



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](#) 

Master's Thesis

The Automatic Statistician: A Relational Perspective

Yunseong Hwang

Department of Electrical and Computer Engineering
(Computer Engineering)

Graduate School of UNIST

2016

The Automatic Statistician: A Relational Perspective

Yunseong Hwang

Department of Electrical and Computer Engineering
(Computer Engineering)

Graduate School of UNIST


The Automatic Statistician: A Relational Perspective

A thesis
submitted to the Graduate School of UNIST
in partial fulfillment of the
requirements for the degree of
Master of Science

Yunseong Hwang

12. 4. 2015

Approved by



Advisor

Jaesik Choi

The Automatic Statistician: A Relational Perspective

Yunseong Hwang

This certifies that the thesis of Yunseong Hwang is approved.

12. 4. 2015



Advisor: Jaesik Choi



Comittee Member: Sung Ju Hwang



Comittee Member: Gil Jin Jang

Abstract

Gaussian Processes (GPs) provide a general and analytically tractable way of capturing complex time-varying, nonparametric functions. The time varying parameters of GPs can be explained as a composition of base kernels such as linear, smoothness or periodicity in that covariance kernels are closed under addition and multiplication. The Automatic Bayesian Covariance Discovery (ABCD) system constructs natural-language description of time-series data by treating unknown time-series data nonparametrically using GPs. Unfortunately, learning a composite covariance kernel with a single time-series dataset often results in less informative kernels instead of finding qualitative distinct descriptions. We address this issue by proposing a relational kernel learning which can model relationship between sets of data and find shared structure among the time series datasets. We show the shared structure can help learning more accurate models for sets of regression problems with some synthetic data, US top market capitalization stock data and US house sales index data.

Contents

I. Introduction	1
II. Automatic Bayesian Covariance Discovery	3
2.1 Gaussian Process	3
2.2 Compositional Kernel Learning	3
2.2.1 Base Kernels	4
2.2.2 Operations	5
2.2.3 Search grammar	6
2.2.4 Algorithm	6
III. Relational Automatic Bayesian Covariance Discovery	8
3.1 Relational Learning	8
3.2 Basic idea	8
3.3 Model	9
3.4 Algorithm	11
IV. Related Work	13
4.1 Learning Composite Gaussian Process Kernels	13
4.2 Relational Learning for Continuous Data	13
V. Experimental Results	14
5.1 Learning Hyperparameters on Synthetic Data	14
5.2 Stock Market Data	14
5.3 Housing Market Data	17
VI. Conclusion	21

List of Figures

1.1	This figure shows components learned from ABCD (top) and Relational Automatic Bayesian Covariance Discovery (RABCD) (bottom). (a) and (b) represent adjusted closes of GE and learned components by ABCD, respectively. (c) and (d) represent adjusted closes of 3 selected stocks (from top, yellow:XOM, purple:GE, green:MSFT) and learned (common) components by RABCD. The red vertical lines indicate the September 11 attacks which occurred in 2001. RABCD finds and explains the sudden drop after 911 by selectively applying a constant kernel around that period. However, ABCD tries to fit that drop using a (less informative) rapidly varying squared exponential (SE) kernel function. We provide all components learned by ABCD and RABCD in the appendix.	2
2.1	This figure shows sampled functions that are drawn from a GP prior with zero mean function and squared exponential (SE) covariance kernel function. As they share the same covariance kernel function, the shapes of sampled functions show similar smoothness in change (since the squared exponential kernel function encodes smooth function).	4
2.2	Depiction of kernel search process of the ABCD algorithm. The algorithm (1) extends an initial kernel function to multiple possible candidate kernel functions, (2) optimizes hyperparameters of kernel functions to maximize likelihood (3) and finally selects the best kernel function that maximizes Bayesian Information Criterion (BIC).	7
3.1	Graphical representation of (a) Relational Gaussian Processes (RGPs) [1] and (b) RABCD. Here f denotes latent function. The subscripts of f_i and f_j denote sampled latent values for each point, as in $f_i = f(x_i)$. y_i and y_j are observed values for each point. N is the number of data points for each data set. M is the number of different data sets.	9
3.2	Simple depiction of how the RABCD model works. The model gets a set of sequences of observations as input and gives a GP model that represents the multiple sequences by a shared covariance kernel function. And the model decomposes sequences into additive components that show distinct characteristics.	10
3.3	Graphical representation of RABCD model with more steps in generating GP prior. k defines the kernel function's structure but not the values of its hyperparameters. θ completes the function by deciding the function's hyperparameter values. σ_j is multiplied by $k(x, x')$ for each GP prior. This works as normalization of scale variance of each dataset.	10
5.1	The convergence of error between the optimized hyperparameter and original true parameter that was used when generating data. The horizontal axis denotes the number of data sets used in hyperparameter optimization. The vertical axis shows the value of root mean squared error between optimized and true hyperparameters. Each boxplot shows the distribution of errors for each step.	15

5.2	Plotting shape of total 9 number of US stock data. Upper plot is original data and lower plot is normalized data.	16
5.3	Picture that shows expressive power of RABCD by finding the sudden decrease in the most of stocks after 9/11.	18
5.4	Plotting shape of total 6 number of US house price index data. Upper plot is original data and lower plot is normalized data.	19

List of Tables

5.1	The experimental comparisons of GP models with individual ABCD and RABCD in the stock data set (top 6 US stocks in 2001). For each column, NLL means the negative log likelihood, N is the number of data points, P is the number of hyperparameters and BIC is the Bayesian information criterion. ABCD is slightly better in a simple aggregation of NLL only. However, our RABCD outperforms ABCD in the BIC due to a significantly lower 21 parameters (P) compared to 49 parameters in ABCD.	15
5.2	The BIC of ABCD and RABCD in the stock data set. ‘Top 3’, ‘Top 6’ and ‘Top 9’ stocks were selected by their market capitalization ranks in 2011. As shown in Table 5.1, RABCD requires fewer parameters than ABCD. RABCD models trained with 3 stocks and 6 stocks show better performance than individually optimized ABCD models. When 9 stocks are considered, the individual ABCD models show better performance than the single (shared) RABCD model.	17
5.3	A comparison of BIC between the GP model from individual ABCD and the GP model from RABCD, with house data. For each column, NLL means negative log likelihood, N is the number of data points, P is the number of hyperparameters and BIC is the Bayesian information criterion. For each row, RABCD is the result from our model. From New York to San Francisco, those results are from individual ABCD. Total is summation of those individual results.	18
5.4	The BIC of ABCD and RABCD in the housing market data set. ‘Top 2’, ‘Top 4’ and ‘Top 6’ US cities were selected in terms of their city population rank. The BICs of the RABCD models are similar or better than the BICs of individually trained ABCD models.	20

List of Abbreviations

ABCD Automatic Bayesian Covariance Discovery. 1–9, 11, 13–15, 17, 18, 20

BIC Bayesian Information Criterion. 2, 4, 7, 14, 17, 18, 20, 21

CKL Compositional Kernel Learning. 1, 3, 13

GDP Gross Domestic Product. 1

GP Gaussian Process. 1–5, 8–11, 13–15, 18

MKL Multiple Kernel Learning. 13

RABCD Relational Automatic Bayesian Covariance Discovery. 1–4, 8–15, 17, 18, 20, 21

RGP Relational Gaussian Process. 2, 8, 9

SRL Statistical Relational Learning. 13

Chapter I

Introduction

Gaussian Processes (GPs) provide a general and analytically tractable way of capturing complex time-varying, nonparametric functions. The time varying parameters of GPs can be explained as a composition of base kernels such as linear, smoothness or periodicity in that covariance kernels are closed under addition and multiplication. The Automatic Bayesian Covariance Discovery (ABCD) system constructs natural-language description of time-series data by treating unknown time-series data nonparametrically using GPs.

It is important to find data dependencies and the structure in time-series data. GPs represent data in a non-parametric way with a mean function and a covariance kernel function. The covariance kernel function determines correlation patterns between the data points. Therefore, learning a proper kernel is essential to model data points with GPs. Unfortunately, finding an appropriate kernel often requires manual encoding by human experts or reduces to a simple problem estimating parameters of a fixed, predefined kernel structure. Solving the estimation problem would act as optimization, improving performance of the kernel. But still the expressiveness of the kernel and resulting GP model is restricted to the fixed kernel structure.

