VARYING-COEFFICIENT STOCHASTIC DIFFUSION PROCESSES WITH DEEP LEARNING

by

Zihui Wang

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Applied Mathematics

Charlotte

2025

Approved by:

Dr. Jiancheng Jiang

Dr. Qingning Zhou

Dr. Yinghao Pan

Dr. Jake Smithwick

ABSTRACT

ZIHUI WANG. Varying-Coefficient Stochastic Diffusion Processes with Deep Learning. (Under the direction of DR. JIANCHENG JIANG)

Varying-coefficient stochastic diffusion processes provide a flexible way to capture time-varying dynamics in time series. These models allow both the drift and diffusion terms in the stochastic differential equation to evolve over time, reflecting changing market conditions. In this dissertation, we explore different methodologies for modeling time series: nonparametric estimation of varying-coefficient SDE, deep learning for multivariate time series, and high-dimensional regression with LASSO.

For the univariate case, we estimate time-varying drift and diffusion functions using local regression. We establish the asymptotic properties of the model estimators under regularity conditions.

To extend the analysis to multivariate settings, we use a regression-based structure to model relationships among interdependent financial variables and make prediction. Deep learning neural networks are applied to capture nonlinear dependencies and temporal dynamics across multiple series.

As the number of variables increases, the high dimensionality poses challenges for estimation and interpretation. To address this, we incorporate LASSO regularization for variable selection and dimensionality reduction. We investigate the selection consistency of LASSO in high-dimensional time series regression when predictors may include both stationary and nonstationary components.

DEDICATION

To my parents, for their unwavering love and support.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Jiancheng Jiang. Without his expert advice and constructive feedback, I would not have been able to complete this dissertation. His passion for Statistics and dedication to his students have been a constant source of inspiration for me.

I would also like to sincerely thank my committee members, Dr. Qingning Zhou, Dr. Yinghao Pan, and Dr. Jake Smithwick, for their thoughtful feedback, insightful suggestions, and their unwavering support. Their expertise and dedication have helped shape the direction of this research, and I have greatly benefited from their diverse perspectives.

Additionally, I am deeply grateful to Dr. Shaozhong Deng, Dr. Xingjie Li, Dr. Wenyu Gao and Dr. Judith Krauss for their support in both my professional and personal development.

Finally, I would like to acknowledge the financial support I received from UNC Charlotte Graduate School and the Department of Mathematics and Statistics, which allowed me to focus on my research.

TABLE OF CONTENTS

LIST OF TABLES viii

LIST OF FIGURES ix

CHAPTER 1: Introduction 1

1.1. Motivation 1

1.2. Outline of the Dissertation 3

CHAPTER 2: Varying-Coefficient Stochastic Diffusion Model 4

2.1. Model Specification and Notation 4

2.2. Estimating the Functions $\alpha_0(t)$ and $\alpha_1(t)$ 5

2.3. Estimating the Functions $\beta_0(t)$ and $\beta_1(t)$ 6

2.4. Bandwidth Selection 7

2.4.1. Bandwidth Selection for Drift Coefficient Estimation 7

2.4.2. Bandwidth Selection for Volatility Coefficient Estimation 8

CHAPTER 3: Simulations 9

CHAPTER 4: Asymptotic Properties 13

4.1. Notation and Preliminaries 13

4.2. Asymptotic Properties 16

4.2.1. Drift Coefficient Estimator $\widehat{\alpha}(t)$ 16

4.2.2. Volatility Coefficient Estimator $\widehat{\theta}(t)$ 16

CHAPTER 5: Modeling Dependencies Among Financial Time Series Using Deep Learning Regression Frameworks 18

5.1. Introduction and Motivation 18

5.2. Data and Variable Structure 19

5.3. Deep Learning Models 20

5.3.1. Long Short-Term Memory (LSTM) 20

5.3.2. Gated Recurrent Unit (GRU) 23

5.3.3. Convolutional Neural Network (CNN) 26

5.3.4. CNN–LSTM Hybrid Model 28

5.4. Result Evaluation 31

CHAPTER 6: High-Dimensional Time Series Regression and Variable Selection 33

6.1. Introduction and Motivation 33

6.2. Model Setup 34

6.3. Tuning Parameter Selection 35

6.4. Simulation 37

CHAPTER 7: Discussion 43

APPENDIX A: Regularity Conditions 49

APPENDIX B: Proof of Theorem in Section 4.2 51

LIST OF TABLES

TABLE 5.1: Forecasting Performance on S&P 500 Data 31

TABLE 5.2: Ranking of Models 32

TABLE 6.1: Simulation results for Example 6.1: Independent predictors with mixed stationarity ($N = 200$). LASSO performance with λ ranging from 0.06 to 0.17 and cross-validated λ_{cv}. 39

TABLE 6.2: Simulation results for Example 1: Independent predictors with mixed stationarity ($N = 400$). LASSO performance with λ ranging from 0.06 to 0.17 and cross-validated λ_{cv}. 39

TABLE 6.3: Simulation results for Example 6.2: Correlated predictors with mixed stationarity ($N = 200$). LASSO performance under varying correlation levels ρ with λ fixed at 0.1. 41

TABLE 6.4: Simulation results for Example 6.2: Correlated predictors with mixed stationarity ($N = 400$). LASSO performance under varying correlation levels ρ with λ fixed at 0.1. 41

LIST OF FIGURES

FIGURE 3.1: Pointwise 95% envelope (black band), median (red), and true value (blue) for the estimators of $\alpha_0(t)$, $\alpha_1(t)$, $\beta_0(t)$, and $\beta_1(t)$ under the time-homogeneous model in Example 3.1. 10

FIGURE 3.2: Pointwise 95% envelope (black band), median (red), and true function (blue) for the estimators of $\alpha_0(t)$, $\alpha_1(t)$, $\beta_0(t)$, and $\beta_1(t)$ under the varying-coefficient model in Example 3.2. 12

FIGURE 5.1: Long Short-Term Memory (LSTM) network structure. 21

FIGURE 5.2: Gated Recurrent Unit (GRU) network structure 24

FIGURE 5.3: 1D-CNN structure. 26

FIGURE 5.4: CNN–LSTM structure 28

FIGURE 6.1: Rolling-origin cross-validation 36

CHAPTER 1: Introduction

1.1 Motivation

Understanding and forecasting the behavior of financial time series is a central concern in financial econometrics. Early modeling approaches often relied on stochastic differential equations (SDEs) with constant parameters. The geometric Brownian motion model introduced by Black and Scholes [6] is a classic example:

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t, \tag{1.1}$$

where μ and σ are constant drift and volatility terms, and W_t denotes standard Brownian motion. This model forms the foundation for much of modern option pricing theory. However, it assumes that financial processes are stationary and homoskedastic, which empirical data rarely supports.

Researchers have since developed models that relax these assumptions. Stanton [33] proposed a nonparametric approach to estimate drift and diffusion functions of interest rates using discrete-time data. His empirical findings showed significant nonlinearities, especially in the drift of short-term rates, highlighting the limitations of parametric specifications.

Fan and Zhang [13] extended this direction with a rigorous nonparametric framework for estimating time-homogeneous diffusion processes:

$$dX_t = \mu(X_t) \, dt + \sigma(X_t) \, dW_t. \tag{1.2}$$

Their method uses local regression to estimate the drift and diffusion functions without assuming a parametric form. They also introduced diagnostic tools to test the adequacy of existing models. This work builds directly on the foundational contributions of Fan and Gijbels [11], whose monograph on local polynomial modeling established the theoretical underpinnings for kernel-based estimation in time series and diffusion processes.

To accommodate structural breaks and time-varying features in financial data, Fan et al. [14] introduced a semiparametric time-dependent diffusion model:

$$dX_t = \{\alpha_0(t) + \alpha_1(t)X_t\}\, dt + \beta_0(t)X_t^{\beta_1(t)} dW_t, \tag{1.3}$$

where all parameters are allowed to evolve smoothly over time.

While these approaches have improved our understanding of univariate financial dynamics, they often fall short in multivariate settings. Financial markets are composed of many interconnected time series whose interactions evolve over time. Modeling these high-dimensional dependencies is challenging for traditional parametric or even nonparametric models.

This has led to increasing interest in machine learning methods, particularly deep learning. Long Short-Term Memory (LSTM) networks, introduced by Hochreiter and Schmidhuber [21], and Gated Recurrent Units (GRUs), proposed by Cho et al. [10], are specifically designed to capture long-range dependencies in sequential data and have demonstrated strong performance in financial applications [15]. To further enhance forecasting accuracy, hybrid architectures such as CNN-LSTM models have also been developed and applied successfully in financial time series prediction [7, 25].

