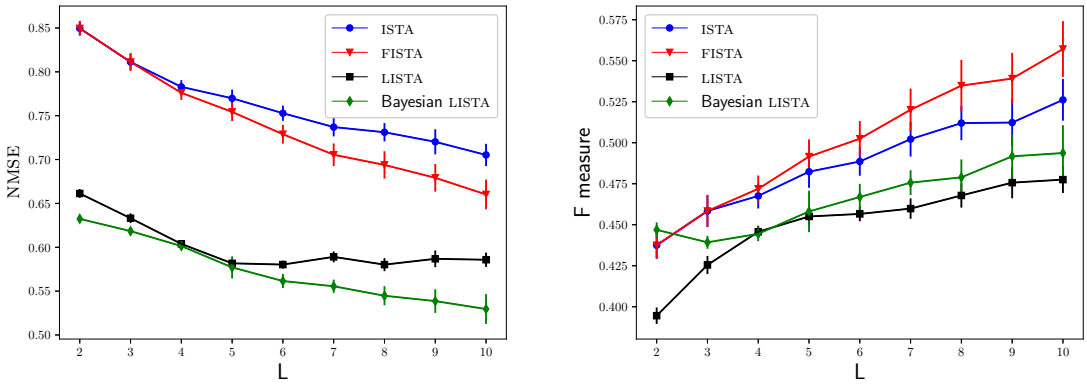
Figure 5.2: Grid optimisation for the shrinkage parameter λ on the synthetic data.

5.6.1 Predictive Performance on Synthetic Data

First, the predictive performance of the proposed Bayesian LISTA is analysed on synthetic data. $N_{\text{train}} = 500$ sparse coefficients vectors $\boldsymbol{\beta}^{(n)}$ are generated, each of size $D = 100$. Coefficients $\boldsymbol{\beta}^{(n)}$ are generated from the spike and slab distribution with truncated slab: each component of $[\beta^{(n)}]_d$ is zero with the probability 0.8 or is from the standard Gaussian distribution without interval $(-0.1, 0.1)$ with the probability 0.2. To simulate sparse observations, the random Gaussian design matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$ is generated. The observations are generated according to the linear regression problem (2.1) with zero-mean Gaussian noise with standard deviation 0.5. The shrinkage parameter is set to $\lambda = 0.1$, which is determined by grid optimisation (Figure 5.2).

In Figure 5.3 predictive performance for different number of layers L is presented. The observation size is set to $K = 50$. The Bayesian LISTA outperforms LISTA in terms of both measures. Although the baselines ISTA and FISTA show better performance in terms of F measure, the Bayesian LISTA has the lowest NMSE.

Figure 5.4 gives the results of predictive performance for different observation sizes K. The number of layers is set as $L = 4$. In the previous experiment Bayesian and classical LISTA show similar results with this number of layers. The results of Figure 5.4 confirms this competitive behaviour between two LISTA networks. Baselines show similar results in



(a) NMSE on validation

(b) F measure on validation

Figure 5.3: Predictive accuracy for different numbers of layers (for neural networks) or iterations (for baselines) on the synthetic data

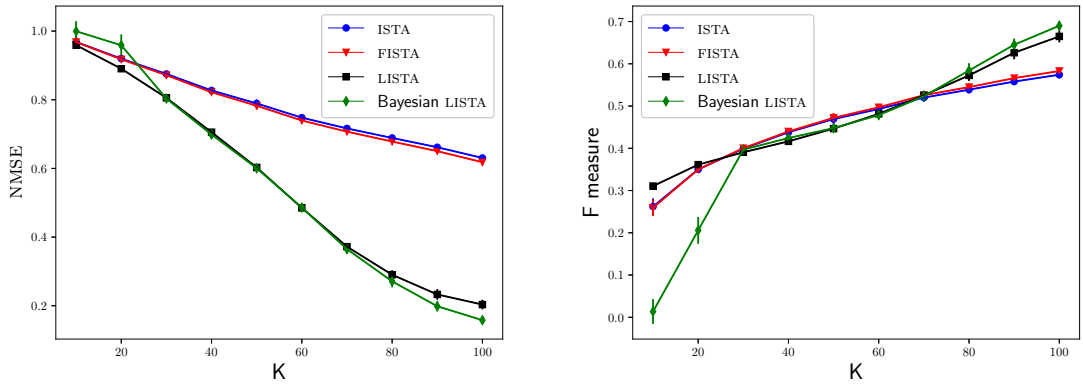
terms of F measure and underperform in terms of NMSE.

Overall, the experiments on synthetic data demonstrate that Bayesian LISTA provides competitive results in terms of predictive accuracy.

5.6.2 Predictive Performance on MNIST Data

Here, the proposed Bayesian LISTA is evaluated in terms of predictive performance on the MNIST dataset (LeCun, Bottou, et al. 1998). The dataset contains images of handwritten digits of size $28 \times 28 = 784$. The design matrix $\mathbf{X}$ is learned on 5000 images with the minibatch online algorithm (Mairal, Bach, Ponce, and Sapiro 2009). The resulting size of $\mathbf{X}$ is $K \times 784$. Then, the observations are generated as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ are flattened images. 100 images are used for training and 100 for validation. The shrinkage parameter λ is fixed as 0.1.

Figures 5.5 and 5.6 present quality on the validation set with dictionaries of size 100 and 250. The experiment with $K = 100$ presents severe conditions for the algorithms: the very limited size of the training dataset combined with the small dimensionality of observations. The Bayesian LISTA is able to learn under these conditions, outperforming LISTA that demonstrates poor results. Under better conditions of the second experiment



(a) NMSE on validation

(b) F measure on validation

Figure 5.4: Predictive performance for different sizes of observations on the synthetic data

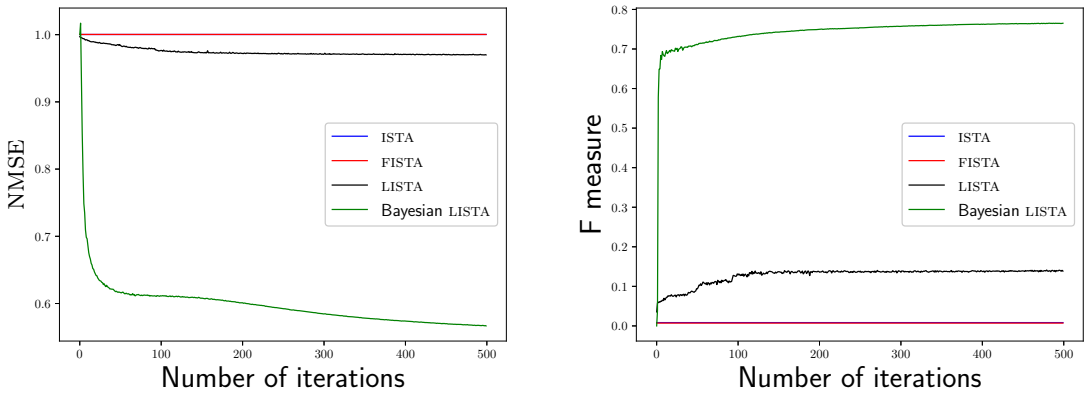
with $K = 250$ both LISTA networks converge to the similar results. However, the Bayesian LISTA demonstrates remarkably better convergence rate. Both baselines are unable to perform well in these experiments.

The proposed Bayesian LISTA network also estimates the posterior distribution for $\boldsymbol{\beta}$. Figure 5.8 shows samples from the posterior for one of the validation data points and Figure 5.7 shows the parameters of this posterior.

5.6.3 Active Learning

To demonstrate a potential scenario that can benefit from uncertainty estimates of the Bayesian LISTA, the active learning example (Settles 2009) is considered. The active learning area researches ways to select new training subsets to reduce the total number of required supervision. One of the popular approaches in active learning is uncertainty sampling when the data with the least certain predictions is chosen to obtain a new label for. Uncertainty is usually measured with entropy. In case of the spike and slab distributed data points there is no closed form for entropy. Therefore, variance from Lemma 3 is used as a measure of uncertainty.

The data in this example is the same MNIST dataset as in Section 5.6.2 with learnt dictionary of size $K = 100$. The train data of size 50, the pool data of size 500, and the test



(a) NMSE on validation

(b) F measure on validation

Figure 5.5: Predictive performance for different numbers of iterations on the MNIST data with dictionary size $K = 100$

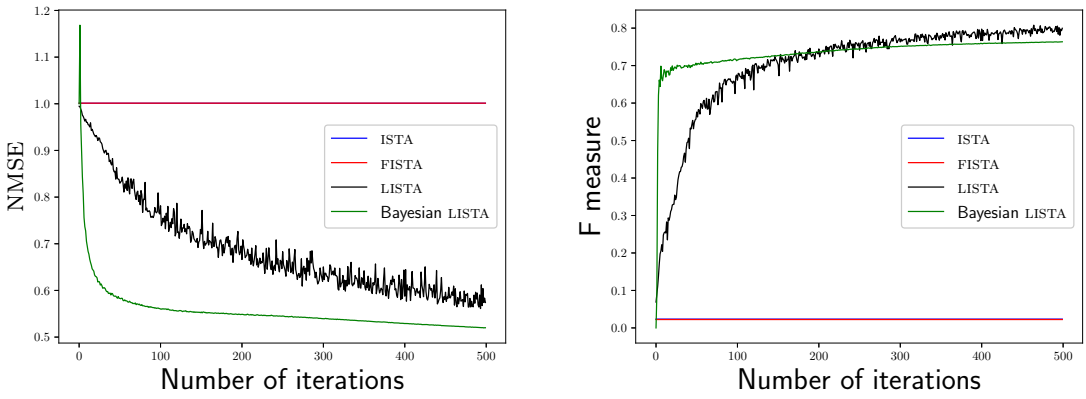
data of size 100 are used. The algorithm learns on the train data for 50 iterations and it is evaluated on the test data. To actively collect a next data point from the pool, the algorithm is used to predict a point with the highest uncertainty. The selected point is moved from the pool to the train data and the algorithm performs additional 10 iterations on the updated train data. Overall, 10 pool additions are performed. After every addition the performance is measured on the test data.

The actively updated approach of adding new points from the pool is compared with the random approach that picks a new data point from the pool at random. The overall procedure is repeated for 20 times with new randomly selected initial datasets.

Figure 5.9 demonstrates performance of the active and non-active methods of updates with the Bayesian LISTA. The active approach with uncertainty sampling steadily demonstrates better results in terms of both quality measures. This means that the posterior distribution learnt by the Bayesian LISTA is adequately estimated.