The covariance kernels of GPs are known to be closed under addition and multiplication [2]. Thus, a sequence of complex real-valued variables could be explained by a compositional kernel with base kernels and kernel operations. Recently, kernel operation grammars and a framework for an automatic discovery of a compositional kernel have been proposed [3]. This framework is flexible and interpretable in that the framework automatically discovers a complex composition of interpretable base kernels. Once a compositional kernel is found, individual base component can be explained in a human readable form. This capability of decomposition into multiple components and interpretability shed light on finding shared structure given multiple sets of data.

Finding a shared structure in multiple sequences may reveal the common covariance structures of the sequences beyond the patterns in a single sequence. As a typical example in economics, the multivariate view is central where each variable is normally viewed in the context of relationships to other variables, like exchange rate affecting Gross Domestic Product (GDP).

We propose a Relational Automatic Bayesian Covariance Discovery (RABCD) (explained in Chapter 3) to find a shared covariance kernel for multiple sets of data sequences. Our algorithm discovers a shared kernel which can explain the causes of changes in multiple data sets. Since GPs with the learned kernel are non-parametric, we can represent any data which is known to have the same relationship with the training data.

This paper is organized as follows. Chapter 2 introduces Gaussian Processes (GPs) and the Automatic Bayesian Covariance Discovery (ABCD) system. Chapter 3 presents our main contribution and algorithm, Relational Automatic Bayesian Covariance Discovery (RABCD). Chapter 4 discusses related

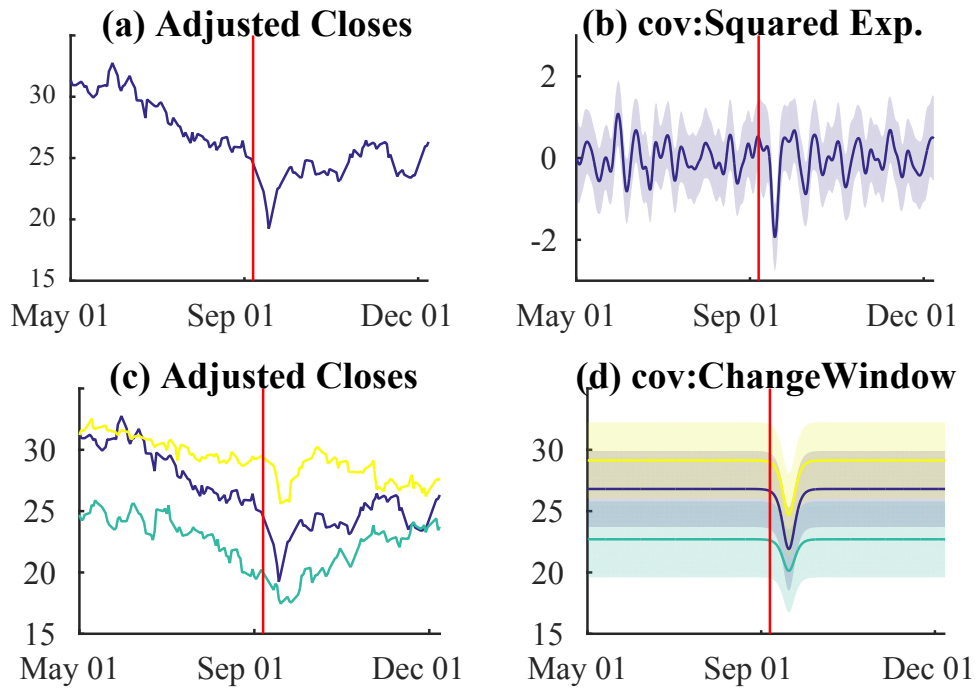


Figure 1.1: This figure shows components learned from ABCD (top) and RABCD (bottom). (a) and (b) represent adjusted closes of GE and learned components by ABCD, respectively. (c) and (d) represent adjusted closes of 3 selected stocks (from top, yellow:XOM, purple:GE, green:MSFT) and learned (common) components by RABCD. The red vertical lines indicate the September 11 attacks which occurred in 2001. RABCD finds and explains the sudden drop after 911 by selectively applying a constant kernel around that period. However, ABCD tries to fit that drop using a (less informative) rapidly varying squared exponential (SE) kernel function. We provide all components learned by ABCD and RABCD in the appendix.

work. Chapter 5 shows experimental results with one synthetic and two real-world data sets followed by conclusions in Chapter 6.

Chapter II

Automatic Bayesian Covariance Discovery

Here, we briefly explain GP models, and then introduce the ABCD system [3], which is an extension of the Compositional Kernel Learning (CKL) [2].

2.1 Gaussian Process

GPs are distributions over functions such that any finite set of function evaluations, $(f(x_1), f(x_2), \dots, f(x_N))$ form a multivariate Gaussian distribution [4]. As a multivariate Gaussian distribution is specified by its mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, a GP is specified by its mean function, $\boldsymbol{\mu}(x) = \mathbb{E}[f(x)]$ and covariance kernel function, $k(x, x') = \text{Cov}(f(x), f(x'))$. Evaluations of each function for the GPs on a finite set of points are equivalent to the mean vector and the covariance matrix for the multivariate Gaussian distribution, like $\boldsymbol{\mu}_i = \boldsymbol{\mu}(x_i)$ and $\Sigma_{ij} = k(x_i, x_j)$. These mappings enable GPs utilize the multivariate Gaussian distribution form in infinite input space. Log-likelihood of target values $\mathbf{y} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)]^T$ given inputs $\mathbf{X} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)]^T$ is like the following:

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}(\mathbf{y} - \mathbf{m})\boldsymbol{\Sigma}^{-1}(\mathbf{y} - \mathbf{m}) - \frac{1}{2} \log |\boldsymbol{\Sigma}| - \frac{N}{2} \log 2\pi \quad (2.1)$$

where $\mathbf{m}_i = \boldsymbol{\mu}(\mathbf{x}_i)$, $\Sigma_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ and N is number of data points.

Throughout this paper, we will use the following notations for GPs. If a function or evaluations of the function f are drawn from a GP specified by its mean function $\boldsymbol{\mu}(x)$ and covariance kernel function $k(x, x')$:

$$f \sim \mathcal{GP}(\boldsymbol{\mu}(x), k(x, x')) \quad (2.2)$$

We may occasionally omit the arguments of the mean and covariance kernel functions, which are x and x' , for simplicity, for example just saying k as a kernel function. For zero-mean GPs, we put 0 in the place of mean function, which means $\boldsymbol{\mu}(x) = 0 \forall x$. The covariance kernel functions often have its hyperparameters which are free parameters. For example, the squared exponential kernel function $k(x, x') = \sigma^2 \exp(-|x - x'|^2/\ell)$ has two hyperparameters, σ and ℓ . Throughout this paper, we will use notation $k(x, x'; \boldsymbol{\theta})$ that denotes a kernel function that has a vector $\boldsymbol{\theta}$ as hyperparameter vector, which is composed of possible hyperparameters of the kernel function.

2.2 Compositional Kernel Learning

CKL constructs and finds richer kernels which are composed of several base kernels and operations. In theory, any positive definite kernels are closed under addition and multiplication [2] and can be base kernels. Here, each base kernel expresses each distinctive feature such as linearity or periodicity.

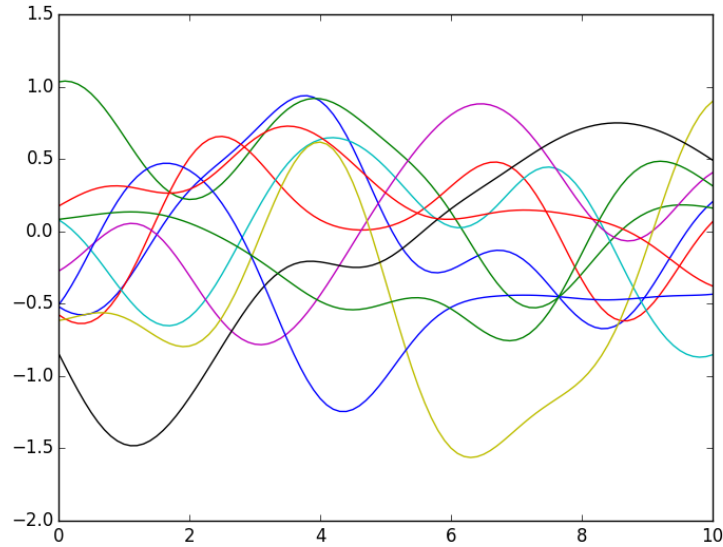


Figure 2.1: This figure shows sampled functions that are drawn from a GP prior with zero mean function and squared exponential (SE) covariance kernel function. As they share the same covariance kernel function, the shapes of sampled functions show similar smoothness in change (since the squared exponential kernel function encodes smooth function).

