



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Doctoral Thesis

PROBABILISTIC MODEL DISCOVERY  
RELATIONAL LEARNING AND SCALABLE INFERENCE

Anh Tong

Department of Computer Science and Engineering

Ulsan National Institute of Science and Technology

2021

PROBABILISTIC MODEL DISCOVERY  
RELATIONAL LEARNING AND SCALABLE INFERENCE

Anh Tong

Department of Computer Science and Engineering

Ulsan National Institute of Science and Technology

# Probabilistic Model Discovery Relational Learning and Scalable Inference

A thesis submitted to  
Ulsan National Institute of Science and Technology  
in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy

Anh Tong

01/06/2021

Approved by

*Kwang In Kim*

Kwang In Kim (Jan 6, 2021 19:36 GMT+9)

---

Advisor  
Kwang In Kim

# Probabilistic Model Discovery

## Relational Learning and Scalable Inference

Anh Tong

This certifies that the thesis of Anh Tong is approved.

01/06/2021

Signature

  
Kwang In Kim (Jan 6, 2021 19:36 GMT+9)

---

Advisor: Kwang In Kim

Signature



---

Jaesik Choi: Thesis Committee Member #1

Signature



---

Se Young Chun: Thesis Committee Member #2


Signature

  
Kee-Eung Kim (Jan 6, 2021 16:39 GMT+9)

---

Kee-Eung Kim: Thesis Committee Member #3

Signature

  
Sung-Phil Kim (Jan 6, 2021 19:04 GMT+9)

---

Sung-Phil Kim: Thesis Committee Member #4

## Abstract

This thesis studies interesting problems in compositionality for machine learning models under some settings including relational learning, scalability and deep models. Compositionality is the terminology describing the process of building small objects to complex ones. Bringing this concept into machine learning is important because it appears in many aspects from infinitesimal atomic to planetary structures. In this thesis, machine learning models center around Gaussian process of which covariance function is compositionally constructed. The proposed approach builds methods that can explore compositional model space automatically and efficiently as well as strives to address the interpretability for obtained models.

The aforementioned problems are both important and challenging. Considering multivariate or relational learning is de facto in time series analysis for many domains. However, the existing methods of compositional learning are inapplicable to extend to such a setting since the explosion in model space makes it infeasible to use. Learning compositional structures is already a time-consuming task. Although there are existing approximation methods, they do not work well for compositional covariances. This makes it even harder to propose a scalable approach without sacrificing model performances. Finally, analyzing hierarchical deep Gaussian processes is notoriously difficult especially when incorporating different covariance functions. Previous work focuses on a single case of covariance function and is difficult to generalize for many other cases.

The goal of this thesis is to propose solutions to the given problems. The first contribution of this thesis is a general framework for modeling multiple time series which provides descriptive relations between time series. Second, this thesis presents efficient probabilistic approaches to address the model search problem which previously is done by exhaustive enumerating evaluation. Furthermore, a scalable inference for Gaussian process is proposed, providing accurate approximation with guarantees of error bounds. Last but not least, to address the existing issues in deep Gaussian process, this thesis presents a unified theoretical framework to explain the pathology in deep Gaussian processes with better error bounds for various kernels compared to existing work and rates of convergence.



# Table of contents

<b>List of figures</b>	<b>vii</b>
<b>List of tables</b>	<b>xi</b>
<b>Notation</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Thesis scope . . . . .	1
1.2 Challenges . . . . .	2
1.3 The contributions of thesis . . . . .	3
1.3.1 A general framework for modeling covariance structure in multiple time series . . . . .	3
1.3.2 A scalable method for learning compositional kernel functions . . . . .	4
1.3.3 A theoretical understanding of extension to deep Gaussian process . . . . .	5
1.4 The outline of this thesis . . . . .	5
1.5 Publication notes . . . . .	5
<b>2 Gaussian process and the Automatic Statistician</b>	<b>7</b>
2.1 Weight-space view . . . . .	7
2.2 Function space . . . . .	8
2.3 Covariance function . . . . .	9
2.4 The Automatic Statistician System . . . . .	12
<b>3 Global relational kernel learning with local variations</b>	<b>15</b>
3.1 Introduction . . . . .	15
3.2 Review of Relational Kernel Learning . . . . .	16
3.3 Semi-Relation Kernel Learning . . . . .	17
3.4 Experimental Results . . . . .	20



3.4.1	Data sets . . . . .	20
3.4.2	Quantitative evaluations . . . . .	21
3.4.3	Qualitative Comparisons . . . . .	22
3.5	Related work and final remark . . . . .	23
<b>4</b>	<b>Selective compositional kernel discovery</b>	<b>25</b>
4.1	Introduction . . . . .	25
4.2	Latent Kernel Model . . . . .	27
4.2.1	Indian Buffet Process . . . . .	27
4.2.2	Model definition . . . . .	28
4.2.3	Properties . . . . .	29
4.2.4	Inference algorithm . . . . .	30
4.3	Model discovery in multiple time series . . . . .	33
4.4	Experimental evaluations . . . . .	36
4.4.1	Real-world time series data . . . . .	37
4.4.2	Qualitative results . . . . .	37
4.4.3	Quantitative results . . . . .	40
4.5	Related work and final remark . . . . .	41
<b>5</b>	<b>Kernel selection for Scalable GP</b>	<b>45</b>
5.1	Introduction . . . . .	45
5.1.1	Variational Sparse Gaussian process . . . . .	46
5.1.2	Shrinkage prior . . . . .	47
5.2	Kernel selection with shrinkage prior . . . . .	48
5.2.1	Kernel selection with Horseshoe prior . . . . .	49
5.2.2	Multi-inducing sparse Gaussian process . . . . .	50
5.3	Variational inference with shrinkage prior . . . . .	55
5.4	Detail of variational inference . . . . .	56
5.5	Experimental Evaluations . . . . .	59
5.5.1	Kernel function pool . . . . .	59
5.6	Related work and conclusion . . . . .	64
<b>6</b>	<b>Characterizing Deep Gaussian process</b>	<b>65</b>
6.1	Introduction . . . . .	65
6.2	Moment-generating function of distance quantity . . . . .	67

6.3	Analyzing dynamic systems with chaos theory . . . . .	68
6.4	Squared exponential kernel function . . . . .	69
6.5	Cosine kernel function . . . . .	72
6.6	Periodic kernel function . . . . .	73
6.7	Rational quadratic kernel function . . . . .	74
6.8	Spectral mixture kernel . . . . .	75
6.9	Extension to non-pathological cases . . . . .	76
6.10	Analysis of recurrence relations . . . . .	77
6.10.1	Identify the pathology . . . . .	77
6.10.2	Rate of convergence . . . . .	78
6.11	Experimental results . . . . .	78
6.11.1	Correctness of recurrence relations . . . . .	79
6.11.2	Justifying the conditions of pathology . . . . .	80
6.11.3	Using recurrence relations in DGPs . . . . .	83
6.11.4	High-dimensional data set with zero-mean DGPs . . . . .	84
<b>7</b>	<b>Conclusion and Future work</b>	<b>87</b>
7.1	Summary of contribution . . . . .	87
7.2	Future work . . . . .	87
7.3	Conclusion . . . . .	88
	<b>References</b>	<b>89</b>



# List of figures

2.1	An illustration of posterior when the number of data points increases. The model uncertainty is updated as the more data is added. . . . .	9
3.1	Plot of 9 time series extracted from stock data sets. Time series values are normalized. . . . .	17
3.2	Graphical model of SRKL model. Compared to RKL, SRKL includes an additional distinctive kernel for each data sequence. . . . .	19
3.3	An ablation study on negative log-likelihoods (NLL) of $k_S$ and $k_j$ . (a) Non-overfitting case. The NLLs on $k_j$ and $k_S$ decrease together; (b) Overfitting is observed starting from level 3 in the search grammar. The NLL on $k_S$ keeps decreasing from level 3. On the other hand, the total NLL on $k_j$ increases. . . . .	20
3.4	The first four plots are the posterior of SRKL for four currency exchange rates. The last plot is a shared kernel found by SRKL to explain financial changes. . . . .	21
4.1	Time series of Gold, Oil, NASDAQ, and USD index . . . . .	27
4.2	A toy example. Learning two lines using LKM. This data set consists of two samples generated from the same GP prior with a periodic kernel. . . . .	30
4.3	Comparison between the graphical model of (a) LKM and (b) relation model in the previous chapter. . . . .	31
4.4	An example of expansion in PSE. . . . .	34
4.5	An illustration of the convergence of $\nu$ . . . . .	36

4.6	IBP matrix $\mathbf{Z}$ in epileptic seizure data. The EEG recordings of five activities are considered: seizure ( <b>act1</b> ), located tumor ( <b>act2</b> ), identifying tumor ( <b>act3</b> ), eyes closed ( <b>act4</b> ), eyes open ( <b>act5</b> ). (a) Non-seizure. <i>Left</i> : learned $\mathbf{Z}$ for each activity (black: $z_{nk} = 0$ , white: $z_{nk} = 1$ ); <i>Right</i> : the GP posterior of three time series from <b>act5</b> with the corresponding decomposition. (b) Seizure. <i>Left</i> : learned $\mathbf{Z}$ from <b>act1</b> ; <i>Right</i> : posterior plot of first three time series from <b>act1</b> . The gray background plots indicate $z_{nk} = 0$ . . . . .	36
4.7	An example of pairwise comparison in GONU data set. The upper plots are the posterior distribution of two time series. The remaining plots contain shared components and individual components with descriptions and posteriors $\mathbf{f}_k \mathbf{x}_n$ for each time series. The blank in the individual components means “not available”. . . . .	39
4.8	RMSEs for each data set (9 stocks, 6 houses, 4 currencies, GONU) with corresponding methods. . . . .	40
4.9	Comparing Oil and USD index. This is extracted from the pairwise comparison of GONU data set. . . . .	41
5.1	The graphical model of two models. Solid and dashed lines indicate the connections modeled by two different kernel function $k_1$ and $k_2$ . (a): Sparse inducing GP. The inducing points $u_i$ is introduced as a proxy for the connections between $f_i$ . (b): Our approach. Inducing points are grouped. Each group represents an individual kernel $k_1$ or $k_2$ . . . . .	49
5.2	The posterior distributions between models. Here, $W_2$ is the Wasserstein-2 distance between a model and the true model. The posterior obtained from our approach is close to the true model as well as the full GP model. SVGP model struggles to fit the data. . . . .	51
5.3	Behavior of Horseshoe prior in kernel selection. Both models predicts the test data ( $\star$ ). The bar plots are the weights $w_i$ corresponding to $k_i$ . . . . .	60
5.4	Extrapolation on time series data sets. . . . .	60
5.5	First row: the weights $w_i$ in two cases. Second row: our kernel decomposition for airline data with three most significant components $\mathcal{GP}(\mu_i(\cdot), \Sigma_i(\cdot, \cdot))$ . The weights $w_i$ are showed at the upper-left corners. . . . .	61
5.6	GEFCOM data set. First row is the plot of training data. The next rows are the predictive posterior at test points. Our model outperforms the alternatives in term of root mean square error (RMSE) and test negative log-likelihood (NLL). . . . .	62

6.1	Studying the squared distance, $Z_n$ , between outputs of two consecutive layers. The asymptotic property (middle plot) of the recurrence relation of this quantity between two consecutive layers decides the existence of pathology for a very deep model. Here, $\theta$ indicates kernel hyperparameters. The middle plot is the bifurcation plot providing the state of DGP at very deep layer. The pathology is identified by the zero-value region where $\mathbb{E}[Z_n] \rightarrow 0$ . Note that this bifurcation plot is for illustration purpose only. . . . .	67
6.2	Finding the recurrence relation of the quantity $\mathbb{E}[(f_n(x) - f_n(x'))^2]$ between two consecutive layers. . . . .	67
6.3	Bifurcation plot of the logistic function $u_n = ru_{n-1}(1 - u_{n-1})$ . . . . .	68
6.4	(a): Bifurcation plot of the recurrence relation of SE kernel for $m = 1$ . (b): Contour plot of $u_n$ at layer $n = 300$ and $m = 1$ . The misalignment between the <b>red line</b> ( $\sigma^2/\ell^2 = 1$ ) and the zero-level contour is due to numerical errors. (c): Increase $m > \sigma^2/\ell^2$ to avoid pathology. . . . .	71
6.5	<i>Left</i> : Graphical model of input-connected construction suggested by [Duvenaud et al., 2014; Neal, 1995]. <i>Right</i> : The bifurcation plot of input-connected DGP. . . . .	77
6.6	Contour plots of $\mathbb{E}[Z_n]$ at $n = 300$ with respect to four kernel functions. . . . .	78
6.7	Bifurcation and contour plot of SE kernel for two cases $m = 2, 3$ . (a)-(b): $m = 2$ . (c)-(d): $m = 3$ . . . . .	79
6.8	Bifurcation plot of the recurrence of periodic kernel for $m = 1$ . First row: From left to right, $\ell$ is varied. Second row: $\sigma^2$ is varied. . . . .	79
6.9	RQ: Contour plots of $\mathbb{E}[Z_n]$ at $n = 300$ . The two contour plots share the same zero-value level. So that $\alpha$ does not decide the condition overcome the pathology. . . . .	80
6.10	(a-b) Paths to fixed points for two cases: RQ and SM. Iterations of RQ start from $x = 1.2$ and converge to 0. Those of SM start from $x = 0.6$ and converge to a point near 1. (c) Plot of all recurrence functions $h(x)$ . Note that $x$ is not input data but plays the role of $\mathbb{E}[Z_n]$ . . . . .	80
6.11	$\mathbb{E}[Z_n]$ computed from recurrence vs. empirical estimation of $\mathbb{E}[Z_n]$ for two kernel functions. . . . .	81
6.12	Trace of RMSDs. RMSDs converge to 0 when the pathology occurs. . . . .	81
6.13	High-dimensional SE: $\mathbb{E}[Z_n]$ computed from recurrence vs. empirical estimation of $\mathbb{E}[Z_n]$ . . . . .	81
6.14	SM kernel: $\mathbb{E}[Z_n]$ computed from recurrence vs. empirical estimation of $\mathbb{E}[Z_n]$ . . . . .	82
6.15	Contour plots of RMSDs at layer 100 for three kernels: PER, SM and RQ. . . . .	82

6.16	Dual-axis plot of the trajectory $\mathbb{E}[Z_n]/\sigma^2$ with $n$ running from 1 to $N$ and RMSE. Solid lines indicate the trajectories of $\mathbb{E}[Z_n]/\sigma^2$ projected on the left $y$ -axis. Star markers ( $\star$ ) indicate RMSEs projected on the right $y$ -axis. Dashed lines connect the $\mathbb{E}[Z_n]/\sigma^2$ and RMSE of the same $N$ . Here, the constrain coefficient $c_0 = 0.2$ . . . . .	83
6.17	Standard zero-mean DGPs: Results of Boston housing data set . . . . .	84
6.18	Constrained DGPs: Results of Boston housing data set . . . . .	84
6.19	Standard zero-mean DGPs: Results of diabetes data set . . . . .	85
6.20	Constrained DGPs: Results of diabetes data set . . . . .	85
6.21	(a-b) Loss landscape of two models. (c) Classification accuracy with respect to the number of layers, $N$ , and constrain coefficients, $c_0$ . . . . .	86

# List of tables

3.1	Negative log-likelihoods (NLLs), Bayesian Information Criteria (BIC), and root mean square errors (RMSEs) of CKL, RKL and SRKL . . . . .	21
4.1	RMSEs and NMLPs for each data set with corresponding methods (5 independent runs per method). In most cases, LKM has lower RMSEs and NMLPs compared to those of existing methods. . . . .	43
5.1	Extrapolation performance in UCI benchmarks. Results are aggregated from 10 independent runs. . . . .	59
5.2	Description of UCI data sets . . . . .	63
5.3	Description of heart, liver, pima data set . . . . .	63
5.4	Classification error (in %) on three data sets. . . . .	64





# Notation

$x$	scalar
$\mathbf{x}$	vector
$\mathbf{A}$	matrix
$\mathcal{N}$	Gaussian distribution
IG	Inverse Gamma distribution
$\mathcal{GP}$	Gaussian Process
$\mathbb{E}$	Expectation
KL	Kullback-Leibler divergence
CKL	Compositional Kernel Learning
RKL	Relational Kernel Learning
SRKL	Semi-Relational Kernel Learning
IBP	Indian Buffet Process
LKM	Latent Kernel Model
PSE	Partial Set Expansion
SVGP	Sparse Variational Gaussian Process
MultiSVGP	Multi-inducing Sparse Variational Gaussian process
DGP	Deep Gaussian Process
RMSE	Root Mean Squared Error
NLL	Negative Log-Likelihood



# Chapter 1

## Introduction

Compositionality is one of the important concepts to equip to machine learning models. It can be understood as the process of building structures from small and simple to complex and rich. Compositionality can be the key to bringing creativity to machines by allowing them to learn new models or generate new data. This concept can be recognized in many things around us from the way that the smallest atoms combine to form molecules to the macro level where galaxies are constituted from planets. In machine learning data, compositionality can be found in natural language processing where sentences are created from words. Another example is image data where it can contain multiple objects, i.e., trees, roads and cars. The question is how to learn or explore a certain type of composition efficiently since there are challenges due to the cardinality of compositional model space as well as the difficulty of model selection problems.

### 1.1 Thesis scope

This thesis aims to tackle the problem of learning composition structures in a way that is done in automatic manners and takes interpretability into account. The main studying model that this thesis focuses on is Gaussian process. Gaussian process is a flexible probabilistic model presenting several attractive properties including universal approximation and uncertainty quantification [Rasmussen and Williams, 2005]. To find an appropriate Gaussian process model, one may consider two aspects: model selection and model search. Under the Bayesian approach, models are compared based on model evidence which balances the trade-off between model fit and model complexity and reflects the notion of Occam's razor on preference simpler models over complex ones. Bayesian Information Criterion is a simple Laplace approximation of model evidence that is a key to evaluate models in existing work. However, the question of what measurement quantity is the most relevant is still controversial. In terms of model search, one

can create Gaussian process models based on some basic kernel functions and a set of rules and operators. Although generated models are capable to learn complex data well, the space in which they lie is open-ended.

This thesis considers the following problems. First, multiple time series are considered as the main target data on which an automated machine learning framework is built, resulting in interpretable models. The goal is to extract relations between these time series under compositionality representation. This is considered as the relational learning for compositional models. The second aspect is to improve the scalability that this thesis develops efficient methods to explore compositional model space for large-scale data. Finally, this thesis extends theoretical studies of a hierarchical deep version of Gaussian process [Damianou and Lawrence, 2013]. This approach is considered as a functional composition between hierarchical layers.

## 1.2 Challenges

Given a glimpse of problem settings, there are certain challenges. For the multiple time series, it is non-trivial to have a direct extension from existing models that work on single data settings. Although consider multivariate settings is a sensible approach since making use of relations between data can improve model generalization, such relations are rather complex. It is difficult to discover the most appropriate model that can characterize every individual data as well as capture relations between the multiple data.

The second challenge is that the space of compositional models is discrete and open-ended. Due to its discrete combinatorial nature, brute-force search by enumerating all possible models is infeasible. The existing approach of using greedy search tactics is still time-consuming. This problem can be cast into combinatorial or discrete optimization. However, solutions are not ready yet.

The third challenge is that the current framework is only applicable to small-scale data. In order to scale up for large-size data sets, incorporation with existing approximate Gaussian process methods is a natural extension. However, considering complex models or kernel functions, there exists some degradation in approximation power.

Lastly, there is an existing issue of deep Gaussian process model that, in some conditions, the model collapses at very deep layers. The behavior of this model is not well-studied for many kernel functions yet. It is necessary to have a comprehensive investigation before doing any further model selection task.

## 1.3 The contributions of thesis

This thesis proposes two models for learning covariance structure for multiple time series: Semi-Relational Kernel Learning (SRKL) and Latent Kernel Model (LKM). Semi-Relational Kernel Learning (Chapter 3) is an extension of Relation Kernel Learning (RKL), emphasizing learning global kernel structures but allows variants in individual time series. Latent Kernel Model (Chapter 4) no longer relies on global sharing assumptions. LKM automatically extracts shared information, indicates what kernel structures are different. LKM is a general model and maintains that all kernel structures are interpretable while some parts of SRKL are not. Under the treatment of Indian Buffet Process prior (IBP), LKM allows extracting the relations between multiple time series.

The second contribution (Chapter 5) is a scalable algorithm for learning compositional kernels. This chapter devises Multi-inducing Sparse Variational Gaussian Process (MultiSVGP) which is a new sparse Gaussian process model, aiming to improve approximation capacity for complex compositional kernel functions. MultiSVGP maintains a group of inducing points where each member in this group is responsible for an additive kernel component in compositional kernel functions. This approach can be demonstrated to have a better error bound compared to the traditional sparse Gaussian process. In the combination with MultiSVP, a Bayesian approach utilizing a shrinkage prior is used for selecting kernel functions.

The third contribution (Chapter 6) is a unified framework to analyze the pathological issue in deep Gaussian process. This framework presents a guideline for a given kernel function by considering the statistical characteristics when the kernel function takes distributional inputs. As a result, five common kernel functions are studied. The rate of convergence for each kernel is discussed. To avoid disastrous failure in learning DGP, this chapter proposes a regularization which constrains kernel hyperparameters staying in safe regions such that the pathology can be avoided.

### 1.3.1 A general framework for modeling covariance structure in multiple time series

This thesis proposes a general framework for tackling ubiquitous multiple data. A previous model, Relation Kernel Learning [Hwang et al., 2016], motivated by many existing works in statistical relational learning [Choi et al., 2010, 2015; Getoor and Taskar, 2007; Wang and Domingos, 2008], models a group of time series with an assumption that all of the time series are globally correlated. One can find that there are many real-world cases. For example, many

stocks share the same up-and-down pattern because they are influenced by the same law of finance or causal relations. The difference among a group of time series is the magnitude and scale among these. Relation Kernel Learning uses a single kernel function to describe this.

However, this model seems to underfit. To address this, this thesis proposes Semi-Relational Kernel Learning in Chapter 3 which maintains global kernel assumption like RKL but allows an individual kernel function for each time series. The responsibility of this individual kernel function is to fit the residual that the global kernel function may not fit the time series well. The output of this model still benefits from global kernel assumption, resulting in that the found kernel structure capture the informative description shared among time series. Moreover, this model achieves better predictive performance comparing to RKL and the existing approach [Duvenaud et al., 2013; Lloyd et al., 2014].

The previous setting is restricted in terms of the data collection that should guarantee the assumption where all time series should be strongly correlated. In practice, it takes a lot of care to gather such information. The question is how to make a model that does not require such data preparation. More importantly, how to make a model automatically return the relation between time series rather than fixating a relation assumption between them? To tackle this question, this thesis presents Latent Kernel Learning in Chapter 4. Indian Buffet Process (IBP) prior is used to model the binary latent variables that capture relations between time series.

### 1.3.2 A scalable method for learning compositional kernel functions

This thesis develops a scalable approximate Gaussian process in the specific case of compositional kernel learning. Gaussian process is known to be not scalable due to computation complexity  $\mathcal{O}(n^3)$  with  $n$  is the number of data. This is the computation cost for a single model. Searching among a huge number of unscalable models becomes impossible when the size of data increases. Chapter 5 presents Multi-inducing Sparse Variational Gaussian Process (MultiSVGP) which mitigates the cubic time complexity of Gaussian process, maintains good approximations for the large-sized data sets. MultiSVGP extends sparse Gaussian process methods [Hensman et al., 2013; Snelson and Ghahramani, 2006] for compositional kernel functions by dividing the responsibility of inducing points according to individual additive kernel function in the kernel sum. Interestingly, because of the strategy on structuring inducing points, the error bound of the approximate distribution compared to the true posterior distribution is smaller than that of the traditional approach in sparse Gaussian process. To facilitate the search procedure, a probabilistic kernel selection combines with MultiSVGP based on a shrinkage prior called Horseshoe prior. As the results, the proposed approach yields 25 times faster than [Duvenaud

et al., 2013; Lloyd et al., 2014] and 4-10 times faster than alternative approaches using sparse Gaussian process [Kim and Teh, 2018]. Experiments further verify that the proposed model outperforms the state-of-the-art Gaussian process methods in extrapolation tasks.

### 1.3.3 A theoretical understanding of extension to deep Gaussian process

In the vision to perform kernel selection for a new class of models, deep Gaussian processes, in which Gaussian process layers are hierarchically stacked in a similar manner with deep neural networks. The first step to overcome existing issues in learning deep Gaussian process. Chapter 6 establishes the theoretical foundation by analyzing the characteristics of deep Gaussian process models given a kernel function. Five common kernel functions including squared exponential function, cosine function, periodic function, rational quadratic function and spectral mixture kernel function. The key findings of the analysis include the condition to avoid pathology for each kernel function and the rate of convergence to fixed points. Also, the result shows the spectral mixture kernel function does not. From the theoretical analysis, this chapter provides a regularization technique to alleviate the difficulty in learning deep Gaussian process. It is done by applying constraint which forcefully avoids the unsafe pathological kernel parameters. Empirical experiments demonstrate that we can learn zero-mean deep Gaussian process models while existing work fails to train such models.

## 1.4 The outline of this thesis

Chapter 2 provides backgrounds related to Gaussian processes and an introduction of the Automatic Statistician framework. The main contributions of the thesis are described in Chapter 3, 3, 5 and 6. Chapter 3 presents Semi-Relational Kernel Learning. Chapter 4 generalizes Semi-Relational Kernel Learning. Chapter 5 proposes a better approximation method, Multi-inducing Sparse Variational Gaussian Process with a theoretical guarantee as well as kernel selection with a shrinkage prior. Chapter 6 gives theoretical analyses of deep Gaussian process on the condition to avoid pathology. The thesis ends with Chapter 7 containing a summarization and open directions for future work.

## 1.5 Publication notes

This thesis is composed from (or a part of) the following publications with revision and adaption:

- Second model in Hwang et al. [2016]: Chapter 3.



- Tong and Choi [2019]: Chapter 4
- Tong et al. [2021]: Chapter 5.
- Tong and Choi [2021]: Chapter 6.

## Chapter 2

# Gaussian process and the Automatic Statistician

This chapter presents an introduction of Gaussian process (GP) by providing the weight-space view, then transitioning to functional view. Several basic kernel functions are introduced. What follows is the Automatic Statistician framework.

### 2.1 Weight-space view

Consider a data set  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  where  $\mathbf{x}$  is an input vector of dimension  $D$  and  $y$  is the corresponding scalar output.

In the linear regression problem, one may consider the following

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}, \quad y = f(\mathbf{x}) + \varepsilon,$$

where  $\varepsilon$  is a white noise. In Bayesian setting, the parameter  $\mathbf{w}$  is placed an prior. In this problem, we assume the prior over  $\mathbf{w}$  is a Gaussian distribution as

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

In many machine learning problems, working in the input space is not enough. Projecting input into a new space is more favorable for models, e.g. easy to find a linear separator. Let  $\phi$  be the feature map transforming  $\mathbf{x}$  to  $\phi(\mathbf{x})$ . The goal is to find the linear model over this new feature space.

$$f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}, \quad y = f(\mathbf{x}) + \varepsilon,$$

With this hypothesis, the mean and covariance of function evaluations  $f(\mathbf{x})$  can be obtained easily by

$$\begin{aligned} \mathbb{E}[f(\mathbf{x})] &= 0, \\ \mathbb{E}[f(\mathbf{x})^\top f(\mathbf{x}')] &= \phi(\mathbf{x})^\top \phi(\mathbf{x}'), \end{aligned} \tag{2.1.1}$$

To sum up, this Bayesian linear regression models center around the probabilistic derivation or understanding over weights  $\mathbf{w}$ . However, one can move to a functional view where the probabilistic attention is placed on the function evaluation  $f(\mathbf{x})$ . The above equations are a first glimpse for this transition. For example, the dot product representation usually defines a kernel function  $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$ .

## 2.2 Function space

**Definition 2.2.1.** A Gaussian process is a set of random variables in which any of its subset follows a multivariate Gaussian distribution.

Gaussian process [Rasmussen and Williams, 2005] is a prior over function

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

where  $m(\mathbf{x})$  is the mean function, usually set to 0, and  $k(\mathbf{x}, \mathbf{x}')$  is the kernel function.

It is clear that the mean and covariance coincide with Equation (2.1.1). By the definition of Gaussian process, they are written as

$$\begin{aligned} \mathbb{E}[f(\mathbf{x})] &= m(\mathbf{x}) = 0 \\ \text{Cov}(f(\mathbf{x}), f(\mathbf{x}')) &= \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] - \underbrace{\mathbb{E}[f(\mathbf{x})]\mathbb{E}[f(\mathbf{x}')]}_0 = \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}') \end{aligned}$$

In contrast to the weight-space view in the previous section, the weights do not explicitly appear to represent data. On the other hand, Gaussian process directly make the assumption over  $f(\mathbf{x})$  which is imposed a prior distribution and used to model  $\mathbf{y}$ .

To predict the function value at a test point  $\mathbf{x}_*$ , we use the joint assumption between training data and test data. Let us consider data set with three points  $\mathbf{x}_1, \mathbf{x}_2$  and  $\mathbf{x}_3$ . The joint probability

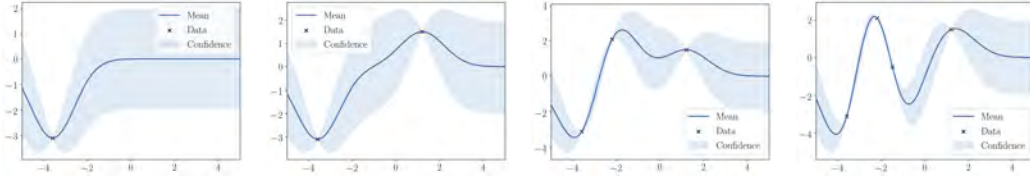


Fig. 2.1 An illustration of posterior when the number of data points increases. The model uncertainty is updated as the more data is added.

distribution of these points and new point  $\mathbf{x}_*$  is

$$\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ f(\mathbf{x}_3) \\ f(\mathbf{x}_*) \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & k(\mathbf{x}_1, \mathbf{x}_3) & k(\mathbf{x}_1, \mathbf{x}_*) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & k(\mathbf{x}_2, \mathbf{x}_3) & k(\mathbf{x}_2, \mathbf{x}_*) \\ k(\mathbf{x}_3, \mathbf{x}_1) & k(\mathbf{x}_3, \mathbf{x}_2) & k(\mathbf{x}_3, \mathbf{x}_3) & k(\mathbf{x}_3, \mathbf{x}_*) \\ k(\mathbf{x}_*, \mathbf{x}_1) & k(\mathbf{x}_*, \mathbf{x}_2) & k(\mathbf{x}_*, \mathbf{x}_3) & k(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix} \right).$$

The predictive distribution for  $f(\mathbf{x}_*)$  can be obtained from the conditional Gaussian distribution

$$\begin{aligned} f(\mathbf{x}_*) | \mathbf{X}, \mathbf{f} &\sim \mathcal{N}(\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)) \\ \text{with } \mu(\mathbf{x}_*) &= \mathbf{k}(\mathbf{x}_*, \mathbf{X}) \mathbf{K}^{-1}(\mathbf{X}, \mathbf{X}) \mathbf{f} \\ \sigma^2(\mathbf{x}_*) &= k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*, \mathbf{X}) \mathbf{K}^{-1}(\mathbf{X}, \mathbf{X}) \mathbf{k}(\mathbf{X}, \mathbf{x}_*) \end{aligned}$$

This predictive distribution is in a simple closed form, providing an attractive way to quantify the uncertainty of model prediction. This property becomes helpful for other methods including Bayesian optimization and Bayesian quadrature.

There is an equivalence between the predictive posterior of Gaussian process and that of Bayesian linear model in the previous section. That is, one can obtain the same posterior when replacing the dot product between two features by a kernel function. The connection is presented clearly in [Rasmussen and Williams \[2005\]](#).

### 2.3 Covariance function

Modeling kernel function is one of important problems in Gaussian process. Because the value of kernel function between  $\mathbf{x}$  and  $\mathbf{x}'$  describes how the corresponding function evaluations  $f(\mathbf{x})$  and  $f(\mathbf{x}')$  are correlated.

In general kernel methods, a function  $k(\cdot, \cdot)$  mapping a pair of input to  $\mathbb{R}$  is defined as a kernel. Let  $\mathcal{K}$  be the integral operator with respect to the kernel function  $k(\mathbf{x}, \mathbf{x}')$ .

$$(\mathcal{K}f)(\mathbf{x}) = \int k(\mathbf{x}, \mathbf{x}')f(\mathbf{x}')d\mu(\mathbf{x}'),$$

with  $\mu$  is a probability measure. A kernel function  $k(\mathbf{x}, \mathbf{x}')$  has two common properties:

- It is symmetric. That is, the kernel function is invariant when exchanging between two inputs in the pair, e.g.,  $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$ .
- A kernel function is positive semi-definite.

$$\int \int k(\mathbf{x}, \mathbf{x}')f(\mathbf{x})f(\mathbf{x}')d\mu(\mathbf{x})d\mu(\mathbf{x}') \geq 0.$$

Suppose that data points  $\{\mathbf{x}_i | i = 1, \dots, n\}$ , the corresponding covariance matrix denotes as  $\mathbf{K}$ , having  $[\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ . The covariance matrix  $\mathbf{K}$  is a symmetric and positive semidefinite matrix.

The family of kernel functions can be divided into two categories: stationary and non-stationary. The stationary kernel function can instead consider input  $\boldsymbol{\tau} = \mathbf{x} - \mathbf{x}'$ .

**Theorem 2.3.1** (Bochner's theorem). *A kernel function  $k$  is a weakly stationary kernel function if and only if it can be represented as*

$$k(\boldsymbol{\tau}) = \int \exp(2\pi i \mathbf{s} \boldsymbol{\tau}) p(\mathbf{s}) d\mathbf{s}.$$

where  $p(\mathbf{s})$  is the spectral density function over  $\mathbf{s}$ ,  $i$  is the imagine unit ( $i^2 = -1$ ).

This theorem can be used to derive random Fourier features [Rahimi and Recht, 2008], or spectral mixture kernel [Wilson and Adams, 2013].

The following contains the description of several common kernel functions:

**Squared exponential kernel function** The squared exponential (SE) kernel function is defined as

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

Given two inputs  $\mathbf{x}$  and  $\mathbf{x}'$ , the covariance is high when inputs are close. The lengthscale hyperparameter  $\ell$  helps to rescale the distance between  $\mathbf{x}$  and  $\mathbf{x}'$ .

**Matérn kernel function** The family of Matérn kernel functions is defined by

$$k(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu}r}{\ell} \right)^\nu K_\nu \left( \frac{\sqrt{2\nu}r}{\ell} \right),$$

where  $r = \|\mathbf{x} - \mathbf{x}'\|_2^2$ ,  $\nu$  and  $\ell$  are positive hyperparameters and  $K_\nu$  is a modified Bessel function. Note that if we send  $\nu \rightarrow \infty$ , the kernel function asymptotically becomes the SE kernel function. In practice, there are cases where  $\nu = \frac{3}{2}$  and  $\nu = \frac{5}{2}$  used because the kernel functions are simple and in closed forms, e.g.,

$$k_{\nu=3/2}(r) = \left( 1 + \frac{\sqrt{3}r}{\ell} \right) \exp \left( -\frac{\sqrt{3}r}{\ell} \right),$$

$$k_{\nu=5/2}(r) = \left( 1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2} \right) \exp \left( -\frac{\sqrt{5}r}{\ell} \right).$$

**Rational quadratic kernel function** The rational quadratic (RQ) is defined as

$$k(\mathbf{x}, \mathbf{x}') = \left( 1 + \frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\alpha\ell^2} \right)^{-\alpha},$$

where  $\alpha$  is the scale parameter. This kernel function is originated from integrating out the lengthscale hyperparameter of SE kernel function in which the inverse of this lengthscale follows a Gamma distribution.