5.7 Summary

In this chapter a new method for propagating uncertainty through the soft-thresholding function is presented. This allows to propose the Bayesian LISTA network, that at every



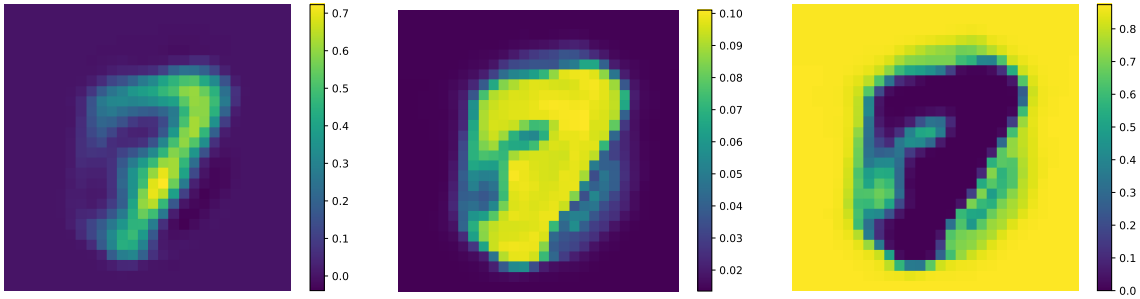
(a) NMSE on validation

(b) F measure on validation

Figure 5.6: Predictive performance for different numbers of iterations on the MNIST data with dictionary size $K = 250$

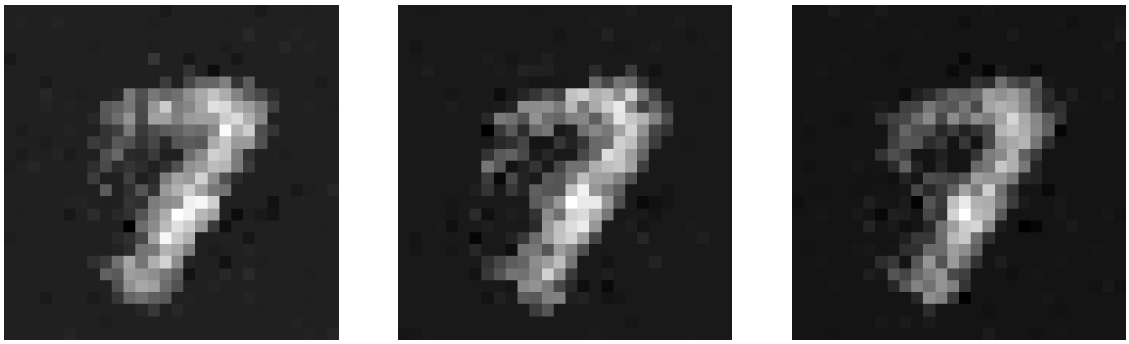
layer takes the input spike and slab distribution, transforms it with Gaussian parameters and outputs the spike and slab distribution. The proposed inference algorithm learns the distributions of the weights and makes the uncertainty estimates of the outputs. The forward propagation in the algorithm is based on the proposed uncertainty propagation method, the backward propagation is based on the probabilistic backpropagation method, that additionally accounts for multidimensionality of inputs and outputs, likelihood of the Bayesian LISTA and its recurrent nature.

Experiments on the synthetic and MNIST datasets demonstrate that the proposed algorithm preserves the predictive accuracy of non-Bayesian methods while also providing posterior estimates. It is also shown that when the training data is very small the proposed algorithm significantly outperforms the classical LISTA in terms of predictive accuracy. Experiments on active learning demonstrate that the proposed Bayesian LISTA gives accurate posterior estimates that can be used to choose the next data point for labelling.



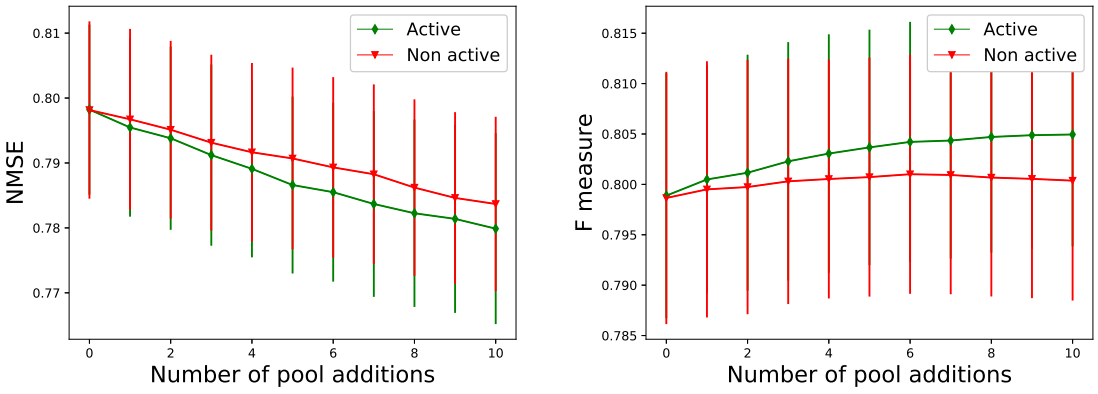
(a) β posterior mean (b) β posterior std (c) β posterior spike indicator

Figure 5.7: Posterior parameters for an image of digit 7



(a) (b) (c)

Figure 5.8: Samples from the posterior for an image of digit 7



(a) NMSE

(b) F measure

Figure 5.9: Performance for the active learning experiment on the MNIST data

Chapter 6

ENSEMBLE KALMAN FILTERS FOR SPARSE GAUSSIAN PROCESSES

In Chapters 3-5 sparsity is considered in the sense that the data contains zeros, that naturally appears in different applications. Another interpretation of sparsity is selection of the most meaningful data points that can improve computational complexity of different numerical algorithms. Gaussian processes (Section 2.3) are used in Bayesian machine learning and signal processing for estimation of unknown functions, which can be useful to model structure as in Chapter 4. However, GPs suffer from high computational complexity, as in a basic form they scale cubically with the number of observations. Several sparse approaches based on inducing points are proposed to handle this problem in a static context. However, these methods lack performance for data that is received sequentially over time. In this chapter, a novel online algorithm for training sparse Gaussian process models from online data is presented, that uses the idea of Bayesian filtering to update values of inducing points.

The chapter is organised in the following way: first, the overview of existing approaches for sparse Gaussian processes is presented in Section 6.1. The overview of the ensemble Kalman filter and the problem of state and parameter estimation within this framework is described in Section 6.2. In Section 6.3 the joint and dual ensemble Kalman filter frameworks for GPs are proposed. In Section 6.4 the experiments are conducted on the synthetic data and UK house price data and the conclusion is presented in Section 6.5.

The materials of the chapter were published as

- Danil Kuzin, Le Yang, Olga Isupova, and Lyudmila Mihaylova (2018). "Ensemble Kalman filtering for online Gaussian process regression and learning". In: *Proceedings of the 21st International Conference on Information Fusion (FUSION)*. IEEE, pp. 39–46. DOI: 10.23919/ICIF.2018.8455785

6.1 Sparse Gaussian Processes

In Bayesian machine learning and signal processing, Gaussian processes (GPs) are used to approximate unknown functions (Rasmussen and Williams 2006) and provide posterior estimates for the mean and variance of the target function in the selected points. The function can be latent, and, in this case, GPs represent the idea of proximity, or structure, when close values of inputs lead to close values of outputs. Another popular application is black-box optimisation with GPs known as Bayesian optimisation. GPs are widely applied for signal processing, examples include audio (Turner and Sahani 2011), communications (Pérez-Cruz et al. 2013), fault detection (Svensson et al. 2015).

GPs are characterised by covariance functions that usually have a set of hyperparameters. The popular examples are stationary functions that depend only on distance between points: squared-exponential, Matérn and exponential covariance functions (Rasmussen and Williams 2006). They provide solutions with different smoothness properties. The hyperparameters are hard to estimate by experts and they are usually learnt within the GP framework, for example, by optimising the marginal likelihood, which leads to local maxima.

GPs are usually represented in a grid of points and it is the source of the main limitation. The required resources are huge: computational time scales cubically with the number of grid points, memory scales quadratically. It is essential to reduce these numbers in order to make GPs applicable for larger datasets or online inference.

During the last decades multiple approaches have been proposed to deal with this problem. The most popular approach is introduction of inducing points (Quiñonero-Candela and Rasmussen 2005), where the locations of grid points are optimised, their amount is reduced with an attempt to maintain good prediction power. Inducing points can be treated as variational parameters and variational inference (Titsias 2009) or expectation propagation (Bui et al. 2017) can be performed for parameter learning and predictions.

Another approach is distributed computation, where local predictions are combined into unified mean and variance predictions. For the GP problem, the dataset can be partitioned with the use of Kd-trees (Shen et al. 2006). Another distributed Bayesian version with sparse approximation is proposed by Gal et al. (2014).

The online procedure for updating GP parameters is proposed by Huber (2014). The mean in the grid points is treated as a state variable, GP hyperparameters and noise are

treated as parameters and for the joint state-parameter vector the unscented Kalman filter is used. The model has been recently used for received-signal-strength estimation (Yin and Gunnarsson 2017; Yin, Zhao, et al. 2017), flow modelling and prediction in sports analytics (Zhao, Yin, et al. 2016).

Other approaches for online updating of the GP hyperparameters include slice sampler (Murray and Adams 2010), sequential Monte Carlo (Svensson et al. 2015), Bayesian Monte Carlo (Osborne et al. 2012).

6.2 Ensemble Kalman Filter Overview

Ensemble Kalman filter (EnKF) was originally discussed by Evensen (1994), and recent overview with different improvement techniques is given by Roth et al. (2017). EnKF uses the Monte Carlo method to generate an ensemble of state sigma points and then this state ensemble is passed through the measurement function to obtain the observation ensemble. It is additionally perturbed with the measurement noise. The mean and variance of the resulting observational distribution together with actual observations are used to update the state. The main computational difference in comparison to the classical Kalman filter is that the covariance matrices are replaced with ensembles that can be less in dimensionality.

The usual approach to parameter estimation is augmenting the state vector with parameter vector thus creating the larger augmented state-parameter vector. It can then be used to perform online estimation within the EnKF framework (Anderson 2001; Evensen 2009). Dual estimation of the state and parameters can replace joint estimation as in classical Kalman filters (Wan and Nelson 1997): for every new observation, first the parameters are updated and then using the updated parameters the state is updated. Dual estimation of the parameters and state for EnKF is considered by Moradkhani et al. (2005).

Other approaches for parameters estimation in EnKF include the maximum likelihood method (DelSole and Yang 2010; Mitchell and Houtekamer 2000) and the Bayesian inference (Stroud and Bengtsson 2007).

6.3 Ensemble Kalman Filter for Gaussian Processes

This chapter proposes the algorithms for the problem of online estimation of the constant unknown continuous function $f(\mathbf{x})$ of the D-dimensional input vector $\mathbf{x} \in \mathbb{R}^D$. The unknown

function is approximated with a GP: the mean $\mathbf{g} \in \mathbb{R}^K$ of the GP is approximated at the K grid points $\mathbf{X}_g \in \mathbb{R}^{K \times D}$ and L_θ parameters of the covariance function $\boldsymbol{\theta} \in \mathbb{R}^{L_\theta}$ are estimated. With mean and parameters of the covariance function it is possible to predict the mean and variance of $f(\mathbf{x}^*)$ at any point $\mathbf{x}^*$.

It is assumed that the observations of the function are available sequentially, at every timestamp $1 \leq n \leq N$, where N is the last observation timestamp. At every iteration n of the algorithm a total of S one-dimensional noisy function observations $\mathbf{y}^{(n)} \in \mathbb{R}^S$ are obtained at random points $\mathbf{X}_{\text{new}}^{(n)}$ as $\mathbf{y}^{(n)} = f(\mathbf{X}_{\text{new}}^{(n)}) + \boldsymbol{\varepsilon}_y$. The variance σ_y^2 of independent noise $\boldsymbol{\varepsilon}_y$ is assumed to be unknown and is estimated at every iteration of the algorithm. The full vector of parameters is therefore $\boldsymbol{\eta} = [\boldsymbol{\theta}, \sigma_y^2] \in \mathbb{R}^L$, where $L = L_\theta + 1$.