However, deep learning models can be difficult to interpret and may overfit the data, especially in complex or high-dimensional settings. To address these issues, regularization techniques like LASSO Tibshirani [36] have been used. LASSO performs both variable selection and shrinkage, making it a valuable tool for analyzing time series data. Recent theoretical developments have investigated its performance in more complex environments, including scenarios involving both stationary and nonstationary predictors, with a focus on its consistency in variable selection.

This dissertation unifies these methods—nonparametric diffusion modeling, deep neural networks, and regularized high-dimensional regression—into a framework for flexible financial forecasting.

1.2 Outline of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 introduces the methodological framework, including the formulation of the varying-coefficient stochastic diffusion model and the estimation method for the drift and volatility components. Chapter 3 presents the simulation results, evaluating the performance of the proposed framework on both simulated and real-world financial datasets. In Chapter 4, we investigate the asymptotic properties of model estimators. In Chapter 5, the deep learning modeling framework is introduced. In Chapter 6, we investigate the LASSO variable selection consistency in high dimensional time series regression model. Concluding remarks are presented in Chapter 7. Proofs of the main results are given in the Appendix.

CHAPTER 2: Varying-Coefficient Stochastic Diffusion Model

In this chapter, we introduce the varying-coefficient stochastic diffusion model. Starting by introducing the model and the necessary notation. Then focusing on the estimation procedure of the time-varying drift and diffusion components of the model inspired by Fan et al. [14].

2.1 Model Specification and Notation

Let $\{X_t, t \geq 0\}$ denote a continuous-time stochastic process observed at discrete time points over a finite time interval. The process evolves according to the following stochastic differential equation:

$$dX_t = \{\alpha_0(t) + \alpha_1(t)X_t\}\, dt + \beta_0(t)X_t^{\beta_1(t)}\, dW_t, \tag{2.1}$$

where W_t is a standard Brownian motion. The functions $\alpha_0(t)$ and $\alpha_1(t)$ define the drift component, capturing the instantaneous expected change in the process, while $\beta_0(t)$ and $\beta_1(t)$ control the diffusion component, representing the volatility structure of the system. Unlike fixed-parameter models, all four coefficients are allowed to vary smoothly with time, providing the model with the flexibility to capture temporal changes in both the trend and volatility.

Several well-known models can be viewed as special cases of this general specification. For instance, when $\alpha_0(t)$, $\alpha_1(t)$, and $\beta_0(t)$ are constant, and $\beta_1(t) = 1/2$, the model reduces to the CIR process. If $\beta_1(t) = 1$ and other coefficients are constant, we recover the GBM process used in the Black-Scholes model.

Assume that the process X_t is observed at discrete, equally spaced time points $\{t_i = i\Delta, i = 0, 1, \ldots, n\}$, with Δ representing the sampling interval.

2.2 Estimating the Functions $\alpha_0(t)$ and $\alpha_1(t)$

The drift coefficients $\alpha_0(t)$ and $\alpha_1(t)$ are estimated following the local regression method Fan and Gijbels [11].

The process X_t follows the stochastic differential equation:

$$dX_t = \{\alpha_0(t) + \alpha_1(t)X_t\}\, dt + \beta_0(t)X_t^{\beta_1(t)}\, dW_t, \tag{2.2}$$

where $\alpha_0(t)$ and $\alpha_1(t)$ are smooth, time-varying drift terms, $\beta_0(t)$ and $\beta_1(t)$ govern the volatility structure, and W_t is standard Brownian motion.

In practical settings, X_t is not observed continuously but at a set of discrete time points $t_1 < t_2 < \cdots < t_{n+1}$. Define $\Delta_i = t_{i+1} - t_i$ and the observed increments:

$$Y_{t_i} := X_{t_{i+1}} - X_{t_i}, \quad i = 1, \ldots, n.$$

Applying a first-order Euler approximation, the continuous-time model can be discretized as:

$$Y_{t_i} \approx \{\alpha_0(t_i) + \alpha_1(t_i)X_{t_i}\}\Delta_i + \beta_0(t_i)X_{t_i}^{\beta_1(t_i)}\sqrt{\Delta_i}\,\varepsilon_{t_i}, \tag{2.3}$$

where $\varepsilon_{t_i} \sim \mathcal{N}(0,1)$ are independent standard normal variables. This approximation allows the drift function to be estimated from discrete data.

Assuming the drift functions are locally constant, they can then be approximated around a target time t_0 as:

$$\alpha_0(t) \approx \alpha_0(t_0), \quad \alpha_1(t) \approx \alpha_1(t_0), \quad \text{for } t \in [t_0 - h, t_0],$$

where $h > 0$ is a bandwidth parameter defining the local neighborhood.

The drift coefficients are estimated by solving the following kernel-weighted least squares problem:

$$\min_{a,b} \sum_{i=1}^{n} \left(\frac{Y_{t_i}}{\Delta_i} - a - bX_{t_i} \right)^2 K_h(t_i - t_0), \tag{2.4}$$

where $K_h(u) = \frac{1}{h} K\left(\frac{u}{h}\right)$ is a kernel function scaled by the bandwidth h. The minimizers $\hat{a}(t_0)$ and $\hat{b}(t_0)$ yield the drift estimates:

$$\hat{\alpha}_0(t_0) = \hat{a}(t_0), \quad \hat{\alpha}_1(t_0) = \hat{b}(t_0).$$

By evaluating this regression at each t_0 over a dense grid, we obtain smooth, time-varying estimates of the drift functions $\alpha_0(t)$ and $\alpha_1(t)$.

2.3 Estimating the Functions $\beta_0(t)$ and $\beta_1(t)$

Following the estimation of the drift component, we now turn to estimating the time-varying volatility function in the varying-coefficient diffusion model. Consider again the discretized version of the model:

$$Y_{t_i} \approx \mu(t_i, X_{t_i})\Delta_i + \sigma(t_i, X_{t_i})\sqrt{\Delta_i}\, \varepsilon_{t_i}, \tag{2.5}$$

where $\varepsilon_{t_i} \sim \mathcal{N}(0, 1)$, and $\sigma(t, X_t)$ is specified as

$$\sigma(t, X_t) = \beta_0(t) X_t^{\beta_1(t)}.$$

Given the estimated drift $\widehat{\mu}(t_i, X_{t_i}) = \widehat{\alpha}_0(t_i) + \widehat{\alpha}_1(t_i) X_{t_i}$, the standardized residuals are defined as:

$$\widehat{E}_{t_i} = \frac{Y_{t_i} - \widehat{\mu}(t_i, X_{t_i})\Delta_i}{\sqrt{\Delta_i}}. \tag{2.6}$$

These residuals approximate the volatility component:

$$\widehat{E}_{t_i} \approx \beta_0(t_i) X_{t_i}^{\beta_1(t_i)} \varepsilon_{t_i}.$$

To estimate $\beta_0(t)$ and $\beta_1(t)$, we employ a local pseudo-likelihood approach. The conditional log-likelihood of $\widehat{E}_{t_i}$, up to an additive constant, is approximated as:

$$\ell(\beta_0, \beta_1; t_0) = -\frac{1}{2}\sum_{i=1}^{n} K_h(t_i - t_0)\left\{\log\left(\beta_0^2 X_{t_i}^{2\beta_1}\right) + \frac{\widehat{E}_{t_i}^2}{\beta_0^2 X_{t_i}^{2\beta_1}}\right\}, \tag{2.7}$$

where $K_h(\cdot)$ is a kernel function with bandwidth h, centered at t_0.

Maximizing this criterion yields the local estimators $\widehat{\beta}_0(t_0)$ and $\widehat{\beta}_1(t_0)$. For fixed β_1, the optimal value of β_0^2 admits the closed-form solution:

$$\widehat{\beta}_0^2(t_0; \beta_1) = \frac{\sum_{i=1}^{n} K_h(t_i - t_0)\widehat{E}_{t_i}^2 |X_{t_i}|^{-2\beta_1}}{\sum_{i=1}^{n} K_h(t_i - t_0)}. \tag{2.8}$$

Hence, the estimation reduces to a univariate optimization over β_1, which simplifies the computation significantly. Once $\widehat{\beta}_1(t_0)$ is obtained, it is substituted into the above expression to compute $\widehat{\beta}_0(t_0)$.

2.4 Bandwidth Selection

In nonparametric estimation of time-varying drift and diffusion functions, the choice of bandwidth plays a central role in determining the accuracy and stability of the estimators. This section discusses the bandwidth selection strategies for both the drift and volatility coefficient estimation procedures.

2.4.1 Bandwidth Selection for Drift Coefficient Estimation

For estimating the drift function $\mu(t, X_t) = \alpha_0(t) + \alpha_1(t)X_t$, we use a one-sided kernel that respects the causal structure of financial time series, relying only on information available up to time t_0. The bandwidth parameter h determines the extent of the local neighborhood over which the regression is performed.