The kernel operations include not only addition and multiplication but also so-called, change-point and change-window operation to deal with some abrupt structural changes along input space.

2.2.1 Base Kernels

Five base kernels are used for making compositional kernels. Each kernel encodes different characteristics of functions, which can be combined to make more expressive kernel functions.

Base Kernel	Kernel function	Encoding Function	Parameters
White Noise (WN)	$\sigma^2 \delta_{x,x'}$	Uncorrelated noise	σ
Constant (C)	σ^2	Constant functions	σ
Linear (LIN)	$\sigma^2 (x - \ell)(x' - \ell)$	Linear functions	σ, ℓ
Squared Exponential (SE)	$\sigma^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$	Smooth functions	σ, ℓ
Periodic (PER)	$\sigma^2 \frac{\exp(\cos \frac{2\pi(x-x')}{p}/\ell^2) - I_0(1/\ell^2)}{\exp(1/\ell^2) - I_0(1/\ell^2)}$	Periodic functions	σ, ℓ, p

where $\delta_{x,x'}$ is Kronecker delta function and $I_0(x)$ is modified Bessel function of the first kind of order zero.

2.2.2 Operations

Addition

The first operation is *addition* which sums multiple kernel functions and makes a new kernel function.

$$k'(x, x') = k_1(x, x') + k_2(x, x').$$

This operation works as a superposition of multiple independent covariance functions. In general, if $f_1 \sim \mathcal{GP}(\mu_1, k_1), f_2 \sim \mathcal{GP}(\mu_2, k_2)$ then $f := f_1 + f_2 \sim \mathcal{GP}(\mu_1 + \mu_2, k_1 + k_2)$.

Following is the proof:

$$\begin{aligned}
 \text{cov}(f(x_i), f(x_j)) &= \mathbb{E}[f(x_i)f(x_j)] - \mathbb{E}[f(x_i)]\mathbb{E}[f(x_j)] \\
 &= \mathbb{E}[(f_1(x_i) + f_2(x_i))(f_1(x_j) + f_2(x_j))] \\
 &\quad - \mathbb{E}[(f_1(x_i) + f_2(x_i))]\mathbb{E}[(f_1(x_j) + f_2(x_j))] \\
 &= \text{cov}(f_1(x_i), f_1(x_j)) + \text{cov}(f_1(x_i), f_2(x_j)) \\
 &\quad + \text{cov}(f_2(x_i), f_1(x_j)) + \text{cov}(f_2(x_i), f_2(x_j)) \\
 &= \text{cov}(f_1(x_i), f_1(x_j)) + \text{cov}(f_2(x_i), f_2(x_j))
 \end{aligned} \tag{2.3}$$

Multiplication

ABCD assumes zero mean for GPs, since marginalizing over an unknown mean function can be equivalently expressed as a zero-mean GPs with a new kernel [3]. Under this assumption, the following *multiplication* operation is also applicable.

$$k'(x, x') = k_1(x, x') \times k_2(x, x').$$

In general, if $f_1 \sim \mathcal{GP}(0, k_1), f_2 \sim \mathcal{GP}(0, k_2)$ then $f := f_1 \times f_2 \sim \mathcal{GP}(0, k_1 \times k_2)$.

Following is the proof:

$$\begin{aligned}
 \text{cov}(f(x_i), f(x_j)) &= \mathbb{E}[f(x_i)f(x_j)] - \mathbb{E}[f(x_i)]\mathbb{E}[f(x_j)] \\
 &= \mathbb{E}[f_1(x_i)f_1(x_j)f_2(x_i)f_2(x_j)] \\
 &\quad - \mathbb{E}[f_1(x_i)f_2(x_i)]\mathbb{E}[f_1(x_j)f_2(x_j)] \\
 &= (f_1(x_i)f_1(x_j) - 0)(f_2(x_i)f_2(x_j) - 0) \\
 &= \text{cov}(f_1(x_i), f_1(x_j)) \times \text{cov}(f_2(x_i), f_2(x_j))
 \end{aligned} \tag{2.4}$$

Change Point

The third operation is the *change-point (CP)* operation. Given two kernel functions, k_1 and k_2 , the new kernel function is represented as follows:

$$\begin{aligned}
 k'(x, x') &= \sigma(x)k_1(x, x')\sigma(x') \\
 &\quad + (1 - \sigma(x))k_2(x, x')(1 - \sigma(x'))
 \end{aligned}$$

where $\sigma(x)$ is a sigmoidal function, that the function output lies between 0 and 1, and ℓ is the change-point in input space. The change-point operation divides function domain (i.e., time) into two sides and applies different kernel function on each side. To generalize, if $f_1 \sim \mathcal{GP}(\mu_1, k_1)$, $f_2 \sim \mathcal{GP}(\mu_2, k_2)$ then $f := \sigma(x)f_1 + (1 - \sigma(x))f_2 \sim \mathcal{GP}(\sigma(x)\mu_1 + (1 - \sigma(x))\mu_2, \sigma(x)k_1\sigma(x') + (1 - \sigma(x))k_2(1 - \sigma(x')))$.

Change Window

Finally, the *change-window (CW)* operation applies the CP operation twice with two different change points ℓ_1 and ℓ_2 . Given two sigmoidal functions $\sigma_1(x; \ell_1)$ and $\sigma_2(x; \ell_2)$ where $\ell_1 < \ell_2$, the new function will be $f := \sigma_1(x)f_1(1 - \sigma_2(x)) + (1 - \sigma_1(x))f_2\sigma_2(x)$, which applies the function f_1 to the window (ℓ_1, ℓ_2) . A composite kernel expression after the change-window operation will be as follows:

$$k'(x, x') = \sigma_1(x)(1 - \sigma_2(x))k_1(x, x')\sigma_1(x')(1 - \sigma_2(x')) \\ + (1 - \sigma_1(x))\sigma_2(x)k_2(x, x')(1 - \sigma_1(x'))\sigma_2(x').$$

2.2.3 Search grammar

ABCD searches a composite kernel based on the search grammar. The search grammar specifies how to develop the current kernel expression by applying the operations with the base kernels. The following rules are examples for typical search grammar:

$$\begin{array}{ll} \mathcal{S} \rightarrow \mathcal{S} + \mathcal{B} & \mathcal{S} \rightarrow \mathcal{S} \times (\mathcal{B} + \mathcal{C}) \\ \mathcal{S} \rightarrow \mathcal{S} \times \mathcal{B} & \mathcal{S} \rightarrow \mathcal{B} \\ \mathcal{B} \rightarrow \mathcal{B}' & \mathcal{S} \rightarrow \mathcal{C} \\ \mathcal{S} \rightarrow \text{CP}(\mathcal{S}, \mathcal{S}) & \mathcal{S} + \mathcal{S}' \rightarrow \mathcal{S} \\ \mathcal{S} \rightarrow \text{CW}(\mathcal{S}, \mathcal{S}) & \mathcal{S} \times \mathcal{S}' \rightarrow \mathcal{S} \\ \mathcal{S} \rightarrow \text{CW}(\mathcal{S}, \mathcal{C}) & \\ \mathcal{S} \rightarrow \text{CW}(\mathcal{C}, \mathcal{S}) & \end{array}$$

where \mathcal{S} represents any kernel subexpression, \mathcal{B} and \mathcal{B}' are base kernels. \mathcal{C} is a constant base kernel. For example, suppose that there is a kernel expression $\mathcal{E} = \mathcal{K}_1 + \mathcal{K}_2 + \mathcal{K}_3$ where \mathcal{K}_i is a kernel expression, which cannot be separated into summands. Then, kernel subexpression \mathcal{S} can be a summation of a non-empty subset among the summands, \mathcal{K}_i . If we apply multiplication grammar on subset $(\mathcal{K}_1 + \mathcal{K}_3)$ with base kernel \mathcal{B}_1 , the expanded kernel expression would be $\mathcal{E}' = \mathcal{K}_2 + (\mathcal{K}_1 + \mathcal{K}_3) \times \mathcal{B}_1$