The dependency between covariance function parameters and observations is non-linear, therefore the nonlinear Kalman filter is used. The ensemble Kalman filter allows to have constant complexity for updates, which is determined by the number of ensemble points, M.

Two versions of ensemble Kalman filter for the online GP learning are proposed, they differ in the way how hyperparameters of the GP are treated: *Dual EnKF* first updates the hyperparameters of the GP and then based on their estimates updates the state; *Joint EnKF* updates hyperparameters of the GP and the state simultaneously with the augmented state-hyperparameter vector.

6.3.1 Dual Ensemble Kalman Filter for Gaussian Processes

This algorithm is further denoted as Dual GP-EnKF. It uses the ensembles of same size M to approximate the distributions of the parameters and state. At every iteration the predicted distributions of the parameters and state are computed, and the observations are predicted. Then, based on the cross-covariance of the parameter and observation ensembles, the Kalman gain is computed and it is used to update the parameter distribution. After this step, new observations are predicted with the updated parameters and then the cross-covariance of the new observations and state is used to update the state. The details of the algorithm are presented below.

Initialisation

Initially, ensembles for parameters $\mathbf{H} \in \mathbb{R}^{M\times L} = [\boldsymbol{\eta}_{(m)} \in \mathbb{R}^{1\times L}]_{1\leq m\leq M}$ and mean $\mathbf{G} \in \mathbb{R}^{M\times K} = [\mathbf{g}_{(m)} \in \mathbb{R}^{1\times K}]_{1\leq m\leq M}$ at the grid points of the GP are generated. The rows of matrices correspond to ensemble members. For parameters that can only be positive, such as variance, logarithms of their values are used in the ensemble. Initial ensembles are generated from the Gaussian distribution: for each ensemble index $1 \leq m \leq M$

$$\boldsymbol{\eta}_{(m)}^{0|0} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_H), \tag{6.1a}$$

$$\mathbf{g}_{(m)}^{0|0} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_G), \tag{6.1b}$$

where $\boldsymbol{\Sigma}_H$, $\boldsymbol{\Sigma}_G$ are the initial covariance matrices for the ensembles. In the experiments, they are assumed to be diagonal.

After the initialisation at every iteration of the algorithm three steps follow: prediction, update for the parameters, update for the state.

Prediction

For the whole running time of the algorithm the estimated function remains constant, while unknown. This can be simulated with the random walk motion model for the parameters and state. Each ensemble member is updated as

$$\boldsymbol{\eta}_{(m)}^{n+1|n} = \boldsymbol{\eta}_{(m)}^{n|n} + \boldsymbol{\varepsilon}_\eta, \tag{6.2a}$$

$$\mathbf{g}_{(m)}^{n+1|n} = \mathbf{g}_{(m)}^{n|n} + \boldsymbol{\varepsilon}_g, \tag{6.2b}$$

where $\boldsymbol{\varepsilon}_\eta \sim \mathcal{N}(\mathbf{0}, \sigma_\eta \mathbf{I})$ and $\boldsymbol{\varepsilon}_g \sim \mathcal{N}(\mathbf{0}, \sigma_g \mathbf{I})$ are the noise variables with corresponding variances.

Assume that S observations are obtained at locations $\mathbf{X}_{\text{new}}^{(n)} = [\mathbf{x}_{\text{new}}^{(n)\ s}]_{s=1}^{S}$. According to the definition of GPs the joint distribution for any finite set of samples is the multivariate Gaussian distribution. Therefore, for each parameter ensemble m the distribution of predicted function values $\widehat{\mathbf{y}}_{(m)} = [\widehat{y}_{1,(m)}, \ldots, \widehat{y}_{S,(m)}]$ at locations $\mathbf{X}_{\text{new}}^{(n)}$ can be obtained as

$$\begin{aligned}\widehat{\mathbf{y}}_{(m)} = K(\mathbf{X}_{\text{new}}^{(n)}, \mathbf{X}_g | \boldsymbol{\theta}_{(m)}^{n+1|n}) \\ \times [K(\mathbf{X}_g, \mathbf{X}_g | \boldsymbol{\theta}_{(m)}^{n+1|n}) + \sigma_{y(m)}^{2,n+1|n} \mathbf{I}]^{-1} \mathbf{g}_{(m)}^{n+1|n},\end{aligned} \tag{6.3}$$

where $K(\mathbf{X}_1, \mathbf{X}_2|\boldsymbol{\theta})$ is the covariance matrix evaluated at every pair of points from $\mathbf{X}_1$, $\mathbf{X}_2$ with parameters $\boldsymbol{\theta}$; $\boldsymbol{\theta}_{(m)}^{n+1|n}$ and $\sigma_{y(m)}^{2,n+1|n}$ are components of the joint parameter vector $\boldsymbol{\eta}_{(m)}^{n+1|n}$. The matrix for all predictions is denoted as $\widehat{\mathbf{Y}} \in \mathbb{R}^{M\times S} = [\widehat{\mathbf{y}}_{(m)}]_{m=1}^{M}$

In EnKF, observations are treated as random variables and the observation ensemble is generated, which has a Gaussian distribution around the actual observation with the predefined covariance σ_{obs}^2

$$\mathbf{y}_{(m)} = \mathbf{y} + \boldsymbol{\varepsilon}_{\text{obs}}, \tag{6.4}$$

where $\boldsymbol{\varepsilon}_{\text{obs}} \sim \mathcal{N}(0, \sigma_{\text{obs}}^2\mathbf{I})$.

Update Parameters

EnKF updates are similar to the usual Kalman filter, with the means and covariances estimated from the ensembles. First, cross covariances of the parameter ensemble and prediction ensemble are computed. Let $\mathbb{E}_i[\cdot]$ denote the expected value with respect to ensembles. Then

$$\overline{\boldsymbol{\eta}}^{n+1|n} = \frac{1}{M}\sum_{m=1}^{M}\boldsymbol{\eta}_{(m)}^{n+1|n}, \tag{6.5a}$$

$$\begin{aligned}\boldsymbol{\Sigma}^{\boldsymbol{\eta}\mathbf{y}} =& \mathbb{E}_i\left[(\mathbf{H}^{n+1|n} - \mathbb{E}_i[\mathbf{H}^{n+1|n}])^\top(\widehat{\mathbf{Y}} - \mathbb{E}_i[\widehat{\mathbf{Y}}])\right]\\ =& \frac{1}{M-1}\sum_{m=1}^{M}(\boldsymbol{\eta}_{(m)}^{n+1|n} - \overline{\boldsymbol{\eta}}^{n+1|n})^\top(\widehat{\mathbf{y}}_{(m)} - \mathbf{y}).\end{aligned} \tag{6.5b}$$

After that, the forecast error covariance matrix of the predictions is computed

$$\begin{aligned}\boldsymbol{\Sigma}^{\mathbf{y}\mathbf{y}} =& \mathbb{E}_i\left[(\widehat{\mathbf{Y}} - \mathbb{E}_i[\widehat{\mathbf{Y}}])^\top(\widehat{\mathbf{Y}} - \mathbb{E}_i[\widehat{\mathbf{Y}}])\right]\\ =& \frac{1}{M-1}\sum_{m=1}^{M}(\widehat{\mathbf{y}}_{(m)} - \mathbf{y})^\top(\widehat{\mathbf{y}}_{(m)} - \mathbf{y}).\end{aligned} \tag{6.6}$$

Then the Kalman gain for correcting parameters can be computed as

$$\mathbf{K}^{\boldsymbol{\eta}} = \boldsymbol{\Sigma}^{\boldsymbol{\eta}\mathbf{y}}(\boldsymbol{\Sigma}^{\mathbf{y}\mathbf{y}} + \sigma_{\text{obs}}^2\mathbf{I})^{-1}. \tag{6.7}$$

The parameters are updated as

$$\boldsymbol{\eta}_{(m)}^{n+1|n+1} = \boldsymbol{\eta}_{(m)}^{n+1|n} + \mathbf{K}^{\boldsymbol{\eta}}(\mathbf{y}_{(m)} - \widehat{\mathbf{y}}_{(m)}). \tag{6.8}$$

Update State

Updates for the state are similar to the updates for parameters, but with the updated values of the parameters.

First, predictions of observations are corrected with the updated parameters using

$$\begin{aligned}\widehat{\mathbf{y}}_{(m)} =& K(\mathbf{X}_{\text{new}}^{(n)}, \mathbf{X}_g | \boldsymbol{\theta}_{(m)}^{n+1|n+1}) \\ & \times [K(\mathbf{X}_g, \mathbf{X}_g | \boldsymbol{\theta}_{(m)}^{n+1|n+1}) + \sigma_{y(m))}^{2,n+1|n} \mathbf{I}]^{-1} \mathbf{g}_{(m)}^{n+1|n}.\end{aligned} \tag{6.9}$$

After that, the cross covariance of the state ensemble and prediction ensemble is updated

$$\overline{\mathbf{g}}^{n+1|n} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{g}_{(m)}^{n+1|n}, \tag{6.10a}$$

$$\begin{aligned}\boldsymbol{\Sigma}^{\mathbf{gy}} =& \mathbb{E}_i \left[(\mathbf{G}^{n+1|n} - \mathbb{E}[\mathbf{G}^{n+1|n}])^\top (\widehat{\mathbf{Y}} - \mathbb{E}[\widehat{\mathbf{Y}}]) \right] \\ =& \frac{1}{M-1} \sum_{m=1}^{M} (\mathbf{g}_{(m)}^{n+1|n} - \overline{\mathbf{g}}^{n+1|n})^\top (\widehat{\mathbf{y}}_{(m)} - \mathbf{y}).\end{aligned} \tag{6.10b}$$

The forecast error covariance matrix of the predictions is computed according to (6.6) and then the Kalman gain for correcting state is

$$\mathbf{K}^{\mathbf{g}} = \boldsymbol{\Sigma}^{\mathbf{gy}} (\boldsymbol{\Sigma}^{\mathbf{yy}} + \sigma_{\text{obs}}^2 \mathbf{I})^{-1}. \tag{6.11}$$

Then the state is updated as

$$\mathbf{g}_{(m)}^{n+1|n+1} = \mathbf{g}_{(m)}^{n+1|n} + \mathbf{K}^{\mathbf{g}} (\mathbf{y}_{(m)} - \widehat{\mathbf{y}}_{(m)}). \tag{6.12}$$

The resulting procedure is given in Algorithm 2.