**Periodic kernel function** The periodic (PER) kernel function is constructed based on the idea of wrapping input  $\mathbf{x}$  into a new feature space  $\mathbf{u}(\mathbf{x}) = [\cos(\mathbf{x}), \sin(\mathbf{x})]^\top$ . Then, the squared exponential kernel function takes the input in this feature space to give the periodic kernel as

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

where  $p$  is a hyperparameter encoding the periodicity.

**Linear kernel** Unlike the previous kernel functions, the linear (LIN) kernel function is a nonstationary kernel. It is defined as following:

$$k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \ell)^\top (\mathbf{x}' - \ell).$$

This kernel function is a variant of dot-product kernel function,  $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$  by introducing a hyperparameter,  $\ell$ , which indicates the shift in location of inputs.

**Kernel construction** Kernel construction can be done by the addition and multiplication. Given two kernel function  $k_1$  and  $k_2$ , one can generate a new kernel function by

$$k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$$

$$k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$$

This is also known as kernel tricks like other kernel methods. The obtained kernel functions satisfy any properties of a kernel function including symmetric positive definite.

## 2.4 The Automatic Statistician System

This section presents the background of the Automatic Statistician system. The Automatic Statistician or **Automatic Bayesian Covariance Discovery (ABCD)** aims to mimic and automate the process of statistical modeling [Duvenaud et al., 2013; Ghahramani, 2015; Grosse et al., 2012; Lloyd et al., 2014; Steinruecken et al., 2019]. There are three main components in this framework: language of models, search procedure and report generations.

**Language of models** The ingredients to construct Gaussian process models include a grammar over kernels with a set of base kernels and kernel operators. The base kernels are: SE (squared exponential), LIN (linear), PER (periodic). The operators consist of + (addition),  $\times$  (multiplication). As composed kernels get more complex, the corresponding generated models become more expressive to fit complex data.

**Search procedure** The search procedure is done in a greedy manner. That is, the language of models generates candidate models. Then, all of them are optimized by maximizing log likelihoods. A model is selected based on the trade-off between model and data complexity. Let  $\mathcal{M}$  be the hypothesis model. Lloyd et al. [2014] selects the most appropriate model by the Bayesian Information Criteria (BIC) [Schwarz, 1978.]:

$$BIC(\mathcal{M}) = \underbrace{-2 \log p(\mathcal{D}|\mathcal{M})}_{\text{data fit}} + \underbrace{|\mathcal{M}| \log N}_{\text{complexity penalty}}.$$

Here,  $|\mathcal{M}|$  is the number of free parameters in the model. BIC is the results of a rough Laplace approximation of model evidence. Then, this selected model is the input of the language of models to create new candidates.

**Automatic generated explanation of models** The compositional kernels resulted from the search procedure are transformed into a sum of products of base kernels. Each structural product of kernels is interpreted under natural-language expressions. For example,

$$\underbrace{\text{SE}}_{\text{smooth change}} \times \underbrace{\text{LIN}}_{\text{linearly varying amplitude}} \times \underbrace{\text{PER}}_{\text{periodic function}}$$

All descriptions are gathered to produce a complete report with visualized plots and human-friendly analyses.





## Chapter 3

# Global relational kernel learning with local variations

In the real world, many events and objects are governed by the same causes. Consequently, data generated from these events and objects usually shares similar patterns. The goal of this chapter is to identify models not only can describe the sharing information between multiple time series but also fit well for each time series.

This chapter presents Semi-Relational Kernel Learning (SRKL) which is my contribution out of two models presented in [Hwang et al., 2016].

### 3.1 Introduction

The recent advance in learning structure covariance which is known as the Automatic Bayesian Covariance Discovery (ABCD) framework, provides powerful Gaussian process models which are able to fit complex real-world data well [Duvenaud et al., 2013; Lloyd et al., 2014]. However, learning such compositional covariance for single (univariate) time series may not be informative enough to describe the actual characteristics or causes of underlying data generative process. Instead, multivariate time series are often considered where we take many variables into account. For example, in economics, exchange rates depends on other variables such as gross domestic products.

To address the above issue, as the part of [Hwang et al., 2016], Relational Kernel Learning (RKL) proposes an approach by marrying statistical relational learning concepts [Belle et al., 2015; Choi et al., 2010, 2011a,b; Wang and Domingos, 2008] and the compositional kernel learning (CKL) from ABCD framework [Duvenaud et al., 2013]. Specifically, in order to model

multiple time series, RKL assumes a global kernel function which is shared among all time series. To deal with the variations in magnitudes between these time series, RKL introduces scale and shift coefficients to each time series which are optimized jointly with kernel hyperparameters. The globally shared kernel function is the main target, is searched in the same manners as CKL. This kernel function is considered as the invariance between all time series. RKL not only strives for compactness and simplicity but also focuses on extract the global pattern among multiple sequences. Therefore, this model provides interesting qualitative results by finding interpretable components which can explain actual causes and events.

In many real-world cases, the assumption made by the RKL model is rather too strong, usually leading to underfit. Because there are possible variations between individual data sequences even though they have a common global structure. This chapter presents a model which is called Semi-Relational Kernel Learning (SRKL). This model solves the underfit shortcoming of RKL while keeping the spirit of RKL by encouraging a shared covariance function between multiple data. By introducing a distinctive kernel function for each time series and  $\cdot$ . The role of this additional kernel function is to fit the residual between data and the globally shared kernel function. The realm of such kind of designing kernel function resembles the recent concept of meta-learning [Finn et al., 2017; Schmidhuber, 1987] and global local forecaster with deep neural network approach [Sen et al., 2019].

This chapter organizes as follows. The relational kernel learning is reviewed to provide a detailed background to motivate the proposed model. Following up, Semi-Relational Kernel Learning which is the main contribution of this chapter, presents individual kernel for each time series. Then, we demonstrate that the proposed model gives more accurate prediction compared to baseline models and RKL.

## 3.2 Review of Relational Kernel Learning

This section reviews the model definition of Relation Kernel Learning (RKL). Before diving into details, we take stock market as an example to illustrate strong correlation between a group of time series. Figure 3.1 is a motivated example for time series having common global pattern of dynamics.

**Model definition** Let us denote  $M$  time series as  $\mathcal{D} = \{d_1, \dots, d_M\}$  where each  $d_j$  represents the  $j$ -th time series. The main assumption on  $\mathcal{D}$  is that these time series resembles each other in terms of dynamic pattern (see 3.1). Relational Kernel Learning (RKL) defines a set of kernel

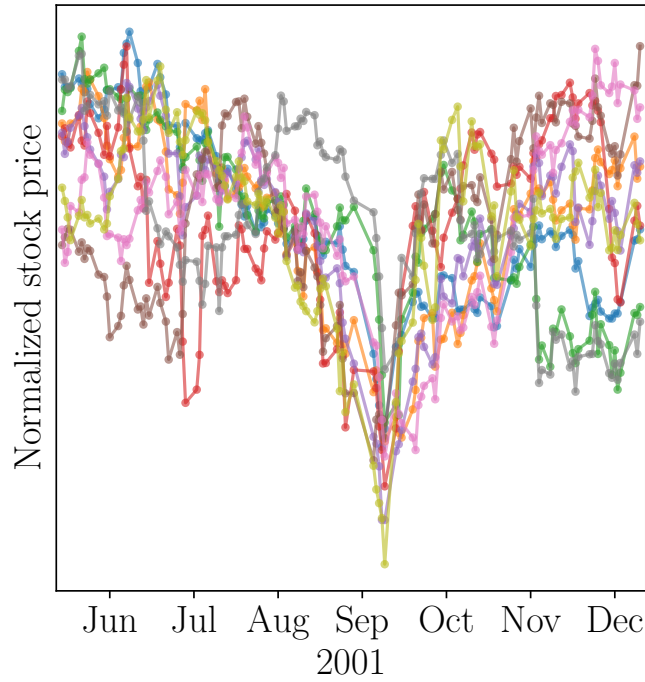


Fig. 3.1 Plot of 9 time series extracted from stock data sets. Time series values are normalized.

functions:

$$\{k_j(\cdot, \cdot) = \sigma_j^2 k_S(\cdot, \cdot) + s_j^2 | d_j \in \mathcal{D}, j = 1, \dots, M\}.$$

This model aims to find the kernel function  $k_S$  which shares among all  $d_j$ . Here  $\sigma_j$  and  $s_j$  are respectively a scale parameter and shift parameter. These parameters are introduced to tackle the difference in magnitudes between time series. All GP hyperparameters and scale and shift parameters are learned by maximizing the log-likelihood which is defined as

$$\log p(\mathcal{D} | k_S, \{\sigma_j\}_{j=1}^M, \{s_j\}_{j=1}^M) = \log \prod_{j=1}^M p(d_j | k_S, \sigma_j, s_j) = \sum_{j=1}^M \log p(d_j | k_S, \sigma_j, s_j).$$

The more detail of this model can be found in [Hwang et al., 2016].

### 3.3 Semi-Relation Kernel Learning

Given a brief introduction of RKL in the previous section, this section presents Semi-relation kernel learning which is the main contribution in this chapter.

## Model definition

Semi-Relational Kernel Learning (SRKL) defines a set of kernel functions as

$$\{k_j(\cdot, \cdot) = \sigma_j^2 k_S(\cdot, \cdot) + k_{d_j} + s_j^2(\cdot, \cdot) | d_j \in \mathcal{D}, j = 1, \dots, M\},$$

where all time series share the same kernel  $k_S$ ,  $k_{d_j}$  is the distinctive kernel function assigned to  $j$ -th sequence. By metaphorically describing time series as trees, ones can say that all of trees have the common shape of trunk part represented by the shared kernel function. The distinctive kernel function is to model the residual or remaining part of time series which can be considered as the small branches and leaves.

Unfortunately, the search can be done to explore both the shared kernel and  $M$  distinctive kernels. This is because the search space exponentiall increases. Here, if  $n$  is the number of possible kernels on each search grammar tree, there are  $\mathcal{O}(n^{M+1})$  number of models in SRKL. To prevent this exhaustive search,  $M$  distinctive kernel functions are not searched by the search grammar by fixed to be the spectral mixture (SM) kernel functions [Wilson and Adams, 2013]

$$k(\tau) = \sum_{q=1}^Q w_q \prod \exp\{-2\pi^2 \tau_p^2 v_q^{(p)}\} \cos(2\pi \tau_p \mu_q^{(p)})$$

where  $Q$  is the number of mixture components,  $\tau = x - x'$  is a  $P$  dimensional vector. Choosing SM kernel function is natural since its expressiveness is suitable to fit the residual of each time series.

Interestingly, the proposed model has a connection to meta-learning problems [Finn et al., 2017; Schmidhuber, 1987]. Meta-learning problems considers several tasks. In some model-based meta-learning approaches, models usually are built from neural networks in which there are shared parameters between tasks and task-specific parameters. This is similar to SRKL in designing kernel functions.

## Learning SRKL

The learning procedure of SRKL is described in Algorithm 1. For each depth  $s$ , the search grammar  $G$  generates composite kernels. For each kernel in the search space, the optimization problem considers (1) shared hyperparamters  $\theta_S$  in the shared kernel function, (2) scale factors  $\sigma_1, \dots, \sigma_M$  which is similar to RKL, distinctive hyperparameters  $\theta_1, \dots, \theta_M$  in distinctive kernel function. By minimizing the negative log likelihood of data on kernels  $\mathbf{K}$ , the optimal hyperparameters and value are obtained to make further comparison. The shared kernel is

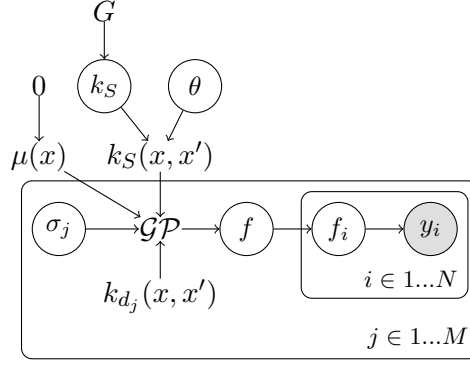


Fig. 3.2 Graphical model of SRKL model. Compared to RKL, SRKL includes an additional distinctive kernel for each data sequence.

---

**Algorithm 1** Semi-Relational Kernel Learning

---

**Require:** data  $\mathcal{D} = \{d_1, \dots, d_M\}$ , grammar  $G$ , maximum depth of search  $s$

- 1: Initialize with empty candidate set  $\mathcal{K} \leftarrow \emptyset$
  - 2: **for**  $i \in 0 \dots s$  **do**
  - 3:    $K_S \leftarrow \text{expand}(G)$
  - 4:    $\Theta \leftarrow \emptyset$
  - 5:   **for**  $k_S \in K_S$  **do**
  - 6:     Initialize  $\theta^0 \leftarrow (\theta_S^0, \theta_1^0, \dots, \theta_M^0, \sigma_1^0, \dots, \sigma_M^0)$
  - 7:      $k_j(\theta^0) \leftarrow k_S(\theta_S^0, \sigma_j^0) + k_{d_j}(\theta_j^0), j = 1 \dots M$
  - 8:      $\theta^* \leftarrow \text{argmin} \sum_{j=1}^M -\log p(\mathcal{D} | k_j(\theta))$
  - 9:      $\Theta \leftarrow \Theta \cup \{(k_S, \theta^*)\}$
  - 10:   **end for**
  - 11:    $(\hat{k}_S, \hat{\theta}) \leftarrow \text{argmin}_{(k_S, \theta) \in \Theta} \text{BIC}(k_S, \mathcal{D})$
  - 12:    $\mathcal{K} \leftarrow \mathcal{K} \cup (\hat{k}_S, \hat{\theta}, \hat{\sigma})$
  - 13: **end for**
  - 14: **return**  $\mathcal{K}$
- 

selected by the BIC score on shared kernel  $K_S$  in which the likelihood is computed by summing all likelihoods in each time series w.r.t. the shared kernel.

There is a compromise between  $k_s$  and  $k_{d_j}$  observed during the learning procedure. When the shared kernel is coarse and not expressive enough at the several initial depths, the distinctive kernel fit the residual gap data and the shared kernel. When the search grammar goes further,  $k_S$  becomes more complex and now  $k_{d_j}$  can accommodate to the residual part which is not fitted by  $k_S$ , yet. As  $k_S$  generated by the search grammar is expressive enough,  $k_{d_j}$  will make no improvement on  $k_j$ . We identify that the overfitting phenomena occurs when the negative log-likelihood made by  $k_j$  takes over the negative log-likelihood made by  $k_S$ . Figure 3.3 provides an example of overfitting in learning SRKL.

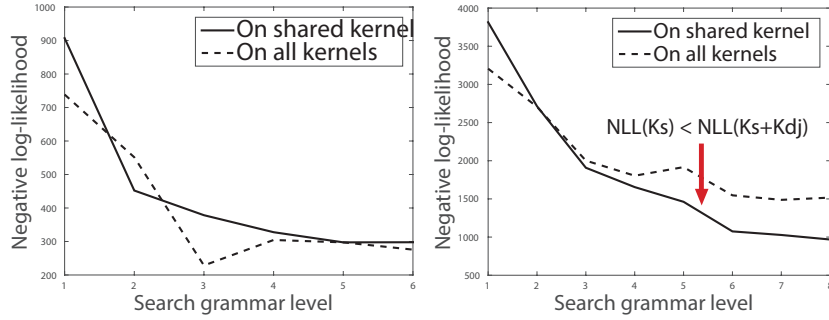


Fig. 3.3 An ablation study on negative log-likelihoods (NLL) of  $k_S$  and  $k_j$ . (a) Non-overfitting case. The NLLs on  $k_j$  and  $k_S$  decrease together; (b) Overfitting is observed starting from level 3 in the search grammar. The NLL on  $k_S$  keeps decreasing from level 3. On the other hand, the total NLL on  $k_j$  increases.

### 3.4 Experimental Results

This section presents the quantitative and qualitative evaluations between RKL, SRKL and ABCD system.

#### 3.4.1 Data sets

##### Stock market

Nine most valuable stocks including GE (General Electric), MSFT (Microsoft), XOM (Exxon Mobil), PFE (Pfizer), C (Citigroup), WMT (Walmart), INTC (Intel Corporation), BP (BP) and AIG (American International Group) are selected based on the market capitalization ranks as of 2001 [von Alten, 2001]. The data sets are retrieved from Yahoo finance [Yahoo Inc., 2015] with time starting from 2001-05-29 to 2001-12-25. Each stock historical data contains 129 data points. The total number of data points is 1161(=129×9). Note that the time period includes the September 11 event. Observe that all stock values show a sudden drop After the 9/11 attacks and gradual recovery as time goes on (see Figure 3.1). Three different learning settings for this data set are considered: STOCK3, STOCK6, and STOCK9.

##### Housing Market

The housing price data set are collected from top-6 selected cities in US including Chicago, Los Angeles, , New York, San Francisco, San Diego and, Phoenix. The selection is based on the population of these cities [United States Census Bureau, 2014]. Time series is retrieved from the beginning of 2004 to the end of 2013 with monthly granularity. Each data set has 120 data points. The total number of data point in the whole data set is 720. In this data set, it is observed that

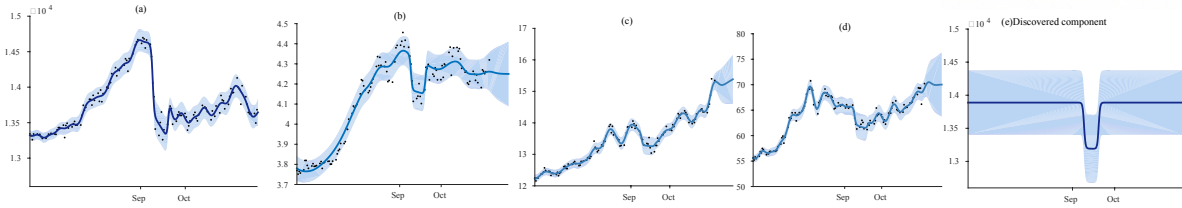


Fig. 3.4 The first four plots are the posterior of SRKL for four currency exchange rates. The last plot is a shared kernel found by SRKL to explain financial changes.

Data set	Negative log likelihood			Bayesian Information Criteria			Root mean square error		
	CKL	RKL	SRKL	CKL	RKL	SRKL	CKL	RKL	SRKL
STOCK3	332.75	311.84	<b>304.05</b>	750.65	<b>665.09</b>	1251.62	0.40	0.78	<b>0.38</b>
STOCK6	<b>972.00</b>	1007.09	988.14	2219.71	<b>2066.18</b>	3333.21	3.69	5.75	<b>1.22</b>
STOCK9	1776.31	1763.96	<b>1757.11</b>	3985.03	<b>3626.00</b>	5633.33	8.35	9.77	<b>4.85</b>
HOUSE2	<b>264.69</b>	304.29	310.38	<b>634.00</b>	<b>634.76</b>	905.76	6.58	<b>2.75</b>	3.12
HOUSE4	594.79	<b>586.81</b>	1249.82	1424.18	<b>1221.88</b>	3326.94	5.84	3.66	<b>2.22</b>
HOUSE6	<b>849.64</b>	891.09	1495.40	2100.62	<b>1876.47</b>	4339.54	7.96	5.33	<b>3.10</b>
CURRENCY4	<b>578.35</b>	617.77	693.76	<b>1165.82</b>	1291.77	2269.17	330.00	282.24	<b>201.56</b>

Table 3.1 Negative log-likelihoods (NLLs), Bayesian Information Criteria (BIC), and root mean square errors (RMSEs) of CKL, RKL and SRKL

in there is a peak around 2007. It is followed by a drop in 2009 due to the subprime mortgage crisis. Three learning settings are considered: HOUSE2, HOUSE4, and HOUSE6.

### Emerging Currency Market

The currency data set contains 4 currency exchange rates including Indonesian Rupiah (IDR), South African Rand (ZAR), Russian Rouble (RUB) and Malaysian Ringgit (MYR). The time index in this data is from 2015-06-28 to 2015-12-30 and are retrieved from Yahoo Finance [Yahoo Inc., 2015]. Specifically, each time series is 132 currency values. A key observation in this data set is that the financial market greatly fluctuated from the end of September 2015 to the beginning of October 2015. This is because of several economic events including FED’s announcement about policy changes in interest rates and China’s foreign exchange reserves falls. We call this data set as CURRENCY4.

#### 3.4.2 Quantitative evaluations

The baseline is the compositional kernel learning (CKL) where each time series is learned individually. The quantitative measurements are summed from the results of each time series. To compare between CKL, RKL, SRKL, three evaluation criteria are considered: negative log-likelihood (NLL) and BIC on training data, and root mean square error on test (extrapolation) data. Table 3.1 presents all experimental results with corresponding criteria.



## Negative log-likelihood and BIC

Table 3.1 shows the negative log-likelihoods and BIC scores in all data sets. It is clear that RKL has better BIC scores in most of data sets. Because it considers a fewer number of hyperparameters by sharing parameters among multiple time series. Similar to RKL, SRKL focuses on finding shared kernel among multiple data. However, SRKL maintains high BIC scores due to the number of hyperparameters in SM kernel. Moreover, the determinant term,  $\log \det(K_S + K_{d_j})$  in the negative log-likelihoods, penalizes more in SRKL than that of other models [Rasmussen and Williams, 2005].

## Extrapolation performance

Extrapolation is measured by the root mean squared error (RMSE) of the prediction on future events. The test data sets of stock data, housing data, currency data contains the next 14 days, 13 months and 13 days of data respectively.

According to Table 3.1, SRKL outperforms on most of data sets although it has higher BIC scores and NLLs. This is because SRKL overcomes the underfit issue in RKL by having SM kernel to complement the shared covariance. While it maintains the generalization which benefits from sharing information between multiple time series.

### 3.4.3 Qualitative Comparisons

RKL and SRKL can find kernel components which are dominant in multiple sequences better than CKL. This is because multiple data contains more information and evidence to decide whether a signal is really dominant. It can be shown by the case of US stock data where CKL cannot extract the drop after the 9/11 but recognizes the drop by a smooth change. On the other hand, RKL can explain this even by a time window. Another example is in currency exchange rate data where SRKL also captures a qualitatively important compositional kernel shortly written as  $CW(SE + CW(WN + SE, WN), CONST)$ . The second change-window kernel indicates a time period from mid September 2015 to mid October 2015 (see Figure 3.4). This can be related to the big financial changes, e.g., announcements on the change in interest rates by FED. CKL captures a change-point on only one currency for Indonesian Rupiah. The other results from CKL do not show this change.

### 3.5 Related work and final remark

The advances in compositional kernel learning have been studied in [Duvenaud et al., 2013; Lloyd et al., 2014] showing a great improvement comparing the the default setting by using only squared exponential in learning Gaussian processes. In contrast to this search-based approach, there are optimization-based methods considering learning multiple kernel from linear combinations of kernels [Bach et al., 2004; Wilson and Adams, 2013].

The idea of learning the relation between multiple objects or multiple data is stem from statistical relational learning [Belle et al., 2015; Choi et al., 2010, 2011b, 2015; Getoor and Taskar, 2007; Wang and Domingos, 2008]. The model proposed in this chapter concerns about the case that the studying data are time series. More importantly, this model further addresses the interpretability

In conclusion, this chapter provides an approach to extract shared information among groups of time series. The residual of each time series is fitted with expressive kernel function. The model demonstrates an outrun performance in prediction in multiple settings. The model explanation is further improved with the consideration of global information among data.



## Chapter 4

# Selective compositional kernel discovery

In many real-world data, there are many data that are not aligned perfectly. This can be due to the way of choosing or collecting data. This chapter investigates regimes that multiple data is not well-prepared or even comes from different sources or domains. The questions are: Can the model find which portions of models indicate the common structure? Can the model give us how much any arbitrary pair within the multiple data are correlated?

The model in this chapter is introduced in [Tong and Choi, 2019], providing a general solution compared to the Semi-Relational Kernel Learning in the previous chapter.

### 4.1 Introduction

There are numerous important real-world applications in time series analysis including signal processing and financial market. When considering multiple correlated data sources, a model that takes a group structure into account often shows competitive predictive performance [Yuan and Lin, 2006]. It is important to study how correlated multiple sequences are. Consequently, there are many practical applications, i.e., visualization and automated writing reports from multiple time series based on their relations which are inherently encoded. However, the task of extracting such important relations among them is non-trivial.

The Automatic Bayesian Covariance Discovery (ABCD) focuses on regression problems using Gaussian process (GP) models [Duvenaud et al., 2013; Ghahramani, 2015; Hwang et al., 2016; Kim and Teh, 2018; Lloyd et al., 2014; Malkomes et al., 2016]. In previous work, selecting GP kernels is done manually with expert or domain knowledge. The ABCD follows an automated

procedure based on pre-defined grammar rules resulting an appropriate compositional kernel function to fit data. The framework outputs descriptive reports which are in form of natural language to explain data. The key strength of compositional covariance structures lies in their expressiveness and interpretability. Cognitive studies [Schulz et al., 2016, 2017] show that compositional functions constructed by ABCD are preferred by humans. Employing such properties of compositional structure, this chapter proposes a kernel composition framework which generate interpretable outputs with improved predictive performance for multiple time series.

One of well-established research areas in which GP models involve in learning multiple time series is multi-task GP [Álvarez et al., 2012; Bonilla et al., 2007; Guarnizo et al., 2015; Titsias and Lázaro-Gredilla, 2011; Wilson et al., 2012]. Yet, among these multi-task GP methods, the approach of learning compositional covariance structure is not considered. As an example, the multi-output GP regression network (GPRN) proposed by Wilson et al. [2012] models data with the linear combination of latent GPs where the linear weights are also GPs. Because the network in this model is complex, it is impossible to perform structure search for all GPs in the network. To make a selected covariance structures interpretable for multiple sequences, we use additive structure kernels. Rather being fixed, these additive kernels are searched over a set of kernels. Indian Buffet Process (IBP) [Griffiths and Ghahramani, 2005, 2011] prior is used to model an binary matrix that indicates the membership of additive kernels to time series. Moreover, we combine learning IBP and a search algorithm to have a better exploration over model space.

The model presented in this chapter suggests a new way to understand multiple time series better via analyzing latent features from IBP and interpretable kernel functions. That is, the output of models provides the relation among time series with human-readable descriptions. This can be a potential application helping decision-making processes in many fields including scientific discovery, financial management.

Compared to the previous chapter which introduces a global *shared* kernel for all time series and individual kernels for each time series, the model proposed in this chapter is more general because strong correlation assumption is no longer the prerequisite, the relation among time series is instead learned by IBP. Figure 4.1 is an example of the data that does not meet the assumption in RKL and SRKL.

This chapter offers the following contributions. The chapter introduces the model dubbed Latent Kernel Model (LKM) and characterize its well-definedness along with an approximate

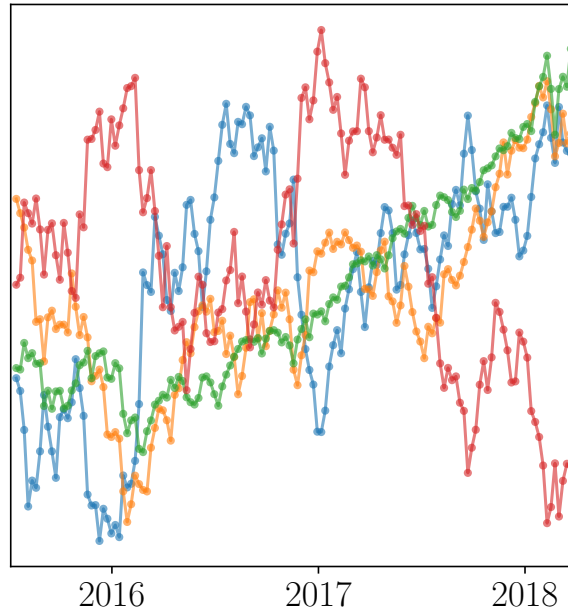


Fig. 4.1 Time series of Gold, Oil, NASDAQ, and USD index

inference algorithm. The chapter proposes a search procedure that extends to multiple time series. Finally, a new type of generated reports for multiple time series is presented.

## 4.2 Latent Kernel Model

This section starts with the Indian Buffet Process (IBP) which is an important building block of the proposed model. Then, the [Latent Kernel Model \(LKM\)](#) is introduced along with theoretical properties and an approximate inference.

### 4.2.1 Indian Buffet Process

The Indian Buffet Process (IBP) is introduced in [[Griffiths and Ghahramani, 2005](#)] and is a probabilistic prior defined over a binary matrix  $\mathbf{Z}$ . Here, the number of rows in  $\mathbf{Z}$  is finite. On the other hand, the number of columns in  $\mathbf{Z}$  can be infinite. The name of this stochastic process is metaphorically inspired by Indian restaurant where (finite) customers enter the restaurant and are served by (infinite) dishes. IBP and Beta-Bernoulli process are closely related [[Thibaux and Jordan, 2007](#)].

$$\theta_k \sim \text{Beta}\left(\frac{\alpha}{K}, 1\right),$$

$$z_{nk} \sim \text{Bernoulli}(\theta_k),$$

IBP is obtained when sending the number of columns,  $K$ , to infinity. In particular  $\lim_{K \rightarrow \infty} p(\mathbf{Z}|\alpha) = 0$ . However,  $[\mathbf{Z}] = \text{lof}(\mathbf{Z})$

$$\lim_{K \rightarrow \infty} p(\mathbf{Z}|\alpha) = \exp\{\alpha H_N\} \frac{\alpha^{K_+}}{\prod_{h>0} K_h!} \prod_{k \leq K_+} \frac{(N - m_k)!(m_k - 1)!}{N!}$$

infinite exchangeable.

An intuitive explanation of IBP is that the matrix represents feature assignments where the element at the  $i$ -th row and the  $j$ -th column expresses the presence or absence of the  $j$ -th feature in the  $i$ -th object.

#### 4.2.2 Model definition

**Notation** Let  $\mathbf{x}_n = (x_{n1}, \dots, x_{nD})^\top$  be the  $n$ -th time series. Here, at the time step  $d$  indexed by  $t_d$ , the value of the  $n$ -th time series is  $x_{nd}$ . The number of time series is  $N$ . The number of data points is  $D$ . Note that  $\mathbf{x}_n, n = 1 \dots N$  be rows constructing matrix  $\mathbf{X}$ . We introduce binary vectors  $\mathbf{z}_n, n = 1 \dots N$  as rows of latent matrix  $\mathbf{Z}$ .

When a set of GP kernels  $\{\mathbf{C}_k\}_{k=1}^K$  is given, the generative procedure modeling time series  $\mathbf{x}_n$  is defined as

$$\begin{aligned} \mathbf{Z} &\sim \text{IBP}(\alpha), \\ \mathbf{f}_n &\sim \mathcal{GP}(\mathbf{0}, \sum_{k=1}^K z_{nk} \mathbf{C}_k), \\ \mathbf{x}_n &\sim \mathcal{N}(\mathbf{f}_n, \sigma_n^2 \mathbf{I}), \end{aligned} \tag{4.2.1}$$

where the IBP concentration parameter is  $\alpha$ . Under this model construction, the observed data  $x_{nd}$  is fitted by a GP latent function variable  $f_n(t_d)$ . Since all  $p(\mathbf{x}_n|\mathbf{z}_n)$  are independent, the  $p(\mathbf{X}|\mathbf{Z})$  is obtained as

$$p(\mathbf{x}_n|\mathbf{z}_n) = |2\pi \mathbf{D}(\mathbf{z}_n)|^{-1/2} \exp\left(-\frac{1}{2} \mathbf{x}_n^\top \mathbf{D}(\mathbf{z}_n)^{-1} \mathbf{x}_n\right), \tag{4.2.2}$$

where  $\mathbf{D}(\mathbf{z}_n) = \sum_{k=1}^K z_{nk} \mathbf{C}_k + \sigma_n^2 \mathbf{I}$ . The entry at  $(n, d)$  of  $N \times K$  matrix  $\mathbf{Z}$  which is denoted as  $z_{nk} \in \{0, 1\}$  indicates the membership whether the  $n$ -th time series has kernel  $\mathbf{C}_k$  in its compositional kernel. By definition of IBP,  $\mathbf{Z}$  can have infinitely many columns as  $K \rightarrow \infty$ . This model can be viewed as a generative process that creates the stochastic kernel  $\mathbf{D}(\mathbf{z}_n)$ . In this chapter, IBP matrix  $\mathbf{Z}$  is learned by an approximate Bayesian inference to reason about kernel structures and relations between time series.

### 4.2.3 Properties

**Well-definedness of LKM** The number of kernels theoretically can go to infinity because of the choice of IBP prior over the matrix  $\mathbf{Z}$  in which the number of columns can be infinitely many. This leads to an important verification of whether  $p(\mathbf{X}|\mathbf{Z})$  results in a well-defined probability distribution for the case that the number of kernels approaches infinity. Griffiths and Ghahramani [2011] presents an detailed analysis for linear factor model (LFM) where feature matrix can be marginalized in  $p(\mathbf{X}|\mathbf{Z})$ . However,  $p(\mathbf{X}|\mathbf{Z})$  in LKM still relies on kernels in its representation. Therefore, it is necessary to justify the well-definedness in LKM.

**Proposition 1.** *With  $\mathbf{Z} \sim \text{IBP}(\alpha)$ , the likelihood of LKM as*

$$p(\mathbf{X}|\mathbf{Z}) = \prod_{n=1}^N p(\mathbf{x}_n|\mathbf{z}_n),$$

with  $p(\mathbf{x}_n|\mathbf{z}_n)$  in Equation 4.2.2 is well-defined.

*Proof.* The main proving technique is to use left-order-from (*lof*) on  $\mathbf{Z}$  which is done by reordering the columns in  $\mathbf{Z}$  by the binary number computed from a column [Griffiths and Ghahramani, 2011]. Because of the commutative properties among  $\mathbf{C}_k$ , the order of  $\mathbf{C}_k$  is changed according to the order of *lof* performing on  $\mathbf{Z}$ . This does not affect  $p(\mathbf{X}|\mathbf{Z})$ .

Specifically, when applying *lof* on  $\mathbf{Z}$ , we obtain  $[\mathbf{Z}^+\mathbf{Z}^0]$  where  $K^+$  nonzero columns is gathered on the left as  $\mathbf{Z}^+$  and the remaining  $\mathbf{Z}_0$  is just  $K^0$  zero columns. Now  $\mathbf{D}(\mathbf{z}_n)$  is represented as  $\mathbf{D}(\mathbf{z}_n) = \sum_{k=1}^{K^+} z_{nk}^+ \mathbf{C}_k + \sigma_n^2 \mathbf{I}$  which only depends on  $\mathbf{Z}_+$ . As  $K \rightarrow \infty$ , IBP maintains the finite number of nonzero columns  $K^+$ . Therefore,  $\mathbf{D}(\mathbf{z}_n)$  is computed as the sum of a finite number of covariances kernels  $\mathbf{C}_k$ . This means that  $p(\mathbf{x}_n|\mathbf{z}_n)$  has a well-defined. This means that  $p(\mathbf{X}|\mathbf{Z})$  is well-defined.  $\square$

IBP prior can be thought of a regularizer preventing the explosion from adding infinite many kernels. Consequently, the resulting GP models remain simple as the IBP matrix is sparse.