2.2.4 Algorithm

Given data and a maximum search depth, the algorithm gives a compositional kernel $k(x, x'; \theta)$. Starting from the WN kernel, the algorithm expands the kernel expression based on the search grammar, optimizes hyperparameters for the expanded kernels, evaluates those kernels given the data and selects the best one among them. This procedure repeats. The next iterative procedure starts with the best composite kernel selected in the previous iteration. The conjugate gradient method is used when optimizing

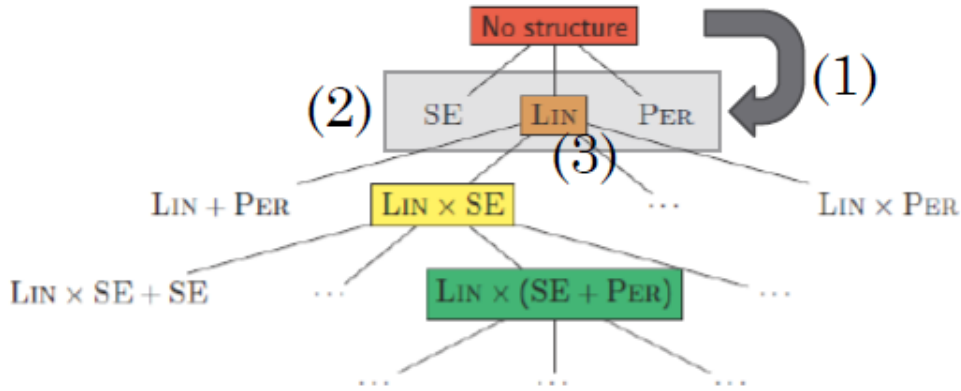


Figure 2.2: Depiction of kernel search process of the ABCD algorithm. The algorithm (1) extends an initial kernel function to multiple possible candidate kernel functions, (2) optimizes hyperparameters of kernel functions to maximize likelihood (3) and finally selects the best kernel function that maximizes BIC.

hyperparameters. Bayesian Information Criterion (BIC) [5] is used for the model evaluation. The BIC of model \mathcal{M} with $|\mathcal{M}|$ number of free parameters and data \mathcal{D} with $|\mathcal{D}|$ number of data points is:

$$\text{BIC}(\mathcal{M}) = -2\log p(\mathcal{D}|\mathcal{M}) + |\mathcal{M}|\log|\mathcal{D}| \quad (2.5)$$

The iteration continues until the specified maximum search depth is reached. During the iteration, the algorithm keeps the best model for the output.

Here is an example of how this algorithm works. Suppose that we start from the WN kernel. Then, we apply operators with some base kernels as described in the expansion grammar, such as $\text{WN} \rightarrow \text{WN} + \text{SE}$, $\text{WN} \rightarrow \text{WN} + \text{LIN}$ and so on. With those expanded kernels, the algorithm optimizes hyperparameters of each expanded kernel. Now we have all the optimized kernels. Finally we compare the optimized ones and select the best kernel in terms of the BIC measure. This procedure repeats until we meet a certain depth of search. As an example, suppose that the best kernel selected in the previous step was $\text{WN} + \text{SE}$. Then we apply kernel expansion again like $\text{WN} + \text{SE} \rightarrow \text{WN} + \text{SE} \times \text{LIN}$ and so on. Then we optimize the hyperparameters, select the best kernel, and then proceed to the next step.

Chapter III

Relational Automatic Bayesian Covariance Discovery

RABCD takes the ABCD system into a concept of relational modeling. Given multiple sets of time-series data, RABCD finds a composite kernel for GPs, which can describe the shared structure of the given datasets.

3.1 Relational Learning

When we say the term relational in machine learning, it means that we want to make use of information between random variables. For example if we know that a random variable x and another random variable y are positively correlated, here the relational information is that there is a positive correlation between the two random variables. And observation of x gives additional information in expecting a value of y , which is the making use of that information. Generally GPs include this kind of relational information between random values in its covariance kernel functions, as the kernel function determines the covariance of two random values given corresponding two input values. So learning a good covariance kernel function is key part when we are dealing with the GPs.

There was a previous approach that adopt relational modeling to GP, called Relational Gaussian Processes (RGPs) [1]. RGPs explicitly brings up a new variable between two random variables that determines relationship between the two, which decides whether those two random variables are positively/negatively correlated. This relational variable works with base kernel from original GPs to give final covariance between two random variable. Our approach is different compared to the RGPs case since our system put more focus on modeling not the relationship between random variables but the relationship of data-wide pattern between different datasets.

3.2 Basic idea

RABCD aims to find a model \mathcal{M} that explains M multiple data sets, $\mathcal{D} = \{d_1, d_2, \dots, d_M\}$ well. To do so, we find a model that maximizes the likelihood, $p(\mathcal{D}|\mathcal{M}) = p(d_1, d_2, \dots, d_M|\mathcal{M})$. Here, \mathcal{M} represents GP models. We assume the conditional independence of the likelihoods of each time-series data d_i . Hence, the log likelihood of the whole data is the sum of the log likelihoods of individual time-series:

$$\begin{aligned} \log p(d_1, \dots, d_M|\mathcal{M}) &= \log \prod_i p(d_i|\mathcal{M}) \\ &= \sum_i \log p(d_i|\mathcal{M}) \end{aligned} \tag{3.1}$$

We assume that there is a single, shared factor that determines the covariance pattern for multiple data sets within the same domain, e.g., stock values of companies in a certain industrial sector. This acts as tying different factors that determine the covariance pattern of each data set as one single factor. By finding the factor, we ultimately wish to figure out the shared compositional covariance kernel. Here, the

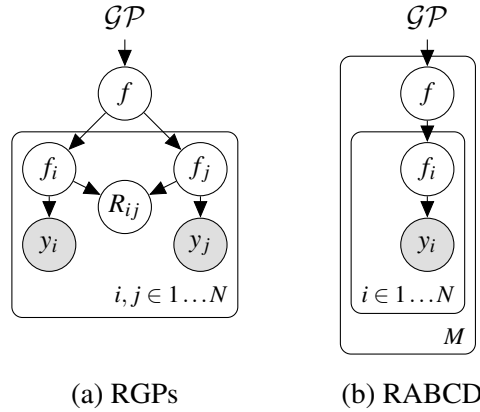


Figure 3.1: Graphical representation of (a) RGPs [1] and (b) RABCD. Here f denotes latent function. The subscripts of f_i and f_j denote sampled latent values for each point, as in $f_i = f(x_i)$. y_i and y_j are observed values for each point. N is the number of data points for each data set. M is the number of different data sets.

term relational model comes from that single factor that ties all the data sets. Finding the shared factor in GPs is reduced to finding a shared covariance kernel function.

This approach enables consideration of each data separately and utilization of existing optimization technique and model evaluation form used for a single dataset. In terms of the optimization technique, ABCD uses conjugate gradient descent algorithm to optimize hyperparameters of covariance kernel function. This requires calculating gradient of log-likelihood. As the log-likelihood of all datasets can be decomposed into summation of log-likelihoods of individual dataset, gradient of the total log-likelihood also can be expressed and calculated by just calculating the gradients individually and adding up all of them.

$$\begin{aligned} \frac{\partial}{\partial \theta} \log p(d_1, \dots, d_M | \mathcal{M}) &= \frac{\partial}{\partial \theta} \sum_i \log p(d_i | \mathcal{M}) \\ &= \sum_i \frac{\partial}{\partial \theta} \log p(d_i | \mathcal{M}) \end{aligned} \quad (3.2)$$

The independence between the datasets lessens computational complexity of calculating inverse of covariance matrix during the inference process of GPs. The independence means zero covariance between the different datasets and makes covariance matrix to be a block diagonal form, making factorization of log-likelihood possible. Assuming that there are M datasets with N number of points for each dataset, computational complexity of calculating inverse of full covariance matrix would be $O(MN^3)$. However if they are independent, we only need to calculate inverse matrices of M number of N by N covariance matrices, which results computational complexity of $M \times O(N^3)$ making faster inference.