The selection of the optimal bandwidth is based on the Average Prediction Error(APE) criterion, which evaluates the out-of-sample prediction performance at a sequence of evaluation points $\{t_1^*, t_2^*, \ldots, t_m^*\}$. Specifically, the APE is defined as:

$$\mathrm{APE}(h) = \frac{1}{m}\sum_{i=1}^{m} \frac{(Y_{t_i^*} - \widehat{Y}_{t_i^*})^2}{\hat{\sigma}^2_{t_i^*}}, \tag{2.9}$$

where $Y_{t_i^*} = X_{t_i^*+\Delta} - X_{t_i^*}$ denotes the observed return, $\widehat{Y}_{t_i^*} = \{\widehat{\alpha}_0(t_i^*) + \widehat{\alpha}_1(t_i^*)X_{t_i^*}\}\Delta$ is the predicted value from the estimated drift, and

$$\hat{\sigma}^2_{t_i^*} = \widehat{\beta}_0^2(t_i^*) X_{t_i^*}^{2\widehat{\beta}_1(t_i^*)}$$

is the estimated conditional variance. The optimal bandwidth is the value of h that minimizes the APE.

2.4.2 Bandwidth Selection for Volatility Coefficient Estimation

For volatility estimation, we adopt a different criterion with respect to the residual-based pseudo-likelihood approach. Given the residuals from the drift estimation:

$$\widehat{E}_{t_i} = \frac{Y_{t_i} - \widehat{\mu}(t_i, X_{t_i})\Delta_i}{\sqrt{\Delta_i}},$$

the local pseudo-likelihood function at t_0 is given by:

$$\mathcal{L}(h) = -\frac{1}{2}\sum_{i=1}^{m} \left\{ \log\left(\widehat{\beta}_0^2(t_i^*) X_{t_i^*}^{2\widehat{\beta}_1(t_i^*)}\right) + \frac{\widehat{E}^2_{t_i^*}}{\widehat{\beta}_0^2(t_i^*) X_{t_i^*}^{2\widehat{\beta}_1(t_i^*)}} \right\}, \tag{2.10}$$

where each $\widehat{\beta}_0(t_i^*)$ and $\widehat{\beta}_1(t_i^*)$ are computed via local pseudo-likelihood maximization. The optimal bandwidth for volatility is chosen as the value of h that maximizes this criterion.

CHAPTER 3: Simulations

To evaluate the performance of the nonparametric estimators for the varying coefficient stochastic diffusion model, we conduct a series of simulation studies. We consider two primary scenarios: 1. A time-homogeneous model where the true coefficient functions are constant over time. 2. A varying-coefficient model in which the coefficients vary smoothly with time.

Example 3.1

We begin by considering a time-homogeneous stochastic diffusion model:

$$dX_t = \{0.0408 - 0.5921X_t\}\,dt + \sqrt{1.6704}X_t^{1.4999}dW_t, \tag{3.1}$$

where W_t is a standard Brownian motion. This model is adopted from Chan et al. [9]. The true parameter values are constant over time, which allows us to examine whether the proposed time-varying estimation framework can correctly identify the absence of time dependence.

We simulate 1578 weekly observations corresponding to the time span from January 5, 1970, to March 31, 2000. The initial value is set to $X_0 = 0.05$.

To evaluate estimator performance, we replicate the simulation 400 times. In each replication, we estimate the drift and diffusion functions using the procedures introduced in Sections 2.2 and 2.3. Specifically, the drift coefficients $\alpha_0(t)$ and $\alpha_1(t)$ are estimated using local constant kernel regression with one-sided kernels, while the volatility parameters $\beta_0(t)$ and $\beta_1(t)$ are estimated by maximizing a localized pseudo-likelihood. The bandwidths for the kernel smoothing are selected based on the average prediction error for the drift and maximum pseudo-likelihood for the diffusion.

For each coefficient function, we construct the pointwise median along with the 2.5th and 97.5th percentiles across the 400 simulations, thus forming a pointwise 95% envelope. Figure 3.1 presents the estimated coefficient functions with the true values.

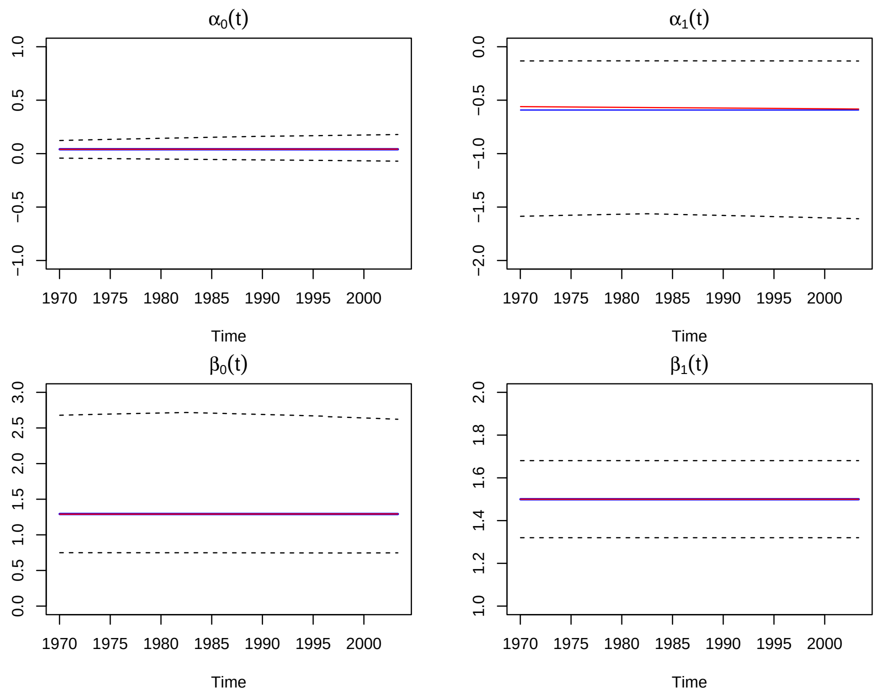

Figure 3.1: Pointwise 95% envelope (black band), median (red), and true value (blue) for the estimators of $\alpha_0(t)$, $\alpha_1(t)$, $\beta_0(t)$, and $\beta_1(t)$ under the time-homogeneous model in Example 3.1.

The simulation results for the constant-coefficient case confirm that the estimators perform well under time-homogeneous conditions. The estimated functions closely align with the true constant values, and the confidence bands remain narrow and stable, reflecting low variability and high estimation precision. These findings provide reassuring evidence that the method remains reliable when applied to simpler, stationary settings.

Example 3.2

Next, we consider a varying-coefficient stochastic diffusion model of the form:

$$dX_t = \{\alpha_0(t) + \alpha_1(t)X_t\}\, dt + \beta_0(t)X_t^{\beta_1(t)} dW_t, \tag{3.2}$$

where the coefficient functions $\alpha_0(t)$, $\alpha_1(t)$, $\beta_0(t)$, and $\beta_1(t)$ are assumed to vary smoothly over time.

To simulate this process, we first fit the model in Equation (3.2) to a real dataset consisting of 1618 weekly observations of U.S. Treasury bill yields from January 4, 1980 to December 31, 2010. Using the estimation methods, we obtain the estimates $\tilde{\alpha}_0(t)$, $\tilde{\alpha}_1(t)$, $\tilde{\beta}_0(t)$, and $\tilde{\beta}_1(t)$ from the real data. These estimated coefficient functions are then treated as the true underlying functions and used to generate synthetic data from the model in Equation (3.2).

A total of 400 independent replications are generated. For each replication, we apply the estimation procedures described in Sections 2.2 and 2.3 to obtain the estimators $\widehat{\alpha}_0(t)$, $\widehat{\alpha}_1(t)$, $\widehat{\beta}_0(t)$, and $\widehat{\beta}_1(t)$.

As in Example 3.1, we construct pointwise 95% envelopes from the empirical distribution of the estimators across simulations, using the 2.5th and 97.5th percentiles. Figure 3.2 presents the estimated coefficient functions along with their envelopes and the true time-varying functions.