**Comparisons with existing models** Consider feature sharing models including [Guarnizo et al., 2015; Titsias and Lázaro-Gredilla, 2011; Wilson et al., 2012] as

$$\mathbf{x}_n = \sum_{k=1}^K w_k \mathbf{f}_k + \epsilon_n,$$

where  $\epsilon_n, n = 1 \dots N$  are Gaussian noises,  $\mathbf{f}_k, k = 1 \dots K$  are shared features. Here, one can understand that  $\mathbf{f}_k$  is a GP realization sampled from  $\mathbf{C}_k$ . The linear weights,  $w_k$ , can have



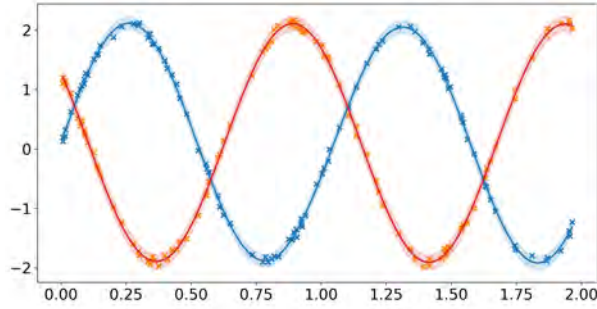


Fig. 4.2 A toy example. Learning two lines using LKM. This data set consists of two samples generated from the same GP prior with a periodic kernel.

different ways to model, for example, spike and slab prior [Titsias and Lázaro-Gredilla, 2011] or GP latents [Wilson et al., 2012].

The LKM is more general and expressive than the feature sharing approaches. Consider the decomposition of posterior distribution in an additive Gaussian process presents as  $\mathbf{f} = \mathbf{f}_1 + \mathbf{f}_2$ , where  $\mathbf{f}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_1)$  and  $\mathbf{f}_2 \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_2)$ . The conditional distribution of  $\mathbf{f}_1$  given the sum  $\mathbf{f}$  is

$$\mathbf{f}_1|\mathbf{f} \sim \mathcal{N}(\mathbf{K}_1^\top(\mathbf{K}_1 + \mathbf{K}_2)^{-1}\mathbf{f}, \mathbf{K}_1 - \mathbf{K}_1^\top(\mathbf{K}_1 + \mathbf{K}_2)^{-1}\mathbf{K}_1).$$

Projecting to the multiple time series setting, the decomposition for the same GP prior can be different for each time series. That is, given a covariance index  $k$  with fixed covariance  $\mathbf{C}_k$ , the posterior  $\mathbf{f}_k|\mathbf{x}_n$  differs as  $\mathbf{x}_n$  changes. Figure 4.2 illustrating on a toy data set can verify this observation. This simple experiment considers generating two sampled sequences from a single periodic GP. LKM is trained on this data with two different periodic kernels  $\mathbf{C}_1$  and  $\mathbf{C}_2$ , resulting  $\mathbf{Z} = [0, 1; 0, 1]$ . This means that LKM can recognize these two samples from one GP.

We also emphasize that the Bayesian approach that is considered in our kernel construction, can be viewed as a stochastic kernel generative process [Jang et al., 2017b].

Compared to the model proposed in the previous chapter, Figure 4.3 shows the difference in term of graphical model. The approach in the previous chapter heavily depends on a global shared kernel for all time series and allows distinctive kernel  $\mathbf{C}_n$  in each time series. Because the spectral mixture kernel [Wilson and Adams, 2013] is used for  $\mathbf{C}_n$  in SRKL, the interpretability is restricted only for the global kernel.

#### 4.2.4 Inference algorithm

**Variational inference** Due to the intractability to obtain the posterior, variational inference [Wainwright and Jordan, 2008] is used where the true posterior  $p(\mathbf{Z}|\mathbf{X})$  is approximated

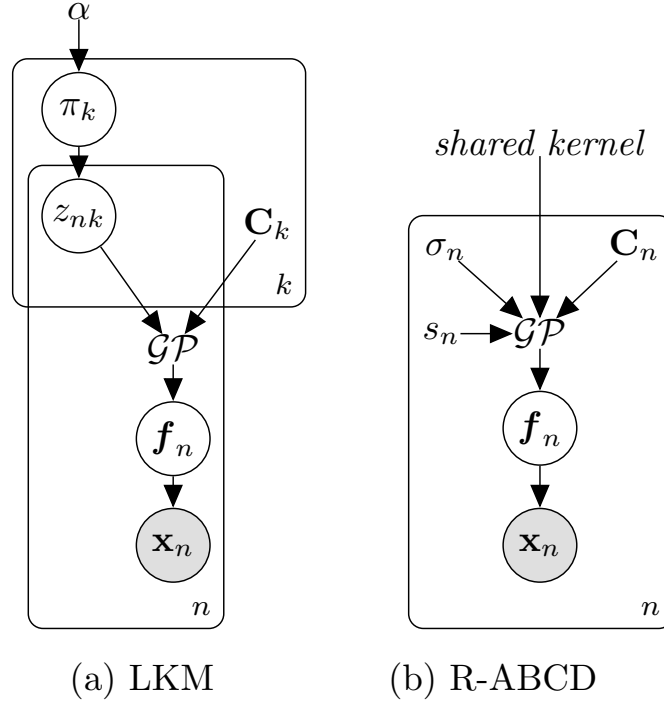


Fig. 4.3 Comparison between the graphical model of (a) LKM and (b) relation model in the previous chapter.

by  $q(\mathbf{Z})$ . That is, we minimize the KL divergence between  $p$  and  $q$  by maximizing the model evidence lower bound (ELBO),

$$\begin{aligned} \log p(\mathbf{X}) &\geq \mathbb{E}[\log p(\mathbf{X}, \mathbf{Z})] + H[q] \\ &= \mathbb{E}[\log p(\mathbf{Z})] + \mathbb{E}[\log p(\mathbf{X}|\mathbf{Z})] + H[q] \triangleq \mathcal{L}. \end{aligned}$$

where  $H[q]$  denotes the entropy of  $q(\mathbf{Z})$  and  $\mathbb{E}$  is the expectation over the approximate posterior distribution  $q(\mathbf{Z})$ . Here, we use the model definition as  $p(\mathbf{X}, \mathbf{Z}) = p(\mathbf{X}|\mathbf{Z})p(\mathbf{Z})$ . The variational distribution  $q(\mathbf{Z})$  is in the mean-field family and factorized into  $q(z_{nk}) = \text{Bernoulli}(z_{nk}; \nu_{nk})$ .

We adopt [Doshi et al. \[2009\]](#) to estimate  $\mathbb{E}[\log p(\mathbf{Z})]$  with two approaches including finite cases and infinite cases. In the finite variational approach, sampling  $\mathbf{Z}$  is done by

$$\begin{aligned} \pi_k &\sim \text{Beta}(\alpha/K, 1), \\ z_{nk} &\sim \text{Bernoulli}(\pi_k). \end{aligned}$$

The variational inference considers an auxiliary variable  $\boldsymbol{\pi}$  of which the approximate distribution is defined as  $\prod_k q(\pi_k)$ . Each  $q(\pi_k)$  is distributed from a Beta distribution,  $\text{Beta}(\tau_{k_1}, \tau_{k_2})$ . Since  $\mathbf{X}$  and  $\boldsymbol{\pi}$  are conditionally independent given  $\mathbf{Z}$ ,  $\mathbb{E}[\log p(\mathbf{X}|\mathbf{Z})]$  does not contain  $\boldsymbol{\pi}$ . Therefore,

we can obtain  $\mathbb{E}[\log p(\mathbf{Z})]$  as

$$\begin{aligned} \mathbb{E}[\log p(\mathbf{Z})] &= \sum_{k=1}^K \left[ \log \frac{\alpha}{K} + \left( \frac{\alpha}{K} - 1 \right) (\psi(\tau_{k_1}) - \psi(\tau_{k_1} + \tau_{k_2})) \right] \\ &\quad + \sum_{k=1}^K \sum_{n=1}^N [\nu_{nk} \psi(\tau_{k_1}) + (1 - \nu_{nk}) \psi(\tau_{k_2}) - \psi(\tau_{k_1} + \tau_{k_2})], \end{aligned}$$

where  $\psi(\cdot)$  is the digamma function.

While in the finite variational approach, stick breaking construction [Teh et al., 2007] is used to sample  $\mathbf{Z}$  as

$$\begin{aligned} v_k &\sim \text{Beta}(\alpha, 1), \\ \pi_k &= \prod_{i=0}^k v_i, \\ z_{nk} &\sim \text{Bernoulli}(\pi_k), \end{aligned}$$

with  $k = 1 \dots \infty$ . Similarly, the variational distribution  $q(\mathbf{v})$  is proposed to approximate  $p(\mathbf{v})$  by independent  $\text{Beta}(\tau_{k_1}, \tau_{k_2})$

$$\begin{aligned} \mathbb{E}[\log p(\mathbf{Z})] &= \sum_{k=1}^K [\log \alpha + (\alpha - 1) (\psi(\tau_{k_1}) - \psi(\tau_{k_1} + \tau_{k_2}))] \\ &\quad + \sum_{k=1}^K \sum_{n=1}^N \left[ \nu_{nk} \left( \sum_{m=1}^k \psi(\tau_{m_1}) - \psi(\tau_{m_1} + \tau_{m_2}) \right) + (1 - \nu_{nk}) \mathbb{E}_{\mathbf{v}} \left[ \log \left( 1 - \prod_{m=1}^k v_m \right) \right] \right], \end{aligned}$$

with  $\mathbb{E}_{\mathbf{v}} \left[ \log \left( 1 - \prod_{m=1}^k v_m \right) \right]$  is further approximated by Taylor expansion.

Now we turn our focus on  $\mathbb{E}[\log p(\mathbf{X}|\mathbf{Z})]$ . Note that the likelihood is defined as

$$p(\mathbf{X}|\mathbf{Z}) = \prod_{n=1}^N p(\mathbf{x}_n|\mathbf{z}_n).$$

We can split  $\mathbb{E}[\log p(\mathbf{X}|\mathbf{Z})]$  into the sum of individual components,  $\mathbb{E}[\log p(\mathbf{x}_n|\mathbf{z}_n)]$ . Observe that  $\mathbb{E}[\log p(\mathbf{x}_n|\mathbf{z}_n)]$  is expensive to compute because we have to compute the expectation in which discrete random variables  $\mathbf{Z}$  are presented. In fact,  $\mathbb{E}[\log p(\mathbf{x}_n|\mathbf{z}_n)]$  is constructed from  $-\frac{1}{2} \mathbf{x}_n^\top \mathbb{E}[\mathbf{D}(\mathbf{z}_n)^{-1}] \mathbf{x}_n$ . This expression is known as the expectation of data-fit term in learning GP. The GP model complexity term,  $\frac{1}{2} \mathbb{E}[\log |2\pi \mathbf{D}(\mathbf{z}_n)|]$ , is the remaining quantity in the likelihood estimation. The main problem involving estimating such a likelihood is the computational

complexity. Specifically, there are  $2^K$  combinatoric configurations: (1)  $p(\mathbf{z}_n = \mathbf{t})\mathbf{D}(\mathbf{t})^{-1}$  for all  $\mathbf{t} \in \{0, 1\}^K$  to estimate the expectation of inverse matrix,  $\mathbb{E}[\mathbf{D}(\mathbf{z}_n)^{-1}]$ ; (2)  $p(\mathbf{z}_n = \mathbf{t}) \log |2\pi\mathbf{D}(\mathbf{t})|$  for all  $\mathbf{t} \in \{0, 1\}^K$  to estimate the expectation of log-determinant,  $\mathbb{E}[\log |2\pi\mathbf{D}(\mathbf{z}_n)|]$ . Therefore, this becomes infeasible to estimate the likelihood due to exponential increase in computation.

**Relaxation** To handle the aforementioned difficulty, we resort to relax the discrete random variables  $z_{nk}$  to a continuous ones. The expectation is further estimated by Monte Carlo integral. Adopting the method in [Maddison et al., 2017], we convert the Bernoulli random variables  $z_{nk} \sim \text{Bernoulli}(\nu_{nk})$  into 2-dimensional continuous random variable  $[\tilde{z}_{nk}, \tilde{z}_{nk}] \sim \text{Concrete}(\nu_{nk}, \lambda)$ , where  $\lambda$  is the temperature parameter. Here, the categorical random variable  $[z_{nk}, 1 - z_{nk}]$  corresponds to the relaxed one  $[\tilde{z}_{nk}, \tilde{z}_{nk}]$ . We only need  $\tilde{z}_{nk}$  which is the relaxed version of  $z_{nk}$ . A sample of  $\tilde{z}_{nk}$  is drawn by first sampling  $g_1$  and  $g_2$  from  $\text{Gumbel}(0, 1)$  and then computing as

$$\tilde{z}_{nk} = \frac{\exp(\frac{\log(\nu_{nk}) + g_1}{\lambda})}{\exp(\frac{\log(\nu_{nk}) + g_1}{\lambda}) + \exp(\frac{\log(1 - \nu_{nk}) + g_2}{\lambda})}.$$

This can be thought of as the Gumbel-Softmax trick [Jang et al., 2017a; Maddison et al., 2017]. After the relaxation, the Monte Carlo estimation of  $\mathbb{E}[\log p(\mathbf{x}_n | \mathbf{z}_n)]$  is obtained by

$$\mathbb{E}[\log p(\mathbf{x}_n | \mathbf{z}_n)] \approx \frac{1}{m} \sum_{i=1}^m \log p(\mathbf{x}_n | \tilde{\mathbf{z}}_n^{(i)}),$$

where  $m$  is the number of Monte Carlo samples,  $\{\tilde{\mathbf{z}}_n^{(i)}\}_{i=1}^m$  are the samples drawn from the Concrete distribution. To this end, the expectation,  $\mathbb{E}[\log p(\mathbf{x}_n | \mathbf{z}_n)]$  no longer depends on the exponential number in terms of  $K$  random variables  $\mathbf{z}_n$  but the number of sample  $M$ . Moreover, the stochastic gradient estimation can be done via the reparameterization trick [Kingma and Welling, 2014; Mohamed et al., 2020; Schulman et al., 2015].

### 4.3 Model discovery in multiple time series

To have a better model exploration, this section presents an approach to search over the space of LKM.

**Search procedure** Due to the fact that LKM model space is huge, the search procedure in this section is done in a greedy manner similar to [Duvenaud et al., 2013; Grosse et al., 2012; Lloyd et al., 2014]. At a certain depth  $d$ , we maintain a collection of additive kernel  $\{\mathcal{S}_d^{(k)} | \mathcal{S}_d^{(k)} = \prod_l \mathcal{B}_d^{(k_l)} \text{ with } \mathcal{B}_d^{(k_l)} \text{ s are base kernels, } k = 1 \dots K\}$ . The required kernel structure

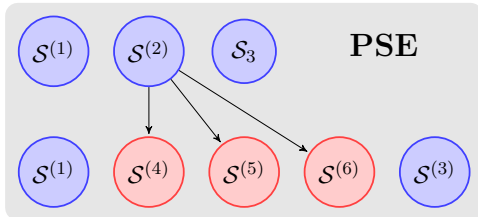


Fig. 4.4 An example of expansion in PSE.

for  $\mathbf{C}_k$  in LKM is assigned correspondingly to  $\mathcal{S}_d^{(k)}$ . At the depth  $(d + 1)$ , the collection will include new additive kernels which are generated from the elements of the collection at the depth  $d$ . The expansion adopts the context-free grammar rules like in Compositional Kernel Learning (CKL) [Duvenaud et al., 2013]. However, suppose that we expand  $\mathcal{S}_d^{(k)}$  into a new kernel which is in an additive form,  $\sum_{m=1}^M \mathcal{S}_{d+1}^{(k_m)}$ , we replace this expansion with  $M$  separated expansions  $\mathcal{S}_d^{(k)} \rightarrow \mathcal{S}_{d+1}^{(k_m)}$ . Instead the sum  $\sum_{m=1}^M \mathcal{S}_{d+1}^{(k_m)}$  is added, we include  $\mathcal{S}_{d+1}^{(k_m)}$  to the collection. This procedure guarantee these new structures satisfy the canonical definition of  $\{\mathcal{S}_d^{(k)}\}$ .

**Partial set expansion (PSE)** Partial set expansion expands the collection  $\mathcal{S}_d^{(k)}$  iteratively and obtain a set of candidates  $\{\mathcal{S}_d^{(k_1)}, \dots, \mathcal{S}_d^{(k_m)}\}$ . A new collection is created from the union of the previous one excluded the selected structure  $\{\mathcal{S}_d^{(k)}\}_{k=1}^K \setminus \{\mathcal{S}_d^{(i)}\}$  and the new candidate structures  $\{\mathcal{S}_d^{(i_1)}, \dots, \mathcal{S}_d^{(i_m)}\}$  (Figure 4.4). The variational inference algorithm (described in Section 4.2.4) will learn the indicator matrix  $\mathbf{Z}$  and GP hyperparameters. The new kernel collection is kept if the learned model has a better BIC score [Schwarz, 1978.]. Otherwise, we rolls back to the previous one. We continue with further expansion described in Algorithm 2.

The main advantages of PSE algorithm are (1) it avoids drastic increases in structure space when performing expansion, (2) it takes consideration into a selection criterion (BIC) and rolls back to the previous model if necessary, (3) the hyperparameter initialization is easier with the fewer number of kernels in PSE.

Our kernel search procedure is a meta search algorithm inspired from oracle machines in computational theory [Papadimitriou, 1994]. The LKM plays a role as an *oracle*. Given a set of kernel structures, one tries to ask the oracle to decide the appropriate structures. The oracle will response an answer as  $\mathbf{Z}$  in our case. Exploiting the returned  $\mathbf{Z}$ , the kernel structures will be elaborated more by performing PSE. The procedure is repeated by making new inquiry based on the expanded structures.

We emphasize that PSE with LKM considers a larger number of kernel structures than those in CKL. Suppose that CKL and our search algorithm have the same found structure at a depth  $d$ . Whereas the CKL’s structure is  $\mathcal{S}_d = \mathcal{S}_d^{(1)} + \dots + \mathcal{S}_d^{(K)}$ , PSE represents it as

---

**Algorithm 2** Search procedure follows partial set expansion with LKM learning
 

---

**Require:** A multiple data set and maximum search depth  $D$ , an initial kernel collection  $\{\mathcal{S}_d^{(k)}\}$

- 1: **for**  $d = 1 \dots D$  **do**
- 2:   **for**  $\mathcal{S}$  in  $\{\mathcal{S}_d^{(k)}\}$  **do**
- 3:     Update  $\{\mathcal{S}_d^{(k)}\} \leftarrow \{\mathcal{S}_d^{(k)}\} \setminus \mathcal{S} \cup \text{expand}(\mathcal{S})$
- 4:     Learn  $\mathbf{Z}$  and GP hyperparameters with LKM
- 5:     **if** there is an improvement in BIC **then**
- 6:       Keep the updated collection  $\{\mathcal{S}_d^{(k)}\}$
- 7:     **else**
- 8:       Rollback to previous collection  $\{\mathcal{S}_d^{(k)}\}$
- 9:     **end if**
- 10:   **end for**
- 11: **end for**

---

a set  $\{\mathcal{S}_d^{(1)}, \dots, \mathcal{S}_d^{(K)}\}$ . Let  $L$  be the largest number of base kernels in  $\mathcal{S}_d^{(k)}$ , and  $R$  be the maximum number of grammar rules per substructure. All possible search candidates in CKL is  $\mathcal{O}(RK2^L + R2^K)$  kernels, while PSE incorporating with LKM considers  $\mathcal{O}(K2^{R2^L+K})$  number of kernels.

We emphasize that PSE with LKM considers a larger number of kernel structures than those in CKL. Suppose that CKL and our search algorithm have the same found structure at a depth  $d$ . While the CKL's structure is  $\mathcal{S}_d = \mathcal{S}_d^{(1)} + \dots + \mathcal{S}_d^{(K)}$ , PSE represents as a set  $\{\mathcal{S}_d^{(1)}, \dots, \mathcal{S}_d^{(K)}\}$ . Let us examine the cardinality of kernel spaces after performing an expansion to the next depth. The procedure is to extract substructures from the current structure, then apply grammar rules on the structure. In CKL, substructures consist of all structures generated from the combinations of  $\mathcal{B}_d^{(k_i)}$  in each individual  $\mathcal{S}_d^{(k)}$  and ones generated by the combination of all  $\mathcal{S}_d^{(k)}$ . The former has  $\mathcal{O}(K \sum_l \binom{L}{l}) = \mathcal{O}(K2^L)$  substructures where  $L$  is the largest number of base kernels in  $\mathcal{S}_d^{(k)}$ . The latter creates  $\mathcal{O}(\sum_k \binom{K}{k}) = \mathcal{O}(2^K)$  combinations. When the maximum number of grammar rules per substructure is  $R$ , the total number of candidates at the depth  $d + 1$  is  $\mathcal{O}(RK2^L + R2^K)$ .

Our approach only applies expansion on individual structure  $\mathcal{S}_d^{(k)}$  via the combinations of  $\mathcal{B}_d^{(k_i)}$ . However, the search space still includes all the cases when substructures are extracted from a combination of  $\mathcal{S}_d^{(k)}$ . For instance, the generation from LIN+PER+SE to (LIN+PER) $\times$ SE+SE in CKL is equivalent to the generation from {LIN, PER, SE} to {LIN $\times$ SE, PER $\times$ SE, SE} in our approach. For the case of PE, the additive kernel set will be expanded into a new one having the number of elements  $R2^L + K$ . With the flexible binary indications (on/off) of  $\mathbf{Z}$ , the number of all possible kernels is  $\mathcal{O}(K2^{R2^L+K})$  when all structures are visited to be expanded.

Although our search algorithm explores a much larger search space than CKL in theory, the prior over  $\mathbf{Z}$  is the bottleneck of our model. In fact, learning  $\mathbf{Z}$  is optimized based on a gradient-based method where the global optimal is not guaranteed.

## 4.4 Experimental evaluations

This section provides the details of data sets and gives qualitative and quantitative results of LKM.

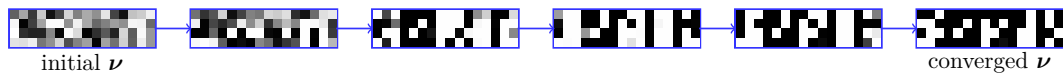


Fig. 4.5 An illustration of the convergence of  $\nu$ .

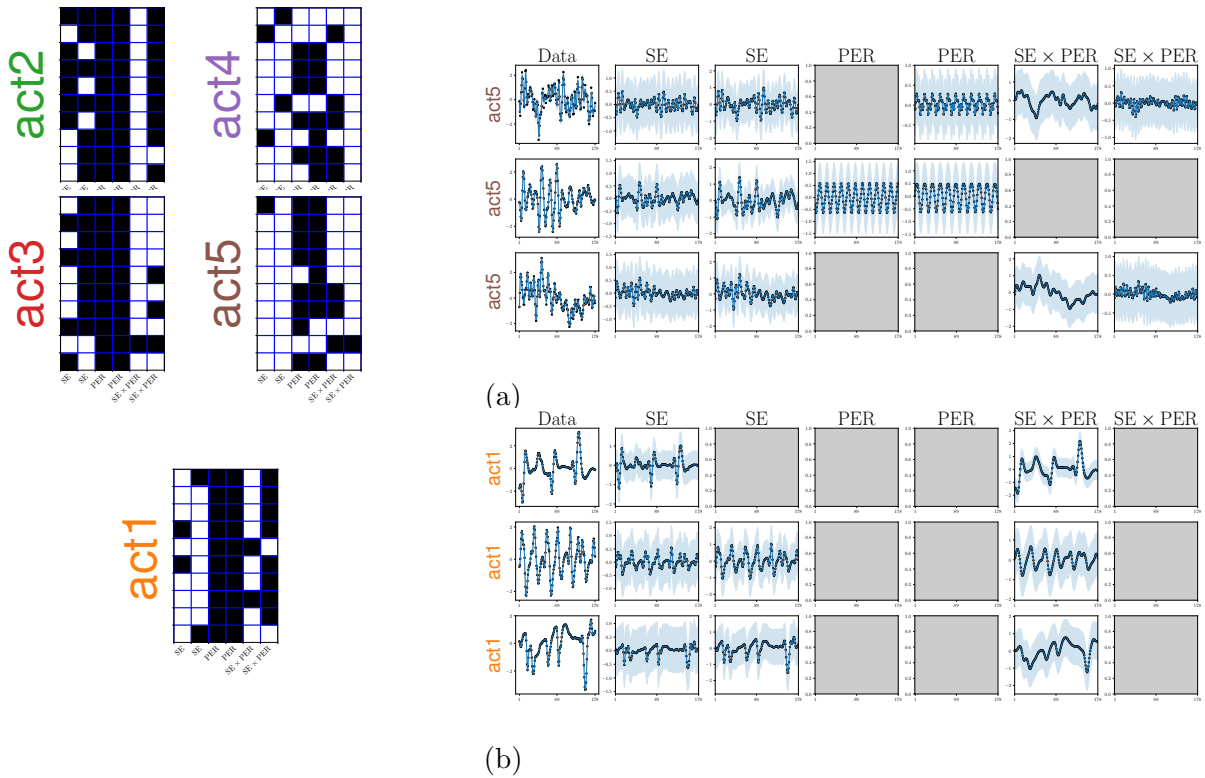


Fig. 4.6 IBP matrix  $\mathbf{Z}$  in epileptic seizure data. The EEG recordings of five activities are considered: seizure (**act1**), located tumor (**act2**), identifying tumor (**act3**), eyes closed (**act4**), eyes open (**act5**). (a) Non-seizure. *Left*: learned  $\mathbf{Z}$  for each activity (black:  $z_{nk} = 0$ , white:  $z_{nk} = 1$ ); *Right*: the GP posterior of three time series from **act5** with the corresponding decomposition. (b) Seizure. *Left*: learned  $\mathbf{Z}$  from **act1**; *Right*: posterior plot of first three time series from **act1**. The gray background plots indicate  $z_{nk} = 0$ .

#### 4.4.1 Real-world time series data

**Strongly correlated data sets** Three data sets are considered: US stock prices, US housing markets and currency exchanges. These data sets are described in previous chapter, containing a multiple time series which strongly correlates each other. These data sets are considered as baseline data sets to compare our method to the relational approach in the previous chapter.

**Heterogeneous data set** To emphasize the ability of LKM to handle more general setting, time series are collected from various domains. It is composed of gold price, crude oil price, NASDAQ composite index, and USD index<sup>1</sup> from 2015 July 1<sup>st</sup> to 2018 July 1<sup>st</sup>. We name this data set as GONU (Gold, Oil, NASDAQ, USD index). Each time series contains 157 data points taken from [Quandl, 2018]. The relations between time series are complex. For example, the gold and oil prices may have a negative correlation where one may increase but the other decreases. There are many financial studies focusing on these time series [Filis et al., 2011; Reboredo et al., 2014].

**Epileptic seizure data set** This data set [Andrzejak et al., 2002] is retrieved from UCI repository Dheeru and Karra Taniskidou [2017a]. It contains EEG recordings of brain activities for 23.6s. Each record is one of five activities including eyes open, eyes closed, identifying the tumor, located the tumor and seizure activity. There are 178 data points for each time series.

#### 4.4.2 Qualitative results

To make machine learning models interpretable, thereby help scientific discovery and decision making, the following experiments demonstrate the potential applications the proposed model.

##### Exploiting information from $\mathbf{Z}$

**Learning  $\mathbf{Z}$**  An ablation study to visualize the convergence of variational parameters  $\nu$  can be found in Figure 4.5. The value of  $\nu_{nk}$  indicates the probability of  $z_{nk} = 1$ . The bigger  $\nu_{nk}$  is, the more likely the kernel  $\mathbf{C}_k$  is chosen to model time series  $\mathbf{x}_n$ .

**Interpreting  $\mathbf{Z}$**  50 time series is randomly selected from the epileptic seizure data where each activity has 10 time series. Because finding a covariance kernel decomposition for a large number of time series is time-consuming, and therefore prohibits kernel structure search, we choose a

---

<sup>1</sup>Data is retrieved from Quandl where codes for these data sets respectively are WGC/GOLD\_DAILY\_USD, FRED/DCOILBRENTU, NASDAQOMX/COMP, FRED/DTWEXM



fixed kernel structures  $\{SE_1, SE_2, PER_1, PER_2, SE_3 \times PER_3, SE_4 \times PER_4\}$ . Figure 4.6 depicts an overview of the model outputs.

There are several interesting observations. The block matrices from  $\mathbf{Z}$  of located tumor and identifying tumor have similar sparsity. Also, having fewer active SE kernels shows that the corresponding EEG recordings do not vary much. On the other hand, there are quickly varying signal described by small lengthscales on the activities of closed eyes and opening eyes. The seizure has a similar sparsity compared to those of closed eyes and opening eyes. However, there does not exist low-frequency periodicity.

We can show that the latent matrix  $\mathbf{Z}$  represents certain relations between time series with the additional information from kernel interpretability. Next, we make use the natural-language description of kernels to produce comparison reports.

### Comparison report

**Overview comparison** By taking the advantage of the learned latent matrix  $\mathbf{Z}$  and the descriptive properties of found GP covariance structures, we generate a human-readable report containing the comparison among time series. For example, the generated text can have formats like

$$“[T_1, \dots, T_m] \text{ share } [description]”$$

where the replacement of  $[T_1, \dots, T_m]$  is a set of time series,  $[description]$  is generated by the found GP structure. Below is extracted from GONU data set.

- Gold, Oil, NASDAQ, USD index share the following property:  
This component is periodic with a period of 1.4 years but with varying amplitude. The amplitude of the function increases linearly away from Apr 2017. The shape of this function within each period has a typical lengthscale of 4.9 days.
- Gold, Oil, USD index share the following property:  
This component is a smooth function with a typical lengthscale of 2.7 weeks.
- NASDAQ has the following property:  
This component is a linear function.

**Pairwise comparison** We provide another type of descriptive comparisons. Given a set of  $N$  time series, the output of our model can generate  $\binom{N}{2}$  reports which compare each pair of time series. These reports give us a more detailed insight than the overview comparison. A report consists of shared components and individual ones between time series. Alongside with

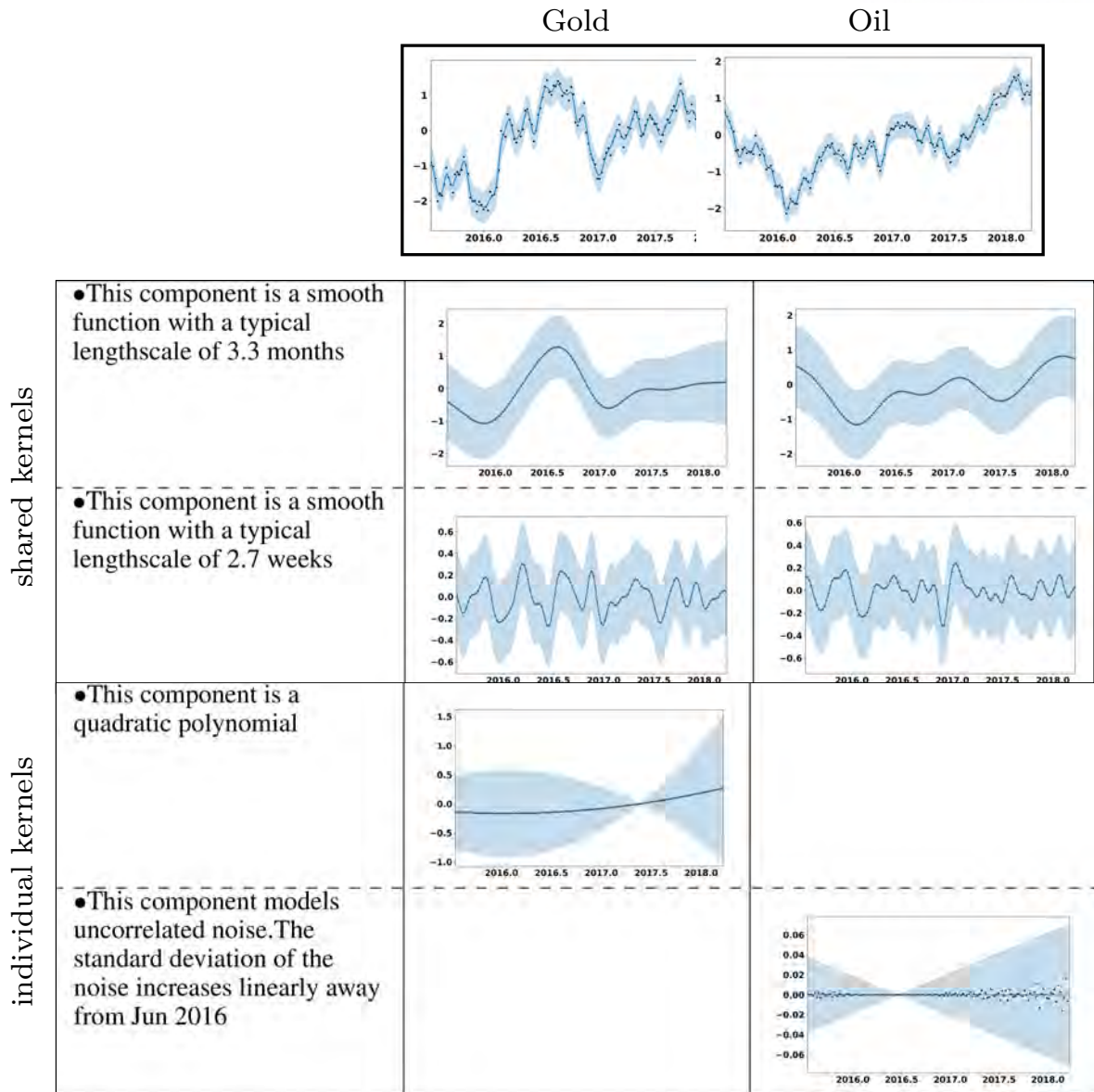


Fig. 4.7 An example of pairwise comparison in GONU data set. The upper plots are the posterior distribution of two time series. The remaining plots contain shared components and individual components with descriptions and posteriors  $\mathbf{f}_k|\mathbf{x}_n$  for each time series. The blank in the individual components means “not available”.

the description of the kernel structure of  $\mathbf{C}_k$ , this type of report presents the corresponding posterior  $\mathbf{f}_k|\mathbf{x}_n$  which will illustrate the variations of GP realizations on different time series (see Figure 4.7).