3.3 Model

Defining a GP requires its mean function $\mu(x)$ and covariance function $k(x, x')$. The mean function is set to be a constant function which gives 0. The covariance function is divided into two parts, functional

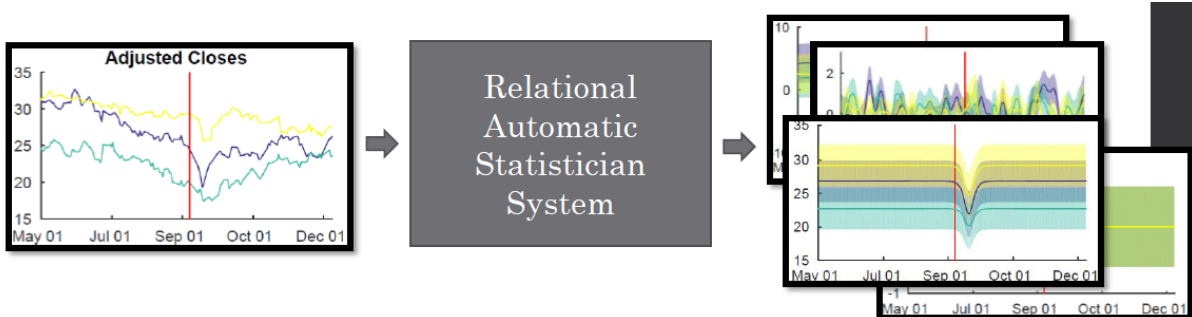


Figure 3.2: Simple depiction of how the RABCD model works. The model gets a set of sequences of observations as input and gives a GP model that represents the multiple sequences by a shared covariance kernel function. And the model decomposes sequences into additive components that show distinct characteristics.

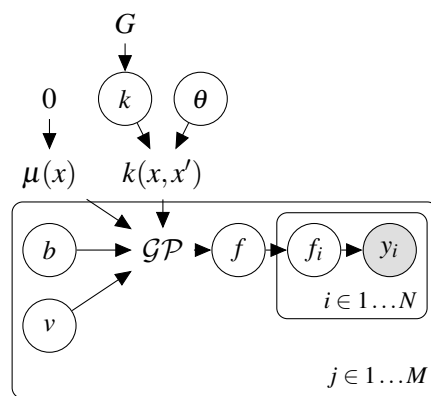


Figure 3.3: Graphical representation of RABCD model with more steps in generating GP prior. k defines the kernel function's structure but not the values of its hyperparameters. θ completes the function by deciding the function's hyperparameter values. σ_j is multiplied by $k(x, x')$ for each GP prior. This works as normalization of scale variance of each dataset.

part k and its parameters θ . Here we say the functional part k is the function structure but the values of its hyperparameters are not specified yet. k is from a context-free language set yielded from context-free grammar G , which defines the search method while expanding the search tree in the kernel search algorithm. Here G contains base kernels like WN and operators like $+$ and \times . After functional part k is decided, θ that is the hyperparameter vector of the kernel function should be decided. In addition to that, we have parameter vector σ where its length is twice of the number of datasets. Each b_j (additive scaling factor) and v_j (multiplicative scaling factor) in $\sigma = [b_1, v_1, \dots, b_M, v_M]$ is added and multiplied to the kernel function $k(x, x'; \theta)$ before defining a GP prior for each data, like $k' = b^2 + v^2 \times k(x, x'; \theta)$. This parameters normalize of scale variance to deal with the data sets with different scale of bias and variance in target value y . Summarizing all the process gives the following model.

$$k \leftarrow G \quad (3.3)$$

$$\mathcal{GP} \leftarrow k, \theta, \sigma \quad (3.4)$$

$$f \leftarrow \mathcal{GP} \quad (3.5)$$

$$\mathbf{y}, \mathbf{X} \leftarrow f \quad (3.6)$$

Here f is the latent function from GP prior. \mathbf{y} and \mathbf{X} are $N \times 1$ and $N \times D$ matrices where N is the number of data points for each data set and D is the input dimension. For y_i which is the i th element of column vector \mathbf{y} , and \mathbf{x}_i which is the i th row vector of matrix \mathbf{X} , $y_i = f(\mathbf{x}_i)$ holds.

Figure 3.3 shows the whole structure of the model. The GP prior is specified for each data set, from 1 to M . A mean function and kernel function are needed to specify a GP prior. The mean function is zero mean function. The covariance kernel function is specified with the kernel structure and hyperparameters. Scaling factor σ for each data set is multiplied by the kernel function and specifies a GP prior. The GP prior for each data set gives latent function f . From that latent function, we get latent values f_i for each data point. Finally y_i is generated and observed which is generated from the latent value f_i with some noise. So for each data set, here j from 1 to M , we have the following distribution:

$$K_{i,j} = b^2 + v^2 \times k(\mathbf{x}_i, \mathbf{x}_j; \theta) \quad (3.7)$$

$$\mathbf{y} \sim \mathcal{N}(0, K) \quad (3.8)$$

Here $\mathcal{N}(0, K)$ denotes a multivariate normal distribution with 0 mean vector and covariance matrix K .

3.4 Algorithm

The algorithm for the RABCD model is described in algorithm 1. This algorithm finds a single covariance kernel that is shared by multiple data sets. The way of finding the covariance kernel follows the basic ABCD algorithm. However the main difference is that when we evaluate each model candidate, we calculate the negative log marginal likelihood of the whole data as a summation of negative log marginal likelihood of each data set. This is possible because our model shares a covariance kernel function through multiple data sets. This sharing kernel function restricts covariance patterns of multiple

data sets to be the same. However this does not specify any correlation of variables across the different data sets. In other words, if we construct a single covariance matrix for all the data, we get a covariance matrix that has a block diagonal form. Then we can factorize the likelihood $p(\mathcal{D}|\mathcal{M})$, where \mathcal{M} is a model and \mathcal{D} is a set of data, into a product of individual likelihoods $\prod_{d \in \mathcal{D}} p(d|\mathcal{M})$ as those individual likelihoods are independent. As we are dealing with the log likelihood, this can be represented as a summation, $\sum_{d \in \mathcal{D}} \log p(d|\mathcal{M})$. After finding the best kernel expression with optimized hyperparameters, scaling parameter σ is added to the model and optimized again with the hyperparameters of the kernel function. During this optimization process, model tries to normalize the target values between the dataset. Hyperparameters are fine-tuned to work with the learned scaling factor.

Algorithm 1 Learning RABCD

Require: initial kernel k which contains initial parameters θ and σ , data $\mathcal{D} = \{d_1, \dots, d_M\}$, expansion grammar G , maximum depth of search s

```

1: procedure LEARNRABCD
2:   for  $i \in 0 \dots s$  do
3:      $K \leftarrow \text{EXPAND}(k, G)$ 
4:      $K \leftarrow \text{OPTIMIZE}(K, \mathcal{D})$ 
5:      $k \leftarrow \text{argmin}_{k \in K} \text{BIC}(k, \mathcal{D})$ 
6:   end for
7:   return  $k$ 
8: end procedure
9: function OPTIMIZE( $K, \mathcal{D}$ )
10:  for  $k \in K$  do
11:     $\theta \leftarrow \text{argmin}_{\theta} \text{NLML}(k, \mathcal{D})$ 
12:  end for
13:  return  $K$ 
14: end function
15: function BIC( $k, \mathcal{D}$ )
16:  return  $2 \times \text{NLML}(k, \mathcal{D}) + \text{LENGTH}(\theta) \times \ln(|\mathcal{D}|)$ 
17: end function
18: function NLML( $k, \mathcal{D}$ )
19:  return  $\sum_{d \in \mathcal{D}} \text{NLMLINDIVIDUAL}(k, d)$ 
20: end function

```

Chapter IV

Related Work

4.1 Learning Composite Gaussian Process Kernels

A composite GP kernel can greatly improve the performance of nonparametric regression models. Manually constructed composite models have shown to fit accurate GP models [4, 6, 7]. Tree-shaped composite kernels have also been used with Support Vector Machine (SVM) [8, 9].