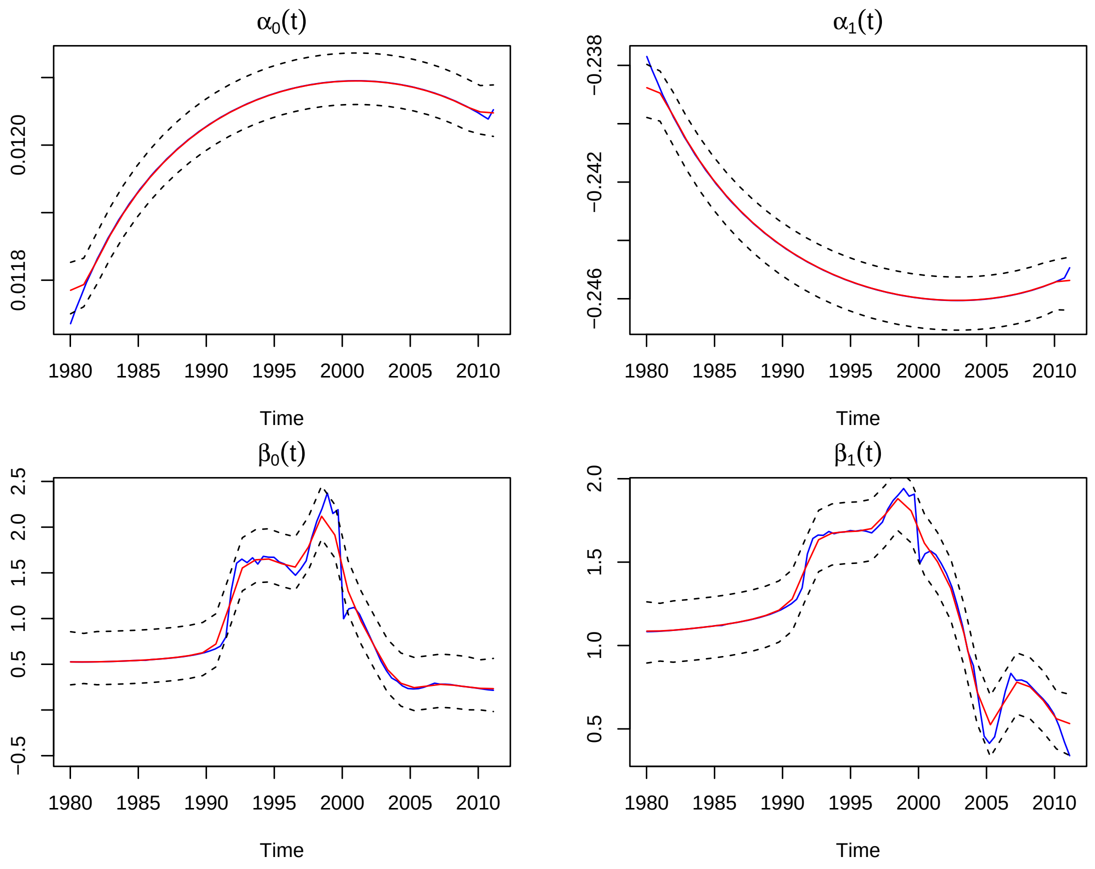

Figure 3.2: Pointwise 95% envelope (black band), median (red), and true function (blue) for the estimators of $\alpha_0(t)$, $\alpha_1(t)$, $\beta_0(t)$, and $\beta_1(t)$ under the varying-coefficient model in Example 3.2.

The simulation results show that the estimators successfully capture the temporal dynamics of the coefficient functions. The true time-varying structures lie well within the estimated confidence envelopes, demonstrating both flexibility and accuracy of the methodology in nonstationary settings.

CHAPTER 4: Asymptotic Properties

This chapter establishes the asymptotic properties of the nonparametric estimators in the varying-coefficient stochastic diffusion model. Specifically, we derive the asymptotic distributions of the estimators for the time-varying drift coefficient $\widehat{\alpha}_0(t)$, $\widehat{\alpha}_1(t)$, and the diffusion coefficient $\widehat{\beta}_0(t)$, $\widehat{\beta}_1(t)$.

4.1 Notation and Preliminaries

- Define the observation times and increments:

$$t_i = t_0 + i\Delta, \quad i = 0, \dots, n, \quad \Delta = \frac{T}{n}, \quad Y_i = X_{t_{i+1}} - X_{t_i}, \quad Y_i^* = \frac{Y_i}{\Delta}.$$

- Define the state and drift parameter vectors:

$$X_i = \begin{pmatrix} 1 \\ \\ X_{t_i} \end{pmatrix}, \quad \alpha(t) = \begin{pmatrix} \alpha_0(t) \\ \\ \alpha_1(t) \end{pmatrix}.$$

- Standardized residual:

$$u_i = \beta_0(t_i) X_{t_i}^{\beta_1(t_i)} \frac{\varepsilon_{t_i}}{\sqrt{\Delta}}, \quad \varepsilon_{t_i} \sim \mathcal{N}(0, 1).$$

- Kernel weights:

$$K_h(t_i - t_0) = \frac{1}{h} K\left(\frac{t_i - t_0}{h}\right), \quad \mu_m = \int_{-1}^{1} u^m K(u) du, \quad R(K) = \int_{-1}^{1} K(u)^2 du.$$

- Local-constant estimator matrices:

$$S_n = \sum_{i=1}^{n} K_h(t_i - t_0) X_i X_i^T, \quad R_n = \sum_{i=1}^{n} K_h(t_i - t_0) X_i Y_i^*, \quad \widehat{\alpha}(t_0) = S_n^{-1} R_n.$$

- Sampling density and design matrix:

$$f_T(t) = \frac{1}{T}, \quad t \in [0, T], \quad I_X(t_0) = f_T(t_0) \mathbb{E}[X_{t_0} X_{t_0}^T].$$

- Volatility structure:

$$\sigma^2(t, x) = \beta_0^2(t) x^{2\beta_1(t)}.$$

- Asymptotic bias and variance for drift coefficient estimators:

$$B(t_0) = \frac{\mu_2}{2 f_T(t_0)} I_X^{-1}(t_0) \alpha''(t_0), \quad \Sigma(t_0) = I_X^{-1}(t_0) \left[R(K) f_T(t_0) \mathbb{E}[X_{t_0} X_{t_0}^T \frac{\sigma^2(t_0, X_{t_0})}{\Delta}] \right] I_X^{-1}(t_0).$$

- Standardized residuals for volatility estimation:

$$\widehat{E}_i = \frac{Y_i - \widehat{\mu}(t_i, X_{t_i}) \Delta}{\sqrt{\Delta}}, \quad \widehat{\mu}(t_i, X_{t_i}) = \widehat{\alpha}_0(t_i) + \widehat{\alpha}_1(t_i) X_{t_i}.$$

- Local pseudo-likelihood:

$$\ell_n(\theta) = -\frac{1}{2} \sum_{i=1}^{n} K_h(t_i - t_0) \left\{ \log \left(\beta_0^2 X_{t_i}^{2\beta_1} \right) + \frac{\widehat{E}_i^2}{\beta_0^2 X_{t_i}^{2\beta_1}} \right\},$$

with $\theta = (\beta_0, \beta_1)^\top$.

- Closed-form for β_0 given β_1:

$$\widehat{\beta}_0^2(t_0; \beta_1) = \frac{\sum_{i=1}^{n} K_h(t_i - t_0) \widehat{E}_i^2 X_{t_i}^{-2\beta_1}}{\sum_{i=1}^{n} K_h(t_i - t_0)}.$$

- Optimized β_1:

$$\widehat{\beta}_1(t_0) = \arg\max_{\beta_1} \ell(\widehat{\beta}_0(t_0;\beta_1),\beta_1;t_0).$$

- Local parameter vectors:

$$\widehat{\theta}(t_0) = \begin{pmatrix} \widehat{\beta}_0(t_0) \\ \widehat{\beta}_1(t_0) \end{pmatrix}, \quad \theta(t_0) = \begin{pmatrix} \beta_0(t_0) \\ \beta_1(t_0) \end{pmatrix}.$$

We write $\theta = (\beta_0,\beta_1)^\top$ as shorthand for $\theta(t_0)$, the local volatility parameter vector evaluated at time t_0.