We bring a brief analysis of GONU data set as an example after taking a quick look over the generated report. For instance, the gold and oil prices share many common characteristics (long and short lengthscale varying), showing a marginally small difference. On the other hand, NASDAQ and USD indices differ each other with many distinctive individual kernels  $\mathbf{C}_k$ s. Interestingly, the negative correlation behavior between the oil and USD indices (i.e. two

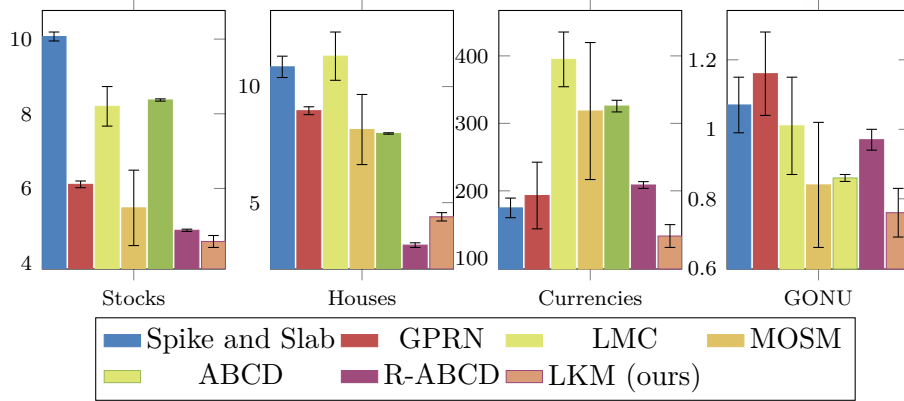


Fig. 4.8 RMSEs for each data set (9 stocks, 6 houses, 4 currencies, GONU) with corresponding methods.

time series often go in opposite directions) can be observed by shared kernels using LKM (see Appendix 4.9). These reports give an easy understanding for ones who do not have knowledge in finance.

### 4.4.3 Quantitative results

**Experiment setup** All experiments are conducted to predict future events (extrapolation) by splitting all data sets and trained with the first 90%, then tested with the remaining 10% as in the standard setting for extrapolation tasks. Root mean square error (RMSE) and Mean Negative Log Likelihood (MNL) [Lázaro-Gredilla et al., 2010] are the main evaluation metrics in all data sets.

**Compare to multi-task GPs** We compare multi-task GP models including ‘Spike and Slab’ model [Titsias and Lázaro-Gredilla, 2011], GP regression network (GPRN) [Nguyen and Bonilla, 2013; Wilson et al., 2012], Linear Model of Coregionalization (LMC) [Álvarez et al., 2012; GPy, since 2012] and Multi-Output Spectral Mixture (MOSM) [Parra and Tobar, 2017]. The result in Table 4.1 and Figure 4.8 indicates that our methods significantly outperform these models. This result could be attributed to that LKM leveraged by PSE selects compositional kernels which are flexible enough to fit complex data.

**Compare to existing kernel composition approaches** We ran ABCD on individual time series then aggregated the results to compare with our models. Our model outperforms ABCD which is known as one of the state-of-the-art GP-based regression methods on univariate

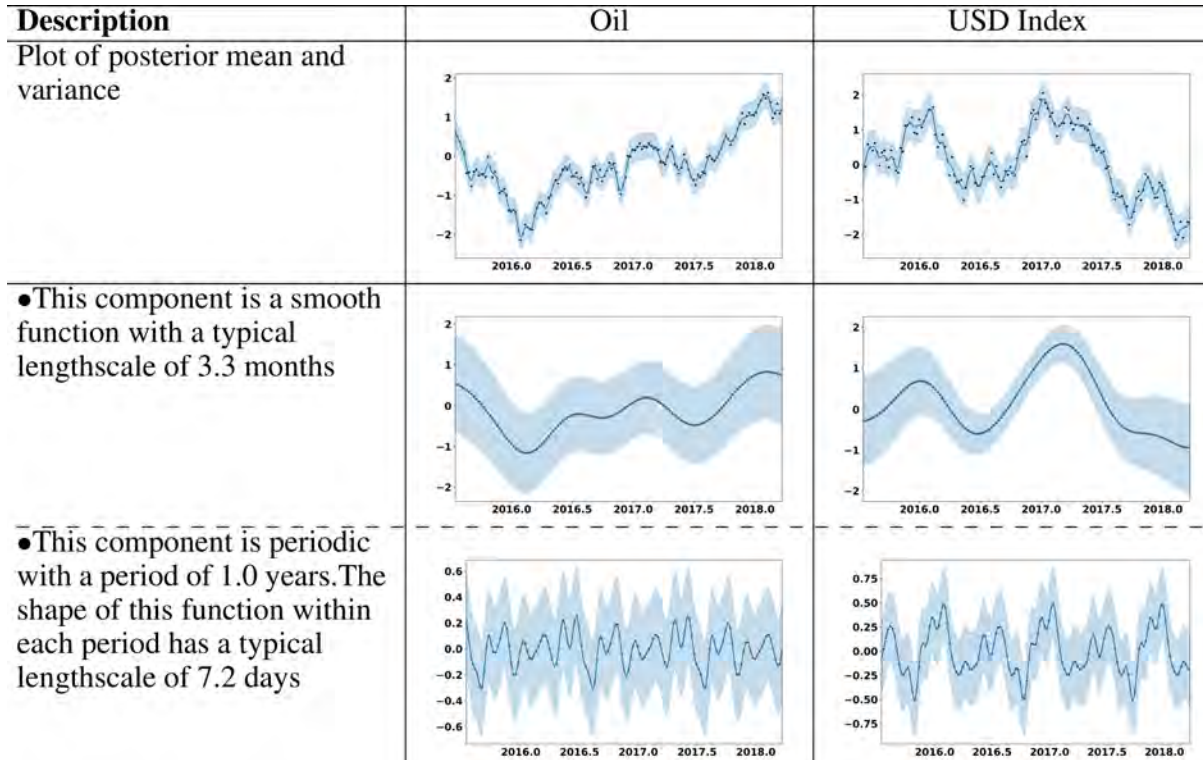


Fig. 4.9 Comparing Oil and USD index. This is extracted from the pairwise comparison of GONU data set.

time series. It proves that our belief about the correlations among multiple time series is plausible.

We then compare with R-ABCD [Hwang et al., 2016]. Rather than making the assumption that all time series share a single global kernel, our model recognizes which structures are shared globally or partially. Quantitatively, LKM shows promising results in prediction tasks. It outruns R-ABCD in most of the data sets (Table 4.1 and Figure 4.8). In a relationally complex data set like GONU, LKM is significantly better while R-ABCD failed as the restriction due to its feature (function) sharing assumption.

Spike and Slab and GPRN models perform better than ABCD and R-ABCD in the currency data set where it contains highly volatile data. Although our model shares some computational procedures with ABCD and R-ABCD, our model is more robust to handle different types of time series data.

## 4.5 Related work and final remark

**Compositional kernel learning** This work is the direct extension of [Hwang et al., 2016] for kernel discovery in multiple time series settings. In contrast, the proposed method, LKM,

does not rely on global shared information between multiple time series. Instead, it can work on a more general setting when multiple time series can be partly shared structures. There are a line of research aiming to improve the kernel learning of [Duvenaud et al., 2013]. The list includes [Han et al., 2019; Kim and Teh, 2018; Lu et al., 2018; Malkomes et al., 2016]. This line of work can be considered as parts of explainable machine learning approaches making data self-interpretable Gunning et al. [2019]. In cognitive studies, [Schulz et al., 2016, 2017] find that the kernels constructed by compositional way are more preferable by human. [Sun et al., 2018] presents a network of compositional kernel function inspired by deep neural networks. However, it does not retain interpretability. In terms of probabilistic construction of kernel functions, [Saad et al., 2019; Schaechtle et al., 2015; Tong and Choi, 2016] provides methods based on universal probabilistic programming languages Carpenter et al. [2017]; Mansinghka et al. [2014]. Work by [Jang et al., 2017b] proposes mixtures of kernels with Lévy prior over spectral densities.

**Multi-task learning** In multiple data settings, there are a number of multi-task Gaussian process learning models [Álvarez et al., 2012; Bonilla et al., 2007; Guarnizo and Álvarez, 2015; Guarnizo et al., 2015; Teh et al., 2005; Titsias and Lázaro-Gredilla, 2011; Wilson et al., 2012]. However, the compositional kernel functions are not investigated in such models. The model interpretability for these models is left unexplored. In terms of learning, LKM is based on Bayesian inference with IBP prior which encourages the sparsity while alternative models are learned by gradient-based approaches.

Convolutional networks [LeCun et al., 1989] and sum-product networks [Poon and Domingos, 2011] are related but distinct models in learning complex function. For example, AND-like and OR-like operation have the intuitively similar mechanisms of multiplication and summation in compositional kernels. Adding to this notion, the model in this chapter focuses on multiple complex functions where sharing kernels can be thought of as AND-like operation among multiple time series.

	9 stocks		6 houses		4 currencies		GONU	
	RMSE	MNLP	RMSE	MNLP	RMSE	MNLP	RMSE	MNLP
Spike and Slab	10.07 $\pm$ 0.12	2.87 $\pm$ 0.05	10.85 $\pm$ 0.46	6.92 $\pm$ 0.09	174.71 $\pm$ 14.52	4.09 $\pm$ 0.10	1.07 $\pm$ 0.08	2.36 $\pm$ 0.11
GPRN	6.11 $\pm$ 0.09	2.78 $\pm$ 0.14	8.96 $\pm$ 0.17	6.64 $\pm$ 0.46	193.13 $\pm$ 49.40	4.24 $\pm$ 0.20	1.16 $\pm$ 0.12	2.46 $\pm$ 0.28
LMC	8.20 $\pm$ 0.53	2.24 $\pm$ 0.23	11.31 $\pm$ 1.04	5.90 $\pm$ 0.46	394.83 $\pm$ 40.54	4.90 $\pm$ 0.15	1.01 $\pm$ 0.14	<b>1.43<math>\pm</math>0.11</b>
MOSM	5.48 $\pm$ 1.01	2.97 $\pm$ 0.01	8.15 $\pm$ 1.51	5.90 $\pm$ 0.20	318.26 $\pm$ 101.52	3.93 $\pm$ 0.15	0.84 $\pm$ 0.18	3.13 $\pm$ 1.06
ABCD	8.37 $\pm$ 0.03	2.58 $\pm$ 0.05	7.98 $\pm$ 0.03	5.61 $\pm$ 0.05	325.58 $\pm$ 8.64	4.47 $\pm$ 0.04	0.86 $\pm$ 0.01	2.21 $\pm$ 0.03
R-ABCD	4.88 $\pm$ 0.03	1.95 $\pm$ 0.05	<b>3.17<math>\pm</math>0.10</b>	6.07 $\pm$ 0.09	208.32 $\pm$ 5.02	<b>3.62<math>\pm</math>0.03</b>	0.97 $\pm$ 0.03	2.01 $\pm$ 0.10
LKM	<b>4.58<math>\pm</math>0.16</b>	<b>1.87<math>\pm</math>0.10</b>	4.37 $\pm$ 0.16	<b>5.54<math>\pm</math>0.40</b>	<b>133.00<math>\pm</math>16.92</b>	<b>3.61<math>\pm</math>0.16</b>	<b>0.76<math>\pm</math>0.07</b>	1.90 $\pm$ 0.25

Table 4.1 RMSEs and MNLPs for each data set with corresponding methods (5 independent runs per method). In most cases, LKM has lower RMSEs and MNLPs compared to those of existing methods.



## Chapter 5

# Kernel selection for Scalable GP

Although the Automatic Statistician framework showcases its strength to fit data well as well as provides attractive model explanations, one of the challenges is that the framework cannot scale with large-size data. To deal with this problem, this chapter presents a novel approach to scale up the learning Gaussian processes when the kernel function is compositional and additive. To further accelerate the kernel search procedure, model selection is done with the combination of a shrinkage approach where Horseshoe prior is used.

This chapter contains the model descriptions and experimental results presented at [Tong et al. \[2021\]](#).

### 5.1 Introduction

Recently, there have been many advances in automating model learning with statistical methods. Still, there are challenges on how to make the automatic procedure efficient and scalable.

The Automatic Statistician framework [[Duvenaud et al., 2013](#); [Ghahramani, 2015](#); [Grosse et al., 2012](#); [Lloyd et al., 2014](#); [Steinruecken et al., 2019](#)] aims to address challenges on automating model discovery. The framework follows search procedures over a model space, listing all compositional Gaussian Process (GP) kernel functions generated from compositional grammar rules and base kernels. The obtained models can therefore create human-readable explanations as dissecting explainable components in the compositional kernel models. However, existing methods have to run a time-consuming task because the search space is huge.

Recently, inspired by deep neural networks, [[Sun et al., 2018](#)] proposes a differential extension of compositional learning. Although this model considers expressive kernel functions for GPs and thereby demonstrates good predictive performances, it is less interpretable compared to existing kernel learning methods, e.g. [[Lloyd et al., 2014](#)].



This chapter presents a new kernel composition learning method which focuses on seeking a sparse composition of kernels with a shrinkage prior, Horseshoe prior, which is proven to be effective in learning sparse signal [Bhadra et al., 2019; Carvalho et al., 2009]. To preserve the interpretability in models, we devise compositional kernels by using additive kernels which can be decipher in natural language.

To make models scalable, this chapter proposes a new approximate posterior for GP, Multi-inducing Sparse Variational GP (MultiSVGP). Previous work on sparse inducing GP [Hensman et al., 2013; Snelson and Ghahramani, 2006] gives an approximation for GP for a kernel function in general settings considering a single set of inducing points. However, this can be further improved for the specific case of additive compositional kernels by considering a group of inducing points and assigning an individual member of inducing points responsible for an additive kernel in our approximating posterior. This idea is justified that the error bound of our approximation with compositional kernels is better than that of the sparse inducing GP. Experiments demonstrate that the proposed model can capture appropriate kernel structures for time series from small-scaled data sets to large-scaled data sets. In general, our model runs faster than existing compositional kernel learning methods [Kim and Teh, 2018; Lloyd et al., 2014] five to twenty times while maintaining similar accuracy. On extrapolation task, our model outperforms alternative models as well as improves additive GPs [Duvenaud et al., 2011] with our kernel selection.

The chapter offers the following contributions: an improved GP approximation method for additive compositional kernels; a probabilistic kernel selection using Horseshoe prior so that the learned models can capture the inductive kernel structure in data and maintain interpretability.

### 5.1.1 Variational Sparse Gaussian process

The history of sparse Gaussian process methods dated back from the work of Snelson and Ghahramani [2006]. It can be considered as a sibling of Nyström approximation [Williams and Seeger, 2001]. The central idea of sparse Gaussian process is to introduce *pseudo inducing points*,  $\mathbf{u}$ , which are distributed jointly with Gaussian process latent variable  $\mathbf{f}$  under a Gaussian distribution. The number of inducing points,  $M$ , is much smaller than the number of data points,  $N$ . The computational complexity of sparse Gaussian process is  $\mathcal{O}(NM^2)$  for each learning iteration. There is a line of research on improving and understanding sparse Gaussian processes [Bauer et al., 2016; Bui et al., 2017; Burt et al., 2019; Lee et al., 2017; Walder et al., 2008].

Given a data set  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ , a sparse variational Gaussian process (SVGP) is defined by

$$\begin{aligned} f(\cdot) &\sim \mathcal{GP}(0, k(\cdot, \cdot)), \\ y_i | f, \mathbf{x}_i &\sim p(y_i | f(\mathbf{x}_i)), \end{aligned}$$

where  $p(y_i | f(\mathbf{x}_i))$  is a likelihood which can be Gaussian in regression tasks and categorical in classification tasks. To approximate the posterior, the variational approach considers  $M$  inducing points  $\mathbf{u} = \{u_i\}_{i=1}^M$  at locations  $\{\mathbf{z}_i\}_{i=1}^M$ , forming the variational distribution as

$$\begin{aligned} \mathbf{u} &\sim \mathcal{N}(\mathbf{m}, \mathbf{S}), \\ f(\cdot) | \mathbf{u} &\sim \mathcal{GP}(\mu(\cdot), \Sigma(\cdot, \cdot)), \end{aligned}$$

where the mean and covariance are obtained as

$$\begin{aligned} \mu(\cdot) &= \mathbf{k}_u(\cdot)^\top \mathbf{K}_{\mathbf{uu}}^{-1} \mathbf{u}, \\ \Sigma(\cdot, \cdot) &= k(\cdot, \cdot) - \mathbf{k}_u(\cdot)^\top \mathbf{K}_{\mathbf{uu}}^{-1} \mathbf{k}_u(\cdot), \end{aligned} \tag{5.1.1}$$

with  $\mathbf{k}_u(\cdot) = [k(\mathbf{z}_i, \cdot)]_{i=1}^M$  and  $\mathbf{K}_{\mathbf{uu}}$  is the covariance of  $\mathbf{u}$ .

The variational inference maximizes the evidence lower bound (ELBO) given as following [Hensman et al., 2013]

$$\mathcal{L} = \sum_i \mathbb{E}_{q(f(\cdot))} [\log p(y_i | f(\mathbf{x}_i))] - \text{KL}[q(\mathbf{u}) || p(\mathbf{u})].$$

Here  $q(f(\cdot))$  is a Gaussian, having mean as  $\mathbf{k}_u(\cdot)^\top \mathbf{K}_{\mathbf{uu}}^{-1} \mathbf{m}$  and covariance as  $k(\cdot, \cdot) - \mathbf{k}_u(\cdot)^\top \mathbf{K}_{\mathbf{uu}}^{-1} (\mathbf{S} - \mathbf{K}_{\mathbf{uu}}) \mathbf{K}_{\mathbf{uu}}^{-1} \mathbf{k}_u(\cdot)$ . The objective  $\mathcal{L}$  can be optimized using stochastic gradient descent.

### 5.1.2 Shrinkage prior

In many statistical model learning, we often encounter the problem of sparse variable selection. Some well-known methods are proposed to tackle the problem, including Lasso regularization [Tibshirani, 1996], spike and slab prior [Mitchell and Beauchamp, 1988] and Horseshoe prior [Carvalho et al., 2009].

**Spike and slab prior** The spike and slab prior over  $\mathbf{w}$  is defined as

$$\begin{aligned} v_i &\sim \mathcal{N}(0, \sigma_w^2), \\ s_i &\sim \text{Bernoulli}(\pi_s), \\ w_i &= v_i s_i. \end{aligned}$$

This belongs to the two-group models. That is, the event  $\{w_i = 0\}$  happens with probability  $(1 - \pi_s)$ ; and nonzero  $w_i$  distributed according to a Gaussian prior with probability  $\pi_s$ . There is existing work using the prior for kernel learning [Titsias and Lázaro-Gredilla, 2011].

**Horseshoe prior** The horseshoe prior [Carvalho et al., 2009] introduces a way to sample a sparse vector  $\beta$  as

$$\begin{aligned} \beta_i &\sim \mathcal{N}(0, \tau^2 \lambda_i^2), \quad i = 1 \dots m \\ \lambda_i &\sim \mathcal{C}^+(B), \quad i = 1 \dots m \\ \tau &\sim \mathcal{C}^+(A), \end{aligned}$$

where  $\mathcal{C}^+(\cdot)$  is the half-Cauchy distribution,  $A$  and  $B$  are the scale parameters. Here,  $\tau$  is the global shrinkage parameter,  $\lambda_i$  is the local shrinkage parameter. In contrast to the spike and slab prior, horseshoe prior is a continuous shrinkage one. It has Cauchy-like tails which allow signals to at large values. On the other hand, the infinite spike near zero keeps  $w_i$  around the origin. Ghosh et al. [2019] uses Horseshoe prior for the weight selection in Bayesian deep neural networks.

Compared to Horseshoe prior, the spike and slab prior exhibits a substantial computational burden as the dimension of sparse vectors increases.

## 5.2 Kernel selection with shrinkage prior

This section presents our main contributions: (1) kernel selection with Horseshoe prior and (2) our approximate GP for compositional kernels.

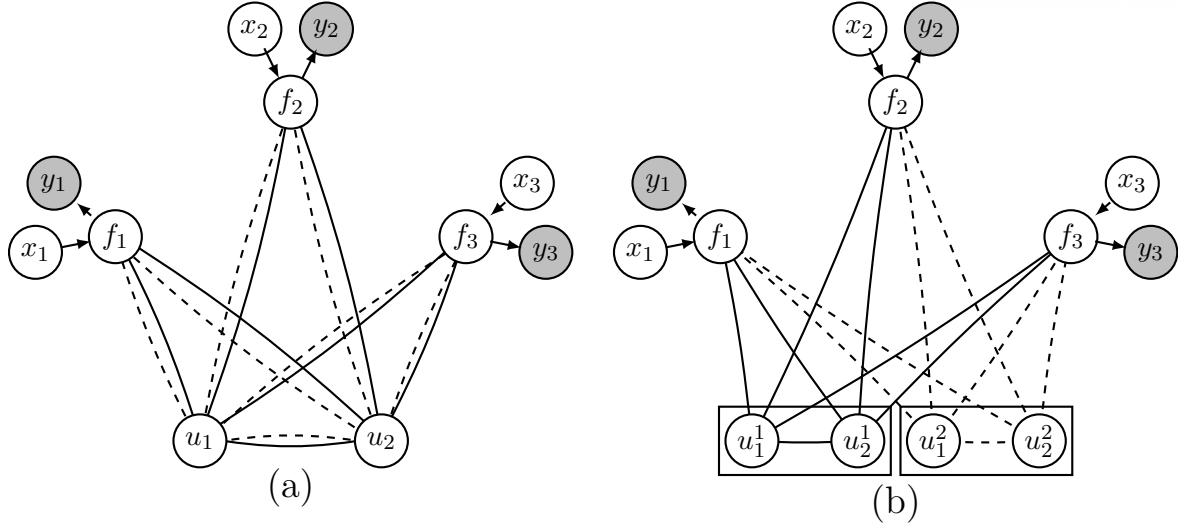


Fig. 5.1 The graphical model of two models. Solid and dashed lines indicate the connections modeled by two different kernel function  $k_1$  and  $k_2$ . (a): Sparse inducing GP. The inducing points  $u_i$  is introduced as a proxy for the connections between  $f_i$ . (b): Our approach. Inducing points are grouped. Each group represents an individual kernel  $k_1$  or  $k_2$ .

### 5.2.1 Kernel selection with Horseshoe prior

Consider the full GP model with kernel construction based on the following generative procedure:

$$f(\cdot)|\mathbf{w} \sim \mathcal{GP}(0, \tilde{k}(\cdot, \cdot)), \quad (5.2.1)$$

where  $\tilde{k}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^m w_i^2 k_i(\mathbf{x}, \mathbf{x}')$  is constructed from  $m$  kernel functions  $k_i(\mathbf{x}, \mathbf{x}')$ . We introduce a probabilistic prior over the weights  $\mathbf{w} = [w_{1:m}]$ ,  $p(\mathbf{w})$  motivated from the Horseshoe prior. That is, we add the covariance term  $k_i(\mathbf{x}, \mathbf{x}')$  to the step sampling  $\beta_i$  in Horseshoe generative procedure. This makes  $\beta_i$  equivalent to  $f_i(\mathbf{x})$  in GP, i.e.,

$$\beta_i \sim \mathcal{N}(0, \tau^2 \lambda_i^2) \Rightarrow f_i(\mathbf{x}) \sim \mathcal{GP}(0, \tau^2 \lambda_i^2 k_i(\mathbf{x}, \mathbf{x}')).$$

When considering the multivariate normal distribution  $\mathbf{f}_i \sim \mathcal{N}(\mathbf{0}, \tau^2 \lambda_i^2 \mathbf{K}_i)$  with kernel matrix  $\mathbf{K}_i$  computed from  $k_i(\mathbf{x}, \mathbf{x}')$ , the *multivariate* version of Horseshoe variable,  $\beta_i \sim \mathcal{N}(\mathbf{0}, \tau^2 \lambda_i^2 \mathbf{I})$ , is a special case of  $\mathbf{f}_i$  when  $\mathbf{K}_i$  is the identity matrix. This generalization is natural, equipping the sparsity among  $\{f_i(\mathbf{x})\}_{i=1}^m$ . Denoting  $w_i^2 = \tau^2 \lambda_i^2$  and assuming that  $\{f_i(\mathbf{x})\}_{i=1}^m$  are mutually independent, we can get  $f(\mathbf{x}) = \sum_{i=1}^m f_i(\mathbf{x}) \sim \mathcal{GP}(0, \tilde{k}(\mathbf{x}, \mathbf{x}'))$ .

The assumption on sparsity among kernel functions  $k_i$  encourages simple kernels which agree with model selection principles like Occam’s razor in Rasmussen and Ghahramani [2001] and BIC in Lloyd et al. [2014].

### 5.2.2 Multi-inducing sparse Gaussian process

To motivate the proposed approach, we will first provide a naive model directly obtained from the sparse Gaussian process. Then, we present our main model.

**SVGP with compositional kernels** Given inducing points  $\mathbf{u}$ , we formulate the corresponding sparse GP model as

$$\begin{aligned}
 f(\cdot)|\mathbf{w}, \mathbf{u} &\sim \mathcal{GP}(\tilde{\mu}(\cdot), \tilde{\Sigma}(\cdot, \cdot)), \\
 \tilde{\mu}(\cdot) &= \tilde{\mathbf{k}}_{\mathbf{u}}^{\top}(\cdot) \tilde{\mathbf{K}}_{\mathbf{u}\mathbf{u}}^{-1} \mathbf{u}, \\
 \tilde{\Sigma}(\cdot, \cdot) &= \tilde{k}(\cdot, \cdot) - \tilde{\mathbf{k}}_{\mathbf{u}}^{\top}(\cdot) \tilde{\mathbf{K}}_{\mathbf{u}\mathbf{u}}^{-1} \tilde{\mathbf{k}}_{\mathbf{u}}(\cdot).
 \end{aligned} \tag{5.2.2}$$

Here, we denote that  $\tilde{\mathbf{K}}_{\mathbf{u}\mathbf{u}} = \sum_{i=1}^m w_i^2 \mathbf{K}_{i\mathbf{u}\mathbf{u}}$ , and  $\mathbf{K}_{i\mathbf{u}\mathbf{u}}$  is the covariance of  $\mathbf{u}$  computed from kernel function  $k_i(\cdot, \cdot)$ .

**Multi-inducing sparse variational GP** Given  $\mathbf{w}$ , we define the model via the combination of conditional posterior distributions:

$$f(\cdot)|\mathbf{U}, \mathbf{w} \sim \mathcal{GP} \left( \sum_{i=1}^m w_i \mu_i(\cdot; \mathbf{u}_i), \sum_{i=1}^m w_i^2 \Sigma_i(\cdot, \cdot; \mathbf{u}_i) \right), \tag{5.2.3}$$

where

$$\begin{aligned}
 \mu_i(\cdot) &= \mathbf{k}_{\mathbf{u}_i}(\cdot)^{\top} \mathbf{K}_{\mathbf{u}_i \mathbf{u}_i}^{-1} \mathbf{u}_i, \\
 \Sigma_i(\cdot, \cdot) &= k_i(\cdot, \cdot) - \mathbf{k}_{\mathbf{u}_i}(\cdot)^{\top} \mathbf{K}_{\mathbf{u}_i \mathbf{u}_i}^{-1} \mathbf{k}_{\mathbf{u}_i}(\cdot).
 \end{aligned} \tag{5.2.4}$$

Here  $\mathbf{k}_{\mathbf{u}_i}(\cdot) = [k_i(\mathbf{u}_i, \cdot)]^{\top}$ , and  $\mathbf{K}_{\mathbf{u}_i \mathbf{u}_i}$  is the covariance of  $\mathbf{u}_i$  w.r.t. kernel  $k_i$ . For convenience, we omit the notation  $\mathbf{u}_i$  in  $\mu_i$  and  $\Sigma_i$ .

Compared to the model in Equation 5.2.2,  $m$  inducing groups of inducing points,  $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^m$  are used. Each  $\mathbf{u}_i$  is responsible for a kernel function  $k_i$ , consisting of  $M_i$  inducing points  $\mathbf{u}_i = [u_{1:M_i}^{(i)}]$  at inducing location  $\mathbf{Z}_i = [\mathbf{z}_{1:M_i}^{(i)}] \in \mathcal{Z}_i$ . The number of inducing points,  $M_i$ , should be much smaller than the number of data points in the data set,  $N$ , (i.e.  $M_i \ll N$ ).

Figure 5.1 compares between the graphical model of SVGP and that of our proposed approach. In simple words, each member in inducing groups is assigned to a single kernel structural representation. We dub this model as Multi-inducing Sparse Variational Gaussian Process (MultiSVGP).

**Discussion** It is obvious that the conditional distribution in Equation 5.2.3 is not equivalent to the conditional distribution of SVGP in Equation 5.2.2 since the inverse of the matrix sum is not equal to the sum of inverse matrices. Our proposed conditional distribution treats each kernel independently with separate inducing points while the condition in SVGP contains correlations between the kernels which are often complicated under the matrix inverse operator.

**Better fit** We hypothesize that the sum of conditional Gaussians is still able to learn from data well compared to other GP models. This can be confirmed by a small experiment in which data are generated from a true model with kernel function  $SE_1 + SE_2 + PER_1$ . We then fit the data with full GP, SVGP model and our proposed approach. Figure 5.2 shows the posterior distributions of these models along with their Wasserstein-2 distance to the true model. We can see that the posterior obtained from our assumption can have good approximation.

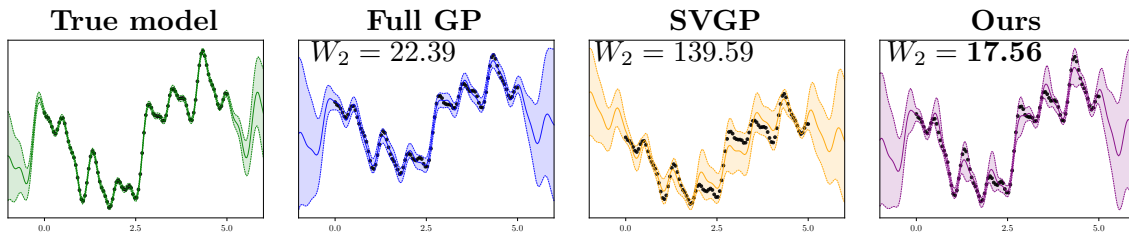


Fig. 5.2 The posterior distributions between models. Here,  $W_2$  is the Wasserstein-2 distance between a model and the true model. The posterior obtained from our approach is close to the true model as well as the full GP model. SVGP model struggles to fit the data.

**Interpretation using inter-domain variational Gaussian Process** Inter-domain Gaussian process [Lázaro-Gredilla and Figueiras-Vidal, 2009] can formulate features which can lie in difference domains. The following adopts the similar approach to explain the proposed model.

Let us consider  $m$  Gaussian processes which are associated to different kernels:

$$g_i(\mathbf{x}) \sim \mathcal{GP}(0, k_i(\mathbf{x}, \mathbf{x}')), \quad i = 1 \dots m. \quad (5.2.5)$$

Because Gaussian process is close under any linear transformation. We consider the following linear transformation

$$u_i(\cdot) = \int \phi_i(\mathbf{x}, \cdot) g_i(\mathbf{x}) d\mathbf{x}.$$

The choice of  $\phi_i(\mathbf{x}, \mathbf{z})$  is a Dirac function included the information of which inducing point group  $\tilde{\mathbf{z}}$  is in:

$$\phi_i(\mathbf{x}, \mathbf{z}) = \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_i\} \delta(\mathbf{z} - \mathbf{x}).$$

Choosing Dirac delta function  $\delta(\cdot)$  is similar to traditional sparse Gaussian process. Whereas  $\mathbb{I}\{\mathbf{z} \in \mathcal{Z}_i\}$  provides the membership information of inducing points in the group. We combine all  $g_i(\mathbf{x})$  to get the model in Equation 5.2.1:

$$f(\mathbf{x}) = \sum_{i=1}^m w_i g_i(\mathbf{x}) \sim \mathcal{GP} \left( 0, \sum_{i=1}^m w_i^2 k_i(\mathbf{x}, \mathbf{x}') \right).$$

Followed by Lázaro-Gredilla and Figueiras-Vidal [2009], the corresponding (cross-) covariance between  $\mathbf{U}$  and  $\mathbf{f}$  can be obtained as

$$\begin{aligned} k_{\mathbf{uf}}(\mathbf{z}, \mathbf{x}) &= \sum_{i=1}^m w_i \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_i\} k_i(\mathbf{z}, \mathbf{x}), \\ k_{\mathbf{uu}}(\mathbf{z}, \mathbf{z}') &= \sum_{i=1}^m \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_i\} \mathbb{I}\{\mathbf{z}' \in \mathcal{Z}_i\} k_i(\mathbf{z}, \mathbf{z}'). \end{aligned} \tag{5.2.6}$$

From the posterior mean and covariance in Equation 5.1.1, we get the same formula in Equation 5.2.3.

Here, we compare the approximate quality of MultiSVGP and that of SVGP. Here, we argue that our approximate posterior can at least as good as SVGP. Let  $\hat{P}$  be the true posterior of GP,  $Q_{\text{multi}}$  be the variational approximation of MultiSVGP in Equation 5.2.3 and  $Q_{\text{single}}$  be the variational approximation of SVGP in Equation 5.2.2.

The case<sup>1</sup>  $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$  is considered. Let  $\lambda_i, \lambda_i^{(1)}$ , and  $\lambda_i^{(2)}$  be the  $i$ -th operator eigenvalue w.r.t.  $k, k_1$ , and  $k_2$ . MultiSVGP has  $M$  inducing points in each inducing group. SVGP has  $M$  inducing points. The error bound for the SVGP is

$$\text{KL}(Q_{\text{single}} || \hat{P}) \leq \frac{C_{\text{single}}}{2\sigma_n^2 \delta} \left( 1 + \frac{\|\mathbf{y}\|_2^2}{\sigma_n^2} \right),$$

---

<sup>1</sup>without loss of generality,  $w_i$  is set to 1

with probability at least  $1 - \delta$  [Burt et al., 2019]. Here,  $C_{\text{single}} = \sum_{i=M+1}^{\infty} \lambda_i$ , and  $\sigma_n^2$  is a Gaussian noise variance. In MultiSVGP, the KL divergence between  $\hat{P}$  and  $Q_{\text{multi}}$  is bounded by the following proposition.

**Proposition 2.** *Given  $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$ , with probability at least  $1 - \delta$ , we have*

$$\text{KL}(Q_{\text{multi}}||\hat{P}) \leq \frac{C_{\text{multi}}}{2\sigma_n^2\delta} \left( 1 + \frac{\|\mathbf{y}\|_2^2}{\sigma_n^2} \right),$$

with  $C_{\text{multi}} = N \sum_{i=1}^M (\lambda_i - \lambda_i^{(1)} - \lambda_i^{(2)}) + N \sum_{j=M+1}^{\infty} \lambda_j$ .

Furthermore, it is true that  $C_{\text{multi}} \leq C_{\text{single}}$ , making the upper bound of  $\text{KL}(Q_{\text{multi}}||\hat{P})$  is smaller or equal than the upper bound of  $\text{KL}(Q_{\text{single}}||\hat{P})$ .