6.3.2 Liu-West Filter

The evolution of the parameter distribution in (6.2) leads to its over-diffuse. The Liu-West filter (Liu and West 2001) uses kernel density estimation, it can be used to estimate the predicted distribution of the parameters so that the resulting distribution converges to the true distribution. It is parametrised with a discount factor $\delta_{lw} \in (0, 1]$, that is usually taken from the interval $[0.95, 0.99]$. With introduction of additional parameters

$$a_{lw} = \frac{3\delta_{lw} - 1}{2\delta_{lw}}, \tag{6.13a}$$

$$h_{lw}^2 = 1 - a_{lw}^2. \tag{6.13b}$$

Algorithm 2 Dual GP-EnKF algorithm

Initialise (6.1)

for $n = 1$ to N **do**

Predict

for $m = 1$ to M **do**

Predict parameters and state (6.2)

Predict observations (6.3)

Compute noisy trajectories (6.4)

end for

Update parameters

Compute cross covariance of parameter ensemble and prediction ensemble (6.5)

Compute forecast error covariance matrix of the predictions (6.6)

Compute Kalman gain for correcting parameters (6.7)

Update parameters (6.8)

Update state

Predict observations with updated parameters (6.9)

Compute cross covariance of state ensemble and prediction ensemble (6.10)

Compute forecast error covariance matrix of the predictions (6.6)

Compute Kalman gain for correcting state (6.11)

Update state (6.12)

end for

the evolution of the parameter density is

$$\boldsymbol{\eta}_{(m)}^{n+1|n} = a_{lw}\boldsymbol{\eta}_{(m)}^{n|n} + (1 - a_{lw})\overline{\eta}^{n|n} + \boldsymbol{\varepsilon}_{lw}, \tag{6.14}$$

where $\boldsymbol{\varepsilon}_{lw} \sim \mathcal{N}\left(\mathbf{0}, \sqrt{h_{lw}^2 \operatorname{Var} \boldsymbol{\eta}_{n|n}}\right)$ The algorithm is further denoted as Liu-West Dual GP-EnKF.

6.3.3 Computational Complexity of Dual Ensemble Kalman Filter for Gaussian Processes

At the prediction step the most demanding operation is prediction of observations, that requires inversion of the covariance matrix for each ensemble member, that is $\mathcal{O}(MK^3)$. At the update steps it is computation of Kalman gains, that is $\mathcal{O}(S^3) + \mathcal{O}(LS^2)$ for the parameters and $\mathcal{O}(S^3)+\mathcal{O}(KS^2)$ for the state. If the number S of observations is greater than the dimensionality L of hyperparameters, then the resulting computational time complexity for the Dual GP-EnKF is $\mathcal{O}(N(MK^3 + S^3 + KS^2))$.

The classical GP without inducing points that stores all previous observations and recomputes predictions at every time step n has $\mathcal{O}(S^3n^3)$ computational complexity due to the covariance matrix growing in size as Sn at every dimension. The resulting computational time complexity for the classical GP is then $\mathcal{O}(S^3N^3)$. Note that the computational complexity of the dual ensemble Kalman Filter is linear with respect to the number N of time steps.

6.3.4 Joint Ensemble Kalman Filter for Gaussian Processes

It is also possible to estimate parameters of the model by augmenting the state vector $\mathbf{g}$ with the parameter vector $\boldsymbol{\eta}$: the augmented state is $\boldsymbol{s} = [\mathbf{g}; \boldsymbol{\eta}]$. The algorithm is further denoted as Joint GP-EnKF. The details are presented in Algorithm 3.

Initialisation

Initially, an ensemble for the augmented state $\mathbf{S} \in \mathbb{R}^{M\times(L+K)} = [\mathbf{s}_{(m)}]_{1\leq m\leq M}$ is generated. For each $1 \leq m \leq M$

$$\boldsymbol{\eta}_{(m)}^{0|0} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_S), \tag{6.15a}$$

where $\boldsymbol{\Sigma}_S$ is the initial covariance matrices for the ensembles. After the initialisation the algorithm iterates prediction and update steps.

Prediction

Similar to the dual EnKF, the random walk assumption for the motion model of the augmented state is assumed. Each ensemble member is updated as

$$\mathbf{s}_{(m)}^{n+1|n} = \mathbf{s}_{(m)}^{n|n} + \boldsymbol{\varepsilon}_s, \tag{6.16}$$

where $\boldsymbol{\varepsilon}_s \sim \mathcal{N}(\mathbf{0}, \sigma_s \mathbf{I})$ is the noise variable with corresponding variance.

The predictions are made in the same way as in (6.3) and observations are noised as in (6.4).

Update

Updates for the augmented state are similar to the updates for the state in dual EnKF. The cross covariance of the augmented state ensemble and prediction ensemble is estimated as

$$\bar{\mathbf{s}}^{n+1|n} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{s}_{(m)}^{n+1|n}, \tag{6.17a}$$

$$\begin{aligned} \boldsymbol{\Sigma}^{\mathbf{sy}} &= \mathbb{E}_i \left[(\mathbf{S}^{n+1|n} - \mathbb{E}[\mathbf{S}]^{n+1|n})^\top (\widehat{\mathbf{Y}} - \mathbb{E}[\widehat{\mathbf{Y}}]) \right] \\ &= \frac{1}{M-1} \sum_{m=1}^{M} (\mathbf{s}_{(m)}^{n+1|n} - \bar{\mathbf{s}}^{n+1|n})^\top (\widehat{\mathbf{y}}_{(m)} - \mathbf{y}). \end{aligned} \tag{6.17b}$$

After that, the forecast error covariance matrix of the predictions is computed as (6.6) and then the Kalman gain for correcting augmented state

$$\mathbf{K}^{\mathbf{s}} = \boldsymbol{\Sigma}^{\mathbf{sy}} (\boldsymbol{\Sigma}^{\mathbf{yy}} + \sigma_y \mathbf{I})^{-1}. \tag{6.18}$$

Then the augmented state is updated as

$$\mathbf{s}_{(m)}^{n+1|n+1} = \mathbf{s}_{(m)}^{n+1|n} + \mathbf{K}^{\mathbf{s}} (\mathbf{y}_{(m)} - \widehat{\mathbf{y}}_{(m)}). \tag{6.19}$$

6.4 Experiments

In this section the performance of the proposed algorithms is evaluated on both synthetic and real data. Three versions of the EnKF for online GP parameters estimation are assessed: Dual GP-EnKF, Liu-West Dual GP-EnKF, and Joint GP-EnKF. The developed algorithms are compared with the classical GP regression without online updates in terms of both

Algorithm 3 Joint GP-EnKF algorithm

Initialise (6.15)

for $n = 1$ to N **do**

Predict

for $m = 1$ to M **do**

Predict augmented state (6.16)

Predict observations (6.3)

Compute noisy trajectories (6.4)

end for

Update augmented state

Compute cross covariance of augmented state ensemble and prediction ensemble (6.17)

Compute forecast error covariance matrix of the predictions (6.6)

Compute Kalman gain for correcting augmented state (6.18)

Update augmented state (6.19)

end for

computational time and predictive accuracy. At every iteration n the classical GP regression is applied on all historical data.

For quantitative evaluation of the predictive accuracy, two quality metrics are used on held-out test data $[\mathbf{x}_{test}, \mathbf{y}_{test}]$:

Log marginal likelihood of the model.

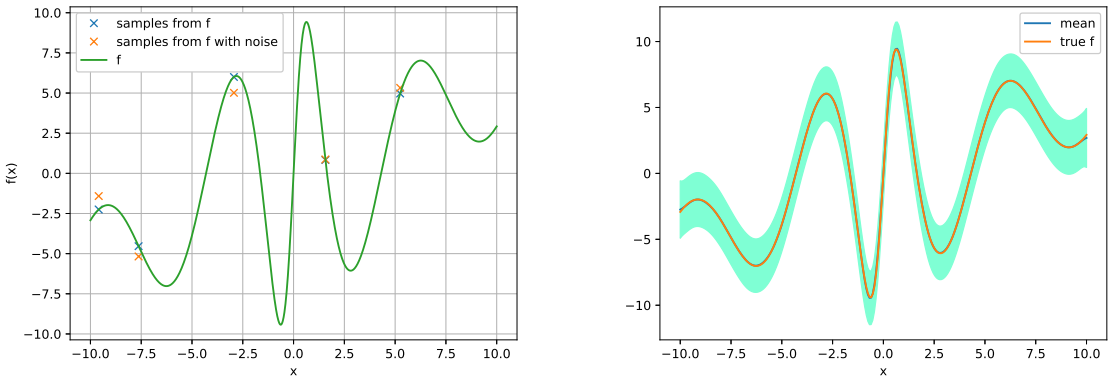
$$\begin{aligned} \log p(\mathbf{y}_{test}) = &-\frac{1}{2}\mathbf{y}^\top (K(\mathbf{x}_{test}, \mathbf{x}_{test}) + \sigma_y^2 \mathbf{I})^{-1} \mathbf{y}_{test} \\ &- \frac{1}{2}\log |K(\mathbf{x}_{test}, \mathbf{x}_{test}) + \sigma_y^2 \mathbf{I}| - \frac{N_{\text{test}}}{2}\log 2\pi, \end{aligned} \tag{6.20}$$

where N_{test} is the total number of test data points.

Mean normalised squared error of predictions (NMSE).

$$\text{NMSE} = \frac{1}{N_{\text{test}}} \sum_{s=1}^{N_{\text{test}}} \frac{\sqrt{(y_s - f^*(\mathbf{x}_{test\,s}))^2}}{|y_s|}, \tag{6.21}$$

where y_s is the observed value of the function at the test data point $\mathbf{x}_{test\,s}$, and $f^*(\mathbf{x}_{test\,s})$ is the predicted function value at the test data point.



(a) The target function used in the synthetic data experiment. As an example, the sample from the function used as the input for the algorithm at one iteration is displayed.

(b) The classical GP mean with two standard deviations, parameters optimised on the full history of observations.

Figure 6.1: Target function and classical GP approximation for the synthetic data

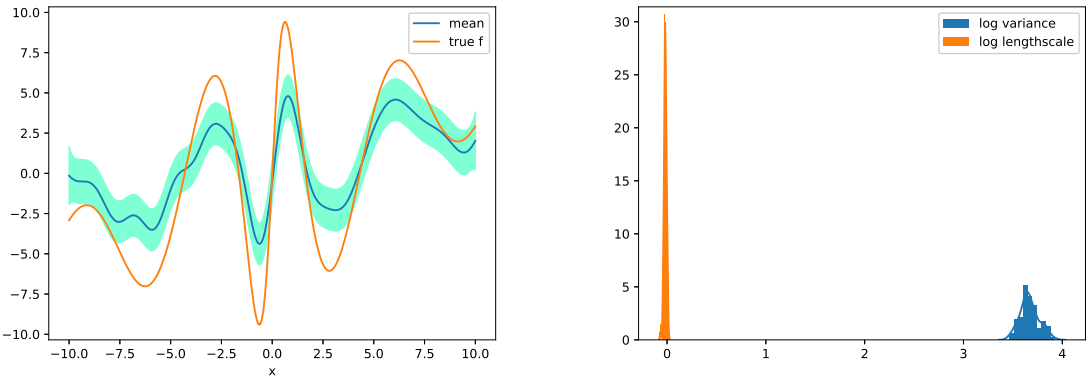
6.4.1 Synthetic Data

The algorithms are firstly evaluated on the synthetic data. The target function for this data is $f(x) = \frac{x}{2} + \frac{25x}{1+x^2}\cos(x)$ (Figure 6.1a). The experiments are conducted with the following parameters: the observation noise is $\sigma_y^2 = 0.01$, size of grid is $K = 51$, the covariance function is squared-exponential and it has two hyperparameters, $\boldsymbol{\theta} =$ [variance, lengthscale], size of each ensemble is $M = 100$, sample size is $S = 5$ and the total number of iterations is $N = 200$.