- Score function:

$$\psi_i(\theta) = \begin{pmatrix} \dfrac{\partial}{\partial\beta_0}\ell_i(\theta) \\ \dfrac{\partial}{\partial\beta_1}\ell_i(\theta) \end{pmatrix} = \begin{pmatrix} -\dfrac{1}{\beta_0} + \dfrac{\widehat{E}_{t_i}^2}{\beta_0^3 X_{t_i}^{2\beta_1}} \\ -\log X_{t_i} + \dfrac{\widehat{E}_{t_i}^2 \log X_{t_i}}{\beta_0^2 X_{t_i}^{2\beta_1}} \end{pmatrix}.$$

- Observed information matrix (negative Hessian):

$$h_i(\theta) = -\begin{pmatrix} \dfrac{\partial^2}{\partial\beta_0^2}\ell_i(\theta) & \dfrac{\partial^2}{\partial\beta_0\partial\beta_1}\ell_i(\theta) \\ \dfrac{\partial^2}{\partial\beta_1\partial\beta_0}\ell_i(\theta) & \dfrac{\partial^2}{\partial\beta_1^2}\ell_i(\theta) \end{pmatrix} = \begin{pmatrix} \dfrac{1}{\beta_0^2} - \dfrac{3\widehat{E}_{t_i}^2}{\beta_0^4 X_{t_i}^{2\beta_1}} & \dfrac{2\widehat{E}_{t_i}^2 \log X_{t_i}}{\beta_0^3 X_{t_i}^{2\beta_1}} \\ \dfrac{2\widehat{E}_{t_i}^2 \log X_{t_i}}{\beta_0^3 X_{t_i}^{2\beta_1}} & \dfrac{2\widehat{E}_{t_i}^2 \log^2 X_{t_i}}{\beta_0^2 X_{t_i}^{2\beta_1}} \end{pmatrix}.$$

- Fisher information and variance of the score function:

$$I_\theta(t_0) = \mathbb{E}\left[h_i(\theta)\right], \quad V_\theta(t_0) \;=\; \mathrm{Var}\Big(\,\psi_i\big(\theta(t_0)\big)\Big)$$

- Log-likelihood sum and local reparameterization:

$$\ell_n(\theta) = \sum_{i=1}^n K_h(t_i - t_0)\,\ell_i(\theta), \quad \delta = \sqrt{nh}\,(\theta-\theta_0), \quad a_n = \frac{1}{\sqrt{nh}}.$$

Convergence in probability and distribution are denoted by $\xrightarrow{P}$ and $\xrightarrow{d}$, respectively.

4.2 Asymptotic Properties

In this section, we establish the asymptotic properties of the proposed estimators for the varying-coefficient stochastic diffusion model. The regularity conditions required for these results are listed in Appendix A, and all proofs are provided in Appendix B.

4.2.1 Drift Coefficient Estimator $\widehat{\alpha}(t)$

Recall from the previous section that the drift coefficient estimator:

$$\widehat{\alpha}(t_0) = \begin{pmatrix} \widehat{\alpha}_0(t_0) \\ \widehat{\alpha}_1(t_0) \end{pmatrix} = S_n^{-1} R_n, \quad S_n = \sum_{i=1}^{n} K_h(t_i - t_0)\, X_i X_i^\top, \quad R_n = \sum_{i=1}^{n} K_h(t_i - t_0)\, X_i Y_i^*.$$

Theorem 4.1 Under Assumptions A1–A8, we have:

$$\sqrt{nh}\left[\widehat{\alpha}(t_0) - \alpha(t_0) - h^2 B(t_0)\right] \xrightarrow{d} \mathcal{N}(0, \Sigma(t_0)),$$

where

$$B(t_0) = \frac{\mu_2}{2 f_T(t_0)} I_X(t_0)^{-1} \alpha''(t_0), \quad \Sigma(t_0) = I_X(t_0)^{-1}\left[R(K) f_T(t_0)\, \mathbb{E}\left\{X_{t_0} X_{t_0}^\top \frac{\sigma^2(t_0, X_{t_0})}{\Delta}\right\}\right] I_X(t_0)^{-1}$$

4.2.2 Volatility Coefficient Estimator $\widehat{\theta}(t)$

The diffusion coefficient estimator $\widehat{\theta}(t_0) = (\widehat{\beta}_0(t_0), \widehat{\beta}_1(t_0))^\top$ is obtained by maximizing the local pseudo–log–likelihood

$$\ell(\beta_0, \beta_1; t_0) = -\frac{1}{2} \sum_{i=1}^{n} K_h(t_i - t_0)\left[\log\bigl(\beta_0^2 X_{t_i}^{2\beta_1}\bigr) + \frac{\widehat{E}_i^2}{\beta_0^2 X_{t_i}^{2\beta_1}}\right],$$

where $\widehat{E}_i$ denotes the standardized residual.

Theorem 4.2 Under Assumptions A1–A8, we have

$$\sqrt{n\,h}\left[\widehat{\theta}(t_0) \;-\; \theta(t_0) \;-\; h^2\, B_\theta(t_0)\right] \;\xrightarrow{d}\; \mathcal{N}\bigl(0,\; \Sigma_\theta(t_0)\bigr),$$

where

$$B_\theta\left(t_0\right) = \frac{\mu_2}{2f_T\left(t_0\right)} I_\theta\left(t_0\right)^{-1}\left[\frac{d^2}{dt^2} E\left(\partial_\theta \ell_i(\theta(t))\right)\right]_{t=t_0},$$

$$\Sigma_\theta(t_0) \;=\; I_\theta(t_0)^{-1}\, V_\theta(t_0)\, I_\theta(t_0)^{-1},$$

with

$$I_\theta(t_0) = E\Big[-\partial^2_{\theta\theta^\top}\,\ell_i\big(\theta(t_0)\big)\Big], \quad V_\theta(t_0) = Var\Big[\partial_\theta\,\ell_i\big(\theta(t_0)\big)\Big], \quad \mu_2 = \int u^2 K(u)\,du.$$

Remarks.

- Both $\widehat{\alpha}(t_0)$ and $\widehat{\theta}(t_0)$ converge at the nonparametric rate $\sqrt{n\,h}$.
- Each estimator carries an $O(h^2)$ bias. In particular

$$E\big[\widehat{\alpha}(t_0)\big] = \alpha(t_0) + h^2\,B(t_0) + o(h^2), \quad E\big[\widehat{\theta}(t_0)\big] = \theta(t_0) + h^2\,B_\theta(t_0) + o(h^2).$$

- The asymptotic variance-covariance matrices

$$\Sigma(t_0) = I_X(t_0)^{-1}\Big[R(K)\,f_T(t_0)\,E\{X_{t_0}X_{t_0}^\top\sigma^2(t_0,X_{t_0})/\Delta\}\Big]I_X(t_0)^{-1}, \quad \Sigma_\theta(t_0) = I_\theta(t_0)^{-1}\,V_\theta(t_0)\,I_\theta$$

can be consistently estimated by plugging in

$$\widehat{I}_X(t_0) = \frac{1}{nh}\sum_{i=1}^{n} K_h(t_i - t_0)\,X_iX_i^\top, \quad \widehat{\sigma}^2(t_i, X_{t_i}) = \widehat{\beta}_0(t_0)^2\,X_{t_i}^{2\widehat{\beta}_1(t_0)},$$

and similarly using

$$\widehat{I}_\theta(t_0) = -\frac{1}{nh}\sum_{i=1}^{n} K_h(t_i - t_0)\,\frac{\partial^2}{\partial\theta\partial\theta^\top}\,\ell_i\big(\widehat{\theta}(t_0)\big), \quad \widehat{V}_\theta(t_0) = \frac{1}{nh}\sum_{i=1}^{n} K_h(t_i - t_0)\,\widehat{\psi}_i\,\widehat{\psi}_i^\top,$$

with $\widehat{\psi}_i = \partial_\theta\ell_i(\widehat{\theta}(t_0))$.

CHAPTER 5: Modeling Dependencies Among Financial Time Series Using Deep Learning Regression Frameworks

5.1 Introduction and Motivation

In the previous chapter, we focused on modeling the temporal evolution of a single financial process using a varying-coefficient stochastic diffusion model. While this univariate approach offers flexibility in capturing nonlinear dynamics and time-varying volatility, it does not address the multivariate nature of financial markets. In practice, financial variables like interest rates, earnings, and dividends often exhibit interdependencies over time.

To capture such relationships, we adopt a multivariate regression framework in which the target variable—such as the S&P 500 index—is modeled as a function of several explanatory financial processes. This framework aims not only to improve predictive accuracy but also to uncover how different variables interact and jointly influence financial outcomes, particularly in the context of financial forecasting.

Formally, we consider the following model:

$$Y_t = f(X_{1t}, X_{2t}, \ldots, X_{pt}) + \epsilon_t,$$

where Y_t represents the target financial process at time t, $X_{1t}, \ldots, X_{pt}$ denote p explanatory financial variables, and ϵ_t is an error term. The function $f(\cdot)$ is unknown and potentially highly nonlinear.

We evaluate several neural network architectures—including LSTM, GRU, CNN, CNN-LSTM and traditional time series models such as AR and ARIMA to assess their ability to learn these dependencies and forecast financial outcomes. Given the complexity and nonlinearity of such relationships, we estimate $f(\cdot)$ using deep learning neural networks. These models can approximate complex functional forms without imposing restrictive parametric structures.

5.2 Data and Variable Structure

The data used in this chapter is sourced from the data used in Campbell and Yogo [8] and Hong et al. [22]. It contains monthly observations of key financial and macroeconomic indicators frequently used in return prediction studies. The target variable is the monthly closing value of the S&P 500 index (S&P 500). The explanatory variables include:

- Earnings: the earnings-price (EP) ratio for the S&P 500 index
- Dividends: the dividend-price (DP) ratio for the S&P 500 index
- TB3MS: the 3-month U.S. Treasury bill secondary market rate, used as a proxy for the short-term risk-free rate
- Lagged S&P 500: a 5-month lag of the S&P 500 index, included to account for autocorrelation and momentum effects

All variables are recorded at monthly frequency. After removing missing values and applying the 5-period lag to the S&P 500 index, the dataset contains a total of 1,003 monthly observations.