*Proof.* In Lemma 1 of Burt et al. [2019], we have

$$\text{KL}(Q_{\text{multi}}||\hat{P}) \leq \frac{t}{2\sigma_n^2} \left( 1 + \frac{\|\mathbf{y}\|_2^2}{\sigma_n^2 + t} \right).$$

Here  $t = \text{trace}(\mathbf{K}_{\text{ff}} - \underbrace{\mathbf{K}_{\text{uf}}^{\top} \mathbf{K}_{\text{uu}}^{-1} \mathbf{K}_{\text{uf}}}_{\mathbf{Q}_{\text{ff}}})$ .  $\mathbf{K}_{\text{uf}}$  is the cross-covariance matrix between inducing points  $\mathbf{U}$  and  $\mathbf{f}$ .  $\mathbf{K}_{\text{uu}}$  is the covariance matrix of  $\mathbf{U}$ .

Recall the covariance between  $\mathbf{U}$  and  $\mathbf{f}$  in Equation 5.2.6:

$$\begin{aligned} k_{\text{uf}}(\mathbf{z}, \mathbf{x}) &= \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_1\} k_1(\mathbf{z}, \mathbf{x}) + \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_2\} k_2(\mathbf{z}, \mathbf{x}), \\ k_{\text{uu}}(\mathbf{z}, \mathbf{z}') &= \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_1\} \mathbb{I}\{\mathbf{z}' \in \mathcal{Z}_1\} k_1(\mathbf{z}, \mathbf{z}') + \mathbb{I}\{\mathbf{z} \in \mathcal{Z}_2\} \mathbb{I}\{\mathbf{z}' \in \mathcal{Z}_2\} k_2(\mathbf{z}, \mathbf{z}'), \end{aligned}$$

where  $\mathbb{I}(\mathbf{z} \in \mathcal{Z}_i) = 1$  if  $\mathbf{z} \in \mathcal{Z}_i$ , otherwise it is equal 0. Then, we can compute

$$\begin{aligned} \mathbf{K}_{\text{uf}} &= \text{Cov} \left( \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix}, \mathbf{f} \right) = \begin{bmatrix} \mathbf{K}_{\mathbf{u}_1\mathbf{f}} \\ \mathbf{K}_{\mathbf{u}_2\mathbf{f}} \end{bmatrix}, \\ \mathbf{K}_{\text{uu}} &= \text{Cov} \left( \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} \right) = \begin{bmatrix} \mathbf{K}_{\mathbf{u}_1\mathbf{u}_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{\mathbf{u}_2\mathbf{u}_2} \end{bmatrix}, \end{aligned}$$

where  $\mathbf{K}_{\mathbf{u}_i\mathbf{f}}$  is the covariance matrix computed from  $k_1(\cdot, \cdot)$  with  $\mathbf{u}_1 \in \mathcal{Z}_1$ , and  $\mathbf{K}_{\mathbf{u}_1\mathbf{u}_1}$  is the covariance matrix of  $\mathbf{u}_1 \in \mathcal{Z}_1$  computed from  $k_1(\cdot, \cdot)$ . The same is applied to  $\mathbf{K}_{\mathbf{u}_2\mathbf{f}}$  and  $\mathbf{K}_{\mathbf{u}_2\mathbf{u}_2}$ .

From  $\mathbf{Q}_{\text{ff}} = \mathbf{K}_{\text{uf}}^{\top} \mathbf{K}_{\text{uu}}^{-1} \mathbf{K}_{\text{uf}}$  and block matrix multiplication, we can obtain

$$\mathbf{Q}_{\text{ff}} = \mathbf{K}_{\mathbf{u}_1\mathbf{f}}^{\top} \mathbf{K}_{\mathbf{u}_1\mathbf{u}_1}^{-1} \mathbf{K}_{\mathbf{u}_1\mathbf{f}} + \mathbf{K}_{\mathbf{u}_2\mathbf{f}}^{\top} \mathbf{K}_{\mathbf{u}_2\mathbf{u}_2}^{-1} \mathbf{K}_{\mathbf{u}_2\mathbf{f}}.$$



Let  $\psi_i^{(1)}$  and  $\psi_i^{(2)}$  be the eigenfunctions of the covariance operators w.r.t.  $k_1(\mathbf{x}, \mathbf{x}')$  and  $k_2(\mathbf{x}, \mathbf{x}')$ . Similar to [Burt et al. \[2019\]](#), we interpret the individual terms in  $\mathbf{Q}_{\mathbf{ff}}$  under the eigenfeature representation as

$$[\mathbf{K}_{\mathbf{fu}_1} \mathbf{K}_{\mathbf{u}_1 \mathbf{u}_1}^{-1} \mathbf{K}_{\mathbf{u}_1 \mathbf{f}}]_{c,r} = \sum_{i=1}^M \lambda_i^{(1)} \psi_i^{(1)}(\mathbf{x}_c) \psi_i^{(1)}(\mathbf{x}_r),$$

$$[\mathbf{K}_{\mathbf{fu}_2} \mathbf{K}_{\mathbf{u}_2 \mathbf{u}_2}^{-1} \mathbf{K}_{\mathbf{u}_2 \mathbf{f}}]_{c,r} = \sum_{i=1}^M \lambda_i^{(2)} \psi_i^{(2)}(\mathbf{x}_c) \psi_i^{(2)}(\mathbf{x}_r).$$

By Mercer's theorem and with  $\psi_i(\mathbf{x})$  be the eigenfunctions of covariance operator for  $k(\mathbf{x}, \mathbf{x}')$ , we have

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \psi_i(\mathbf{x}) \psi_i(\mathbf{x}').$$

Then the entry at  $(n, n)$  of  $\mathbf{K}_{\mathbf{ff}} - \mathbf{Q}_{\mathbf{ff}}$  is

$$[\mathbf{K}_{\mathbf{ff}} - \mathbf{Q}_{\mathbf{ff}}]_{n,n} = \sum_{i=1}^{\infty} \lambda_i \psi_i^2(\mathbf{x}_n) - \left( \sum_{i=1}^M \lambda_i^{(1)} (\psi_i^{(1)})^2(\mathbf{x}_n) + \sum_{i=1}^M \lambda_i^{(2)} (\psi_i^{(2)})^2(\mathbf{x}_n) \right).$$

From this, we can have the expectation of  $t$

$$\mathbb{E}_{\mathbf{x}}[t] = N \sum_{i=1}^{\infty} \lambda_i - N \left( \sum_{i=1}^M \lambda_i^{(1)} + \sum_{i=1}^M \lambda_i^{(2)} \right).$$

Here the eigenfunction terms are disappeared. Because  $\mathbb{E}[\psi_i^2(\mathbf{x})] = \int \psi^2(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = 1$ . Similarly,  $\mathbb{E}[(\psi_i^{(1)})^2(\mathbf{x})] = \mathbb{E}[(\psi_i^{(2)})^2(\mathbf{x})] = 1$ .

According to [Burt et al. \[2019\]](#), we apply the Markov's inequality, we have, with probability at least  $1 - \delta$ ,

$$\text{KL}(Q_{\text{multi}} || \hat{P}) \leq \frac{C_{\text{multi}}}{2\sigma^2\delta} \left( 1 + \frac{\|\mathbf{y}\|_2^2}{\sigma_n^2} \right),$$

where  $C_{\text{multi}} = N \sum_{i=1}^M (\lambda_i - \lambda_i^{(1)} - \lambda_i^{(2)}) + N \sum_{j=M+1}^{\infty} \lambda_j$ .

Comparing to  $C_{\text{single}} = \sum_{i=M+1}^{\infty} \lambda_i$ , we use the result in [Tao \[2010\]](#); [Wielandt \[1955\]](#) where

$$\sum_{i=1}^M \lambda_i \leq \sum_{i=1}^M \lambda_i^{(1)} + \sum_{i=1}^M \lambda_i^{(2)}.$$

We can conclude that the upper bound of  $\text{KL}(Q_{\text{multi}} || \hat{P})$  is smaller than the upper bound of  $\text{KL}(Q_{\text{single}} || \hat{P})$ .  $\square$

This is considered a theoretical justification for the comparison in [Figure 5.2](#).

### 5.3 Variational inference with shrinkage prior

Having an introduction of kernel selection problem in the previous section, this section presents the variational inference method for this model. Here, the variational distribution which is factorized into two parts: the approximate posterior distribution of GP latent variables and that of sparse vector  $\mathbf{w}$ . Let  $q(\mathbf{w})$  be the variational distribution over  $\mathbf{w}$ . The distribution  $q(\mathbf{f}, \mathbf{U}, \mathbf{w}) = q(\mathbf{f}, \mathbf{U})q(\mathbf{w})$  approximates the true posterior. Similar to the approximate posterior construction of SVGP,  $q(\mathbf{f}, \mathbf{U})$  is formulated as  $p(\mathbf{f}|\mathbf{U})q(\mathbf{U})$ , with  $q(\mathbf{U}) = \prod_{i=1}^m q(\mathbf{u}_i)$  and  $q(\mathbf{u}_i)$  is parameterized by  $\mathcal{N}(\mathbf{m}_i, \mathbf{S}_i)$ . The goal is to maximize the evidence lower bound (ELBO) defined as

$$\begin{aligned} \mathcal{L} &= \mathbb{E} \left[ \log \frac{p(\mathbf{y}, \mathbf{f}, \mathbf{U}, \mathbf{w})}{q(\mathbf{f}, \mathbf{u}, \mathbf{w})} \right] \\ &= \mathbb{E}_{p(f(\cdot))} \left[ \mathbb{E}_{q(\mathbf{w})} [\log p(\mathbf{y}|\mathbf{f}, \mathbf{w})] \right] \\ &\quad - \text{KL}(q(\mathbf{U})||p(\mathbf{U})) - \text{KL}(q(\mathbf{w})||p(\mathbf{w})). \end{aligned} \quad (5.3.1)$$

After marginalizing all  $\mathbf{u}_i$ ,  $p(f(\cdot))$  is obtained in the same manner with SVGP [Hensman et al., 2013]. The KL divergence related to  $\mathbf{U}$  is the sum of the KL divergences of  $\mathbf{u}_i$ .

We describe the subroutine for variational inference w.r.t.  $\mathbf{w}$ , represented by Horseshoe variables,  $\tau$  and  $\boldsymbol{\lambda}$ . Because of the flat-tailed property of Half-Cauchy distribution, it is reparameterized by double inverse Gamma distributions [Ghosh et al., 2019; Wand et al., 2011]. That is, if  $a \sim \mathcal{C}^+(b)$ , this corresponds to  $a^2 \sim \text{IG}(1/2, \phi_a^{-1})$  and  $\phi_a \sim \text{IG}(1/2, b^{-1})$  as an auxiliary variable,  $\phi_a$ , is introduced. To this end, including the additional auxiliary variables, the prior contains variables  $\{\tau, \boldsymbol{\lambda}, \phi_\tau, \phi_\lambda\}$  as

$$\begin{aligned} \tau^2 | \phi_\tau &\sim \text{IG}(1/2, \phi_\tau^{-1}), & \phi_\tau &\sim \text{IG}(1/2, A^{-1}), \\ \lambda_i^2 | \phi_{\lambda_i} &\sim \text{IG}(1/2, \phi_{\lambda_i}^{-1}), & \phi_{\lambda_i} &\sim \text{IG}(1/2, B^{-1}). \end{aligned}$$

The mean-field approach is further used to factorize the variational distribution  $q(\tau, \boldsymbol{\lambda}, \phi_\tau, \phi_\lambda)$ . Specifically, the variational distributions of  $\tau$  and  $\lambda_i$  are chosen as log-normal distributions

$$\begin{aligned} q(\tau^2) &= \text{Lognormal}(\tau^2; \mu_\tau, \sigma_\tau^2), \\ q(\lambda_i^2) &= \text{Lognormal}(\lambda_i^2; \mu_{\lambda_i}, \sigma_{\lambda_i}^2). \end{aligned}$$

Whereas  $q(\phi_\tau)$  and  $q(\phi_{\lambda_i})$  remain inverse Gamma distribution.

As we replace  $\mathbf{w}$  with  $\{\tau, \boldsymbol{\lambda}\}$ , we have the expectation  $\mathbb{E}_{q(\tau)q(\boldsymbol{\lambda})} [\log p(\mathbf{y}|\mathbf{f}, \tau, \boldsymbol{\lambda})]$  which is estimated by Monte Carlo integration.  $q(\tau)$  and  $q(\boldsymbol{\lambda})$  using reparameterization tricks for Log

normal distributions [Kingma and Welling, 2014]. In particular, since the product  $\tau^2\lambda_i^2$  is also Log normal, it can be reparameterized by  $\exp(\mu_\tau + \mu_{\lambda_i} + \varepsilon(\sigma_\tau + \sigma_{\lambda_i}))$  with  $\varepsilon \sim \mathcal{N}(0, 1)$ . We provide the detailed derivation of ELBO in Section 5.4.

**Computational complexity** Compared to SVGP using single inducing points, MultiSVGP takes  $\mathcal{O}(m \max_i \{M_i^2\} b)$  at each optimization iteration with minibatch size  $b$ . Again,  $M_i$  is the number of inducing points  $\mathbf{u}_i$ .

## 5.4 Detail of variational inference

**Prior** Recall the prior over  $\tau, \boldsymbol{\lambda}, \phi_\tau, \phi_\lambda$  after reparameterization is

$$\begin{aligned}\tau^2 | \phi_\tau &\sim \text{IG}(\tau^2 | 1/2, \phi_\tau^{-1}), \\ \phi_\tau &\sim \text{IG}(\phi_\tau | 1/2, A^{-1}), \\ \lambda_i^2 | \phi_{\lambda_i} &\sim \text{IG}(\lambda_i^2 | 1/2, \phi_{\lambda_i}^{-1}), \quad i = 1 \dots m, \\ \phi_{\lambda_i} &\sim \text{IG}(\phi_{\lambda_i} | 1/2, B^{-1}).\end{aligned}$$

**Variational distribution** The variational distributions of  $\tau$  and  $\lambda_i$  are in the form of log normal distribution.

$$\begin{aligned}q(\tau^2) &= \text{Lognormal}(\tau^2 | m_\tau, \sigma_\tau^2) \\ q(\lambda_i^2) &= \text{Lognormal}(\lambda_i^2 | m_{\lambda_i}, \sigma_{\lambda_i}^2), \quad i = 1 \dots m.\end{aligned}$$

On the other hand, the variational distributions of the auxiliary variables  $\phi_\tau$  and  $\phi_{\lambda_i}$  remain as Inverse Gamma distributions

$$\begin{aligned}q(\phi_\tau) &= \text{IG}(\phi_\tau | s_\tau, r_\tau) \\ q(\phi_{\lambda_i}) &= \text{IG}(\phi_{\lambda_i} | s_{\lambda_i}, r_{\lambda_i}), \quad i = 1 \dots m.\end{aligned}$$

**KL divergence** As  $\mathbf{w} = \{\tau, \boldsymbol{\lambda}, \phi_\tau, \phi_\lambda\}$ , the KL divergence  $\text{KL}(q(\mathbf{w})||p(\mathbf{w}))$  becomes

$$\begin{aligned}
& \text{KL} \left( q(\tau^2)q(\phi_\tau) \prod_i q(\lambda_i^2)q(\phi_{\lambda_i}) || p(\tau^2|\phi_\tau)p(\phi_\tau) \prod_i p(\lambda_i|\phi_{\lambda_i})p(\lambda_i) \right) \\
&= H[q(\tau^2)] + H[q(\phi_\tau)] + \sum_i H[q(\lambda_i^2)] + \sum_i H[q(\phi_{\lambda_i})] + \\
& \mathbb{E}_{q(\tau^2)q(\phi_\tau)}[\log p(\tau^2|\phi_\tau)] + \mathbb{E}_{q(\phi_\tau)}[\log p(\phi_\tau)] + \\
& \sum_i \mathbb{E}_{q(\lambda_i)q(\phi_{\lambda_i})}[\log p(\lambda_i^2|\phi_{\lambda_i})] + \sum_i \mathbb{E}_{q(\phi_{\lambda_i})}[\log p(\phi_{\lambda_i})].
\end{aligned} \tag{5.4.1}$$

where  $H[\cdot]$  denotes the entropy of a distribution.

Individual terms will be explained as following. The entropy terms will be computed as

$$\begin{aligned}
H[q(\tau^2)] &= \mu_\tau + \frac{1}{2} \log(2\pi e\sigma_\tau^2), \\
H[q(\lambda_i^2)] &= \mu_{\lambda_i} + \frac{1}{2} \log(2\pi e\sigma_{\lambda_i}^2).
\end{aligned}$$

The expectations of log prior can be derived as

$$\begin{aligned}
& \mathbb{E}_{q(\tau^2)q(\phi_\tau)}[\log p(\tau^2|\phi_\tau)] \\
&= \mathbb{E}_{q(\tau^2)q(\phi_\tau)} \left[ \log \text{IG}(\tau^2|1/2, \phi_\tau^{-1}) \right] \\
&= \mathbb{E}_{q(\tau^2)q(\phi_\tau)} \left[ -\frac{1}{2} \log \phi_\tau - \log \Gamma(1/2) - \frac{3}{2} \log(\tau^2) - \frac{1}{\tau^2 \phi_\tau} \right] \\
&= -\frac{1}{2} \mathbb{E}_{q(\phi_\tau)}[\log \phi_\tau] - \log \Gamma(1/2) - \frac{3}{2} \mathbb{E}_{q(\tau^2)}[\log(\tau^2)] - \mathbb{E}_{q(\tau^2)}[\tau^{-2}] \mathbb{E}_{\phi_\tau}[\phi_\tau^{-1}],
\end{aligned}$$

where the individual terms can be calculated as

$$\begin{aligned}
\mathbb{E}_{q(\phi_\tau)}[\log \phi_\tau] &= \log r_\tau - \psi(s_\tau), & (\text{Inverse Gamma distribution property}) \\
\mathbb{E}_{q(\phi_\tau)}[\phi_\tau^{-1}] &= \frac{s_\tau}{r_\tau}, & (\text{Inverse Gamma distribution property}) \\
\mathbb{E}_{q(\tau^2)}[\log(\tau^2)] &= \mu_\tau & (\text{compute from log normal distribution}) \\
\mathbb{E}_{q(\tau^2)}[\tau^{-2}] &= \exp(-\mu_\tau + \frac{1}{2}\sigma_\tau^2). & (\text{Log normal distribution property})
\end{aligned}$$

Here,  $\psi(\cdot)$  is the digamma function. Similarly, we can obtain the expectation of log prior w.r.t to  $\lambda_i$

$$\begin{aligned} & \mathbb{E}_{q(\lambda_i^2)q(\phi_{\lambda_i})}[\log p(\lambda_i^2|\phi_{\lambda_i})] \\ &= -\frac{1}{2}\mathbb{E}_{q(\phi_{\lambda_i})}[\log \phi_{\lambda_i}] - \log \Gamma(1/2) - \frac{3}{2}\mathbb{E}_{q(\lambda_i^2)}[\log(\lambda_i^2)] - \mathbb{E}_{q(\lambda_i^2)}[\lambda_i^{-2}]\mathbb{E}_{\phi_{\lambda_i}}[\phi_{\lambda_i}^{-1}]. \end{aligned}$$

We intentionally do not write the explicit form of  $H[q(\phi_\tau)]$ ,  $H[q(\phi_{\lambda_i})]$ ,  $\mathbb{E}_{q(\phi_\tau)}[\log p(\phi_\tau)]$  and  $\mathbb{E}_{q(\phi_{\lambda_i})}[\log p(\phi_{\lambda_i})]$  because the variables  $\phi_\tau$  and  $\phi_{\lambda_i}$  do not follow an optimization but are updated by the following.

**Closed-form update for  $q(\phi_\tau)$  and  $q(\phi_{\lambda_i})$**  Under the mean-field assumption on variational variables  $\tau, \boldsymbol{\lambda}, \phi_\tau, \phi_{\lambda_i}$ , we can obtain the closed-form optimal solution w.r.t. the auxiliary variables  $\phi_\tau, \phi_{\lambda_i}$  Neville et al. [2014]. That is, after each optimization step on other variables, we update  $q(\phi_\tau)$  and  $q(\phi_{\lambda_i})$  by

$$\begin{aligned} q(\phi_\tau) &= \text{IG}(s_\tau = 1, r_\tau = \mathbb{E}[\tau^{-2}] + A^{-2}), \\ q(\phi_{\lambda_i}) &= \text{IG}(s_{\lambda_i} = 1, r_{\lambda_i} = \mathbb{E}[\lambda_i^{-2}] + B^{-2}). \end{aligned} \tag{5.4.2}$$

**Evidence lower bound** Recap that the evidence lower bound is in the following form:

$$\mathcal{L} = \mathbb{E}_{p(f(\cdot))} \left[ \mathbb{E}_{q(\tau, \boldsymbol{\lambda})} [\log p(\mathbf{y}|\mathbf{f}, \tau, \boldsymbol{\lambda})] \right] - \text{KL}(q(\mathbf{U})||p(\mathbf{U})) - \text{KL}(q(\tau, \boldsymbol{\lambda})||p(\tau, \boldsymbol{\lambda})). \tag{5.4.3}$$

Note that the expectation w.r.t  $\tau, \boldsymbol{\lambda}$  is estimated by Monte Carlo integration. During training, we draw one sample  $\tau_S$  and  $\boldsymbol{\lambda}_S$  by the reparameterization trick for the product  $\tau_S \lambda_{i_S} = \exp(\mu_\tau + \mu_{\lambda_i} + \varepsilon(\sigma_\tau + \sigma_{\lambda_i}))$  where  $\varepsilon \sim \mathcal{N}(0, 1)$ .

The following algorithm describes our variational inference

---

**Algorithm 3** Variational inference for MultiSVGP with Horseshoe prior

---

**Require:** Data  $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ , a set of kernel function  $\{k_i(\mathbf{x}, \mathbf{x}')\}_{i=1}^m$   
Initialize kernel hyperparameters, variational parameters  $\{\mu_\tau, \sigma_\tau^2, \mu_{\lambda_i}, \sigma_{\lambda_i}^2, s_\tau, r_\tau, s_{\lambda_i}, r_{\lambda_i}\}$   
**for** within a number of iterations **do**  
  Sample a minibatch  $(\mathbf{x}_b, \mathbf{y}_b)$   
  Sample  $\tau_S, \boldsymbol{\lambda}_S$  with  $\tau_S \lambda_{i_S} = \exp(\mu_\tau + \mu_{\lambda_i} + \varepsilon(\sigma_\tau + \sigma_{\lambda_i}))$  where  $\varepsilon \sim \mathcal{N}(0, 1)$   
  Compute  $\mathbb{E}_{p(f(\mathbf{x}_b))} \left[ \mathbb{E}_{q(\tau, \boldsymbol{\lambda})} [\log p(\mathbf{y}_b|\mathbf{f}, \tau, \boldsymbol{\lambda})] \right] \approx \mathbb{E}_{p(f(\mathbf{x}_b))} [\log p(\mathbf{y}_b|\mathbf{f}, \tau_S, \boldsymbol{\lambda}_S)]$   
  Compute  $\text{KL}(q(\mathbf{U})||p(\mathbf{U}))$  as the sum of  $\text{KL}(q(\mathbf{u}_i)||p(\mathbf{u}_i))$   
  Compute  $\text{KL}(q(\tau, \boldsymbol{\lambda})||p(\tau, \boldsymbol{\lambda}))$  by Equation 5.4.1  
  Compute ELBO  $\mathcal{L}$  based on Equation 5.4.3  
  Perform an optimization step for ELBO  $\mathcal{L}$   
  Update  $q(\phi_\tau)$  and  $q(\phi_{\lambda_i})$  by Equation 5.4.2  
**end for**

---

Table 5.1 Extrapolation performance in UCI benchmarks. Results are aggregated from 10 independent runs.

	RMSE				Test log-likelihood			
	SVGP-SE	No prior	GP-NKN	Ours	SVGP-SE	No prior	GP-NKN	Ours
boston	7.30 $\pm$ 0.21	7.24 $\pm$ 0.27	5.53 $\pm$ 0.49	<b>5.41</b> $\pm$ 0.10	-3.72 $\pm$ 0.07	-3.72 $\pm$ 0.10	-3.77 $\pm$ 0.26	<b>-3.24</b> $\pm$ 0.11
concrete	9.64 $\pm$ 0.14	8.70 $\pm$ 1.05	<b>6.44</b> $\pm$ 0.19	7.39 $\pm$ 0.42	-3.54 $\pm$ 0.01	-3.45 $\pm$ 0.08	<b>-3.10</b> $\pm$ 0.01	-3.33 $\pm$ 0.06
energy	0.83 $\pm$ 0.07	0.69 $\pm$ 0.18	0.41 $\pm$ 0.03	<b>0.37</b> $\pm$ 0.05	-1.11 $\pm$ 0.03	-1.07 $\pm$ 0.08	<b>-0.54</b> $\pm$ 0.04	-0.76 $\pm$ 0.05
kin8nm	0.11 $\pm$ 0.00	0.11 $\pm$ 0.08	<b>0.09</b> $\pm$ 0.00	<b>0.09</b> $\pm$ 0.01	0.71 $\pm$ 0.01	0.74 $\pm$ 0.02	<b>1.02</b> $\pm$ 0.05	0.89 $\pm$ 0.01
wine	<b>0.62</b> $\pm$ 0.00	<b>0.62</b> $\pm$ 0.01	0.67 $\pm$ 0.01	<b>0.63</b> $\pm$ 0.01	-1.04 $\pm$ 0.00	-1.04 $\pm$ 0.00	<b>-1.01</b> $\pm$ 0.01	-1.04 $\pm$ 0.01
yacht	1.45 $\pm$ 0.10	1.22 $\pm$ 0.44	0.46 $\pm$ 0.05	<b>0.36</b> $\pm$ 0.05	-1.91 $\pm$ 0.14	-1.67 $\pm$ 0.46	<b>-0.63</b> $\pm$ 0.02	-0.83 $\pm$ 0.12

## 5.5 Experimental Evaluations

This section first sets up the choices for compositional kernels. We then test how the Horseshoe prior for kernel selection on synthetic data as well as time series data. Finally, we demonstrate our model in both regression and classification tasks. The source code is developed based on [Matthews et al. \[2017\]](#).

### 5.5.1 Kernel function pool

Now, we present our approach in designing kernel structures for  $\{k_i(\mathbf{x}, \mathbf{x}')\}$ . A kernel function is constructed as a form of multiplicative kernel  $\prod_{i=1}^{\alpha} \mathcal{B}_i$  where  $\mathcal{B}_i$  is a base kernel taking from SE, LIN, PER. In our experiment, the kernel pool is composed of all possible kernels having the multiplicative order up to 2. We allow duplication in kernels structure. The total number of kernels in the pool is 24. Each kernel in the pool remains interpretable and can be described by natural language explanations [[Lloyd et al., 2014](#)].

**Hyperparameter initialization** [[Kim and Teh, 2018](#); [Vanhatalo et al., 2013](#)] suggests two types of hyperparameter initialization: weak prior and strong prior. Unlike existing approaches requiring multiple restarts, we made sure our kernel pool covers both of them.

**Behavior of Horseshoe prior** To see how Horseshoe prior behaves for kernel selection problem, we created a synthetic data  $(x_i, y_i)_{i=1}^{100}$  with  $x_i \in [-5, 5]$  and  $y_i$  generated from kernel  $\text{PER}_1 + \text{SE} \times \text{PER}_2$ . We train our model and compare to the case where there is no prior on weights  $\mathbf{w}$ . Figure 5.3 shows our model spike at the relevant kernel structure (SE  $\times$  PER) while the model with no prior mistakenly assign weights for local variations (SE).

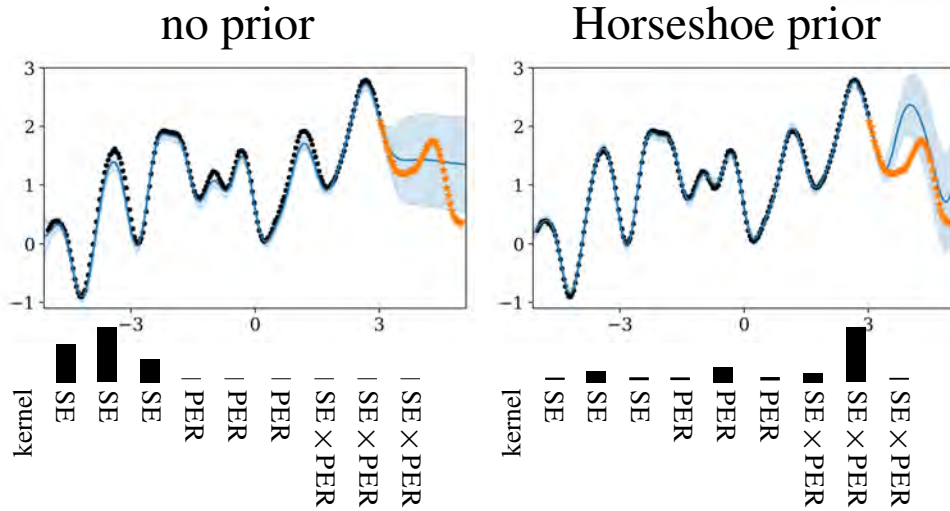


Fig. 5.3 Behavior of Horseshoe prior in kernel selection. Both models predicts the test data ( $\star$ ). The bar plots are the weights  $w_i$  corresponding to  $k_i$ .

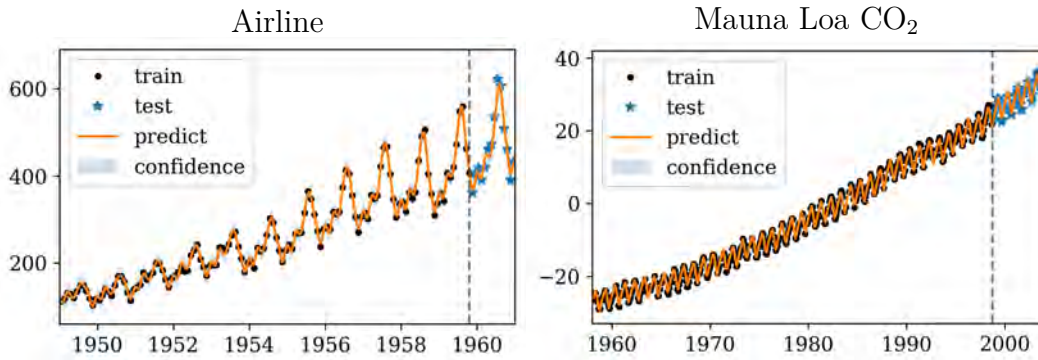


Fig. 5.4 Extrapolation on time series data sets.

**Small-sized 1D regression** We verify our model on small-sized data sets: airline passenger volume, Mauna Loa CO<sub>2</sub> concentration. Figure 5.4 shows that our model can fit the data well. In the airline data set, the obtained kernel includes PER  $\times$  SE, LIN and SE while [Lloyd et al., 2014] reports LIN + PER  $\times$  SE  $\times$  LIN + SE and a heteroscedastic noise. Also, in the mauna data set, the model can explain the trend and periodicity in data. Our model can reduce the running time to less than 0.5 hour comparing to 10 – 12 hours like [Duvenaud et al., 2013; Lloyd et al., 2014] or 2.5 – 4 hours like [Kim and Teh, 2018]. Figure 5.5 provides the visualization of weights  $\mathbf{w}$  found by our model comparing to the model without imposing any prior. This figure also gives the decomposition from  $\mathcal{GP}(\sum w_i \mu_i(\cdot), \sum w_i^2 \Sigma(\cdot, \cdot))$  corresponding with the most important components.

**Medium-sized 1D regression** We test our model on GEFCOM data set from the Global Energy Forecasting Competition [Tao Hong, Pierre Pinson, and Shu Fan, 2014]. The data set

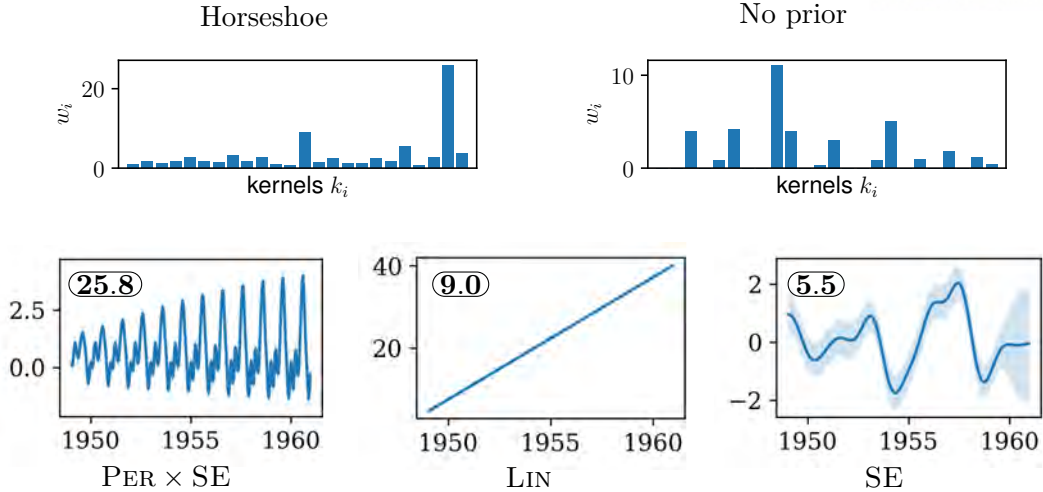


Fig. 5.5 First row: the weights  $w_i$  in two cases. Second row: our kernel decomposition for airline data with three most significant components  $\mathcal{GP}(\mu_i(\cdot), \Sigma_i(\cdot, \cdot))$ . The weights  $w_i$  are showed at the upper-left corners.

has  $N = 38,070$  data points containing hourly records of energy load from January 2004 to June 2008. We randomly take 90% of the data set for training and held out 10% as test data. We compare our model with SVGP with no prior and SVGP with Softmax [Teng et al., 2020].

Figure 5.6 compares the predictive posteriors on the test set. It is clear that our model fits better, giving more accurate predictions as well as uncertainty estimation. The approach in [Teng et al., 2020] takes the second places. The inducing points are associated with complicated kernel function, not divided for each additive kernel. Therefore, the approximate capacity of this model is still more restricted than ours due to Proposition 2.

Our model found  $SE_1 \times PER_1 + SE_2 + SE_3$  as the kernel structure for this data. This agrees with the kernel function in [Lloyd, 2013] which is manually chosen. Also, our PER kernel has periodicity 1.001 days which also can describe the property that there are peaks in the morning and evening. This is aligned with the result reported in [Kim and Teh, 2018].

**Higher-dimension regression** We conducted experiments on UCI data sets [Dheeru and Karra Taniskidou, 2017b] including boston, concrete, energy, kin8nm, wine and yatch (see Table 5.2 for detailed descriptions). We consider baseline models: GP-NKN [Sun et al., 2018], SVGP with no shrinkage prior over  $\mathbf{w}$  (no prior), and SVGP with SE kernel (SVGP-SE). To justify the extrapolation performance, we projected data onto the principal component of data and sorted data according to the projection [Sun et al., 2018]. From sorted indices, test data is taken from top  $1/15$  and bottom  $1/15$  of the data, the remaining is train data. We measure the root mean square error (RMSE) and test log-likelihood in each model.