Figure 6.1b shows the function estimate given by the classical GP regression. Since the total number of observations is sufficiently large, the classical GP enables to reconstruct ideal predictions of the function.

The performance of the proposed approaches is given in Figure 6.2–6.4. As one can observe the Joint GP-EnKF (Figure 6.2) correctly estimates peaks of the target function, but it has large predictive errors for most of the observations. The Joint GP-EnKF learns the consistent ensemble estimates of the hyperparameters, i.e. their variance is not large.

The Dual GP-EnKF (Figure 6.3) provides predictions that are more accurate that the predictions by the Joint GP-EnKF, but still there are several locations where the true target



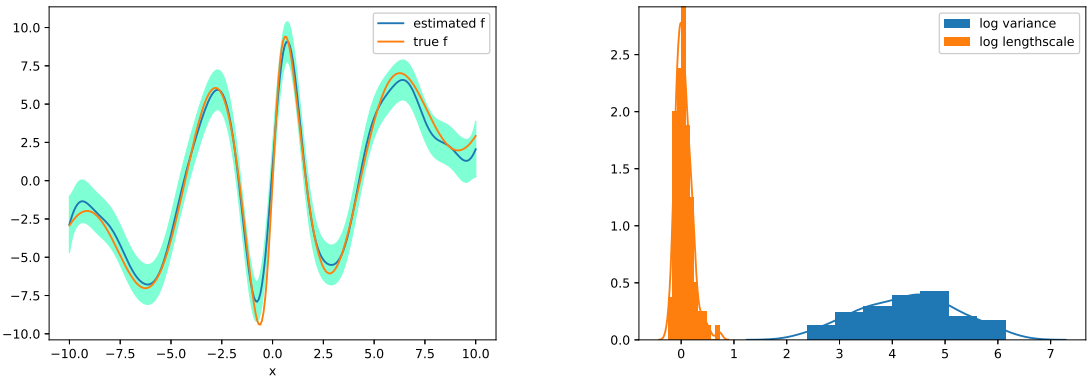
(a) Estimated mean, two standard deviation interval (b) Final distributions for the estimated GP parameters of the GP and target function

Figure 6.2: Performance of the Joint GP-EnKF on the synthetic data

function values lie outside of the two standard deviations of the prediction. The ensemble of the Dual GP-EnKF has the low variance for the logarithm of the lengthscale parameter of the covariance function and the high variance for the estimates of the signal variance parameter.

The Liu-West Dual GP-EnKF (Figure 6.4) is applied with the discount factor $\delta_{lw} = 0.99$. The algorithm makes predictions that are very close to the true values of the target function. The ensemble of the Liu-West Dual GP-EnKF has better estimations of hyperparameter than both Dual and Joint GP-EnKFs.

The procedure is repeated for 10 Monte Carlo runs with different random seeds. The results below are presented as average among these 10 Monte Carlo runs. In the Figure 6.5 the history of quality measures is given over time. Their final values together with the computational time are presented in Table 6.1. While the Joint GP-EnKF is the fastest method, the NMSE of both Dual GP-EnKF methods is lower with the Liu-West Dual GP-EnKF providing the best results. The classical GP provides the lowest NMSE, however, it has the computational time more than 10 times higher than the slowest of the proposed approaches. In terms of the likelihood all methods show similar results with Joint GP-EnKF slightly outperforming the other two algorithms.



(a) Estimated mean, two standard deviation interval (b) Final distributions for the estimated GP parameters of the GP and target function

Figure 6.3: Performance of the Dual GP-EnKF on the synthetic data

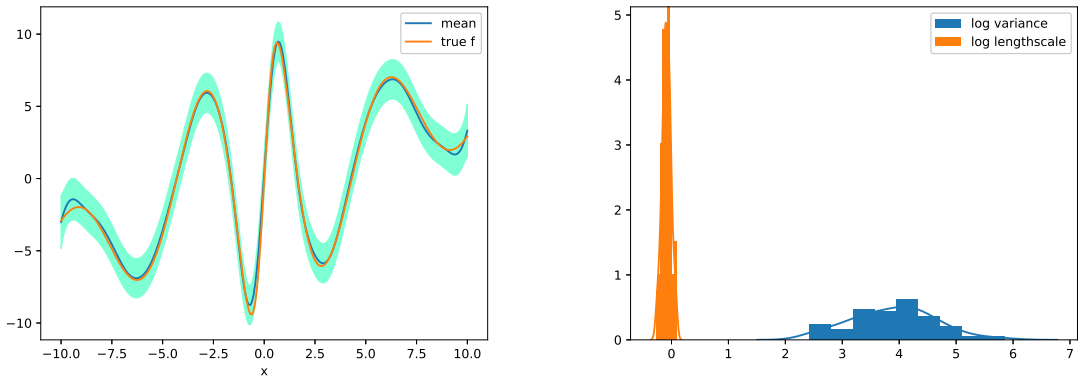
6.4.2 House Prices

The Dual GP-EnKF approach is further evaluated on the real data. In this example the HM Land Registry Price Paid Data[1] is considered. The subset of all flats and maisonettes sold in 2017 is selected and parameter estimation is performed to predict mean prices corresponding to the locations of properties. Longitude and latitude values for every location have been calculated based on the postcode. Therefore, in this experiment, every single input $\mathbf{x}$ is two-dimensional.

[1]https://data.gov.uk/dataset/land-registry-monthly-price-paid-data/

Table 6.1: Performance on the synthetic data at $N = 200$

Method	NMSE	Log Likelihood	Time (s)
Joint GP-EnKF	0.64	$\mathbf{-155.10}$	**7.23**
Dual GP-EnKF	0.48	-187.19	13.68
Liu-West Dual GP-EnKF	**0.19**	-161.61	15.60
Classical GP	0.02	-155.88	186.20

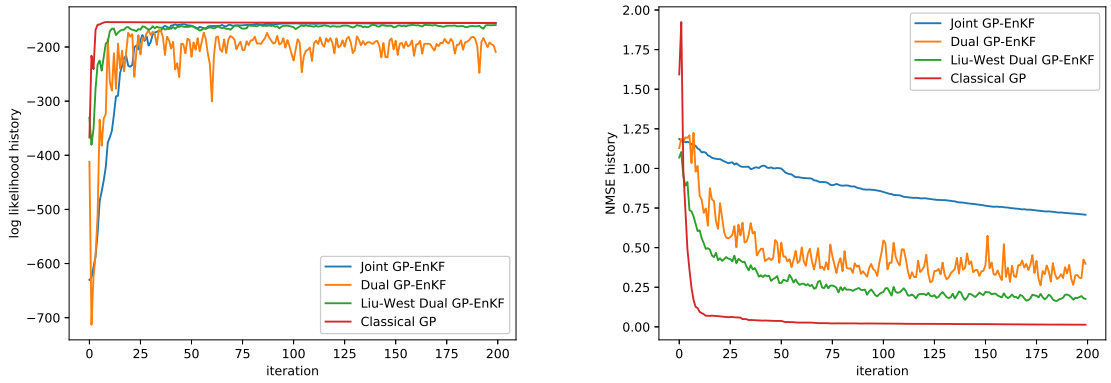


(a) Estimated mean, two standard deviation interval (b) Final distributions for the estimated GP parameters of the GP and target function

Figure 6.4: Performance of the Liu-West Dual GP-EnKF on the synthetic data

A total of $N = 20$ iterations have been performed with two-dimensional grid of size $K = 25 \times 25 = 625$. At every iteration, $S = 100$ samples of the logarithms of standardised prices are used to update parameters and mean in the grid points. The ensemble consists of $M = 200$ members. The covariance function is stationary squared-exponential.

Figure 6.6 demonstrates the results after the first and final iterations. It is clear that the prices have converged close to real values, identifying such areas as London and Oxford as places with higher prices. Though there are spikes of the mean in the sea, the corresponding covariance values that describe uncertainty of predictions in these points are high. Note that the used squared-exponential covariance function is one of the simplest covariance functions in terms of complexity of modelling dependencies of function values at different data points. The stationary squared-exponential covariance function does not depend on locations. Therefore, the results can potentially be further improved if the squared-exponential covariance function is considered together with non-stationary covariance functions to obtain more precise estimates for covariance difference between sea and land locations.



(a) History of log likelihood

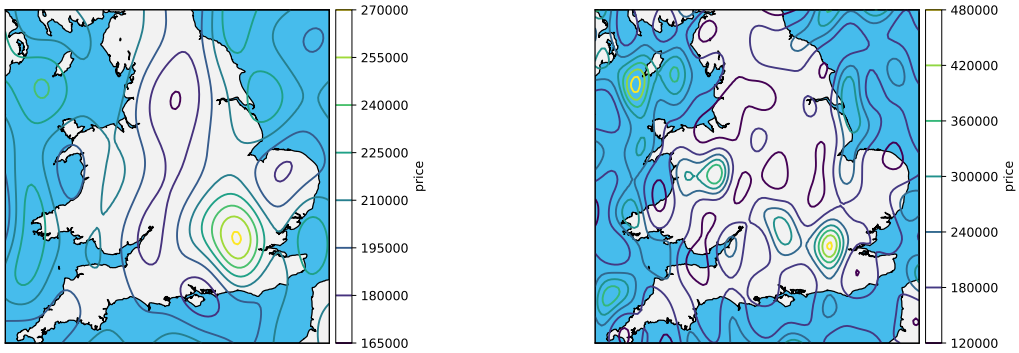
(b) History of NMSE

Figure 6.5: History of quality measures

6.5 Summary

This chapter proposes two ensemble Kalman filters for online Gaussian process regression and learning. The mean and hyperparameters of the GP are interpreted as the state and parameters of the ensemble Kalman filter, respectively. The ensemble Kalman filter updates are utilised to recursively improve estimates of both state and parameters. Two versions of the ensemble updates are proposed: Joint GP-EnKF where the update step of the EnKF is applied for the augmented vector-parameter vector and Dual GP-EnKF where the update step is split to first update the parameters and then based on new estimates of the parameters the state is updated. For the Dual GP-EnKF the Liu-West filter (Liu and West 2001) updates are additionally developed for further improvement of the estimates.

The proposed ensemble Kalman filter approach for the GP has a linear computational complexity with respect to the number of sequential observations, it depends mainly on the dimensionality of the observations at each timestamp and internal parameters of the filter. For the large volume of data acquired sequentially it can significantly reduce the computational time in comparison to the classical GP regression that scales cubically with respect to the number of observations. Starting from a sufficient number of observations cubic complexity makes the usual GP not applicable for this large-scale data. The proposed ensemble Kalman filter is applicable to any number of sequential observations given that at



(a) Mean prices after the first batch of data is presented to the algorithm

(b) Mean prices after the 20th batch of data is presented to the algorithm

Figure 6.6: Mean estimates of the prices with Dual GP-EnKF

each timestamp the dimensionality of observations is feasible.

The experiments both on synthetic and real data show that the proposed ensemble Kalman filter approaches for Gaussian process estimation provide satisfactory predictive accuracy using significantly less computational time in comparison to the GP regression without online updates. Among the proposed approaches the Liu-West Dual GP-EnKF filter demonstrates the best results in terms of the predictive accuracy slightly underperforming the Joint GP-EnKF in terms of the computational time.

Chapter 7

CONCLUSION

This chapter presents the overview of main contributions presented in the thesis and provides potential directions for future research in the area of Bayesian sparsity.