Before model training, each series is scaled to the $[0, 1]$ range using MinMax normalization to ensure stable training and comparable input magnitudes across variables. The dataset is split chronologically, with the first 80% (802 observations) used for training and the remaining 20% (201 observations) for testing. After accounting for the 10-step input window required by the models, this results in 792 training sequences and 191 testing sequences.

Each input sample consists of the 10 most recent monthly observations for all explanatory variables, and the model is trained to predict the S&P 500 value one month ahead. That is, we use the sequence $\{X_{t-9}, X_{t-8}, \ldots, X_t\}$ to forecast Y_{t+1}, where X_t is the vector of normalized explanatory variables at time t, and Y_{t+1} is the corresponding S&P 500 index value in the next month.

5.3 Deep Learning Models

In this section, we explore 4 neural network architectures chosen for their ability to capture different aspects of financial time series behavior. Recurrent models like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) are well suited for sequential modeling and long-range temporal dependencies. Convolutional Neural Networks (CNNs) efficiently extract short-term local patterns, while hybrid models (CNN-LSTM) combine both local and sequential learning. These models are trained on the same inputs and evaluated under uniform settings for direct comparison.

5.3.1 Long Short-Term Memory (LSTM)

The Long Short-Term Memory (LSTM) network, introduced by Hochreiter and Schmidhuber [21], was developed to overcome the limitations of standard recurrent neural networks (RNNs), particularly their inability to capture long-range dependencies due to vanishing and exploding gradients. This makes LSTM networks especially effective for modeling financial time series, which often exhibit persistent temporal correlations and structural shifts.

LSTM units maintain two forms of memory: a hidden state and a cell state. These are updated at each time step through a gated mechanism that regulates the flow of information, allowing the network to learn what to retain, update, or discard across time. Specifically, the LSTM cell employs three gates: a forget gate, an input gate, and an output gate. These gates coordinate how new input, past memory, and past hidden information are combined to update the cell state and compute the output.

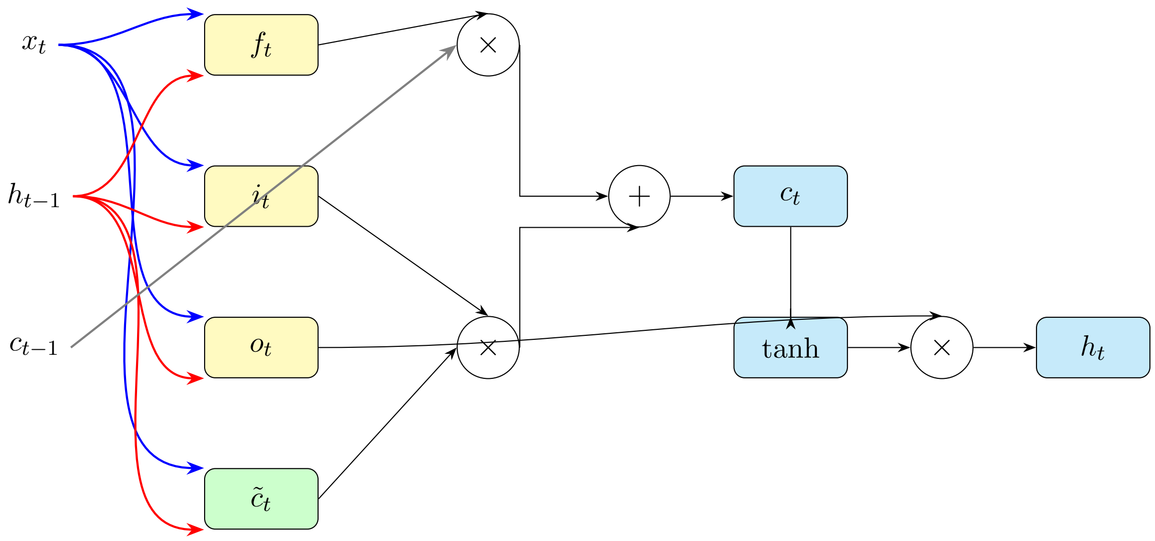

Figure 5.1: Long Short-Term Memory (LSTM) network structure.

The architecture and information flow of an LSTM cell are illustrated in Figure 5.1. At time t, given an input vector x_t, the previous hidden state h_{t-1}, and the previous cell state c_{t-1}, the LSTM performs the following computations:

1. Forget gate:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

   Controls which components of c_{t-1} are retained or forgotten.

2. Input gate and candidate update:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \quad \tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c)$$

   Determines how much of the new candidate memory $\tilde{c}_t$ is added to the cell state.

3. Cell state update:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

   Produces the updated internal memory.

4. Output gate and hidden state:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \quad h_t = o_t \odot \tanh(c_t)$$

   Generates the hidden state that serves as the output of the current cell and input to the next.

where

- $x_t \in \mathbb{R}^m$: Input vector at time t
- $h_t \in \mathbb{R}^d$: Hidden state
- $c_t \in \mathbb{R}^d$: Cell state
- $f_t, i_t, o_t \in (0,1)^d$: Forget, input, and output gate activations
- $\tilde{c}_t \in \mathbb{R}^d$: Candidate update
- $W_. \in \mathbb{R}^{d \times m}$, $U_. \in \mathbb{R}^{d \times d}$, $b_. \in \mathbb{R}^d$: Trainable parameters
- $\sigma(z) = \frac{1}{1+e^{-z}}$: Sigmoid activation
- $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$: Hyperbolic tangent activation
- $\odot$: Element-wise multiplication

In our implementation, we used the following setup:

- One LSTM layer with 50 units
- ReLU activation function
- Dropout layer with rate 0.2 to prevent overfitting
- Dense output layer with one neuron for regression

- Optimizer: Adam
- Loss function: Mean Squared Error (MSE)
- Up to 50 training epochs with early stopping (patience = 5)
- Batch size: 32

This architecture was selected to balance model complexity and training stability. The use of dropout and early stopping was motivated by concerns about overfitting, particularly due to the relatively limited sample size in our dataset. The number of hidden units was chosen based on common practice and preliminary tuning.

5.3.2 Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU), introduced by Cho et al. [10], is a simplified variant of the Long Short-Term Memory (LSTM) network. GRUs were developed to mitigate the vanishing gradient problem while offering a more streamlined architecture. Unlike LSTMs, GRUs do not maintain a separate cell state—instead, the entire memory is encoded in a single hidden state vector h_t, which is updated through two gates: the update gate and the reset gate.

This simplification reduces the number of trainable parameters and computational complexity, making GRUs particularly suitable for settings with limited training data or real-time inference constraints. Despite their reduced structure, GRUs often achieve comparable performance to LSTMs in sequence modeling tasks such as financial forecasting, where modeling long- and short-term dependencies is critical.

The architecture and information flow of an GRU network are illustrated in Figure 5.2. At each time step t, given the input vector x_t and the previous hidden state h_{t-1}, the GRU performs the following computations:

1. Update gate:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

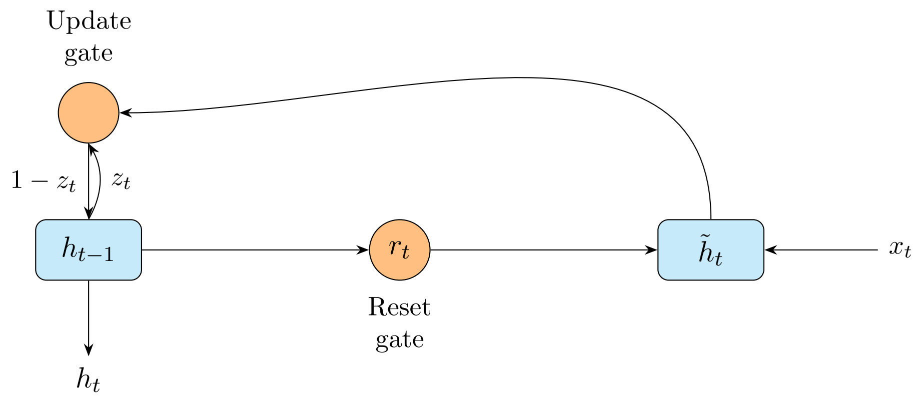

Figure 5.2: Gated Recurrent Unit (GRU) network structure

The update gate controls the balance between retaining the previous hidden state and incorporating new information. A value of $z_t \approx 1$ leads to replacing the memory with the new candidate, while $z_t \approx 0$ retains past memory.