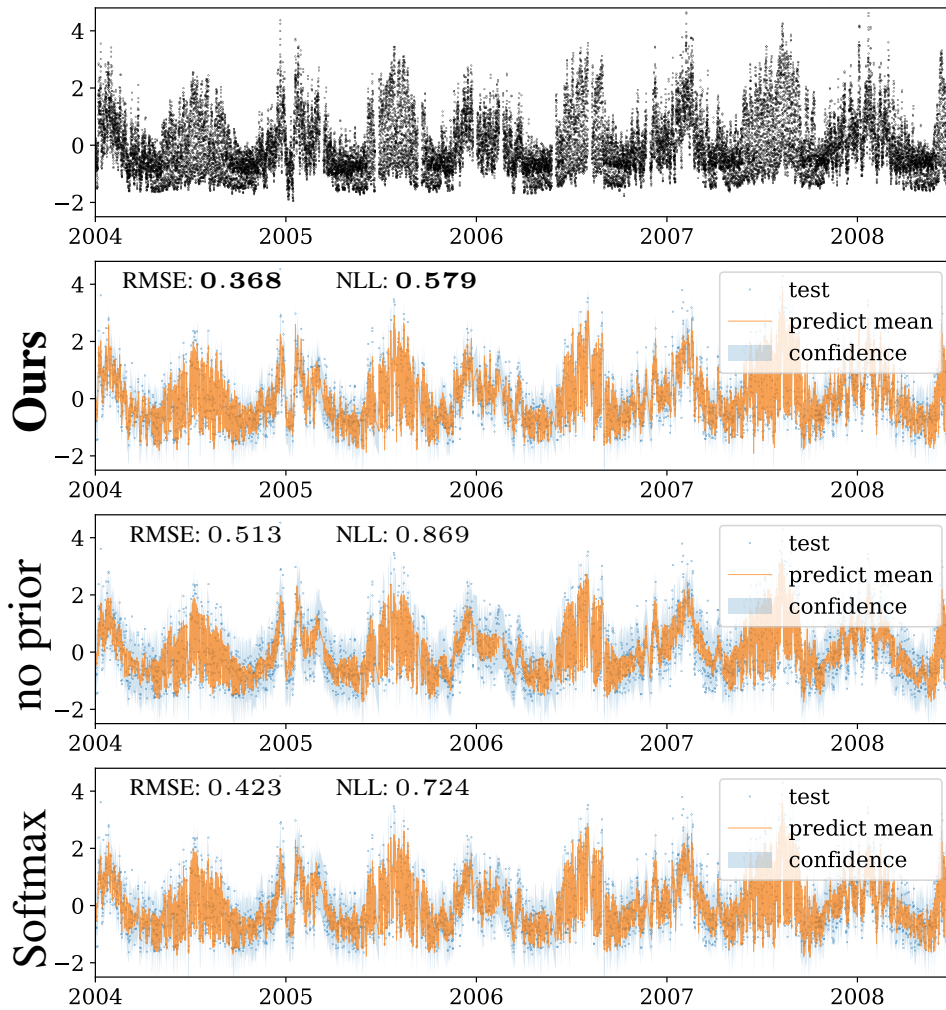


Fig. 5.6 GEFCOM data set. First row is the plot of training data. The next rows are the predictive posterior at test points. Our model outperforms the alternatives in term of root mean square error (RMSE) and test negative log-likelihood (NLL).

Table 5.1 shows that our model has a competitive extrapolation capability comparing to GP-NKN. Roughly, our model has better performance in terms of RMSE for most of data sets, except concrete data set. In boston data set, our model performs well for the predictive log-likelihood. Still, GP-NKN consistently outperforms others in this measurement. This is because this model is still considered as a full GP model retaining good uncertainty quantification while the remaining methods including ours are sparse GPs. However, GP-NKN takes significantly more time to train, e.g. in kin8nm. Although the model with no prior has a highly complex kernel, it fails to this extrapolation task. On the other hand, our model with shrinkage prior demonstrates the effect of regularization in kernel selection, resulting in better predictions.

**Improving additive GPs** [Duvenaud et al., 2011] propose additive kernels for GPs to prevent the local property of kernel functions taking all input dimensions [Bengio et al., 2006]. The

Table 5.2 Description of UCI data sets

Data set	# data $N$	Dimension $D$	Description
boston	506	13	Boston housing price
concrete	1030	8	Predict concrete compressive strength
energy	768	8	Predict energy efficiency for buildings
kin8nm	8192	8	Kinematics of an 8 link robot arm
wine	1599	22	Wine quality data set
yacht	308	7	Prediction of residuary resistance of sailing yachts

Table 5.3 Description of heart, liver, pima data set

Data set	# data $N$	Dimension $D$	Description
heart	303	13	Predict the presence of heart disease
liver	345	6	Predict liver disorders
pima	768	8	Pima Indians Diabetes Database

additive kernel is the sum of lower-dimensional kernel functions which depend on a subset of input variables.

Suppose  $D$  is the dimension of data. Let  $S_D = \{1, \dots, D\}$  be the index set of dimensions. We adopt this approach and consider the  $d$ -order additive kernel

$$k_d(\mathbf{x}, \mathbf{x}') = \sum_{\{i_1, \dots, i_d\} \subseteq S_D} w_{i_1 \dots i_d}^2 \prod_{i \in \{i_1, \dots, i_d\}} k(\mathbf{x}[i], \mathbf{x}'[i]).$$

Unlike [Duvenaud et al., 2011] treating weights  $w_{i_1 \dots i_d}$  equally in the same order  $d$ , we learn  $\mathbf{w} = [w_{i_1 \dots i_d}]$  with our model. We conduct the experiment in three data sets: heart, liver, pima<sup>2</sup> [Duvenaud et al., 2011] for classification task. Table 5.3 provides the description of these three data sets. The kernel type used here is SE kernel. The data sets are randomly split into training (90% of data) and test (10% of data) sets. We first run the model in Duvenaud et al. [2011] to obtain the most important order  $d$ . From  $d$ , we proceed learning  $w_{i_1 \dots i_d}$ . One limitation is that this setting is not scalable w.r.t  $D$  as the number of kernels,  $\binom{D}{d}$ , increases exponentially. Table 5.4 shows that our model can improve the accuracy of additive GPs by selecting appropriate kernels. On the other hand, the model without any prior even hurts the prediction. In the previous regression task, our model performs poorly in concrete data set since the 1-order additive kernels is the best fit for this data according to [Duvenaud et al., 2011]. We retrained the model and obtained an improved result with  $6.90_{\pm 0.05}$  in RMSE, pushing the result closer to that of GP-NKN.

<sup>2</sup>taken from <https://github.com/duvenaud/additive-gps>

Table 5.4 Classification error (in %) on three data sets.

	Additive GPs	No prior	Our model	#kernels $\binom{D}{d}$
Heart	18.15 $\pm$ 4.56	16.00 $\pm$ 1.41	<b>14.00</b> $\pm$ 2.11	$\binom{13}{1} = 13$
Liver	30.36 $\pm$ 8.37	40.29 $\pm$ 6.93	<b>27.43</b> $\pm$ 2.52	$\binom{6}{3} = 20$
Pima	23.99 $\pm$ 3.46	29.87 $\pm$ 3.72	<b>20.52</b> $\pm$ 1.65	$\binom{8}{6} = 28$

## 5.6 Related work and conclusion

There is a large body of work [Duvenaud et al., 2013; Kim and Teh, 2018; Lloyd et al., 2014] establishing the foundation of model discovery for Gaussian processes. [Grosse et al., 2012] presents work on unsupervised learning for the case of matrix decomposition. Then, [Duvenaud et al., 2013; Ghahramani, 2015; Lloyd et al., 2014] extend to supervised settings with Gaussian process (GP) models. [Sun et al., 2018] build complex kernels under network architectures having an additive layer where any two kernels are summed and followed by a product layer where kernels are multiplied. [Kim and Teh, 2018] adopt search procedure but smartly avoid the learning model by finding bounds of likelihood. However, the search is still time-consuming and is done heuristically greedy manner. A complete review as well as a guideline for automatic systems can be found in [Steinruecken et al., 2019]. While inheriting the spirits of existing work on kernel compositions, this chapter focuses on scaling up the system in terms of data size and efficient model selection.

A recent work [Teng et al., 2020] shares some similarity to our work. The paper presents a probabilistic approach to select among models with a Softmax-like assumption for choosing a model. The main idea in this chapter is to select a single model out of a manual fixed set of candidate models. Our model generating kernels combinatorially considers a bigger model space than that of [Teng et al., 2020]. On the other hand, models [Teng et al., 2020] use single inducing points for compositional kernels. Its sparse GP approximation can be limited compared to our MultiSVGP. Another work [Malkomes et al., 2016] attempts to extract a model out of candidate ones using surrogate models, e.g., Bayesian optimization. The surrogate models are based on the distance between GP models.

There is existing work proposing probabilistic priors, e.g., spike-and-slab prior, on multi-task GP [Titsias and Lázaro-Gredilla, 2011]. However, the approach does not scale well with the number of data and suffers from computational burden by the choice of probabilistic priors.

## Chapter 6

# Characterizing Deep Gaussian process

This chapter investigates a new type of model, Deep Gaussian processes, that multiple Gaussian process layers are hierarchically stacked. In contrast to the previous chapters where many theoretical properties of Gaussian processes are well-understood, deep Gaussian processes are fresh and required to take a deeper look at how the models behave, especially, with respect to different kinds of kernel functions. To this end, this chapter aims to analyze the characteristic of deep Gaussian process for five types of basic kernel functions such that the degradation can be avoided. The theoretical insights tailor the decision to choose kernel function in deep Gaussian processes. Experiments justify the theoretical results and suggest a regularization approach to train deep Gaussian processes.

### 6.1 Introduction

Deep Gaussian process (DGP) [Damianou and Lawrence, 2013] is a new promising class of models which are constructed by a hierarchical composition of Gaussian processes. The strength of this model lies in its capacity to have richer representation power from the hierarchical construction and its robustness to overfitting from the probabilistic modeling. Therefore, there have been extensive studies [Bui et al., 2016; Cutajar et al., 2017; Dai et al., 2016; Havasi et al., 2018; Hensman and Lawrence, 2014; Lu et al., 2020; Salimbeni and Deisenroth, 2017; Salimbeni et al., 2019; Ustyuzhaninov et al., 2020] contributing to this research area.

There exists a pathology, stating that the increase in the number of layers degrades the learning power of DGP [Duvenaud et al., 2014]. That is, the functions produced by DGP priors

become flat and cannot fit data. It is important to develop theoretical understanding of this behavior, and therefore to have proper tactics in designing model architectures and parameter regularization to prevent the issue. Existing work [Duvenaud et al., 2014] investigates the Jacobian matrix of a given model which can be analytically interpreted as the product of those in each layer. Based on the connection between the manifold of a function and the spectrum of its Jacobian, the authors show the degree of freedom is reduced significantly at deep layers. Another work [Dunlop et al., 2018] studies the ergodicity of the Markov chain to explain the pathology.

To explain such phenomena, we study a quantity which measures the distance of any two layer outputs. We present a new approach that makes use of the statistical properties of the quantity passing from one layer to another. Therefore, our approach accurately captures the relations of the distance quantity between layers. By considering kernel hyperparameters, our method recursively computes the relations of two consecutive layers. Interestingly, the recurrence relations provide a tighter bound than that of Dunlop et al. [2018] and reveal the rate of convergence to fixed points. Under this unified approach, we further extend our analysis to five popular kernels which are not analyzed yet before. For example, the spectral mixture kernels do not suffer the pathology. We further provide a case study in DGP, showing the connection between our recurrence relations and learning DGPs.

**Deep Gaussian process** We study DGPs in composition formulation where GP layers are stacked hierarchically. An  $N$ -layer DGP is defined as

$$\mathbf{f}_N \circ \mathbf{f}_{N-1} \circ \cdots \circ \mathbf{f}_1(\mathbf{x}),$$

where, at layer  $n$ , for dimension  $d$ ,  $f_n^{(d)}|\mathbf{f}_{n-1} \sim \mathcal{GP}(0, k_n(\cdot, \cdot))$  independently. Note that the GP priors have the mean functions set to zero. The nonzero-mean case is discussed later (Section 6.9). We shorthand  $\mathbf{f}_n \circ \mathbf{f}_{n-1} \circ \cdots \circ \mathbf{f}_1(\mathbf{x})$  as  $\mathbf{f}_n(\mathbf{x})$  and write  $k_n(\mathbf{f}_{n-1}(\mathbf{x}), \mathbf{f}_{n-1}(\mathbf{x}'))$  as  $k_n(\mathbf{x}, \mathbf{x}')$ . Let  $m$  be the number of output of  $\mathbf{f}_n$ . All layers have the same hyperparameters.

**Theorem 6.1.1** (Dunlop et al. [2018]). *Assume that  $k(\mathbf{x}, \mathbf{x}')$  is given by the squared exponential kernel function with variance  $\sigma^2$  and lengthscale  $\ell^2$  and that the input  $\mathbf{x}$  is bounded. Then if  $\sigma^2 < \ell^2/m$ ,*

$$\mathbb{P}(\|\mathbf{f}_n(\mathbf{x}) - \mathbf{f}_n(\mathbf{x}')\|_2 \xrightarrow[n \rightarrow \infty]{} 0 \quad \text{for all } \mathbf{x}, \mathbf{x}' \in \mathcal{D}) = 1$$

where  $\mathbb{P}$  denotes the law of process  $\{f_n\}$ .

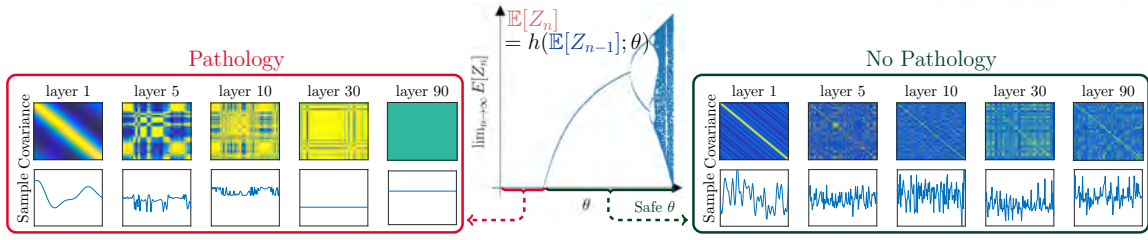


Fig. 6.1 Studying the squared distance,  $Z_n$ , between outputs of two consecutive layers. The asymptotic property (middle plot) of the recurrence relation of this quantity between two consecutive layers decides the existence of pathology for a very deep model. Here,  $\theta$  indicates kernel hyperparameters. The middle plot is the bifurcation plot providing the state of DGP at very deep layer. The pathology is identified by the zero-value region where  $\mathbb{E}[Z_n] \rightarrow 0$ . Note that this bifurcation plot is for illustration purpose only.

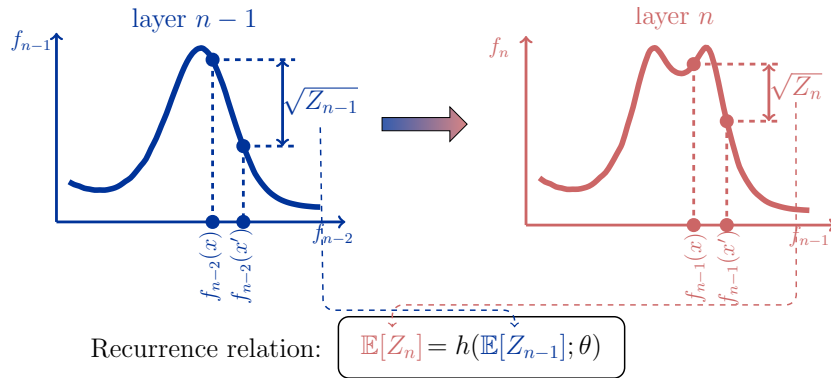


Fig. 6.2 Finding the recurrence relation of the quantity  $\mathbb{E}[(f_n(x) - f_n(x'))^2]$  between two consecutive layers.

This theorem tells us the criterion that the event of vanishing in output magnitude happens infinitely often with probability 1.

## 6.2 Moment-generating function of distance quantity

We are interested in quantifying the expectation of the squared Euclidean distance between any two outputs of a layer and thereby study the dynamics of this quantity from a layer to the next layer. Figure 6.2 shows that we can make use of the found recurrence relations to study the pathology of DGPs.

For any input pair  $\mathbf{x}$  and  $\mathbf{x}'$ , we define such quantity at layer  $n$  as  $Z_n = \|\mathbf{f}_n(\mathbf{x}) - \mathbf{f}_n(\mathbf{x}')\|_2^2 = \sum_{d=1}^m (f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}'))^2$ . When the previous layer  $\mathbf{f}_{n-1}$  is given, the difference between any  $f_n^{(d)}(\mathbf{x})$  and  $f_n^{(d)}(\mathbf{x}')$  is Gaussian,

$$(f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}')) | \mathbf{f}_{n-1} \sim \mathcal{N}(0, s_n).$$

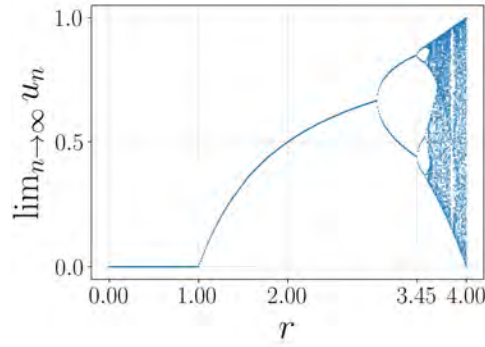


Fig. 6.3 Bifurcation plot of the logistic function  $u_n = ru_{n-1}(1 - u_{n-1})$ .

Here  $s_n = k_n(\mathbf{x}, \mathbf{x}) + k_n(\mathbf{x}', \mathbf{x}') - 2k_n(\mathbf{x}, \mathbf{x}')$  which is obtained from subtracting two dependent Gaussians. We can normalize the difference between  $f_n^{(d)}(\mathbf{x})$  and  $f_n^{(d)}(\mathbf{x}')$  by a factor  $\sqrt{s_n}$  to obtain the form of standard normal distribution as

$$\frac{(f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}'))}{\sqrt{s_n}} | \mathbf{f}_{n-1} \sim \mathcal{N}(0, 1).$$

Since all dimensions  $d$  in a layer are independent, we can say that  $\frac{Z_n}{s_n} | \mathbf{f}_{n-1} \sim \chi_m^2$ , is distributed according to the Chi-squared distribution with  $m$  degrees of freedom.

One useful property of the Chi-squared distribution is that the moment-generating function of  $\frac{Z_n}{s_n} | \mathbf{f}_{n-1}$  can be written in an analytical form, with  $t \leq 1/2$ ,

$$M_{\frac{Z_n}{s_n} | \mathbf{f}_{n-1}}(t) = \mathbb{E} \left[ \exp \left( t \frac{Z_n}{s_n} \right) | \mathbf{f}_{n-1} \right] = (1 - 2t)^{-m/2}. \quad (6.2.1)$$

We shall see that the expectation of the distance quantity  $Z_n$  is computed via a kernel function which, in most cases, involves exponentiations. Given that the input of this kernel is governed by a distribution, i.e.,  $\chi^2$ , the moment-generating function becomes convenient to obtain our desired expectations.

Figure 6.2 depicts our approach to extract a function  $h(\cdot)$  which models the recurrence relation between  $\mathbb{E}[Z_n]$  and  $\mathbb{E}[Z_{n-1}]$ . This is also the main theme of this chapter.

### 6.3 Analyzing dynamic systems with chaos theory

Recurrence maps representing dynamic transitions between DGP layers are nonlinear. Studying the dynamic states and convergence properties for nonlinear recurrences is not as well-established as those of linear recurrences. As an example, given a simple nonlinear model like the logistic map:  $u_n = ru_{n-1}(1 - u_{n-1})$ , its dynamic behaviors can be complicated [May, 1976].

Recurrent plots or bifurcation plots have been used to analyze the behavior of chaotic systems. The plots are produced by simulating and recording the dynamic states up to very large time points. This tool allows us to monitor the qualitative changes in a system, illustrating fixed points asymptotically, or possible visited values. Other techniques, e.g. transient chaos [Poole et al., 2016], recurrence relations [Schoenholz et al., 2017] have been used to study deep neural networks.

We take the logistic map as an example to understand a recurrence relation. Figure 6.3 is the bifurcation plot of the logistic map. This logistic map is used to describe the characteristics of a system which models a population function. We can see that the plot reveals the state of the system, showing whether the population becomes extinct ( $0 < r < 1$ ), stable ( $1 < r < 3$ ), or fluctuating ( $r > 3.4$ ) by seeing the parameter  $r$ .

## 6.4 Squared exponential kernel function

The squared exponential kernel (SE) is defined in the form of

$$\text{SE}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2 / 2\ell^2\right). \quad (6.4.1)$$

**Theorem 6.4.1** (DGP with SE). *Given a triplet  $(m, \sigma^2, \ell^2)$ ,  $m \geq 1$  such that the following sequence converges to 0:*

$$u_n = 2m\sigma^2 \left(1 - (1 + u_{n-1}/m\ell^2)^{-m/2}\right), \quad (6.4.2)$$

*Then,  $\mathbb{P}(\|\mathbf{f}_n(\mathbf{x}) - \mathbf{f}_n(\mathbf{x}')\|_2 \xrightarrow[n \rightarrow \infty]{} 0 \text{ for all } \mathbf{x}, \mathbf{x}' \in \mathcal{D}) = 1$ .*

*Proof.* Note that we do not directly have access to  $\mathbb{E}[Z_n]$  but  $\mathbb{E}[Z_n|\mathbf{f}_{n-1}]$  because of the Markov structure of the DGP construction. Getting  $\mathbb{E}[Z_n]$  is done via  $\mathbb{E}[Z_n|\mathbf{f}_{n-1}]$  where we use the law of total expectation  $\mathbb{E}[Z_n] = \mathbb{E}_{\mathbf{f}_{n-1}}[\mathbb{E}[Z_n|\mathbf{f}_{n-1}]]$ .

Now, we study the term  $\mathbb{E}[Z_n|\mathbf{f}_{n-1}]$ :

$$\begin{aligned} \mathbb{E}[Z_n|\mathbf{f}_{n-1}] &= \mathbb{E}\left[\sum_{d=1}^m (f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}'))^2 | \mathbf{f}_{n-1}\right] \\ &= 2m\sigma^2 - 2mk_n(\mathbf{x}, \mathbf{x}'). \end{aligned} \quad (6.4.3)$$

The second equality is followed by  $\mathbb{E}[(f_n^{(d)}(\mathbf{x}))^2] = \mathbb{E}[(f_n^{(d)}(\mathbf{x}'))^2] = \sigma^2$  and  $\mathbb{E}[f_n^{(d)}(\mathbf{x})f_n^{(d)}(\mathbf{x}')] = k_n(\mathbf{x}, \mathbf{x}')$ . Recall that we write  $k_n(\mathbf{x}, \mathbf{x}') = k_n(\mathbf{f}_{n-1}(\mathbf{x}), \mathbf{f}_{n-1}(\mathbf{x}'))$ . By the definition of SE kernel,



we have

$$\mathbb{E}[Z_n | \mathbf{f}_{n-1}] = 2m\sigma^2 \left( 1 - \exp\left(-\frac{Z_{n-1}}{2\ell^2}\right) \right).$$

Applying the law of total expectation, we have

$$\mathbb{E}[Z_n] = 2m\sigma^2 \left( 1 - \mathbb{E} \left[ \exp\left(-\frac{Z_{n-1}}{2\ell^2}\right) \right] \right).$$

Again, we can only compute  $\mathbb{E}[\exp(-\frac{Z_{n-1}}{2\ell^2})] = \mathbb{E}_{\mathbf{f}_{n-2}}[\mathbb{E}[\exp(-\frac{Z_{n-1}}{2\ell^2}) | \mathbf{f}_{n-2}]]$ . The expectation will be computed by the formula of the moment-generating function with respect to  $\frac{Z_{n-1}}{s_{n-1}} | \mathbf{f}_{n-2}$  where  $t = -\frac{s_{n-1}}{2\ell^2}$  in Equation (6.2.1). Choosing this value also satisfies the condition  $t \leq 1/2$ . Now, we have

$$\begin{aligned} \mathbb{E}[Z_n] &= 2m\sigma^2 \left( 1 - \mathbb{E} \left[ \left( 1 + s_{n-1}/\ell^2 \right)^{-m/2} \right] \right) \\ &\leq 2m\sigma^2 \left( 1 - \left( 1 + \mathbb{E}[s_{n-1}]/\ell^2 \right)^{-m/2} \right). \end{aligned} \quad (6.4.4)$$

Here, Jensen's inequality is used as  $(1+x)^{-a}$  is convex for any  $x > 0$ . By Equation (6.4.3), we have

$$\frac{\mathbb{E}[Z_{n-1} | \mathbf{f}_{n-2}]}{m} = 2\sigma^2 - 2k_{n-1}(\mathbf{x}, \mathbf{x}') = s_{n-1}.$$

Replacing  $s_{n-1}$  in Equation (6.4.4) and applying the law of total expectation for the case of  $Z_{n-1}$ , we obtain recurrence relation between layer  $n-1$  and layer  $n$  is

$$\mathbb{E}[Z_n] \leq 2m\sigma^2 \left( 1 - \left( 1 + \mathbb{E}[Z_{n-1}]/m\ell^2 \right)^{-m/2} \right).$$

Using the Markov inequality, for any  $\epsilon$ , we can bound  $\mathbb{P}(Z_n \geq \epsilon) \leq \frac{\mathbb{E}[Z_n]}{\epsilon^2}$ .

At this point,  $u_n$  defined in Equation (6.4.2) is considered as the upper bound of  $\mathbb{E}[Z_n]$ . We condition that  $\{u_n\}$  converges to 0, then  $\{\mathbb{E}[Z_n]\}$  converges to 0 as well. By the first Borel-Cantelli lemma, we have  $\mathbb{P}(\limsup_{n \rightarrow \infty} Z_n \geq \epsilon) = 0$ , which leads to the conclusion in the same manners as Dunlop et al. [2018].  $\square$

**Analyzing the recurrence** Figure 6.4a illustrates the bifurcation plot of Equation (6.4.2) with  $m = 1$ . The non-zero contour region in Figure 6.4b tells us that  $\sigma^2/\ell^2$  should be smaller than 1 to escape the pathology. When  $m > 1$ , Figure 6.4c shows that if  $m > \sigma^2/\ell^2$ ,  $u_n$  does not approach to 0, implying the condition to prevent the pathology. This result is consistent with Theorem 6.1.1 in Dunlop et al. [2018].

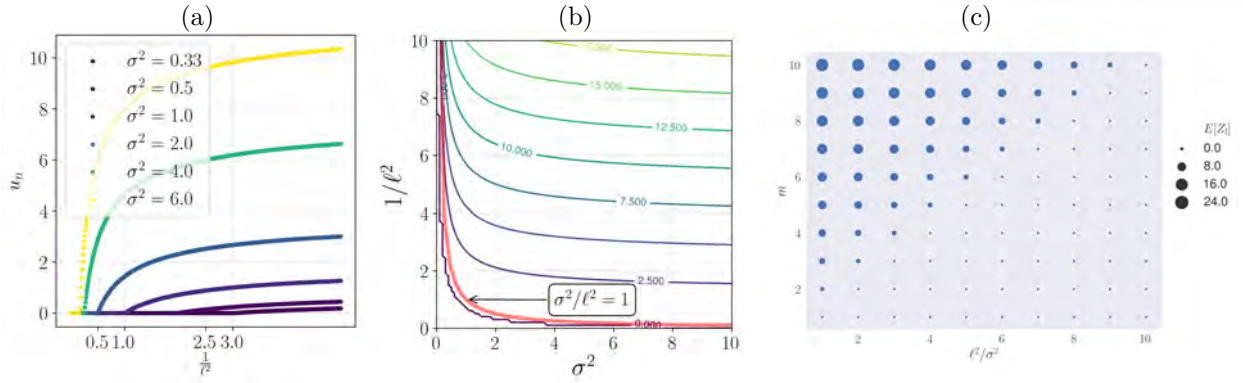


Fig. 6.4 (a): Bifurcation plot of the recurrence relation of SE kernel for  $m = 1$ . (b): Contour plot of  $u_n$  at layer  $n = 300$  and  $m = 1$ . The misalignment between the red line ( $\sigma^2/\ell^2 = 1$ ) and the zero-level contour is due to numerical errors. (c): Increase  $m > \sigma^2/\ell^2$  to avoid pathology.

**Discussion** Note that the relation between  $\mathbb{E}[Z_n]$  and  $\mathbb{E}[Z_{n-1}]$  presents a tighter bound than existing work [Dunlop et al., 2018]. If we construct the recurrence relation based on [Dunlop et al., 2018],  $\mathbb{E}[Z_n]$  is bounded by

$$\mathbb{E}[Z_n] \leq \frac{m\sigma^2}{\ell^2} \mathbb{E}[Z_{n-1}]. \quad (6.4.5)$$

One can show that  $(1+x)^a \geq 1-ax$ ,  $a < 0$ ,  $x > 0$ , implying

$$2m\sigma^2(1 - (1 + \mathbb{E}[Z_{n-1}]/(m\ell^2))^{-m/2}) \leq m\sigma^2\mathbb{E}[Z_{n-1}]/\ell^2.$$

In fact, a numerical experiment shows that our bound of  $\mathbb{E}[Z_n]$  is found to be close to the true  $\mathbb{E}[Z_n]$ . That is, we can see the trajectory of  $\mathbb{E}[Z_n]$  for every layer of a given model of which the depth is not necessary to be infinitely many.

One can reinterpret the recurrence relation for each dimension  $d$  as

$$\mathbb{E}[Z_n^{(d)}] \leq 2\sigma^2 \left( 1 - \left( 1 + \mathbb{E}[Z_{n-1}^{(d)}]/\ell^2 \right)^{-m/2} \right),$$

where  $\mathbb{E}[Z_n^{(d)}] = \frac{\mathbb{E}[Z_n]}{m}$  with  $Z_n^{(d)} = (f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}'))^2$ .

**A guideline to obtain a recurrence relation** Given a specific kernel function, one may follow these steps to acquire the corresponding recurrence relation: (1) considering the form of kernel input where it may be distributed according to either the Chi-squared distribution or its variants (presented in the next sections); (2) checking whether there is a way to represent the kernel function under representations such that statistical properties of kernel inputs are known;

(3) caring about the convexity of the function after choosing a proper setting (as we bound the expectation with Jensen's inequality in the proof of Theorem 6.4.1).

## 6.5 Cosine kernel function

The cosine kernel (COS) function takes inputs as the distance between two points instead of the squared distance like in the case of SE kernel. We will mainly work with  $\sqrt{Z_n}$  in this subsection.

The cosine kernel function  $k(\mathbf{x}, \mathbf{x}') = \text{COS}(\mathbf{x}, \mathbf{x}')$  which is defined as

$$\text{COS}(\mathbf{x}, \mathbf{x}') = \sigma^2 \cos(\pi \|\mathbf{x} - \mathbf{x}'\|_2/p).$$

Starting with Equation (6.4.3) and using the definition of COS kernel, we have

$$\begin{aligned} \mathbb{E}[Z_n | \mathbf{f}_{n-1}] &= 2m\sigma^2 - 2m\sigma^2 \cos(\pi \sqrt{Z_{n-1}}/p) \\ &= 2m\sigma^2 - m\sigma^2 \exp(i\pi \sqrt{Z_{n-1}}/p) \\ &\quad - m\sigma^2 \exp(-i\pi \sqrt{Z_{n-1}}/p). \end{aligned}$$

Here, Euler's formula is used to represent  $\cos(\cdot)$  and  $i$  is the imaginary unit ( $i^2 = -1$ ). To obtain  $\mathbb{E}[Z_n]$ , we use the law of total expectation and compute the two following expectations:  $\mathbb{E}[\exp(i\pi \sqrt{Z_{n-1}}/p) | \mathbf{f}_{n-2}]$  and  $\mathbb{E}[\exp(-i\pi \sqrt{Z_{n-1}}/p) | \mathbf{f}_{n-2}]$ . From  $\frac{Z_n}{s_n} | \mathbf{f}_{n-1} \sim \chi_m^2$ , we have  $\sqrt{\frac{Z_n}{s_n}} | \mathbf{f}_{n-1} \sim \chi_m$ , is distributed according to the Chi distribution. This observation follows the first step in the guideline. The characteristic function of the Chi distribution for random variable  $\sqrt{\frac{Z_n}{s_n}} | \mathbf{f}_{n-1}$  is

$$\begin{aligned} \varphi_{\sqrt{Z_n/s_n} | \mathbf{f}_{n-1}}(t) &= \mathbb{E} \left[ \exp \left( it \sqrt{Z_n/s_n} \right) \right] \\ &= {}_1F_1 \left( \frac{m}{2}, \frac{1}{2}, \frac{-t^2}{2} \right) + it\sqrt{2} \frac{\Gamma((m+1)/2)}{\Gamma(m/2)} {}_1F_1 \left( \frac{m+1}{2}, \frac{3}{2}, \frac{-t^2}{2} \right). \end{aligned}$$

where  ${}_1F_1(a, b, z)$  is Kummer's confluent hypergeometric function. A generalized hypergeometric function is in the form of

$${}_pF_q(a_1, a_2, \dots, a_p; b_1, b_2, \dots, b_q; z) = \sum_{n=0}^{\infty} \frac{a_1^{\bar{n}} a_2^{\bar{n}} \dots a_p^{\bar{n}} z^n}{b_1^{\bar{n}} b_2^{\bar{n}} \dots b_q^{\bar{n}} n!}.$$

Here,  $x^{\bar{n}}$  is the *rising factorial* defined as

$$x^{\bar{n}} = \prod_{k=0}^{n-1} (x + k).$$

This is considered as the second step in the guideline. Back to our process of finding the recurrence function, we consider the case  $\sqrt{\frac{Z_{n-1}}{s_{n-1}}} \mathbf{f}_{n-2} \sim \chi_m$ . By choosing  $t = \pm \frac{\pi\sqrt{s_{n-1}}}{p}$  for its characteristic function, we can obtain

$$\mathbb{E}[Z_n] = 2m\sigma^2 \left( 1 - {}_1F_1\left(\frac{m}{2}, \frac{1}{2}, -\frac{\pi^2}{2p^2} \mathbb{E}[Z_{n-1} | \mathbf{f}_{n-2}]\right) \right).$$

This is because the imaginary parts of  $\varphi(t = \frac{\pi\sqrt{s_{n-1}}}{p})$  and  $\varphi(-\frac{\pi\sqrt{s_{n-1}}}{p})$  are canceled out.

As the third step in the guideline, we perform a sanity check about the convexity of  ${}_1F_1$ . Only with  $m = 1$ ,  ${}_1F_1(\frac{1}{2}, \frac{1}{2}, -\frac{t^2}{2}) = \exp(-\frac{t^2}{2})$  is convex. Our result in this case is restricted to  $m = 1$ . Now, we can state that the recurrence relation is

$$u_n = 2\sigma^2 \left( 1 - \exp(-\pi^2 u_{n-1} / 2p^2) \right). \quad (6.5.1)$$

## 6.6 Periodic kernel function

The periodic kernel (PER) resembles to COS kernel, and is written in the form of

$$\text{PER}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{2 \sin^2(\pi \|\mathbf{x} - \mathbf{x}'\|_2 / p)}{\ell^2}\right).$$

In this case, we do not have an exact recurrence under equality. Instead, we find the lower bound of  $\mathbb{E}[Z_l]$ . It is done by using  $e^x \geq 1 + x$ :

$$\exp\left(-\frac{2 \sin^2(\pi r / p)}{\ell^2}\right) \geq g(\cos(\frac{2\pi r}{p})).$$

where  $r = \|\mathbf{x} - \mathbf{x}'\|_2$ , and  $g(\cos(2\pi r / p)) = 1 - \ell^{-2} + \ell^{-2} \cos(2\pi r / p)$ . We can see that the PER kernel now is bounded in terms of COS kernels and use the readily obtained result of COS kernel to get the recurrence.