A weighted sum of base kernels is limited but also effective when building a composite kernel with composite kernels are given. Multiple Kernel Learning (MKL) [10, 11, 12] find optimal weights in polynomial time when the component kernels and parameter are pre-specified.

Recently, an algorithm for learning a composite kernel (CKL) [2], and a system for an automatic explanation of the learned composite kernel (ABCD) [3] have been proposed. Our relational learning (ABCD) is based on the ABCD system. However, our RABCD expands the ABCD for multiple time-series data and improves the interpretability of the individual time-series data by finding a shared relational structure.

4.2 Relational Learning for Continuous Data

Relational GPs can represent the relationships among multiple time-series data sets [1, 13, 14]. These models are based on pre-specified base kernels instead of composite kernel representations. A straightforward extension of relational GP with a composite kernel does not work unless the time-series data sets share the same hyperparameter. Our model is more flexible to handle relational data sets by introducing GPs with both shared and non-shared hyperparameters.

There is a large body of work attempting to represent the probabilistic knowledge formalisms in Statistical Relational Learning (SRL) [15]. SRL models for continuous variables [16, 17, 18] and lifted inference algorithms have been proposed [19]. Unfortunately, learning composite relational models from complex continuous data is still hard, and thus models are defined manually.

Chapter V

Experimental Results

In this section, we will first show that our model can find the true hyperparameter from the data that is generated from the model with our assumption, a shared kernel with shared hyperparameters. Then we will show how our model actually works with real world stock data and house price index data, in comparison with the original ABCD model.

5.1 Learning Hyperparameters on Synthetic Data

First we have a covariance kernel function k that is already learned from the original ABCD algorithm. In composite kernel form, it's represented as $SE \times PER \times LIN + LIN + SE + WN$. We generate 20 data sets from a GP prior with kernel k . Each data set has a different number of data points. After sampling the data, we try to optimize the hyperparameters of k for $1, 2, \dots, 20$ data sets chosen out of 20 sampled data sets, with the original hyperparameters as a starting point. We do this optimization 10 times for each number of data sets, from 1 to 20, with different combination of choice.

We calculate the root mean squared error (norm distance) between the true hyperparameter and the optimized hyperparameter vector. Figure 5.1 shows the result. The values along the horizontal axis means the number of data sets which are used in the optimization. Each boxplot shows the distribution of the error with 10 different results for each number of data sets used. It shows that as we increase the number of data sets in parameter optimization, the error between the optimized and the true hyperparameter decreases.

5.2 Stock Market Data

We ran our RABCD algorithm on US stock mark data. We select the 9 most valuable stocks, GE, MSFT, XOM, PFE, C, WMT, INTC, BP and AIG based on the market capitalization rank in 2001 [20]. The adjusted closes of stocks from 2001-05-29 to 2001-12-25 were collected from Yahoo finance [21]. Each stock's historical adjusted close in that period consists of 129 points. Thus, the whole number of points is $129 \times 9 = 1161$ points. The period that we collected data from includes September 11 attack. In terms of the general shape of the data the time series signals show similar long-term change through the whole period. After the 9/11 attacks, all the stock values show a steep drop but gradually recovers as time goes on.

Table 5.1 compares the fitness of the ABCD and RABCD models. The results with line heading RABCD are from our model. The others are from the original ABCD. The results with line heading Total is calculated by considering the whole GP priors that are learned for each data set as a single model. So NLL, N, P values of individual models are added up and the total BIC of the ABCD model as

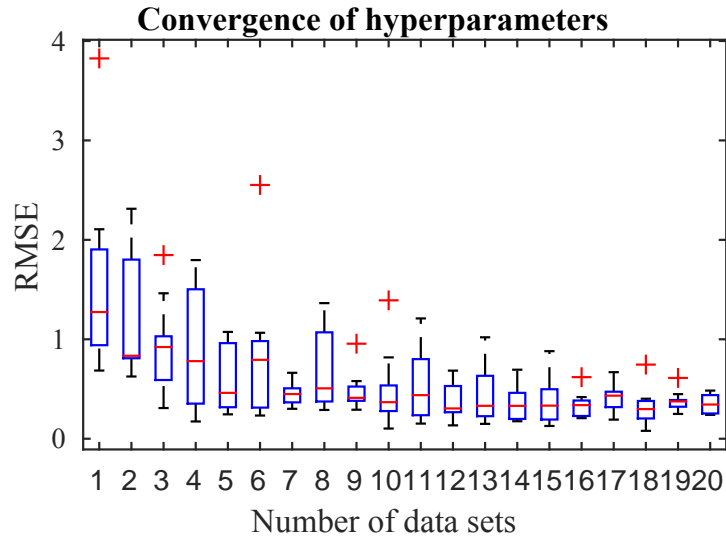


Figure 5.1: The convergence of error between the optimized hyperparameter and original true parameter that was used when generating data. The horizontal axis denotes the number of data sets used in hyperparameter optimization. The vertical axis shows the value of root mean squared error between optimized and true hyperparameters. Each boxplot shows the distribution of errors for each step.

Model-Stock		NLL	N	P	BIC
ABCD	GE	116.36	129	7	266.75
	MSFT	111.17	129	6	251.51
	XOM	82.24	129	9	208.23
	PFE	62.91	129	12	184.14
	C	424.23	129	10	897.07
	WMT	149.96	129	5	324.23
	Total	946.89	774	49	2219.71
RABCD-Total		1001.04	774	21	2141.76

Table 5.1: The experimental comparisons of GP models with individual ABCD and RABCD in the stock data set (top 6 US stocks in 2001). For each column, NLL means the negative log likelihood, N is the number of data points, P is the number of hyperparameters and BIC is the Bayesian information criterion. ABCD is slightly better in a simple aggregation of NLL only. However, our RABCD outperforms ABCD in the BIC due to a significantly lower 21 parameters (P) compared to 49 parameters in ABCD.