7.1 Contributions

The sparse methods are important components of machine learning as they allow to reduce computational complexity and increase interpretability of the results. They are also used on their own in signal processing to approximate signals from the reduced number of measurements. This thesis develops new Bayesian machine learning algorithms to deal with the sparse regression problem. The developed algorithms are applied for EEG source localisation and video compression problems.

Several different interpretations of sparsity are used in this thesis. In the first part, the sparse Bayesian regression is explored and extended for structured data. In the following part, the completely new approach for sparse regression based on Bayesian neural networks is presented. In the last part, the sparsity in Gaussian processes is explored within the context of Bayesian filtering.

In Chapter 3, the problem of compressive background subtraction in video is considered. Two weak Bayesian sparsity methods, based on the Bayesian compressive sensing framework, demonstrate improved computational time compared to the frequentist methods. The multitask Bayesian compressive sensing method improves the performance for the multitask setup, where multiple similar problems are solved in parallel. This chapter demonstrates the potential advantages of sparse Bayesian modelling and raises the problem of structure modelling for sparse models.

Chapter 4 explores the problem of structure modelling within the sparse Bayesian regression. The strong hierarchical sparse model is presented, that discovers time-evolving

structures within the coefficient domain. Structure modelling is achieved with a two-level GP, which allows to approximate structures of an arbitrary shape. The algorithm can operate with online data, drastically reducing computational complexity in this case. The Bayesian inference method is based on EP. The performance of the model is demonstrated on compressive background subtraction and EEG data.

A different approach for sparse regression based on Bayesian neural networks is developed in Chapter 5. The neural network approach for sparse regression reduces time required for predictions while requiring training. The Bayesian neural network approach attempts to solve some of the problems related to the frequentist neural network approach: it can quantify the uncertainty and reduce the overfitting problem. This is achieved by introducing uncertainty for weights of the neural network and propagating the input and uncertainty added with weights through the network to get the distribution of the output. The probabilistic backpropagation algorithm allows to update the distributions of weights based on the training data. The examples demonstrate that the performance of the model is similar to the frequentist model, while the additional uncertainty estimations measures the confidence of the model and it can be useful in different scenarios, such as active learning.

Gaussian processes achieve good results for structure modelling, but they require huge resources for posterior inference. In Chapter 6 the approach to reduce computational complexity for GP regression with online data is introduced. It uses inducing points and recomputes the posterior distribution at these points with Bayesian filtering. Several different versions of EnKF-based filters are developed, which deal with updates. The results are demonstrated on the house price data.

7.2 Directions for Future Work

This thesis presents several new directions of research in sparse Bayesian modelling, that are described in this section.

7.2.1 Variational Inference

The algorithms for Bayesian inference in Chapter 4 and Chapter 5 are based on the expectation propagation method. Another popular inference algorithm is variational inference (Titsias and Lázaro-Gredilla 2011), that approximates posterior by optimising a different form of

KL-divergence, thus leading to different solutions. Usually variational inference poorly scales for increasing number of data points. The models could be reformulated as conditionally conjugate models with local and global variables, where the global variables are the parameters for all data points and the local variables are specific for each data point. For such problems there exists a stochastic variational inference algorithm that allows to speed up updates for global variables (Hoffman, Blei, et al. 2013): the conventional variational inference algorithm uses all data points to update approximations for global variables, while the stochastic algorithm uses only subsets of data points. The corresponding variational algorithms may be developed for models proposed in this thesis to compare their performance with the proposed algorithms.

7.2.2 Expectation Propagation

Similarly to variational inference, expectation propagation also has scaling issues, as the memory overhead increases with the number of data points. For the algorithm introduced in Section 4.4.1, it can be noticed that storage of approximating factors, $\widetilde{\psi}_C$, is required for all factors, thus leading to large memory requirements. To avoid this problem, it is possible to store only the average approximating factor (Li et al. 2015). The algorithms in Chapter 4 and Chapter 5 could be updated to stochastic versions to reduce the memory requirements, especially for large training datasets in neural networks.

7.2.3 Neural Networks Architecture

Several different nonlinearities suitable for the LISTA model have been proposed since the development of the original algorithm (Borgerding et al. 2017), which provide better estimates. The corresponding Bayesian formulations may be developed for the algorithm in Chapter 5 that can improve the learning speed or the posterior approximation quality.

7.2.4 Sparse Gaussian Processes with Bayesian Filtering

While the Chapter 6 provides several filtering algorithms for sparse GPs, they all use the predefined locations of the inducing points. These locations can also be treated as latent variables and therefore they can be learned within the filtering framework as well.

Squared-exponential covariance function was used for experiments, but it can be replaced with other functions to achieve better performance for the nonstationary data.

BIBLIOGRAPHY

Abadi, Martin, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. (2016). "Tensorflow: a system for large-scale machine learning." In: *OSDI*. Vol. 16, pp. 265–283.

Ahn, Sungjin, Anoop Korattikara, and Max Welling (2012). "Bayesian posterior sampling via stochastic gradient Fisher scoring". In: *Proceedings of the 29th International Conference on Machine Learning (ICML)*. Ed. by J. Langford and J. Pineau, pp. 1591–1598.

Akalin-Acar, Zeynep and Nevzat G Gençer (2004). "An advanced boundary element method (BEM) implementation for the forward problem of electromagnetic source imaging". In: *Physics in Medicine & Biology* vol. 49, issue 21, p. 5011.

Andersen, Michael Riis, Aki Vehtari, Ole Winther, and Lars Kai Hansen (2017). "Bayesian inference for spatio-temporal spike and slab priors". In: *The Journal of Machine Learning Research* vol. 18, issue 139, pp. 1–58.

Anderson, Jeffrey L (2001). "An ensemble adjustment Kalman filter for data assimilation". In: *Monthly Weather Review* vol. 129, issue 12, pp. 2884–2903.

Andrews, David F and Colin L Mallows (1974). "Scale mixtures of normal distributions". In: *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 99–102.

Armagan, Artin (2009). "Variational bridge regression". In: *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by D. van Dyk and M. Welling, pp. 17–24.

Armagan, Artin, David B Dunson, and Jaeyong Lee (2013). "Generalized double Pareto shrinkage". In: *Statistica Sinica* vol. 23, issue 1, p. 119.

Arvaneh, Mahnaz, Cuntai Guan, Kai Keng Ang, and Chai Quek (2011). "Optimizing the channel selection and classification accuracy in EEG-based BCI". In: vol. 58, issue 6, pp. 1865–1873.

Bach, Francis, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski (2012a). "Optimization with sparsity-inducing penalties". In: *Foundations and Trends in Machine Learning* vol. 4, issue 1, pp. 1–106.

Bach, Francis, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, et al. (2012b). "Structured sparsity through convex optimization". In: *Statistical Science* vol. 27, issue 4, pp. 450–468.

Baillet, Sylvain and Line Garnero (1997). "A Bayesian approach to introducing anatomo-functional priors in the EEG/MEG inverse problem". In: *IEEE Transactions on Biomedical Engineering* vol. 44, issue 5, pp. 374–385.

Baillet, Sylvain, John C Mosher, and Richard M Leahy (2001). "Electromagnetic brain mapping". In: *IEEE Signal Processing Magazine* vol. 18, issue 6, pp. 14–30.

Baraniuk, Richard G, Volkan Cevher, Marco F Duarte, and Chinmay Hegde (2010). "Model-based compressive sensing". In: *IEEE Transactions on Information Theory* vol. 56, issue 4, pp. 1982–2001.

Baraniuk, Richard, Mark Davenport, Ronald DeVore, and Michael Wakin (2008). "A simple proof of the restricted isometry property for random matrices". In: *Constructive Approximation* vol. 28, issue 3, pp. 253–263.

Beck, Amir and Marc Teboulle (2009). "A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring". In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 693–696.

Bhattacharya, Anirban, Debdeep Pati, Natesh S Pillai, and David B Dunson (2015). "Dirichlet–Laplace priors for optimal shrinkage". In: *Journal of the American Statistical Association* vol. 110, issue 512, pp. 1479–1490.

Bishop, Christopher M (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

Borgerding, Mark, Philip Schniter, and Sundeep Rangan (2017). "AMP-Inspired Deep Networks for Sparse Linear Inverse Problems". In: *IEEE Transactions on Signal Processing* vol. 65, issue 16, pp. 4293–4308.

Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein (2011). "Distributed optimization and statistical learning via the alternating direction method of multipliers". In: *Foundations and Trends in Machine Learning* vol. 3, issue 1, pp. 1–122.

Brochu, Eric, Vlad M Cora, and Nando De Freitas (2010). "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: *arXiv preprint arXiv:1012.2599*.

Bui, Thang D., Josiah Yan, and Richard E. Turner (2017). "A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation". In: *The Journal of Machine Learning Research* vol. 18, issue 104, pp. 1–72.

Candes, Emmanuel J, Justin K Romberg, and Terence Tao (2006). "Stable signal recovery from incomplete and inaccurate measurements". In: *Communications on Pure and Applied Mathematics* vol. 59, issue 8, pp. 1207–1223.

Candes, Emmanuel J and Terence Tao (2005). "Decoding by linear programming". In: *IEEE Transactions on Information Theory* vol. 51, issue 12, pp. 4203–4215.

Candès, Emmanuel J and Michael B Wakin (2008). "An introduction to compressive sampling". In: *IEEE Signal Processing Magazine* vol. 25, issue 2, pp. 21–30.

Carvalho, Carlos M, Nicholas G Polson, and James G Scott (2010). "The horseshoe estimator for sparse signals". In: *Biometrika* vol. 97, issue 2, pp. 465–480.

Cevher, Volkan, Aswin Sankaranarayanan, Marco F Duarte, Dikpal Reddy, Richard G Baraniuk, and Rama Chellappa (2008). "Compressive sensing for background subtraction". In: *Proceedings of the 10th European Conference on Computer Vision (ECCV)*. Ed. by D. Forsyth, P. Torr, and A. Zisserman, pp. 155–168.

Chen, Wei, David Wipf, Yu Wang, Yang Liu, and Ian J Wassell (2016). "Simultaneous Bayesian Sparse Approximation With Structured Sparse Models". In: *IEEE Transactions on Signal Processing* vol. 64, issue 23, pp. 6145–6159.

Chipman, Hugh, Edward I George, Robert E McCulloch, Merlise Clyde, Dean P Foster, and Robert A Stine (2001). "The practical implementation of Bayesian model selection". In: *Model selection*. Ed. by P. Lahiri. Institute of Mathematical Statistics, pp. 65–116.

Damianou, Andreas C, Michalis K Titsias, and Neil D Lawrence (2016). "Variational inference for latent variables and uncertain inputs in Gaussian processes". In: *The Journal of Machine Learning Research* vol. 17, issue 1, pp. 1425–1486.

Daubechies, Ingrid (1992). *Ten Lectures on Wavelets*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Daubechies, Ingrid, Michel Defrise, and Christine De Mol (2004). "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint". In: *Communications on Pure and Applied Mathematics* vol. 57, issue 11, pp. 1413–1457.

Deisenroth, Marc and Shakir Mohamed (2012). "Expectation propagation in Gaussian process dynamical systems". In: *Advances in Neural Information Processing Systems 25 (NIPS)*. Ed. by F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, pp. 2609–2617.