The function  $g$  is obtained based on  $\exp(x) \geq 1 + x$ :

$$g(\cos(\frac{2\pi r}{p})) = 1 - \frac{1}{\ell^2} + \frac{1}{\ell^2} \cos(\frac{2\pi r}{p}).$$

The bound of  $\mathbb{E}[Z_n]$  is recursively computed from  $\mathbb{E}[Z_{n-1}]$ :

$$\mathbb{E}[Z_n] \leq h(\mathbb{E}[Z_{n-1}]),$$

where

$$h(\mathbb{E}[Z_{n-1}]) = \frac{2m\sigma^2}{\ell^2} \left( 1 - {}_1F_1\left(\frac{m}{2}, \frac{1}{2}, -\frac{2\pi^2}{p^2}\mathbb{E}[Z_{n-1}]\right) \right).$$

## 6.7 Rational quadratic kernel function

Now, we study the rational quadratic (RQ) kernel. This kernel is obtained from the SE kernel by marginalizing the inverse lengthscale of SE kernel [Rasmussen and Williams \[2005\]](#):

$$\text{RQ}(\mathbf{x}, \mathbf{x}') = \left( 1 + \|\mathbf{x} - \mathbf{x}'\|^2 / (2\alpha\ell^2) \right)^{-\alpha}.$$

We use the power series expansion  $(1+x)^{-\alpha} = \sum_{k=0}^{\infty} \binom{-\alpha}{k} x^k$  for this kernel

$$\mathbb{E}[Z_n] = 2\sigma^2 - 2\sigma^2 \sum_{k=0}^{\infty} \binom{-\alpha}{k} \frac{\mathbb{E}[Z_{n-1}^k | \mathbf{f}_{l-2}]}{(2\alpha\ell^2)^k}.$$

Next, we use the high-order moment of Chi-squared distribution. As  $\frac{Z_{n-1}}{s_{n-1}} | \mathbf{f}_{n-2} \sim \chi_m^2$ , it is known that  $\mathbb{E} \left[ \frac{Z_{n-1}^k}{s_{n-1}^k} | \mathbf{f}_{n-2} \right] = 2^k \frac{\Gamma(k + \frac{m}{2})}{\Gamma(\frac{m}{2})}$ . Then, we can obtain the corresponding recurrence relation as

$$\begin{aligned} & \mathbb{E} \left[ \left( 1 + \frac{Z_{n-1}}{2\alpha\ell^2} \right)^{-\alpha} | \mathbf{f}_{n-2} \right] \\ &= \sum_{k=0}^{\infty} \binom{-\alpha}{k} \frac{\mathbb{E}[Z_{n-1}^k | \mathbf{f}_{n-2}]}{(2\alpha\ell^2)^k} \\ &= \sum_{k=0}^{\infty} \frac{(-1)^k \alpha^{\bar{k}} 2^k \frac{\Gamma(k + \frac{m}{2})}{\Gamma(\frac{m}{2})} \mathbb{E}[Z_{n-1} | \mathbf{f}_{n-2}]^k}{k! (2\alpha\ell^2)^k} && \text{(use high-order moment of Chi-squared)} \\ &= \sum_{k=0}^{\infty} \alpha^{\bar{k}} \left( \frac{m}{2} \right)^{\bar{k}} \frac{(-1)^k \mathbb{E}[Z_{n-1} | \mathbf{f}_{n-2}]^k}{(\alpha\ell^2)^k k!} && \text{(by a property of rising factorial)} \\ &= {}_2F_0\left(\alpha; \frac{m}{2}; \frac{-\mathbb{E}[Z_{n-1} | \mathbf{f}_{n-2}]}{\alpha\ell^2}\right). && \text{(by definition of hypergeometric function)} \end{aligned}$$

Consequently, we obtain the recurrence between layers as

$$u_n = 2m \left( 1 - {}_2F_0\left(\alpha; \frac{m}{2}; \frac{-u_{n-1}}{\alpha\ell^2}\right) \right),$$

where  ${}_2F_0(\cdot; \cdot; \cdot)$  is one of the hypergeometric functions. This  ${}_2F_0(\cdot; \cdot; \cdot)$  function has a close connection to the exponential integral function  $Ei(x) = \int_{-x}^{+\infty} \frac{e^{-t}}{t} dt$ . This is related to the way of constructing the RQ kernel from SE kernel.

## 6.8 Spectral mixture kernel

We consider the case the spectral mixture kernel has one-dimensional inputs and one mixture.

We rewrite the kernel function as:

$$\begin{aligned} \exp(-2\pi^2\sigma^2r^2) \cos(2\pi\mu r) &= \frac{1}{2} \{ \exp(-2\pi^2\sigma^2r^2 + 2\pi\mu ir) + \exp(-2\pi^2\sigma^2r^2 - 2\pi\mu ir) \} \\ &= \frac{1}{2} \exp\left(-\frac{\mu^2}{2\sigma^2}\right) \left\{ \exp\left(-2\pi^2\sigma^2\left(r - \frac{i\mu}{2\pi\sigma^2}\right)^2\right) + \exp\left(-2\pi^2\sigma^2\left(r + \frac{i\mu}{2\pi\sigma^2}\right)^2\right) \right\}. \end{aligned}$$

This leads to our change in variables in the main text where we denote

$$w^2 = \exp\left(-\frac{\mu^2}{2\sigma^2}\right), \quad v^2 = 2\pi^2\sigma^2, \quad u = \frac{i\mu}{2\pi\sigma^2}.$$

Because  $\frac{(\sqrt{Z_{n-1}} \pm i\mu/(2\pi\sigma^2))^2}{s_{n-1}}$  is distributed according to a non-central Chi-square distribution with degree of freedom 1 and the noncentrality parameter  $\lambda = -\mu^2/(4\pi^2\sigma^4s_{n-1})$ . The moment-generating function is

$$M_{\chi_1^2}(t) = \mathbb{E} \left[ \exp \left( t \frac{(\sqrt{Z_{n-1}} \pm i\mu/(2\pi\sigma^2))^2}{s_{n-1}} \right) \right] = (1 - 2t)^{-1/2} \exp\left(\frac{\lambda t}{1 - 2t}\right).$$

Choosing  $t = -2\pi^2\sigma^2s_{n-1}$ , we have

$$\mathbb{E} \left[ \exp \left( -2\pi^2\sigma^2 \left( \sqrt{Z_{n-1}} \pm \frac{i\mu}{2\pi\sigma^2} \right)^2 \right) \right] = (1 + 4\pi^2\sigma^2s_{n-1})^{-1/2} \exp \left( \frac{\frac{\mu^2}{2\sigma^2}}{1 + 4\pi^2\sigma^2s_{n-1}} \right)$$

We can obtain the recurrence relation as

$$\begin{aligned} u_n &= 2 \left\{ 1 - \exp\left(-\frac{\mu^2}{2\sigma^2}\right) \exp \left( \frac{\frac{\mu^2}{2\sigma^2}}{1 + 4\pi^2\sigma^2u_{n-1}} \right) (1 + 4\pi^2\sigma^2u_{n-1})^{-1/2} \right\} \\ &= 2 \left\{ 1 - \exp \left( -\frac{2\pi^2\mu^2u_{n-1}}{1 + 4\pi^2\sigma^2u_{n-1}} \right) (1 + 4\pi^2\sigma^2u_{n-1})^{-1/2} \right\}. \end{aligned}$$

In the case of high-dimensional DGPs, the SM kernel takes  $m$ -dimensional inputs. Because all dimensions are independent, we can obtain the expectation rely on the probabilistic independence between input dimensions to obtain the expectation as the product of the expectation in each

dimension. Hence, we can have the recurrence in the form:

$$u_n = 2m \left\{ 1 - \exp \left( -\frac{2m\pi^2\mu^2 u_{n-1}}{1 + 4\pi^2\sigma^2 u_{n-1}} \right) (1 + 4\pi^2\sigma^2 u_{n-1})^{-m/2} \right\}.$$

Note that we assume all dimensions share the same parameter  $\sigma^2$  and  $\mu$ .

## 6.9 Extension to non-pathological cases

We use our approach to analyze two cases including *nonzero-mean* DGPs and *input-connected* DGPs where there is no pathology occurring.

**Nonzero-mean DGPs** Let  $f_n^{(d)}(\mathbf{x}) \sim \mathcal{GP}(\mu_n(\mathbf{x}), k_n(\mathbf{x}, \mathbf{x}'))$  with the mean function  $\mu_n(\mathbf{x})$ , the difference between two outputs,  $(f_n^{(d)}(\mathbf{x}) - f_n^{(d)}(\mathbf{x}')) \sim \mathcal{N}(\nu_n, s_n)$  with  $\nu_n = \mu_n(\mathbf{x}) - \mu_n(\mathbf{x}')$ . This leads to  $\frac{Z_n}{s_n} | \mathbf{f}_n \sim \chi_m^2$ , the *non-central* Chi-squared distribution with the non-central parameter  $\lambda = m\nu_n^2$ .

Since we already provide an analysis involving the non-central Chi-squared distribution with spectral mixture kernels, no pathology of nonzero-mean DGPs can be shown by our analysis (Section 4.3). That is, there is no pathology as  $\lambda > 0$ . When  $\lambda = 0$ , this case falls back to zero-mean or constant-mean. Mean functions greatly impact the recurrence relation because  $\lambda$  is inside an exponential function.

To the best of our knowledge, this is the first analytical explanation for the nonexistence of pathology in nonzero-mean DGPs. In practice, there is existing work choosing mean functions [Salimbeni and Deisenroth, 2017]. [Dunlop et al., 2018] briefly makes a connection between nonzero-mean DGPs and stochastic differential equations. However, there is no clear answer given for this case, yet.

**Input-connected DGPs** Previously, [Duvenaud et al., 2014; Neal, 1995] suggest to make each layer connect to input. The corresponding dynamic system is

$$u_n = 2m\sigma^2(1 - (1 + u_{n-1}/m\ell^2)^{-m/2}) + c,$$

with  $c$  is computed from the kernel function taking input data  $\mathbf{x}$ . By seeing its bifurcation plot in Figure 6.5, we can reconfirm the solution from [Duvenaud et al., 2014; Neal, 1995]. That is,

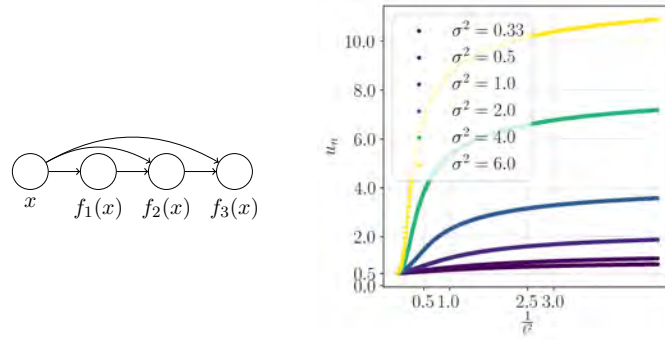


Fig. 6.5 *Left*: Graphical model of input-connected construction suggested by [Duvenaud et al., 2014; Neal, 1995]. *Right*: The bifurcation plot of input-connected DGP.

$u_n$  converges to the value which is greater than zero, and avoids the pathology. However, the convergence rate of  $\mathbb{E}[Z_n]$  stays the same.

## 6.10 Analysis of recurrence relations

This section explains the condition of hyperparameters that causes the pathology for each kernel function. Then we discuss the rate of convergence for the recurrence functions.

### 6.10.1 Identify the pathology

Figure 6.6 shows contour plots based on our obtained recurrence relations. This will help us identify the pathology for each case.

**Cos kernel** Similar to SE, the condition to escape the pathology is  $\pi^2 \sigma^2 / p^2 > 1$ . Figure 6.7 provides the bifurcation plot and contour plots for the case  $m > 1$ .

**Per kernel** If we increase  $\ell$ , then we should decrease the periodic length  $p$  to prevent the pathology (see Figure 6.8).

**RQ kernel** The behavior of this kernel resembles that of SE. We also observe that the change in the hyperparameter  $\alpha$  does not affect the condition to avoid the pathology (Figure 6.9).

**SM kernel** Interestingly, this kernel does *not* suffer the pathology. If  $(\sigma^2, \mu)$  goes to  $(0, 0)$ ,  $\mathbb{E}[Z_n]$  approaches to 0. However,  $\mathbb{E}[Z_n]$  is never equal to 0 since both  $\sigma^2$  and  $\mu$  are positive.



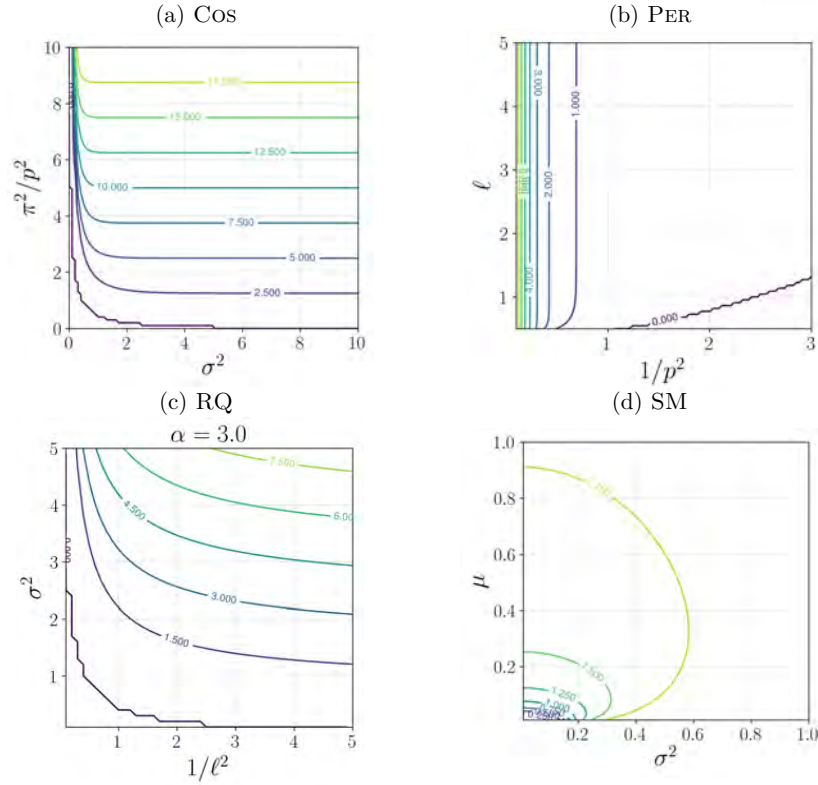


Fig. 6.6 Contour plots of  $\mathbb{E}[Z_n]$  at  $n = 300$  with respect to four kernel functions.

### 6.10.2 Rate of convergence

Recall that  $h(\cdot)$  is the function modeling the recurrence relation between  $\mathbb{E}[Z_n]$  and  $\mathbb{E}[Z_{n-1}]$ . According to Banach fixed-point theorem [Khamsi, 2001], the rate of convergence is decided by the Lipchitz constant of  $h(\cdot)$ ,  $L = \sup h'(\cdot)$ . The more curved the functions are, the faster the convergence rates are (see Figure 6.10a and 6.10b). Figure 6.10c compares the recurrence relation under the function  $h(x)$ . Specifically, for SE, the rate of convergence to a fixed point depends on the dimension parameter  $m$ . In general, SM has the fastest convergence rate among all. On the other hand, the class of RQ kernels has the slowest rate.

Understanding the convergence rate to a fixed point of recurrence relations can be helpful. For example, if a dynamic system corresponding to a DGP model quickly reaches its fixed point, it may be not necessary to have a very deep model. This can give an intuition for designing architectures in DGP given a kernel.

## 6.11 Experimental results

This section verifies our theoretical claims empirically. Firstly, we investigate the correctness of recurrence relations. Then, we check the condition avoiding pathology. Furthermore, we provide

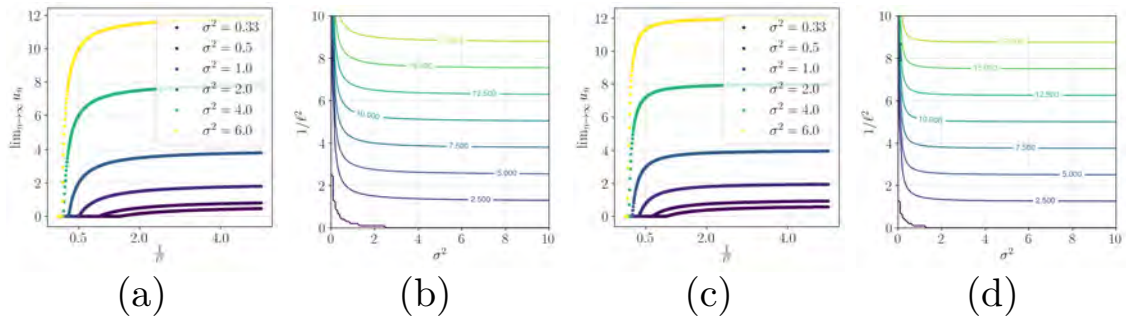


Fig. 6.7 Bifurcation and contour plot of SE kernel for two cases  $m = 2, 3$ . (a)-(b):  $m = 2$ . (c)-(d):  $m = 3$ .

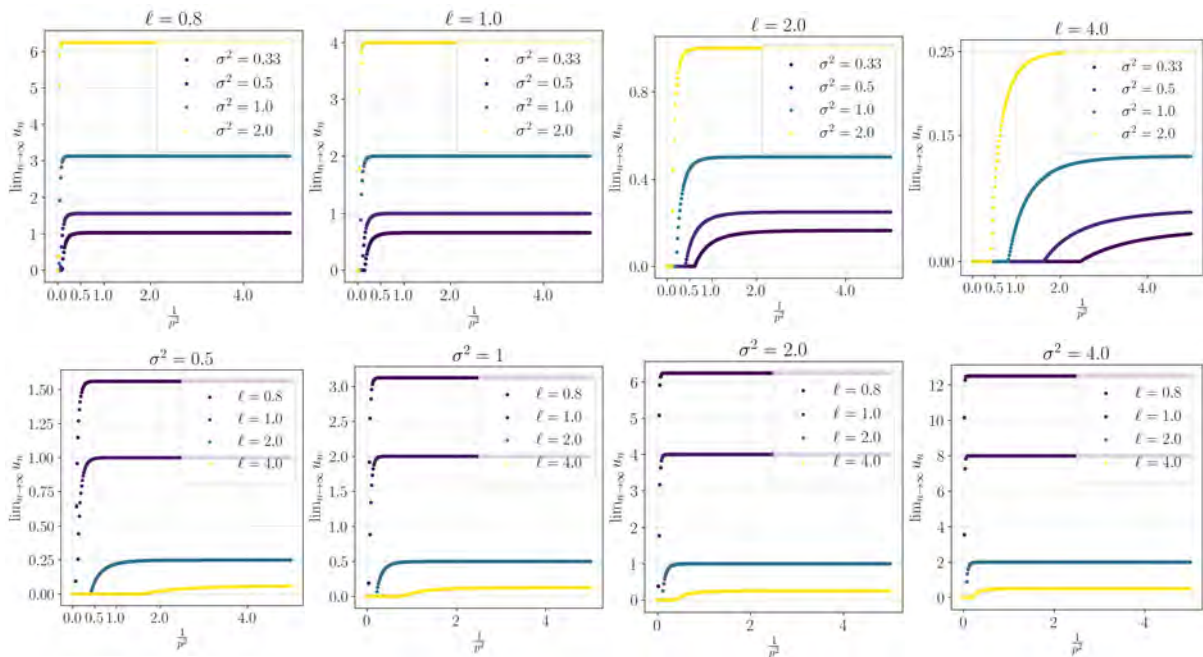


Fig. 6.8 Bifurcation plot of the recurrence of periodic kernel for  $m = 1$ . First row: From left to right,  $\ell$  is varied. Second row:  $\sigma^2$  is varied.

case studies in real-world data sets. All kernels and models are developed based on GPyTorch library [Gardner et al., 2018].

### 6.11.1 Correctness of recurrence relations

We set up a DGP model with 10 layers with SE kernel. The inputs are  $x_0 = 0$  and  $x_1 = 1$ . We will track the value  $Z_n = \|\mathbf{f}_n(x_0) - \mathbf{f}_n(x_1)\|_2^2$  for  $n = 1 \dots 10$ . Given a kernel  $k(x, x')$ , we can exactly compute the expectations  $\mathbb{E}[Z_n]$ . From the model, we collect 2000 samples for each layer  $n$  to obtain the empirical expectation of  $\mathbb{E}[Z_n]$ . Then, we would like to compare the true and empirical estimates. Figure 6.11 plots the comparisons for SE kernel and SM kernel. This numerical experiment supports the claim that our estimation  $\mathbb{E}[Z_n]$  is tight and even close

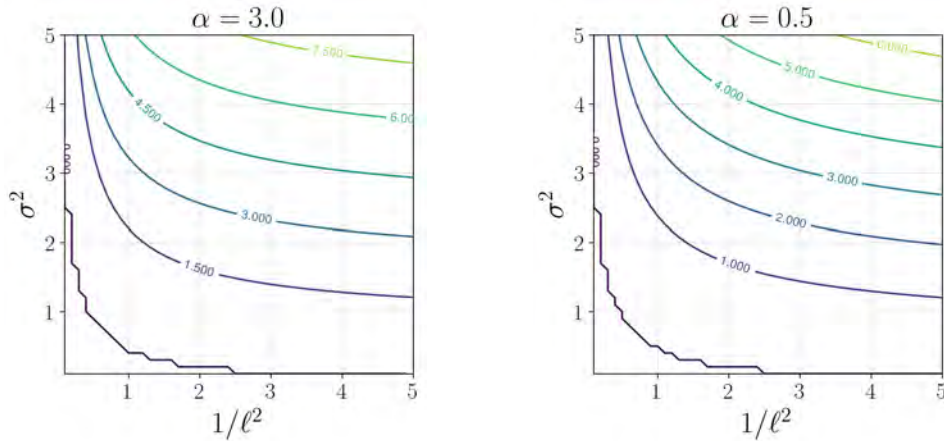


Fig. 6.9 RQ: Contour plots of  $\mathbb{E}[Z_n]$  at  $n = 300$ . The two contour plots share the same zero-value level. So that  $\alpha$  does not decide the condition overcome the pathology.

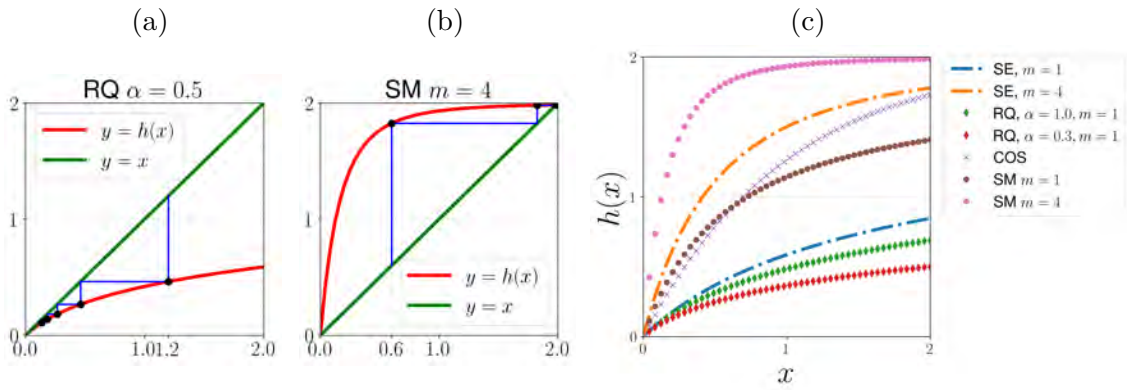


Fig. 6.10 (a-b) Paths to fixed points for two cases: RQ and SM. Iterations of RQ start from  $x = 1.2$  and converge to 0. Those of SM start from  $x = 0.6$  and converge to a point near 1. (c) Plot of all recurrence functions  $h(x)$ . Note that  $x$  is not input data but plays the role of  $\mathbb{E}[Z_n]$ .

to the true estimation. On the other hand,  $\mathbb{E}[Z_n]$  computed based on Dunlop et al. [2018] in Equation (6.4.5) grows exponentially, and cannot fit in Figure 6.11. The additional plots with different settings of hyperparameter and  $m$  can be found in Figure 6.13 and 6.14

### 6.11.2 Justifying the conditions of pathology

From  $N_{\text{data}}$  inputs, we generate the outputs of DGPs and measure the root mean squared distance (RMSD) among the outputs  $\text{RMSD}(n) = \sqrt{\frac{1}{N_{\text{data}}(N_{\text{data}}-1)} \sum_{i \neq j} \|\mathbf{f}_n(\mathbf{x}_i) - \mathbf{f}_n(\mathbf{x}_j)\|_2^2}$ . We record this quantity as we increase  $n$ . We replicate the procedure 30 times to aggregate the statistics of  $\text{RMSD}(n)$ . Here, we only consider the case  $m = 1$ .

**SE kernel** We set up models in one dimension with inputs of each model in range  $(-5, 5)$  with  $N_{\text{data}} = 100$ . The kernel hyperparameter  $\sigma^2$  is set to 1 while  $1/\ell^2$  runs from 0.1 to 5.

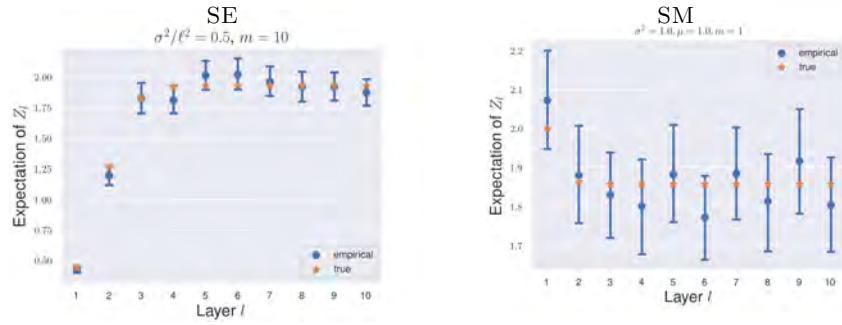


Fig. 6.11  $\mathbb{E}[Z_n]$  computed from recurrence vs. empirical estimation of  $\mathbb{E}[Z_n]$  for two kernel functions.

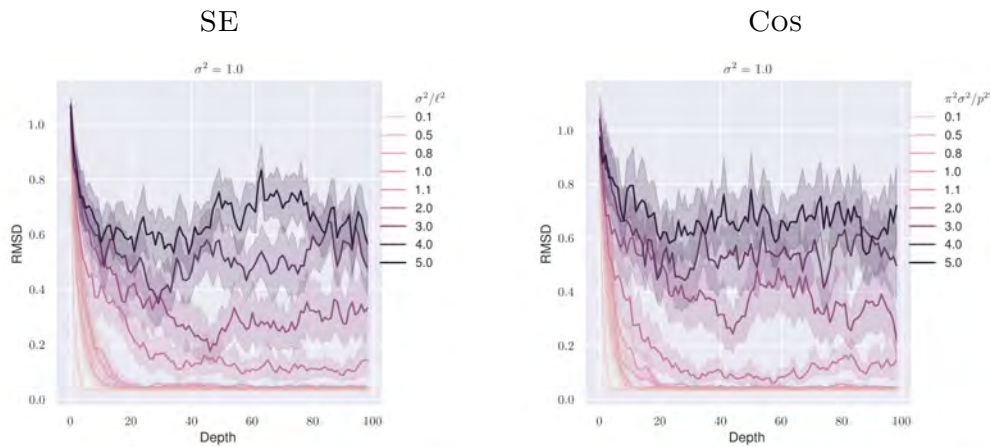


Fig. 6.12 Trace of RMSDs. RMSDs converge to 0 when the pathology occurs.

Figure 6.12a shows the trace of RMSD computed up to layer 100. When  $\sigma^2/\ell^2 > 1$ , the models start escaping the pathology.

**Cos kernel** With a similar setup to that of SE, Figure 6.12b shows that when  $\pi^2\sigma^2/p^2 > 1$ , the models do not suffer the pathology.

**Per kernel** Since the PER kernel has three hyperparameters,  $\sigma^2, \ell^2, p$ , we fix  $\sigma^2$ , and vary  $\ell^2$  and  $p$ . In this case, we collected the RMSDs at layer 100. We then compare the contour plot of

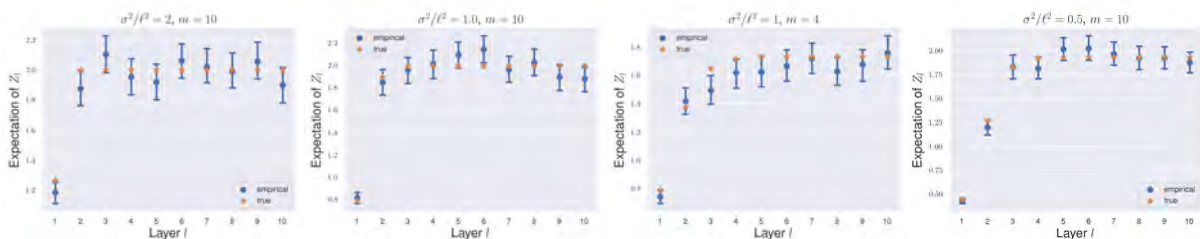


Fig. 6.13 High-dimensional SE:  $\mathbb{E}[Z_n]$  computed from recurrence vs. empirical estimation of  $\mathbb{E}[Z_n]$ .

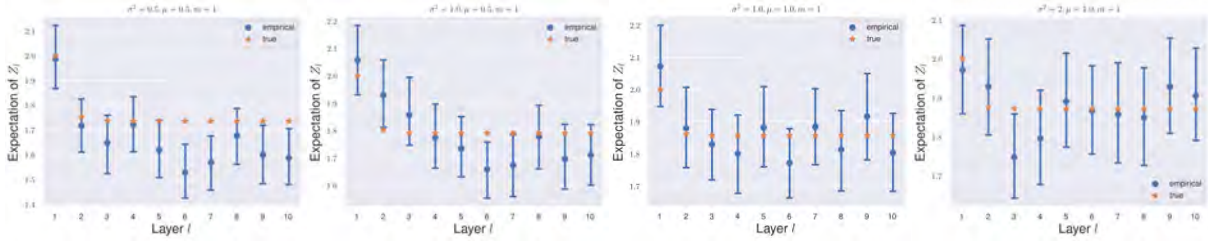


Fig. 6.14 SM kernel:  $\mathbb{E}[Z_n]$  computed from recurrence vs. empirical estimation of  $\mathbb{E}[Z_n]$ .

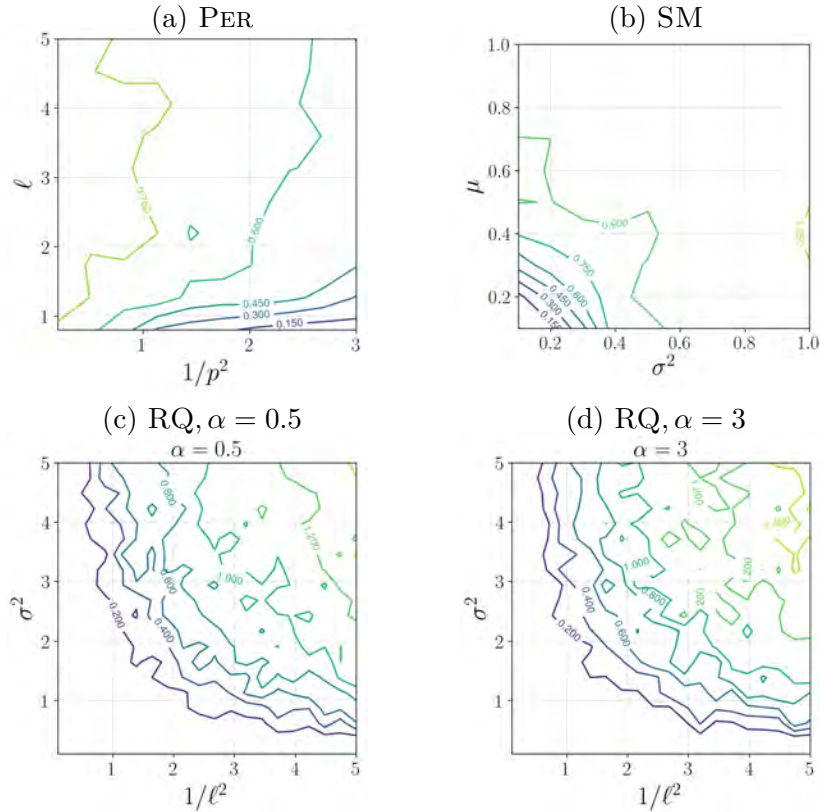


Fig. 6.15 Contour plots of RMSDs at layer 100 for three kernels: PER, SM and RQ.

these RMSDs with the values of the lower bound of  $\mathbb{E}[Z_n]$  computed when  $n$  is large. We can find a similarity between Figure 6.15a and Figure 6.6b. The lower left of both plots has low values, identified as the region that causes the pathology.

**RQ kernel** Analogous to PER, only the RMSDs at layer 100 are gathered. We chose two different values of  $\alpha = \{0.5, 3\}$ , and varying values of  $\sigma^2$  and  $\ell^2$ . Figure 6.15c-d shows two contour plots of RMSDs for the two settings of  $\alpha$ . Both of the two plots share the same area of which the contour level is close to 0.

**SM kernel** This kernel shows no sight of pathology (Figure 6.15b). We can find the similarity between this plot with the contour plot of  $\mathbb{E}[Z_n]$  in Figure 6.6d.