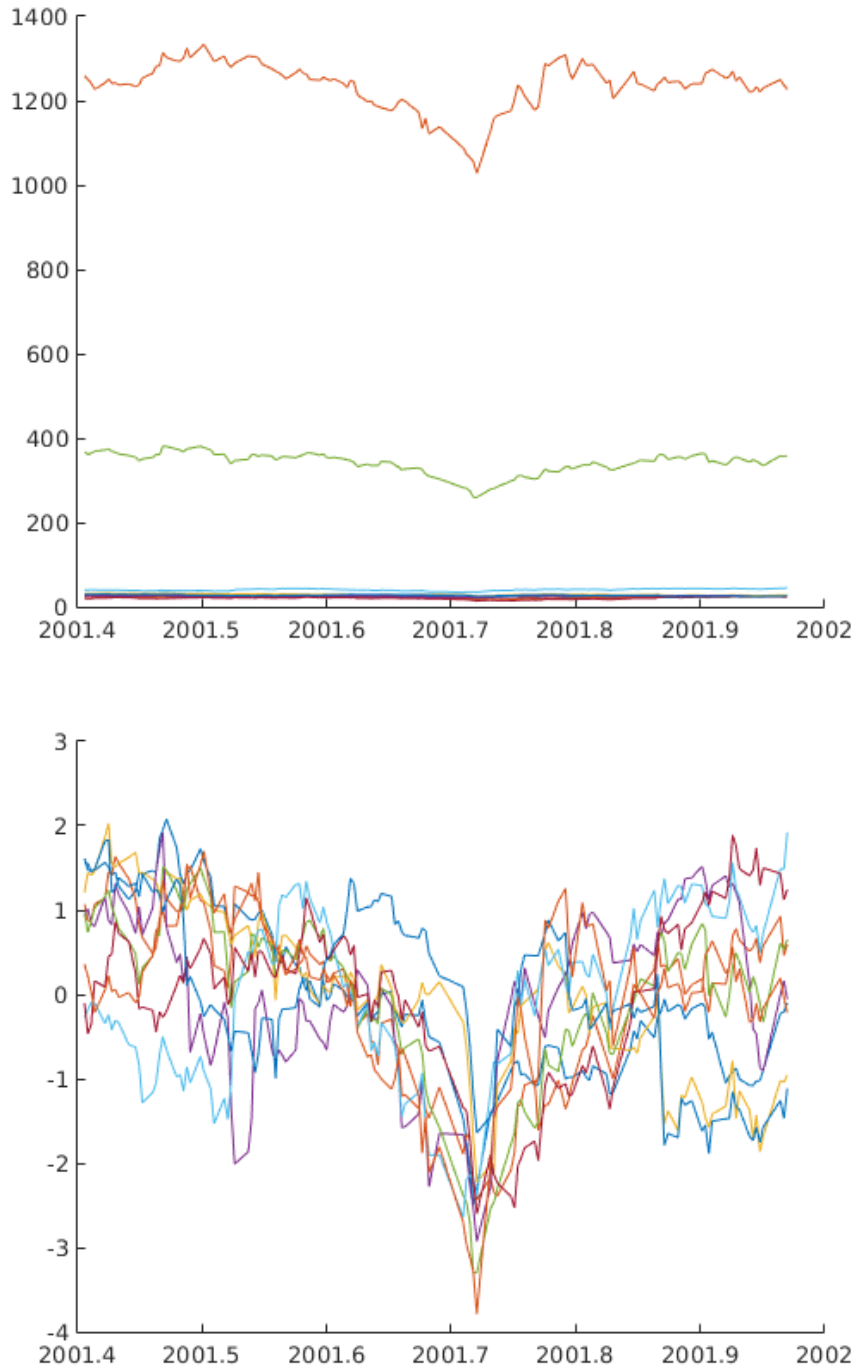


Figure 5.2: Plotting shape of total 9 number of US stock data. Upper plot is original data and lower plot is normalized data.

SET	N	RABCD		ABCD	
		P	BIC	P	BIC
Top 3 stocks	387	13	686.05	22	750.65
Top 6 stocks	774	21	2141.76	49	2219.71
Top 9 stocks	1161	38	4167.40	73	3985.03

Table 5.2: The BIC of ABCD and RABCD in the stock data set. ‘Top 3’, ‘Top 6’ and ‘Top 9’ stocks were selected by their market capitalization ranks in 2011. As shown in Table 5.1, RABCD requires fewer parameters than ABCD. RABCD models trained with 3 stocks and 6 stocks show better performance than individually optimized ABCD models. When 9 stocks are considered, the individual ABCD models show better performance than the single (shared) RABCD model.

a single model was calculated using those values. In terms of negative log likelihood, applying ABCD individually gives better results. However if we compare the BIC, our model achieves better results as our model has a reduced number of free parameters by sharing them through the data sets.

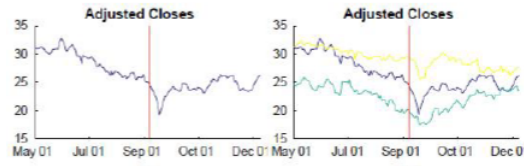
Table 5.2 compares the fitness of the models with different numbers of data sets used in training. As the number of data sets increases, the number of parameters needed also increases. However RABCD shows less need of parameters through the whole setup since our model shares parameters through the data. RABCD also performs well in terms of the BIC for TOP3 and TOP6. However as the number of data sets increases, the performance of the original ABCD model surpasses our model, at TOP9. This is because our model fits the general structure but not individual specific ones. As the number of data sets increases, the learned structure cannot fully explain the individual specific patterns. And this accumulated errors in fitness, which is the NLL, offsets the advantages of the BIC which is from the reduced number of parameters.

5.3 Housing Market Data

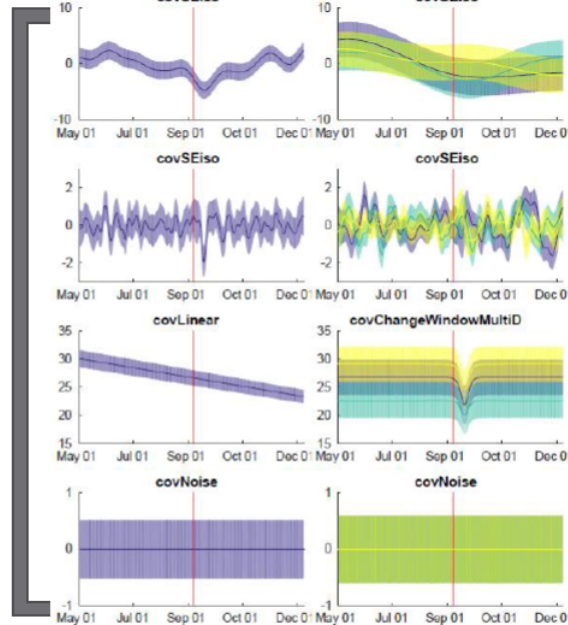
We applied our model to housing market data as a second experiment on real world data. US house price index data were retrieved from S&P Dow Jones Indices [22]. We selected 6 cities: New York, Los Angeles, Chicago, Pheonix, San Diego and San Francisco. These cities were selected based on the US city population rank [23]. Those cities were the top 6 cities in terms of their population, among the cities that had house price index data. The indices are seasonally adjusted. The period is from the start of year 2004 to the end of year 2013 with monthly granularity, with total 120 points for each data set and 720 for all the data sets. In terms of the shape of the data, there are smooth peaks around 2007 and drops until 2009. There is only small change until 2012 and after that they monotonically increases. The experimental results are quite similar to the one with stock values.

Table 5.3 shows the fitness of models between our model and the ABCD model. Similar to the stock market data case, the first line is our model and the others are from the ABCD. Our model shows better results, reduced number of free parameters and smaller BIC compared to Total case. However the

Input data



Components



Detecting changepoint

Figure 5.3: Picture that shows expressive power of RABCD by finding the sudden decrease in the most of stocks after 9/11.

Model-City		NLL	N	P	BIC
ABCD	New York	120.13	120	10	288.13
	Los Angeles	162.94	120	9	368.96
	Chicago	134.73	120	13	331.69
	Pheonix	101.76	120	11	256.17
	San Diego	155.64	120	12	368.73
	San Francisco	174.45	120	6	377.63
	Total	849.64	720	61	2100.62
RABCD-Total		910.63	720	23	1972.58

Table 5.3: A comparison of BIC between the GP model from individual ABCD and the GP model from RABCD, with house data. For each column, NLL means negative log likelihood, N is the number of data points, P is the number of hyperparameters and BIC is the Bayesian information criterion. For each row, RABCD is the result from our model. From New York to San Francisco, those results are from individual ABCD. Total is summation of those individual results.