2. Reset gate:

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

The reset gate determines how much of the past information to forget before computing the candidate activation. A reset gate close to zero effectively resets the memory for the current time step.

3. Candidate activation:

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$

This is the candidate hidden state that proposes a new memory update based on the current input and selectively forgotten past.

4. Hidden state:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

The updated hidden state is computed as a convex combination of the previous memory and the candidate activation, weighted by the update gate. This interpolation allows the GRU to adaptively decide between preserving past memory and integrating new context.

where

- $x_t \in \mathbb{R}^m$: Input vector at time t
- $h_t \in \mathbb{R}^d$: Hidden state (also serves as GRU output)
- $z_t, r_t \in (0,1)^d$: Update and reset gates
- $\tilde{h}_t \in \mathbb{R}^d$: Candidate hidden state
- $W_\cdot \in \mathbb{R}^{d\times m}$, $U_\cdot \in \mathbb{R}^{d\times d}$, $b_\cdot \in \mathbb{R}^d$: Trainable weights and biases
- $\sigma(z) = \frac{1}{1+e^{-z}}$: Sigmoid activation function
- $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$: Hyperbolic tangent activation function
- $\odot$: Element-wise multiplication

In our implementation, we used the following setup:

- One GRU layer with 50 units
- ReLU activation function
- Dropout layer with rate 0.2 to prevent overfitting
- Dense output layer with one neuron for regression
- Optimizer: Adam
- Loss function: Mean Squared Error (MSE)
- Up to 50 training epochs with early stopping (patience = 5)

- Batch size: 32

This architecture was selected as a simplified alternative to LSTM, offering similar representational power with reduced computational complexity. Given the comparable training setup, dropout and early stopping were again applied to guard against overfitting.

5.3.3 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) were originally developed for computer vision but have been successfully adapted for time series forecasting in Borovykh et al. [7], Zeng et al. [39]. In time series applications, CNNs apply 1D convolutional filters across temporal dimensions to extract short-range patterns and local dependencies—such as trend changes or bursts of volatility in financial data.

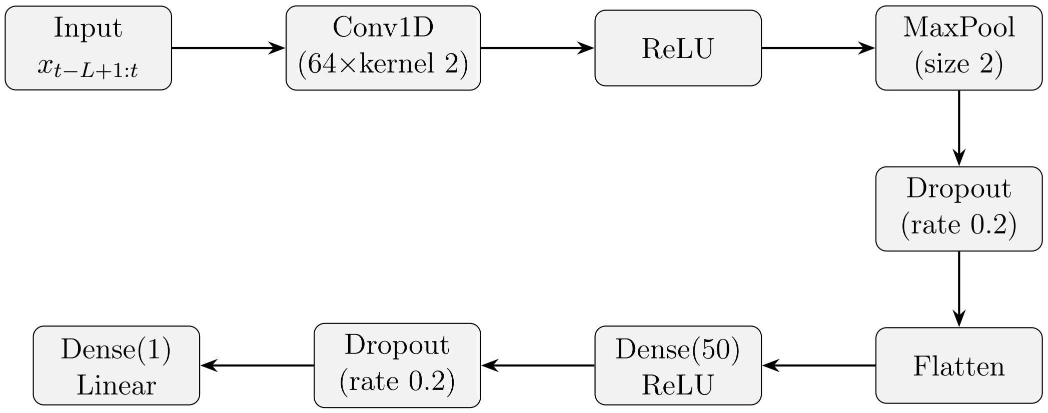

Figure 5.3: 1D-CNN structure.

The architecture of an 1D-CNN are illustrated in Figure 5.3. At each time step t, a one-dimensional convolutional filter of length s slides across the input sequence $x = (x_1, x_2, \dots, x_T)$, capturing local temporal patterns. For the k-th filter, the convolutional output at position t is computed as:

$$h_t^{(k)} = \sigma\left(\sum_{j=0}^{s-1} w_j^{(k)} x_{t+j} + b^{(k)}\right),$$

where $w^{(k)} \in \mathbb{R}^s$ denotes the filter weights, $b^{(k)}$ is a bias term, and $\sigma(\cdot)$ is a non-linear activation function, typically the rectified linear unit (ReLU). Each filter k learns to detect distinct local structures or short-term dependencies within the time series, such as level shifts, bursts, or local trends.

The resulting set of feature maps is then passed through a max pooling layer, which downsamples the output by selecting the most salient responses, thereby reducing dimensionality and providing translational invariance to local fluctuations. These pooled features are subsequently flattened and fed into fully connected (dense) layers, which integrate the learned local representations and map them to the final predictive output.

In our implementation, we used the following setup:

- One 1D convolutional layer with 64 filters and kernel size 2
- ReLU activation function
- Max pooling layer with pool size 2
- Dropout layer with rate 0.2 to prevent overfitting
- Flatten layer followed by a dense layer with 50 units (ReLU)
- Additional dropout layer with rate 0.2 before the final output
- Dense output layer with one neuron for regression
- Optimizer: Adam
- Loss function: Mean Squared Error (MSE)
- Up to 50 training epochs with early stopping (patience = 5)
- Batch size: 32

This architecture was selected for its computational efficiency and strong performance in extracting local temporal features.

5.3.4 CNN–LSTM Hybrid Model

The CNN–LSTM hybrid model combines the spatial pattern extraction capabilities of CNNs with the sequential modeling strength of LSTMs. This hybrid is especially useful for capturing both short-term local fluctuations and long-term temporal dependencies Tian [35], Zeng et al. [39].

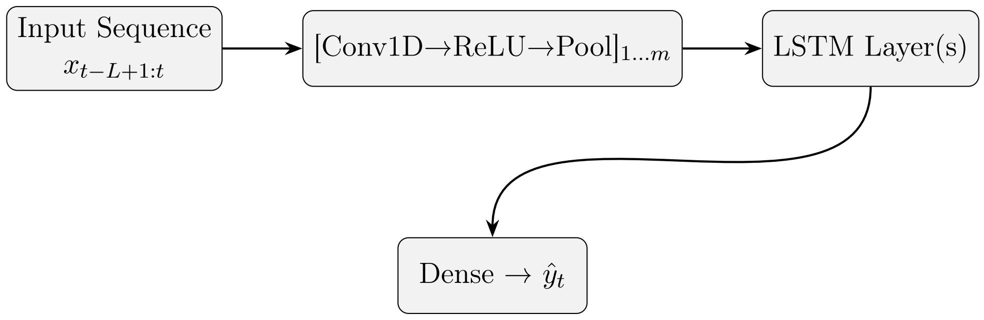
Input Sequence $x_{t-L+1:t}$
[Conv1D→ReLU→Pool]$_{1\ldots m}$
LSTM Layer(s)
Dense → $\hat{y}_t$

Figure 5.4: CNN–LSTM structure

The architecture of CNN–LSTM network are illustrated in Figure 5.4. The CNN–LSTM model combines a 1D convolutional feature extractor with an LSTM to capture both local and long-range temporal dependencies. At each time step t, we first transform the raw series $x = (x_1, \ldots, x_T)$ into a compact feature vector $\tilde{x}_t$ via convolution and pooling, then feed $\tilde{x}_t$ into an LSTM cell. Formally:

1. Convolution and pooling. We apply K one-dimensional filters of length s over a sliding window ending at t:

$$z_t^{(k)} = \sum_{j=0}^{s-1} w_j^{(k)}\, x_{t-j} + b^{(k)}, \quad a_t^{(k)} = \mathrm{ReLU}(z_t^{(k)}), \quad k = 1, \ldots, K.$$

   Next, we downsample by max-pooling over each non-overlapping block of length p:

$$\tilde{x}_i^{(k)} = \max\{\, a_{(i-1)p+1}^{(k)}, \ldots, a_{ip}^{(k)}\}, \quad i = 1, \ldots, \lfloor T/p \rfloor.$$

   We then stack the K-dimensional pooled vectors at the positions aligned to t into the feature vector $\tilde{x}_t \in \mathbb{R}^K$.

2. LSTM update. Given $\tilde{x}_t$, the previous hidden state h_{t-1}, and the previous cell state c_{t-1}, the LSTM performs:

$$\begin{aligned}
&f_t = \sigma(W_f \tilde{x}_t + U_f h_{t-1} + b_f), \\
&i_t = \sigma(W_i \tilde{x}_t + U_i h_{t-1} + b_i), \quad \tilde{c}_t = \tanh(W_c \tilde{x}_t + U_c h_{t-1} + b_c), \\
&c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \\
&o_t = \sigma(W_o \tilde{x}_t + U_o h_{t-1} + b_o), \quad h_t = o_t \odot \tanh(c_t).
\end{aligned}$$

3. Output. The hidden state h_t is then passed to a final dense layer (or directly used) to produce the one-step-ahead forecast $\hat{y}_t$.