Delorme, Arnaud and Scott Makeig (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis". In: *Journal of Neuroscience Methods* vol. 134, issue 1, pp. 9–21.

DelSole, Timothy and Xiaosong Yang (2010). "State and parameter estimation in stochastic dynamical models". In: *Physica D: Nonlinear Phenomena* vol. 239, issue 18, pp. 1781–1788.

Donoho, David L (2006). "Compressed sensing". In: *IEEE Transactions on Information Theory* vol. 52, issue 4, pp. 1289–1306.

Donoho, David L, Arian Maleki, and Andrea Montanari (2009). "Message-passing algorithms for compressed sensing". In: *Proceedings of the National Academy of Sciences* vol. 106, issue 45, pp. 18914–18919.

Duarte, Marco F, Mark A Davenport, Dharmpal Takhar, Jason N Laska, Ting Sun, Kevin F Kelly, and Richard G Baraniuk (2008). "Single-pixel imaging via compressive sampling". In: *IEEE Signal Processing Magazine* vol. 25, issue 2, pp. 83–91.

Evensen, Geir (1994). "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics". In: *Journal of Geophysical Research: Oceans* vol. 99, issue C5, pp. 10143–10162.

Evensen, Geir (2009). "The ensemble Kalman filter for combined state and parameter estimation". In: *IEEE Control Systems* vol. 29, issue 3.

Figueiredo, Mário AT (2003). "Adaptive sparseness for supervised learning". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 25, issue 9, pp. 1150–1159.

Gal, Yarin, Mark Van Der Wilk, and Carl Edward Rasmussen (2014). "Distributed variational inference in sparse Gaussian process regression and latent variable models". In: *Advances*

in Neural Information Processing Systems 27 (NIPS). Ed. by Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, pp. 3257–3265.

George, Edward I and Robert E McCulloch (1993). "Variable selection via Gibbs sampling". In: *Journal of the American Statistical Association* vol. 88, issue 423, pp. 881–889.

Geselowitz, David B (1967). "On bioelectric potentials in an inhomogeneous volume conductor". In: *Biophysical journal* vol. 7, issue 1, p. 1.

Gorban, Alexander N, Eugenij Moiseevich Mirkes, and A Zinovyev (2016). "Piece-wise quadratic approximations of arbitrary error functions for fast and robust machine learning". In: *Neural Networks* vol. 84, pp. 28–38.

GPy (2012). *GPy: A Gaussian process framework in python*. `http://github.com/SheffieldML/GPy`.

Graves, Alex (2011). "Practical variational inference for neural networks". In: *Advances in Neural Information Processing Systems 24 (NIPS)*. Ed. by J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, pp. 2348–2356.

Gregor, Karol and Yann LeCun (2010). "Learning fast approximations of sparse coding". In: *Proceedings of the 27th International Conference on Machine Learning (ICML)*. Ed. by J. Fürnkranz and T. Joachims, pp. 399–406.

Hans, Chris (2009). "Bayesian lasso regression". In: *Biometrika* vol. 96, issue 4, pp. 835–845.

Hastie, Trevor, Robert Tibshirani, and Martin Wainwright (2015). *Statistical learning with sparsity: the lasso and generalizations*. CRC Press.

Hernández-Lobato, José Miguel and Ryan Adams (2015). "Probabilistic backpropagation for scalable learning of Bayesian neural networks". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. Ed. by F. Bach and D. Blei, pp. 1861–1869.

Hernández-Lobato, José Miguel, Daniel Hernández-Lobato, and Alberto Suárez (2015). "Expectation propagation in linear regression models with spike-and-slab priors". In: *Machine Learning* vol. 99, issue 3, pp. 437–487.

Hoffman, Matthew D (2017). "Learning deep latent Gaussian models with Markov chain Monte Carlo". In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Ed. by D. Precup and Y. W. Teh, pp. 1510–1519.

Hoffman, Matthew D, David M Blei, Chong Wang, and John Paisley (2013). "Stochastic variational inference". In: *The Journal of Machine Learning Research* vol. 14, issue 1, pp. 1303–1347.

Hoffman, Matthew D and Andrew Gelman (2014). "The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo". In: *The Journal of Machine Learning Research* vol. 15, issue 1, pp. 1593–1623.

Huber, Marco F (2014). "Recursive Gaussian process: On-line regression and learning". In: *Pattern Recognition Letters* vol. 45, pp. 85–91.

Isupova, Olga, Danil Kuzin, and Lyudmila Mihaylova (2015). "Abnormal behaviour detection in video using topic modeling". In: *USES Conference Proceedings*. The University of Sheffield. DOI: 10.15445/02012015.18.

Isupova, Olga, Danil Kuzin, and Lyudmila Mihaylova (2016). "Dynamic hierarchical Dirichlet process for abnormal behaviour detection in video". In: *Proceedings of the 19th International Conference on Information Fusion (FUSION)*. IEEE, pp. 750–757. URL: https://ieeexplore.ieee.org/document/7527962/.

Isupova, Olga, Danil Kuzin, and Lyudmila Mihaylova (2018). "Learning methods for dynamic topic modeling in automated behavior analysis". In: *IEEE Transactions on Neural Networks and Learning Systems* vol. 29, issue 9, pp. 3980–3993. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2017.2735364.

Isupova, Olga, Lyudmila Mihaylova, Danil Kuzin, Garik Markarian, and Francois Septier (2015). "An expectation maximisation algorithm for behaviour analysis in video". In: *Proceedings of the 18th International Conference on Information Fusion (FUSION)*. IEEE, pp. 126–133. URL: https://ieeexplore.ieee.org/document/7266553/.

Jatoi, Munsif Ali, Nidal Kamel, Aamir Saeed Malik, Ibrahima Faye, and Tahamina Begum (2014). "A survey of methods used for source localization using EEG signals". In: *Biomedical Signal Processing and Control* vol. 11, pp. 42–52.

Ji, Shihao, David Dunson, and Lawrence Carin (2009). "Multitask compressive sensing". In: *IEEE Transactions on Signal Processing* vol. 57, issue 1, pp. 92–106.

Ji, Shihao, Ya Xue, and Lawrence Carin (2008). "Bayesian compressive sensing". In: *IEEE Transactions on Signal Processing* vol. 56, issue 6, pp. 2346–2356.

Kingma, Diederik P, Tim Salimans, and Max Welling (2015). "Variational dropout and the local reparameterization trick". In: *Advances in Neural Information Processing Systems 28 (NIPS)*. Ed. by C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, pp. 2575–2583.

Kingma, Diederik P and Max Welling (2014). "Auto-encoding variational Bayes". In: *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

Koller, Daphne and Nir Friedman (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Kuzin, Danil, Olga Isupova, and Lyudmila Mihaylova (2015). "Compressive sensing approaches for autonomous object detection in video sequences". In: *Proceedings of the Sensor Data Fusion: Trends, Solutions, Applications Workshop (SDF)*. IEEE, pp. 1–6. DOI: 10.1109/SDF.2015.7347706.

Kuzin, Danil, Olga Isupova, and Lyudmila Mihaylova (2017). "Structured sparse modelling with hierarchical GP". In: *Proceedings of the 6th Signal Processing with Adaptive Sparse Structured Representations Workshop (SPARS)*. URL: http://spars2017.lx.it.pt/index_files/papers/SPARS2017_Paper_48.pdf.

Kuzin, Danil, Olga Isupova, and Lyudmila Mihaylova (2018a). "Spatio-temporal structured sparse regression with hierarchical Gaussian process priors". In: *IEEE Transactions on Signal Processing* vol. 66, issue 17, pp. 4598–4611. DOI: 10.1109/TSP.2018.2858207.

Kuzin, Danil, Olga Isupova, and Lyudmila Mihaylova (2018b). "Uncertainty propagation in neural networks for sparse coding". In: *Proceedings of the Third Workshop on Bayesian Deep Learning (NeurIPS)*. URL: http://bayesiandeeplearning.org/2018/papers/47.pdf.

Kuzin, Danil, Olga Isupova, and Lyudmila Mihaylova (2019). "Bayesian neural networks for sparse coding". Accepted at IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Kuzin, Danil, Le Yang, Olga Isupova, and Lyudmila Mihaylova (2018). "Ensemble Kalman filtering for online Gaussian process regression and learning". In: *Proceedings of the 21st International Conference on Information Fusion (FUSION)*. IEEE, pp. 39–46. DOI: 10.23919/ICIF.2018.8455785.

Lawrence, Neil D and Andrew J Moore (2007). "Hierarchical Gaussian process latent variable models". In: *Proceedings of the 24th International Conference on Machine Learning (ICML)*. Ed. by Z. Ghahramani, pp. 481–488.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning". In: *Nature* vol. 521, issue 7553, p. 436.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* vol. 86, issue 11, pp. 2278–2324.

Li, Yingzhen, José Miguel Hernández-Lobato, and Richard E Turner (2015). "Stochastic expectation propagation". In: *Advances in Neural Information Processing Systems 28 (NIPS)*. Ed. by C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, pp. 2323–2331.

Liu, Jane and Mike West (2001). "Combined parameter and state estimation in simulation-based filtering". In: *Sequential Monte Carlo methods in practice*. Springer, pp. 197–223.

Mairal, Julien, Francis Bach, and Jean Ponce (2014). "Sparse modeling for image and vision processing". In: *Foundations and Trends in Computer Graphics and Vision* vol. 8, issue 2-3, pp. 85–283.

Mairal, Julien, Francis Bach, Jean Ponce, and Guillermo Sapiro (2009). "Online dictionary learning for sparse coding". In: *Proceedings of the 26th International Conference on Machine Learning (ICML)*. Ed. by L. Bottou and M. Littman, pp. 689–696.

Mallat, Stephane (2008). *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. 3rd. Academic Press.

Mallat, Stéphane G and Zhifeng Zhang (1993). "Matching pursuits with time-frequency dictionaries". In: *IEEE Transactions on Signal Processing* vol. 41, issue 12, pp. 3397–3415.

Minka, Thomas (2001a). "A family of algorithms for approximate Bayesian inference". PhD thesis. MIT.

Minka, Thomas P (2001b). "Expectation propagation for approximate Bayesian inference". In: *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI)*. Ed. by J. Breese and D. Koller, pp. 362–369.

Minka, Thomas and John Lafferty (2002). "Expectation-propagation for the generative aspect model". In: *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI)*. Ed. by A. Darwiche and N. Friedman, pp. 352–359.

Mitchell, Herschel L and PL Houtekamer (2000). "An adaptive ensemble Kalman filter". In: *Monthly Weather Review* vol. 128, issue 2, pp. 416–433.

Mitchell, Toby J and John J Beauchamp (1988). "Bayesian variable selection in linear regression". In: *Journal of the American Statistical Association* vol. 83, issue 404, pp. 1023–1032.

Moradkhani, Hamid, Soroosh Sorooshian, Hoshin V Gupta, and Paul R Houser (2005). "Dual state–parameter estimation of hydrological models using ensemble Kalman filter". In: *Advances in Water Resources* vol. 28, issue 2, pp. 135–147.

Murphy, K (2012). *Machine learning: a probabilistic approach*. Massachusetts Institute of Technology.

Murray, Iain and Ryan P Adams (2010). "Slice sampling covariance hyperparameters of latent Gaussian models". In: *Advances in Neural Information Processing Systems 23 (NIPS)*. Ed. by J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, pp. 1732–1740.