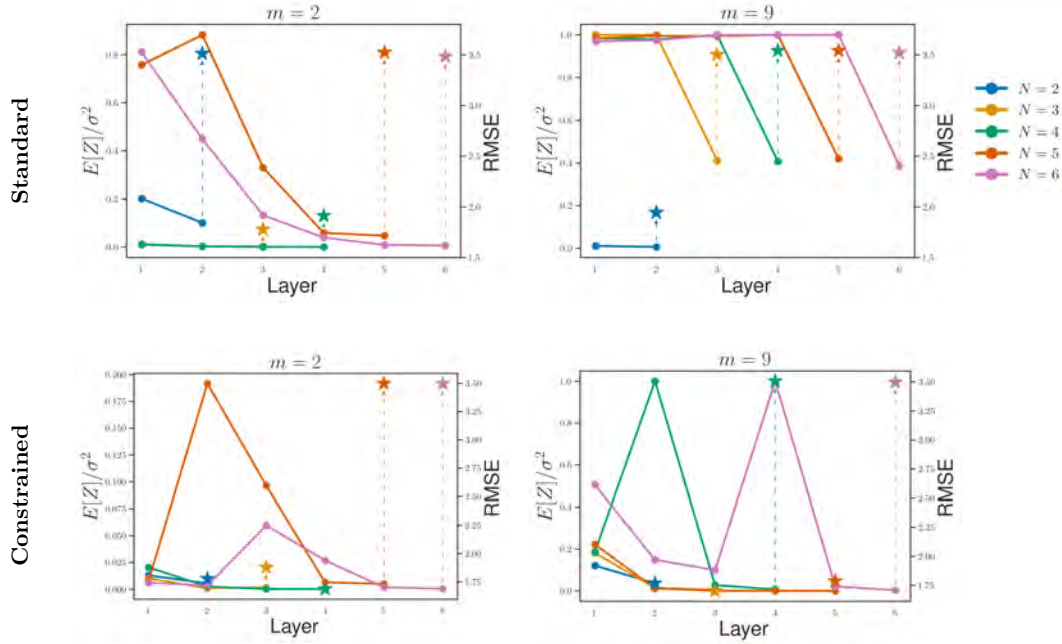


Fig. 6.16 Dual-axis plot of the trajectory  $\mathbb{E}[Z_n]/\sigma^2$  with  $n$  running from 1 to  $N$  and RMSE. Solid lines indicate the trajectories of  $\mathbb{E}[Z_n]/\sigma^2$  projected on the left y-axis. Star markers ( $\star$ ) indicate RMSEs projected on the right y-axis. Dashed lines connect the  $\mathbb{E}[Z_n]/\sigma^2$  and RMSE of the same  $N$ . Here, the constrain coefficient  $c_0 = 0.2$ .

### 6.11.3 Using recurrence relations in DGPs

Here, we use the recurrence relation as a tool to analyze DGP regression models. We learned the models where the number of layers,  $N$ , ranges from 2 to 6 and the number of units per layer,  $m$ , is from 2 to 9. We trained our models on Boston housing data set [Dheeru and Karra Taniskidou \[2017b\]](#) and diabetes data set [Efron et al. \[2004\]](#). For each data set, we train our models with 90% of the data set and hold out the remaining for testing. The inference algorithm is based on [Salimbeni and Deisenroth \[2017\]](#). We considered two settings: (1) standard zero-mean DGPs with SE kernel; (2) the SE kernel hyperparameters are constrained to avoid pathological regions with  $\ell^2 \in (0, c_0 m \sigma^2]$ , constraint coefficient  $0 < c_0 < 1$ .

Figure 6.16 plots the root mean squared errors (RMSEs) and quantity  $\mathbb{E}[Z_n]/\sigma^2$  which describes changes between layers. For the case of standard zero-mean DGPs, we can observe that models can not learn effectively at deeper layers and there are drops in terms of  $\mathbb{E}[Z_n]/\sigma^2$  at the last layer. In the case of constraining hyperparameters, we see fewer drops and the results are improved when comparing to non-constrained cases. It seems that the drop pattern of  $\mathbb{E}[Z_n]/\sigma^2$  correlates to model performances. We provide detailed figures and an additional result on the diabetes data set with a similar observation in [Figure 6.17](#), [6.18](#), [6.19](#) and [6.20](#).

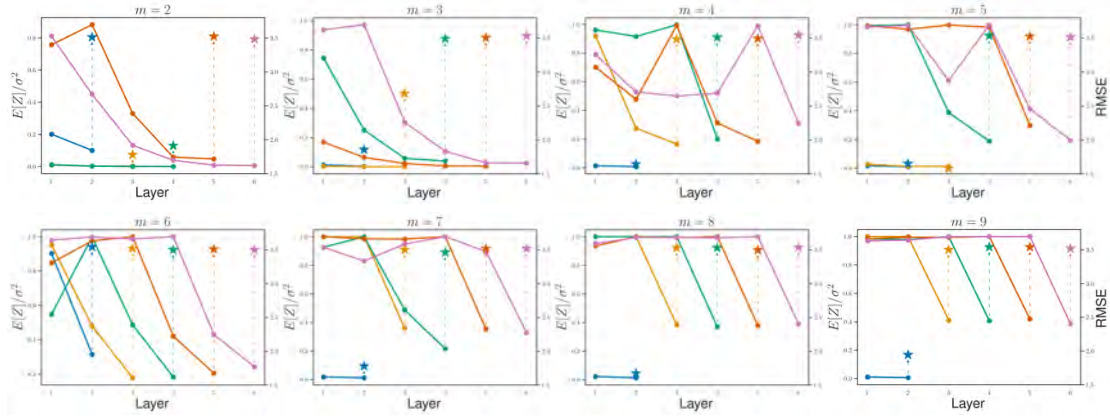


Fig. 6.17 Standard zero-mean DGPs: Results of Boston housing data set

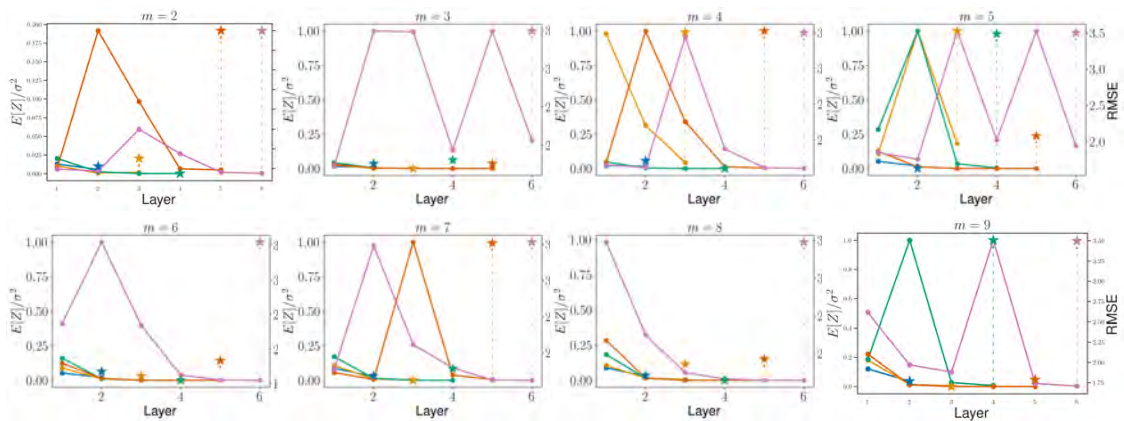


Fig. 6.18 Constrained DGPs: Results of Boston housing data set

#### 6.11.4 High-dimensional data set with zero-mean DGPs

We test on MNIST data set [LeCun and Cortes, 2010] with the two models like previous experiments. The number of units per layer,  $m$ , is chosen as  $m = 30$ . We consider the number of layers,  $N = 2, 3, 4$ .

The standard zero-mean DGP without any regularization fails to learn from data with accuracy  $\approx 10\%$ . This means that the output of this model is just a flat function, making this 10-class classifier have such an accuracy. On the other hand, the constrained zero-mean DGP can alleviate the model performance with accuracy at best 91.21%. Figure 6.21c provides the results with different settings of  $c_0$ .

To have a better understanding of the above models, we visualize the loss landscape Li et al. [2018] of the two cases in Figure 6.21. The standard zero-mean DGP easily falls into unsafe pathological hyperparameters during optimization and cannot escape the unsafe state (see Figure 6.21a). In contrary, the loss landscape of constrained DGPs (Figure 6.21b) shows an improved loss surface. However, we note that it still has a flat region where the optimization

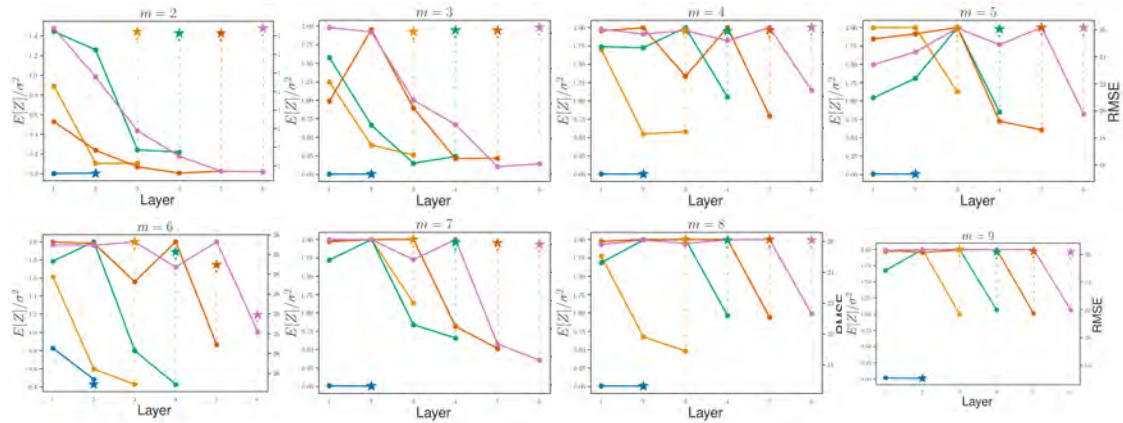


Fig. 6.19 Standard zero-mean DGPs: Results of diabetes data set

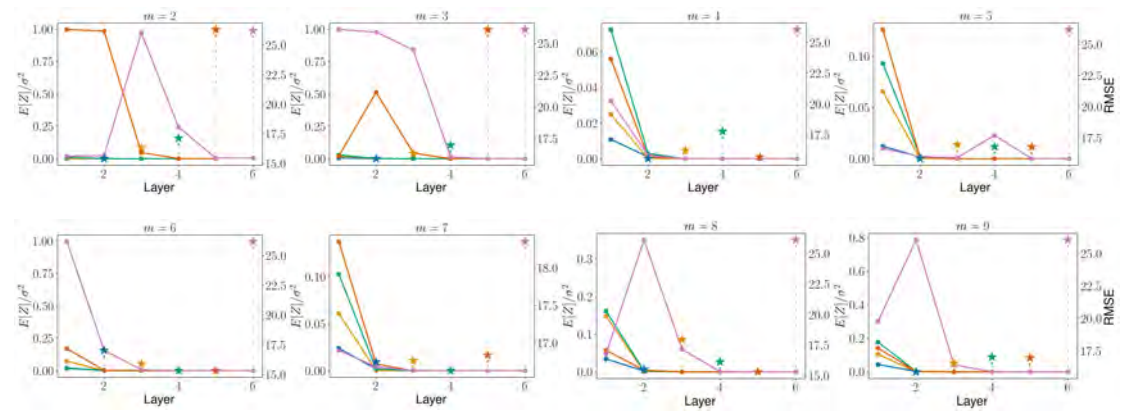


Fig. 6.20 Constrained DGPs: Results of diabetes data set

cannot be improved.

Our result is not as good as the accuracy (98.06%) of nonzero-mean DGPs reported in [Salimbeni and Deisenroth \[2017\]](#). However, we emphasize that the main contribution of our work is not to demonstrate the classifier performance but to show the importance of incorporating the theoretical insights into practice. This shows that learning zero-mean DGPs is potentially possible.



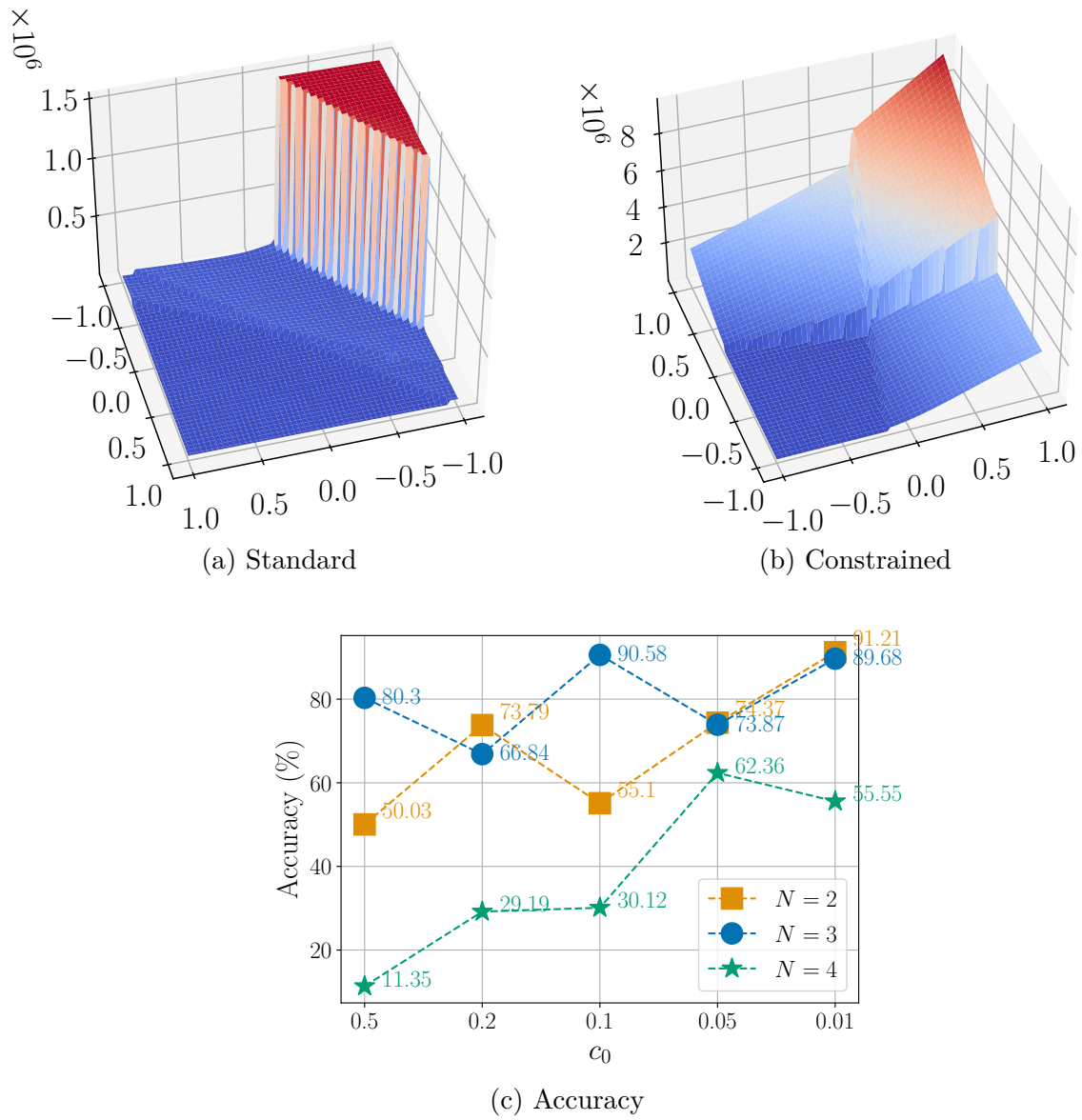


Fig. 6.21 (a-b) Loss landscape of two models. (c) Classification accuracy with respect to the number of layers,  $N$ , and constrain coefficients,  $c_0$ .

## Chapter 7

# Conclusion and Future work

This chapter concludes this thesis with summary and discuss potential follow-up work.

### 7.1 Summary of contribution

One of the main contributions in this thesis is to propose a framework which automatically discovers relational structures for multiple time series. The development of this framework is done in two models: Semi-Relational Kernel Learning and Latent Kernel Model. The former presented in chapter 3 extends Relational Kernel Learning [Hwang et al., 2016]. The latter presented in chapter 4 generalizes the former and allows to train on a wider range of data. Extracting interpretable relations via kernel functions is one of interesting features where one can know how much multiple time series are correlated and what relations are in natural-language reports.

The second contributions of this thesis is a scalable inference for Gaussian processes models of which kernels functions are additive compositional kernels. Along with a new approximate Gaussian process, kernel selection is casted under the hood of learning shrinkage parameters where Horseshoe prior is used.

This thesis also contributes to theoretical insights of deep Gaussian processes. The proposed approach is general, allowing to study a number of kernel functions. Detailed analyses including error bounds and rates of convergence are presented.

### 7.2 Future work

As presented in this thesis, learning discrete structure is usually hard. The approach in this thesis often incorporates probabilistic assumptions over discretized spaces. One promising direction is to

learn the Fourier representation of compositional kernels. This idea is based on the Bochner's theorem where kernel functions can be implicitly inferred from the spectral density in Fourier domain. Preliminary work can be found in [Tompkins et al., 2019]. Yet, the question of how to make such methods interpretable left unanswered.

Deep Gaussian process is a notable extension of Gaussian process by combining Gaussian process layers. Despite the expressiveness of this new model, there are many interesting open questions to establish its foundation: how this model behaves when introducing compositional kernel or injecting inductive biases to this model; practical inference algorithm to select compositional kernel for deep Gaussian processes. These are all challenging problems required careful treatments.

### 7.3 Conclusion

In conclusion, I hope that this thesis can provide a fresh probabilistic perspective of learning compositional covariance structures for multiple time series as well as a guideline for scalable Gaussian processes with complex kernel functions. The theoretical understandings of the existing issues in deep Gaussian processes in this thesis is still preliminary but potentially helpful for building kernel selection for deep Gaussian processes.

# References

- Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3), 2012.
- Ralph G. Andrzejak, Klaus Lehnertz, Florian Mormann, Christoph Rieke, Peter David, and Christian Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 64, 2002. doi: 10.1103/PhysRevE.64.061907.
- Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *ICML*, 2004.
- Matthias Bauer, Mark van der Wilk, and Carl Edward Rasmussen. Understanding probabilistic sparse gaussian process approximations. In *NeurIPS*, pages 1533–1541. 2016.
- Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In *IJCAI*, 2015.
- Yoshua Bengio, Olivier Delalleau, and Nicolas L. Roux. The curse of highly variable functions for local kernel machines. In *NeurIPS*, pages 107–114. 2006.
- Anindya Bhadra, Jyotishka Datta, Nicholas G. Polson, and Brandon Willard. Lasso meets horseshoe: A survey. *Statist. Sci.*, 34(3):405–427, 08 2019.
- Edwin V. Bonilla, Kian Ming Adam Chai, and Christopher K. I. Williams. Multi-task gaussian process prediction. In *NeurIPS*, pages 153–160, 2007.
- Thang D. Bui, José Miguel Hernández-Lobato, Daniel Hernández-Lobato, Yingzhen Li, and Richard E. Turner. Deep gaussian processes for regression using approximate expectation propagation. In *ICML*, pages 1472–1481, 2016.

- Thang D. Bui, Josiah Yan, and Richard E. Turner. A unifying framework for gaussian process pseudo-point approximations using power expectation propagation. *J. Mach. Learn. Res.*, 18: 104:1–104:72, 2017.
- David Burt, Carl Edward Rasmussen, and Mark Van Der Wilk. Rates of convergence for sparse variational Gaussian process regression. In *ICML*, pages 862–871, 2019.
- Bob Carpenter, Andrew Gelman, Matthew Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software, Articles*, 2017.
- Carlos M. Carvalho, Nicholas G. Polson, and James G. Scott. Handling sparsity via the horseshoe. In *AISTATS*, pages 73–80, 2009.
- Jaesik Choi, Eyal Amir, and David J. Hill. Lifted inference for relational continuous models. In *UAI*, pages 126–134, 2010.
- Jaesik Choi, Rodrigo de Salvo Braz, and Hung H. Bui. Efficient methods for lifted inference with aggregate factors. In *AAAI*, page 1030–1036, 2011a.
- Jaesik Choi, Abner Guzmán-Rivera, and Eyal Amir. Lifted relational kalman filtering. In *IJCAI*, pages 2092–2099, 2011b.
- Jaesik Choi, Eyal Amir, Tianfang Xu, and Albert J. Valocchi. Learning relational kalman filtering. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence (AAAI)*, pages 2539–2546, 2015.
- Kurt Cutajar, Edwin V. Bonilla, Pietro Michiardi, and Maurizio Filippone. Random feature expansions for deep Gaussian processes. In *ICML*, volume 70, pages 884–893, 2017.
- Zhenwen Dai, Andreas Damianou, Javier Gonzalez, and Neil D. Lawrence. Variationally auto-encoded deep gaussian processes. In *ICLR*, 2016.
- Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *AISTATS*, volume 31, pages 207–215, 2013.
- Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017a. URL <http://archive.ics.uci.edu/ml>.
- Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017b. URL <http://archive.ics.uci.edu/ml>.

- Finale Doshi, Kurt Miller, Jurgen Van Gael, and Yee Whye Teh. Variational inference for the indian buffet process. In *AISTATS*, 2009.
- Matthew M. Dunlop, Mark A. Girolami, Andrew M. Stuart, and Aretha L. Teckentrup. How deep are deep gaussian processes? *Journal of Machine Learning Research*, 19(54):1–46, 2018.
- David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *ICML*, pages 1166–1174, 2013.
- David K Duvenaud, Hannes Nickisch, and Carl E. Rasmussen. Additive gaussian processes. In *NeurIPS*, pages 226–234. 2011.
- David K. Duvenaud, Oren Rippel, Ryan P. Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *AISTATS*, pages 202–210, 2014.
- Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.
- George Filis, Stavros Degiannakis, and Christos Floros. Dynamic correlation between stock market and oil prices: The case of oil-importing and oil-exporting countries. *International Review of Financial Analysis*, 20(3):152 – 164, 2011.
- Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70, pages 1126–1135, 2017.
- Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *NeurIPS*, 2018.
- Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007. ISBN 0262072882.
- Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553): 452–459, 2015.
- Soumya Ghosh, Jiayu Yao, and Finale Doshi-Velez. Model selection in bayesian neural networks via horseshoe priors. *Journal of Machine Learning Research*, 20(182):1–46, 2019.
- GPY. GPY: A gaussian process framework in python. <http://github.com/SheffieldML/GPY>, since 2012.

- Thomas L. Griffiths and Zoubin Ghahramani. Infinite latent feature models and the indian buffet process. In *NeurIPS*, pages 475–482, 2005.
- Thomas L. Griffiths and Zoubin Ghahramani. The indian buffet process: An introduction and review. *Journal of Machine Learning Research*, 12:1185–1224, 2011.
- R.B. Grosse, R. Salakhutdinov, W.T. Freeman, and J.B. Tenenbaum. Exploiting compositionality to explore a large space of model structures. In *UAI*, pages 306–315, 2012.
- Cristian Guarnizo and Mauricio A. Álvarez. Indian Buffet process for model selection in convolved multiple-output Gaussian processes. *ArXiv e-prints*, 1503.06432, 2015.
- Cristian Guarnizo, Mauricio A. Álvarez, and Álvaro Á. Orozco. Indian buffet process for model selection in latent force models. In *CIARP*, pages 635–642, 2015.
- David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. XAI—explainable artificial intelligence. *Science Robotics*, 4(37), 2019. doi: 10.1126/scirobotics.aay7120. URL <https://robotics.sciencemag.org/content/4/37/eaay7120>.
- Jiyeon Han, Kywoon Lee, Anh Tong, and Jaesik Choi. Confirmatory bayesian online change point detection in the covariance structure of gaussian processes. In *IJCAI*, pages 2449–2455, 2019.
- Marton Havasi, José Miguel Hernández-Lobato, and Juan José Murillo-Fuentes. Inference in deep gaussian processes using stochastic gradient hamiltonian monte carlo. In *NeurIPS*, pages 7517–7527. 2018.
- James Hensman and Neil D Lawrence. Nested variational compression in deep Gaussian processes. *arXiv preprint arXiv:1412.1370*, 2014.
- James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *UAI*, pages 282–290, Arlington, Virginia, USA, 2013.
- Yunseong Hwang, Anh Tong, and Jaesik Choi. Automatic construction of nonparametric relational regression models for multiple time series. In *ICML*, pages 3030–3039, 2016.
- Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017a.
- Phillip A Jang, Andrew Loeb, Matthew Davidow, and Andrew G Wilson. Scalable levy process priors for spectral kernel learning. In *NeurIPS*, pages 3943–3952. 2017b.

- Mohamed Khamisi. An introduction to metric spaces and fixed point theory. 01 2001.
- Hyunjik Kim and Yee Whye Teh. Scaling up the Automatic Statistician: Scalable structure discovery using Gaussian processes. In *AISTATS*, volume 84, pages 575–584, 2018.
- Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.
- Miguel Lázaro-Gredilla and Aníbal R. Figueiras-Vidal. Inter-domain gaussian processes for sparse inference using inducing features. In *NeurIPS*, pages 1087–1095, 2009.
- Miguel Lázaro-Gredilla, Joaquin Quiñero Candela, Carl Edward Rasmussen, and Aníbal R. Figueiras-Vidal. Sparse spectrum gaussian process regression. *J. Mach. Learn. Res.*, 11: 1865–1881, 2010.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4): 541–551, 1989.
- Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL <http://yann.lecun.com/exdb/mnist/>.
- Byung-Jun Lee, Jongmin Lee, and Kee-Eung Kim. Hierarchically-partitioned gaussian process approximation. In *AISTATS*, 2017.
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.
- James Robert Lloyd. GEFCom2012 hierarchical load forecasting: Gradient boosting machines and gaussian processes. *International Journal of Forecasting*, 2013.
- James Robert Lloyd, David Duvenaud, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Automatic construction and natural-language description of nonparametric regression models. In *AAAI*, pages 1242–1250, 2014.
- Chi-Ken Lu, Scott Cheng-Hsin Yang, Xiaoran Hao, and Patrick Shafto. Interpretable deep gaussian processes with moments. In Silvia Chiappa and Roberto Calandra, editors, *AISTATS*, 2020.
- Xiaoyu Lu, Javier Gonzalez, Zhenwen Dai, and Neil Lawrence. Structured variationally auto-encoded optimization. In *ICML*, volume 80, pages 3267–3275, 2018.



- Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *ICLR*, 2017.
- Gustavo Malkomes, Chip Schaff, and Roman Garnett. Bayesian optimization for automated model selection. In *NeurIPS*, pages 2892–2900, 2016.
- Vikash K. Mansinghka, Daniel Selsam, and Yura Perov. Venture: A higher-order probabilistic programming platform with programmable inference. *arXiv preprint*, arXiv:1404.0099, 2014.
- Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke. Fujii, Alexis Boukouvalas, Pablo León-Villagr a, Zoubin Ghahramani, and James Hensman. GPflow: A Gaussian process library using TensorFlow. *J. Mach. Learn. Res.*, pages 1–6, 2017.
- Robert M. May. Simple mathematical models with very complicated dynamics. *Nature*, 261 (5560):459–467, 1976.
- T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.
- Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning, 2020.
- Radford M. Neal. Bayesian learning for neural networks. *PhD thesis*, 1995.
- Sarah E. Neville, John T. Ormerod, and M. P. Wand. Mean field variational bayes for continuous sparse signal shrinkage: Pitfalls and remedies. *Electron. J. Statist.*, 8(1):1113–1151, 2014. doi: 10.1214/14-EJS910.
- Trung Nguyen and Edwin Bonilla. Efficient variational inference for gaussian process regression networks. In *AISTATS*, pages 472–480, 2013.
- Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
- Gabriel Parra and Felipe Tobar. Spectral mixture kernels for multi-output gaussian processes. In *NeurIPS*, pages 6684–6693, 2017.
- Ben Poole, Subhaneil Lahiri, Maithreyi Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *NeurIPS*, pages 3360–3368, 2016.
- Hoifung Poon and Pedro M. Domingos. Sum-product networks: A new deep architecture. In *UAI*, pages 337–346, 2011.

- Quandl. A marketplace for financial data, 2018.
- Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NeurIPS*, pages 1177–1184. 2008.
- Carl Edward Rasmussen and Zoubin Ghahramani. Occam’s razor. In *NeurIPS*, pages 294–300, 2001.
- Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.
- Juan Carlos Reboredo, Miguel A. Rivera-Castro, and Gilney F. Zebende. Oil and us dollar exchange rate dependence: A detrended cross-correlation approach. *Energy Economics*, 42: 132 – 139, 2014.
- Feras A. Saad, Marco F. Cusumano-Towner, Ulrich Schaechtle, Martin C. Rinard, and Vikash K. Mansinghka. Bayesian synthesis of probabilistic programs for automatic data modeling. *Proc. ACM Program. Lang.*, 3(POPL):37:1–37:32, 2019. doi: 10.1145/3290350.
- Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *NeurIPS*, 2017.
- Hugh Salimbeni, Vincent Dutordoir, James Hensman, and Marc Deisenroth. Deep Gaussian processes with importance-weighted variational inference. In *ICML*, pages 5589–5598, 2019.
- Ulrich Schaechtle, Ben Zinberg, Alexey Radul, Kostas Stathis, and Vikash K. Mansinghka. Probabilistic programming with gaussian process memoization. *ArXiv e-prints*, 1512.05665, 2015.
- Jurgen Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 14 May 1987.
- Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *ICLR*, 2017.
- John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient estimation using stochastic computation graphs. In *NeurIPS*, pages 3528–3536. 2015.
- Eric Schulz, Josh Tenenbaum, David K Duvenaud, Maarten Speekenbrink, and Samuel J Gershman. Probing the compositionality of intuitive functions. In *NeurIPS*, pages 3729–3737. 2016.

- Eric Schulz, Joshua B. Tenenbaum, David Duvenaud, Maarten Speekenbrink, and Samuel J. Gershman. Compositional inductive biases in function learning. *Cognitive Psychology*, 99 (Supplement C):44 – 79, 2017.
- G Schwarz. Estimating the dimension of a mode. *The Annals of Statistics*, 6(2), 1978.
- Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *NeurIPS*, pages 4837–4846. 2019.
- Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *NeurIPS*, pages 1257–1264. 2006.
- Christian Steinruecken, Emma Smith, David Janz, James Lloyd, and Zoubin Ghahramani. The Automatic Statistician. In *Automated Machine Learning*. May 2019.
- Shengyang Sun, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger Grosse. Differentiable compositional kernel learning for Gaussian processes. In *ICML*, pages 4828–4837, 2018.
- Terence Tao. *254A, Notes 3a: Eigenvalues and sums of Hermitian matrices*, 2010. URL <https://terrytao.wordpress.com/2010/01/12/254a-notes-3a-eigenvalues-and-sums-of-hermitian-matrices/>.
- Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012, 2014.
- Yee Whye Teh, Matthias W. Seeger, and Michael I. Jordan. Semiparametric latent factor models. In *AISTATS*, 2005.
- Yee Whye Teh, Dilan GrÃ¼n, and Zoubin Ghahramani. Stick-breaking construction for the indian buffet process. In *AISTATS*, pages 556–563, 2007.
- Tong Teng, Jie Chen, Yehong Zhang, and Bryan Kian Hsiang Low. Scalable variational bayesian kernel selection for sparse gaussian process regression. In *AAAI*, pages 5997–6004, 2020.
- Romain Thibaux and Michael I. Jordan. Hierarchical beta processes and the indian buffet process. In *AISTATS*, pages 564–571, 2007.
- R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

- Michalis K. Titsias and Miguel Lázaro-Gredilla. Spike and slab variational inference for multi-task and multiple kernel learning. In *NeurIPS*, pages 2339–2347. 2011.
- Anthony Tompkins, Ransalu Senanayake, Philippe Morere, and Fabio Ramos. Black box quantiles for kernel learning. In *AISTATS*, pages 1427–1437, 2019.
- Anh Tong and Jaesik Choi. Automatic Generation of Probabilistic Programming from Time Series Data. *arXiv e-prints*, art. arXiv:1607.00710, 2016.
- Anh Tong and Jaesik Choi. Discovering latent covariance structures for multiple time series. In *ICML*, pages 6285–6294, 2019.
- Anh Tong and Jaesik Choi. Characterized deep Gaussian processes via nonlinear recurrence systems. *AAAI*, abs/2010.09301, 2021.
- Anh Tong, Toan Tran, Hung Bui, and Jaesik Choi. Learning compositional sparse Gaussian processes with a shrinkage prior. *AAAI*, 2021.
- United States Census Bureau. Population estimates: Historical data. <https://www.census.gov/popest/data/historical/index.html>, 2014. Accessed: 2015-09-15.
- Ivan Ustyuzhaninov, Ieva Kazlauskaitė, Markus Kaiser, Erik Bodin, Neill D. F. Campbell, and Carl Henrik Ek. Compositional uncertainty in deep gaussian processes. In *UAI*, page 206, 2020.
- Jarno Vanhatalo, Jaakko Riihimäki, Jouni Hartikainen, Pasi Jylänki, Ville Tolvanen, and Aki Vehtari. Gpstuff: Bayesian modeling with gaussian processes. *J. Mach. Learn. Res.*, 14(1), 2013.
- Tom von Alten. Top 100 market capitalization. <http://fortboise.org/top100mktcap.html>, 2001. Accessed: 2015-09-15.
- Martin J Wainwright and Michael I Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., 2008. ISBN 1601981848.
- Christian Walder, Kwang In Kim, and Bernhard Schölkopf. Sparse multiscale gaussian process regression. In *ICML*, pages 1112–1119, 2008.
- Matthew P. Wand, John T. Ormerod, Simone A. Padoan, and Rudolf Frühwirth. Mean field variational bayes for elaborate distributions. *Bayesian Anal.*, 6(4):847–900, 12 2011. doi: 10.1214/11-BA631. URL <https://doi.org/10.1214/11-BA631>.

Jue Wang and Pedro M. Domingos. Hybrid markov logic networks. In *AAAI*, pages 1106–1111, 2008.

Helmut Wielandt. An extremum property of sums of eigenvalues. 1955.

Christopher K. I. Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *NeurIPS*, pages 682–688. 2001.

Andrew G. Wilson, David A. Knowles, and Zoubin Ghahramani. Gaussian process regression networks. In *ICML*, 2012.

Andrew Gordon Wilson and Ryan Prescott Adams. Gaussian process kernels for pattern discovery and extrapolation. In *ICML*, pages 1067–1075, 2013.

Yahoo Inc. Yahoo finance - business finance, stock market, quotes, news. <http://finance.yahoo.com/>, 2015. Accessed: 2015-09-15.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006. doi: 10.1111/j.1467-9868.2005.00532.x.

## Acknowledgements

I am fortunate to have Professor Jaesik Choi as my academic advisor. I am grateful for his immense supports and encouragement throughout my PhD journey. Professor Choi always cares for my developments, advising me to stay positive even during difficult times. I have learned from not only his knowledge but also his experience that he shared.

I would like to thank Professor Kwang In Kim for being my administrative advisor. The proposal presentation's discussion with Professor Kwang In Kim helps me to solidify one of the ideas leading to this thesis. I would like to thank Professor Se Young Chun for being in both my proposal and thesis committee as well as for teaching me two fundamental courses at UNIST which are helpful and relevant to my research. I am grateful for having Professor Kee-Eung Kim as one of my thesis committee members and I truly appreciate his thoughtful advice. I would like to thank Professor Sung-Phil Kim for his valuable feedback.

I would like to thank Dr. Hung Bui and Dr. Toan Tran for hosting my internship. I appreciate the time in VinAI as an eye-opening experience that I can broaden my view in different research areas. I always respect their achievements and contributions to the artificial intelligence community in Vietnam.

I also would like to thank SAILab members for being helpful and reliable labmates who I always can count on when asking for favors within our lab environment as well as many other things outside the lab.

I want to thank my friends in my home country for all the encouragements and my friends at UNIST for sharing experience and for hanging out so that my life outside of research is more enjoyable.

Finally, I would like to thank my dearest parents who always support me in pursuing PhD and constantly remind me to stay healthy in any situations. I would like to thank my sister and my brother-in-law for always giving helping hands whenever I need.