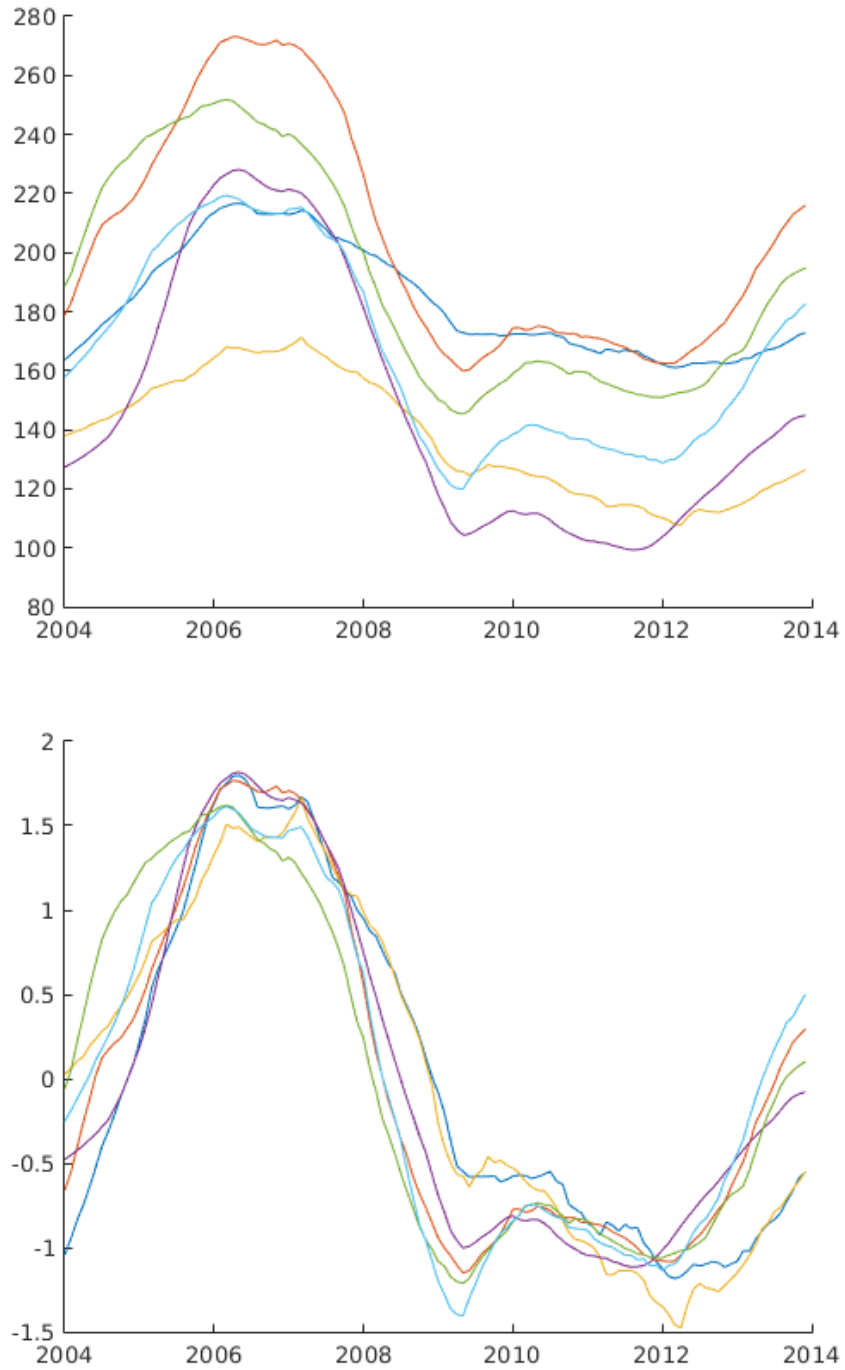


Figure 5.4: Plotting shape of total 6 number of US house price index data. Upper plot is original data and lower plot is normalized data.

SET	N	RABCD		ABCD	
		P	BIC	P	BIC
Top 2 cities	240	12	663.54	20	634.00
Top 4 cities	480	14	1260.05	38	1424.18
Top 6 cities	720	23	1972.58	61	2100.62

Table 5.4: The BIC of ABCD and RABCD in the housing market data set. ‘Top 2’, ‘Top 4’ and ‘Top 6’ US cities were selected in terms of their city population rank. The BICs of the RABCD models are similar or better than the BICs of individually trained ABCD models.

original model still shows better fitness if we only consider NLL. From this experiment we can again confirm that the major contribution of the smaller BIC comes from the reduce number of parameters.

Table 5.4 compares results between our model and the ABCD for different number of data sets. Our model shows reduced number of parameters over all different sets of data since our model shares kernel parameters for multiple data sets. The original model shows better results in terms of BIC for the top 2 indices case. However other than that, our model shows better results in terms of both number of parameters and BIC.

Chapter VI

Conclusion

We proposed a nonparametric Bayesian framework that finds shared structure throughout the multiple sets of data. The resulting Relational Automatic Bayesian Covariance Discovery (RABCD) method provide a way of finding a shared kernel function that can describe multiple data with better BIC. We applied this model to a synthetic data and several real world data, including US stock data and US house price index data, to validate our approach. While this paper assumes conditional independence between the data sets, this Relational Automatic Bayesian Covariance Discovery framework can be generalized in the form of Infinite Tucker Decomposition [24], which is generalization of tensor decomposition with infinite size of core tensor. We hope this is the right direction for the next research topic.

References

- [1] W. Chu, V. Sindhwani, Z. Ghahramani, and S. S. Keerthi, “Relational learning with gaussian processes,” in *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems (NIPS)*, 2006, pp. 289–296.
- [2] D. K. Duvenaud, J. R. Lloyd, R. B. Grosse, J. B. Tenenbaum, and Z. Ghahramani, “Structure discovery in nonparametric regression through compositional kernel search,” in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013, pp. 1166–1174.
- [3] J. R. Lloyd, D. K. Duvenaud, R. B. Grosse, J. B. Tenenbaum, and Z. Ghahramani, “Automatic construction and natural-language description of nonparametric regression models,” in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, 2014, pp. 1242–1250.
- [4] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006.
- [5] G. Schwarz, “Estimating the dimension of a model,” *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.
- [6] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig, “Nonparametric dynamics estimation for time periodic systems,” in *the 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2013, pp. 486–493.
- [7] J. R. Lloyd, “Gefcom2012 hierarchical load forecasting: Gradient boosting machines and gaussian processes,” *International Journal of Forecasting*, vol. 30, no. 2, pp. 369–374, 2014.
- [8] L. Diosan, A. Rogozan, and J. Pécuchet, “Evolving kernel functions for svms by genetic programming,” in *The Sixth International Conference on Machine Learning and Applications (ICMLA)*, 2007, pp. 19–24.
- [9] W. Bing, Z. Wen-qiong, C. Ling, and L. Jia-hong, “A gp-based kernel construction and optimization method for rvm,” in *Computer and Automation Engineering (ICCAE)*, vol. 4, 2010, pp. 419–423.
- [10] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, “Multiple kernel learning, conic duality, and the SMO algorithm,” in *Proceedings of the Twenty-first International Conference on Machine Learning (ICML)*, 2004.
- [11] J. Q. Candela and C. E. Rasmussen, “A unifying view of sparse approximate gaussian process regression,” *Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, 2005.
- [12] A. G. Wilson and R. P. Adams, “Gaussian process kernels for pattern discovery and extrapolation,” in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013, pp. 1067–1075.

- [13] Z. Xu, K. Kersting, and V. Tresp, “Multi-relational learning with gaussian processes,” in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, 2009, pp. 1309–1314.
- [14] A. G. Wilson and Z. Ghahramani, “Generalised wishart processes,” in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011, pp. 736–744.
- [15] L. Getoor and B. Taskar, *Introduction to statistical relational learning*. MIT press, 2007.
- [16] J. Wang and P. M. Domingos, “Hybrid markov logic networks,” in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI)*, 2008, pp. 1106–1111.
- [17] J. Choi, E. Amir, and D. J. Hill, “Lifted inference for relational continuous models,” in *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010, pp. 126–134.
- [18] V. Belle, A. Passerini, and G. Van den Broeck, “Probabilistic inference in hybrid domains by weighted model integration,” in *Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.
- [19] A. Kimmig, L. Mihalkova, and L. Getoor, “Lifted graphical models: a survey,” *Machine Learning*, vol. 99, no. 1, pp. 1–45, 2015.
- [20] T. von Alten, “Top 100 market capitalization,” <http://fortboise.org/top100mktcap.html>, 2001, accessed: 2015-09-15.
- [21] Yahoo Inc., “Yahoo finance - business finance, stock market, quotes, news,” <http://finance.yahoo.com/>, 2015, accessed: 2015-09-15.
- [22] D. Guarino and D. Blitzer, “S&P/Case-Shiller home price tiered index levels,” <https://us.spindices.com/indices/real-estate/sp-case-shiller-20-city-composite-home-price-index>, 2015, accessed: 2015-09-15.
- [23] United States Census Bureau, “Population estimates: Historical data,” <https://www.census.gov/popest/data/historical/index.html>, 2014, accessed: 2015-09-15.
- [24] Z. Xu, F. Yan *et al.*, “Infinite tucker decomposition: Nonparametric bayesian models for multiway data analysis,” *arXiv preprint arXiv:1108.6296*, 2011.

Acknowledgements

I would like to express my gratitude to my supervisor Jaesik Choi for introducing me to the topic as well for the support on the way. Useful comments, remarks and engagement through the learning process of this master thesis were greatly helpful for me to successfully finish my graduate program. Also, I'd like to thank the lab members of Statistical Artificial Intelligence Lab, who gave me many helpful comments and opinions via constructive discussions. Lastly, I would like to thank my loved ones, who have supported me throughout entire process, both by keeping me harmonious and helping me putting pieces together.