Neal, Radford M (1994). "Bayesian learning for neural networks". PhD thesis. University of Toronto.

Osborne, Michael A, Stephen J Roberts, Alex Rogers, and Nicholas R Jennings (2012). "Real-time information processing of environmental sensor network data using bayesian Gaussian processes". In: *ACM Transactions on Sensor Networks (TOSN)* vol. 9, issue 1, p. 1.

Park, Trevor and George Casella (2008). "The bayesian lasso". In: *Journal of the American Statistical Association* vol. 103, issue 482, pp. 681–686.

Pérez-Cruz, Fernando, Steven Van Vaerenbergh, Juan José Murillo-Fuentes, Miguel Lázaro-Gredilla, and Ignacio Santamaria (2013). "Gaussian processes for nonlinear signal processing: An overview of recent advances". In: *IEEE Signal Processing Magazine* vol. 30, issue 4, pp. 40–50.

Polson, Nicholas G and James G Scott (2010). "Shrink globally, act locally: Sparse Bayesian regularization and prediction". In: *Bayesian statistics* vol. 9, pp. 501–538.

Quiñonero-Candela, Joaquin and Carl Edward Rasmussen (2005). "A unifying view of sparse approximate Gaussian process regression". In: *The Journal of Machine Learning Research* vol. 6, issue Dec, pp. 1939–1959.

Rasmussen, Carl Edward and Christopher K. I. Williams (2006). *Gaussian processes for machine learning*. The MIT Press.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic Back-propagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning (ICML)*. Ed. by E. P. Xing and T. Jebara, pp. 1278–1286.

Roth, Michael, Gustaf Hendeby, Carsten Fritsche, and Fredrik Gustafsson (2017). "The Ensemble Kalman filter: a signal processing perspective". In: *EURASIP Journal on Advances in Signal Processing* vol. 2017, issue 1, p. 56.

Sarkka, Simo, Arno Solin, and Jouni Hartikainen (2013). "Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering". In: *IEEE Signal Processing Magazine* vol. 30, issue 4, pp. 51–61.

Schmolck, Alexander and Richard Everson (2007). "Smooth relevance vector machine: a smoothness prior extension of the RVM". In: *Machine Learning* vol. 68, issue 2, pp. 107–135.

Seeger, Matthias W (2008). "Bayesian inference and optimal design for the sparse linear model". In: *The Journal of Machine Learning Research* vol. 9, issue Apr, pp. 759–813.

Seeger, Matthias W and Hannes Nickisch (2011). "Large scale Bayesian inference and experimental design for sparse linear models". In: *SIAM Journal on Imaging Sciences* vol. 4, issue 1, pp. 166–199.

Settles, Burr (2009). *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison.

Shen, Yirong, Matthias Seeger, and Andrew Y Ng (2006). "Fast Gaussian process regression using KD-trees". In: *Advances in Neural Information Processing Systems 19 (NIPS)*. Ed. by B. Schölkopf, J.C. Platt, and T. Hoffman, pp. 1225–1232.

Solin, Arno, Pasi Jylänki, Jaakko Kauramäki, Tom Heskes, Marcel AJ van Gerven, and Simo Särkkä (2016). "Regularizing Solutions to the MEG Inverse Problem Using Space-Time Separable Covariance Functions". In: *arXiv preprint arXiv:1604.04931*.

Soussen, Charles, Jérôme Idier, David Brie, and Junbo Duan (2011). "From Bernoulli–Gaussian deconvolution to sparse signal restoration". In: *IEEE Transactions on Signal Processing* vol. 59, issue 10, pp. 4572–4584.

Sprechmann, Pablo, Ignacio Ramirez, Guillermo Sapiro, and Yonina C Eldar (2011). "C-HiLasso: A collaborative hierarchical sparse modeling framework". In: *IEEE Transactions on Signal Processing* vol. 59, issue 9, pp. 4183–4198.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* vol. 15, issue 1, pp. 1929–1958.

Stroud, Jonathan R and Thomas Bengtsson (2007). "Sequential state and variance estimation within the ensemble Kalman filter". In: *Monthly Weather Review* vol. 135, issue 9, pp. 3194–3208.

Svensson, Andreas, Johan Dahlin, and Thomas B Schön (2015). "Marginalizing Gaussian process hyperparameters using sequential Monte Carlo". In: *Proceedings of the 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, pp. 477–480.

Takhar, Dharmpal, Jason N Laska, Michael B Wakin, Marco F Duarte, Dror Baron, Shriram Sarvotham, Kevin F Kelly, and Richard G Baraniuk (2006). "A new compressive imaging camera architecture using optical-domain compression". In: *Computational Imaging IV*. Vol. 6065. International Society for Optics and Photonics, p. 606509.

Tibshirani, Robert (1996). "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288.

Tipping, Michael E (2001). "Sparse Bayesian learning and the relevance vector machine". In: *The Journal of Machine Learning Research* vol. 1, issue Jun, pp. 211–244.

Tipping, Michael E and Anita C Faul (2003). "Fast marginal likelihood maximisation for sparse Bayesian models". In: *Proceedings of the 9th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by C. M. Bishop and B. J. Frey.

Titsias, Michalis K (2009). "Variational learning of inducing variables in sparse Gaussian processes". In: *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by D. van Dyk and M. Welling, pp. 567–574.

Titsias, Michalis K and Miguel Lázaro-Gredilla (2011). "Spike and slab variational inference for multi-task and multiple kernel learning". In: *Advances in Neural Information Processing Systems 24 (NIPS)*. Ed. by J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, pp. 2339–2347.

Tropp, Joel A and Anna C Gilbert (2007). "Signal recovery from random measurements via orthogonal matching pursuit". In: *IEEE Transactions on Information Theory* vol. 53, issue 12, pp. 4655–4666.

Turner, Richard E and Maneesh Sahani (2011). "Demodulation as probabilistic inference". In: *IEEE Transactions on Audio, Speech, and Language Processing* vol. 19, issue 8, pp. 2398–2411.

Van Gerven, Marcel AJ, Botond Cseke, Floris P De Lange, and Tom Heskes (2010). "Efficient Bayesian multivariate fMRI analysis using a sparsifying spatio-temporal prior". In: *NeuroImage* vol. 50, issue 1, pp. 150–161.

Wainwright, Martin J and Michael I Jordan (2008). "Graphical models, exponential families, and variational inference". In: *Foundations and Trends in Machine Learning* vol. 1, issue 1–2, pp. 1–305.

Wan, Eric A and Alex T Nelson (1997). "Dual Kalman filtering methods for nonlinear prediction, smoothing and estimation". In: *Advances in Neural Information Processing Systems 10 (NIPS)*. Ed. by M.I. Jordan, M.J. Kearns, and S.A. Solla, pp. 793–799.

Wang, Sida and Christopher Manning (2013). "Fast dropout training". In: *Proceedings of the 30th International Conference on Machine Learning (ICML)*. Ed. by S. Dasgupta and D. McAllester, pp. 118–126.

Warnell, Garrett, Sourabh Bhattacharya, Rama Chellappa, and Tamer Başar (2015). "Adaptive-rate compressive sensing using side information". In: *IEEE Transactions on Image Processing* vol. 24, issue 11, pp. 3846–3857.

Welling, Max and Yee W Teh (2011). "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th International Conference on Machine Learning (ICML)*. Ed. by I. Guyon, G. Dror, V. Lemaire, G. Taylor, and D. Silver, pp. 681–688.

West, Mike (1987). "On scale mixtures of normal distributions". In: *Biometrika* vol. 74, issue 3, pp. 646–648.

Wu, Anqi, Mijung Park, Oluwasanmi O Koyejo, and Jonathan W Pillow (2014). "Sparse Bayesian structure learning with dependent relevance determination priors". In: *Advances in Neural Information Processing Systems 27 (NIPS)*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, pp. 1628–1636.

Wu, Qisong, Yimin D Zhang, Moeness G Amin, and Braham Himed (2015). "High-resolution passive SAR imaging exploiting structured Bayesian compressive sensing". In: *IEEE Journal of Selected Topics in Signal Processing* vol. 9, issue 8, pp. 1484–1497.

Xin, Bo, Yizhou Wang, Wen Gao, David Wipf, and Baoyuan Wang (2016). "Maximal sparsity with deep networks?" In: *Advances in Neural Information Processing Systems 29 (NIPS)*. Ed. by D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, and R. Garnett, pp. 4340–4348.

Xu, Yuhang, Qi Yu, Wei Dai, Zoran Cvetković, and Verity M McClelland (2018). "Cortico-Muscular Coherence Enhancement Via Sparse Signal Representation". In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 811–815.

Yang, Jianbo, Xin Yuan, Xuejun Liao, Patrick Llull, David J Brady, Guillermo Sapiro, and Lawrence Carin (2014). "Video compressive sensing using Gaussian mixture models". In: *IEEE Transactions on Image Processing* vol. 23, issue 11, pp. 4863–4878.

Yin, Feng and Fredrik Gunnarsson (2017). "Distributed Recursive Gaussian Processes for RSS Map Applied to Target Tracking". In: *IEEE Journal of Selected Topics in Signal Processing* vol. 11, issue 3, pp. 492–503.

Yin, Feng, Yuxin Zhao, Fredrik Gunnarsson, and Fredrik Gustafsson (2017). "Received-signal-strength threshold optimization using Gaussian processes". In: *IEEE Transactions on Signal Processing* vol. 65, issue 8, pp. 2164–2177.

Yuan, Ming and Yi Lin (2006). "Model selection and estimation in regression with grouped variables". In: *Journal of the Royal Statistical Society. Series B (Methodological)* vol. 68, issue 1, pp. 49–67.

Zhang, Zhilin, Tzyy-Ping Jung, Scott Makeig, Zhouyue Pi, and Bhaskar D Rao (2014). "Spatiotemporal sparse Bayesian learning with applications to compressed sensing of

multichannel physiological signals". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* vol. 22, issue 6, pp. 1186–1197.

Zhang, Zhilin and Bhaskar D Rao (2011). "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning". In: *IEEE Journal of Selected Topics in Signal Processing* vol. 5, issue 5, pp. 912–926.

Zhao, Shiwen, Chuan Gao, Sayan Mukherjee, and Barbara E Engelhardt (2016). "Bayesian group factor analysis with structured sparsity". In: *The Journal of Machine Learning Research* vol. 17, issue 196, pp. 1–47.

Zhao, Yuxin, Feng Yin, Fredrik Gunnarsson, Fredrik Hultkratz, and Johan Fagerlind (2016). "Gaussian processes for flow modeling and prediction of positioned trajectories evaluated with sports data". In: *Proceedings of the 19th International Conference on Information Fusion (FUSION)*. IEEE, pp. 1461–1468.

Zhou, Mingyuan, Haojun Chen, Lu Ren, Guillermo Sapiro, Lawrence Carin, and John W Paisley (2009). "Non-parametric Bayesian dictionary learning for sparse image representations". In: *Advances in Neural Information Processing Systems 22 (NIPS)*. Ed. by Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, pp. 2295–2303.

Zou, Hui and Trevor Hastie (2005). "Regularization and variable selection via the elastic net". In: *Journal of the Royal Statistical Society. Series B (Methodological)* vol. 67, issue 2, pp. 301–320.