Notation:

- $x_t \in \mathbb{R}$: raw time-series value at t.
- $w^{(k)} \in \mathbb{R}^s, b^{(k)} \in \mathbb{R}$: convolutional filter weights and bias for feature k.
- K: number of convolutional filters.
- p: pooling size.
- $\tilde{x}_t \in \mathbb{R}^K$: pooled feature vector at time t.
- $h_t, c_t \in \mathbb{R}^d$: LSTM hidden and cell states.
- $f_t, i_t, o_t \in (0,1)^d$: forget, input, and output gate activations.
- $\tilde{c}_t \in \mathbb{R}^d$: LSTM candidate memory.
- $W_\cdot \in \mathbb{R}^{d \times K}$, $U_\cdot \in \mathbb{R}^{d \times d}$, $b_\cdot \in \mathbb{R}^d$: trainable LSTM parameters.
- $\sigma(z) = 1/(1+e^{-z})$, $\tanh(z) = (e^z - e^{-z})/(e^z + e^{-z})$, $\odot$: element-wise multiplication.

The CNN–LSTM architecture first distills local temporal patterns through convolution and pooling, then models their evolution over time through the gated LSTM dynamics, yielding a powerful hybrid for financial forecasting. In our implementation, we used the following setup:

- One 1D convolutional layer with 64 filters and kernel size 2
- ReLU activation function
- Max pooling layer with pool size 2
- Dropout layer with rate 0.2 after pooling
- LSTM layer with 32 units
- Dropout layer with rate 0.2 after the LSTM
- Dense output layer with one neuron for regression
- Optimizer: Adam
- Loss function: Mean Squared Error (MSE)
- Up to 50 training epochs with early stopping (patience = 5)
- Batch size: 32

This structure was chosen to leverage complementary strengths of both CNNs and LSTMs. Prior work has shown that such hybrid models can outperform standalone architectures in financial prediction tasks by capturing both local and global dynamics.

5.4 Result Evaluation

To assess forecasting performance, we compare each deep learning model to two traditional time series models: $AR(20)$, an autoregressive model applied to differenced S&P 500 returns, and $ARIMA(5, 1, 0)$, an integrated model for capturing trend and autocorrelation.

All models were trained on the first 80% of the dataset and evaluated on the remaining 20%. Forecast accuracy was assessed using three metrics commonly used in financial time series evaluation:

- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Mean Absolute Percentage Error (MAPE)

Table 5.1: Forecasting Performance on S&P 500 Data

Model	RMSE	MAE	MAPE
LSTM	108.8462	91.5796	0.1219
GRU	130.1747	115.3863	0.1464
CNN	91.1347	63.4523	0.0927
CNN–LSTM	90.6981	67.3337	0.0983
AR(20)	266.1643	235.3103	0.3465
ARIMA	346.4359	309.9496	0.4669

The hybrid CNN–LSTM model clearly outperforms every other neural network, posting the lowest RMSE (90.70). The CNN follows closely, achieving the best MAE (63.45) and MAPE (9.27%), which highlights its strength in capturing short-term patterns. The LSTM comes next—its performance (RMSE = 108.85, MAE = 91.58, MAPE = 12.19%) is solid but cannot match the precision of the convolutional methods. The GRU, despite its simpler gating, trails behind both LSTM and the CNNs (RMSE = 130.17, MAE = 115.39, MAPE = 14.64%), suggesting it may be less adept at handling the S&P 500's volatile swings. Likewise, the traditional AR(20) and ARIMA(5,1,0) models deliver substantially larger errors.

To summarize the comparative rankings:

Table 5.2: Ranking of Models

Metric	1st	2nd	3rd
RMSE	CNN–LSTM	CNN	LSTM
MAE	CNN	CNN–LSTM	LSTM
MAPE	CNN	CNN–LSTM	LSTM

This ranking clearly underscores the consistent superiority of the CNN–LSTM architecture across all criteria, followed by the standalone CNN. These findings highlight that the combination of convolutional filters with recurrent memory provides a powerful framework for forecasting non-stationary high noise financial series such as the S&P 500 index.

CHAPTER 6: High-Dimensional Time Series Regression and Variable Selection

6.1 Introduction and Motivation

In the previous chapter, we applied a deep learning framework for forecasting financial time series by modeling nonlinear dependencies among multiple variables. While this showed strong predictive performance, it also highlighted a common challenge in modern time series analysis: managing a large number of explanatory variables without compromising model interpretability or stability.

This issue exists in macroeconomic and financial forecasting, where datasets may include many potential predictors—such as interest rates, inflation rates, corporate earnings, and sentiment indicators. These variables typically exhibit heterogeneous statistical properties: some are stationary, others are nonstationary or even cointegrated. Modeling such high-dimensional, mixed-persistence data introduces both statistical and computational complexities, especially when the number of predictors exceeds the available sample size.

To address these challenges, we employ the Least Absolute Shrinkage and Selection Operator (LASSO), introduced by Tibshirani [36], as a method for variable selection and dimension reduction. LASSO is a popular method in high-dimensional regression due to its ability to shrink coefficients and select variables simultaneously via an ℓ_1 penalty. Its interpretability, scalability, and ability to handle situations where $p \gg n$ make it a natural choice in this context.

While LASSO has been extensively studied in i.i.d. and weakly dependent settings, its behavior in time series regression—particularly when predictors include a mixture of stationary and nonstationary components—is not fully understood. As financial and economic datasets frequently violate the standard assumptions that guarantee

LASSO's consistency. Persistent trends, stochastic volatility, autocorrelations, and structural breaks are common features in such data.

This chapter investigates a central question:

Is LASSO regression consistent for variable selection in high-dimensional time series data when predictors include both stationary and nonstationary components?

To explore this, we examine the LASSO selection consistency under the mixture of stationary and nonstationary variables, varying signal-to-noise ratios, and different correlation structures. We also evaluate its performance on a real-world macro-financial dataset with heterogeneous stationarity properties. These analyses assess both the theoretical robustness and the practical utility of LASSO as a feature selection tool in complex, high-dimensional time series environments.

6.2 Model Setup

Consider the following linear model:

$$Y_t = \beta_0 + \sum_{i=1}^{k} \beta_i X_{i,t} + \sum_{j=k+1}^{k+p} \beta_j X_{j,t} + \epsilon_t, \tag{6.1}$$

where:

- Y_t is the response variable observed over time $t = 1, \ldots, N$,
- $X_{i,t}$, for $i = 1, \ldots, k$, are stationary predictors (e.g., AR(1) processes),
- $X_{j,t}$, for $j = k+1, \ldots, k+p$, are integrated processes of order one (I(1)),
- β_0 is the intercept, and $\beta_i \in \mathbb{R}$ are unknown coefficients,
- $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is an i.i.d. noise term.

The total number of predictors $d = k + p$ may be large relative to the sample size N, creating a high-dimensional situation where $d \gg N$ is possible. We assume that the true coefficient vector $\beta = (\beta_1, \ldots, \beta_d)$ is sparse, meaning only a small subset of predictors have nonzero effects on the response.

To handle this setting, we apply the Least Absolute Shrinkage and Selection Operator (LASSO), which solves the penalized least squares problem:

$$\widehat{\beta}^{\text{lasso}} = \arg\min_{\beta} \left\{ \frac{1}{2N} \sum_{t=1}^{N} \left(Y_t - \beta_0 - \sum_{\ell=1}^{d} \beta_\ell X_{\ell,t} \right)^2 + \lambda \sum_{\ell=1}^{d} |\beta_\ell| \right\}, \tag{6.2}$$

where $\lambda > 0$ is the tuning parameter controlling the sparsity level.

While LASSO is designed to produce sparse models in high-dimensional settings, its behavior in time series applications is complicated by the presence of autocorrelated errors, cross-sectional dependence, and mixed integration orders. Nonstationary regressors, in particular, may distort penalty magnitudes due to their diverging variance over time. These issues motivate a careful examination of LASSO's selection properties in the presence of mixed-persistence predictors.

In the following sections, we investigate whether LASSO can consistently select the important predictors under this model structure.

6.3 Tuning Parameter Selection

In high-dimensional regression, the choice of the regularization parameter λ is critical for balancing model sparsity and predictive performance. This selection becomes particularly important in time series applications, where serial dependence and nonstationary components violate the independence assumptions underlying standard k-fold cross-validation procedures.

To address these issues, we adopt a rolling-origin cross-validation method(also known as walk-forward validation). Specifically, we begin with an initial training set consisting of the first 50% of the observations. Then, we perform five rounds of validation, each time expanding the training window by 10% and using the subsequent 10% of the data as a test set. At each fold, the model is refit on the expanded training data and evaluated on the forward validation block. An illustration of this procedure is provided in Figure 6.1.

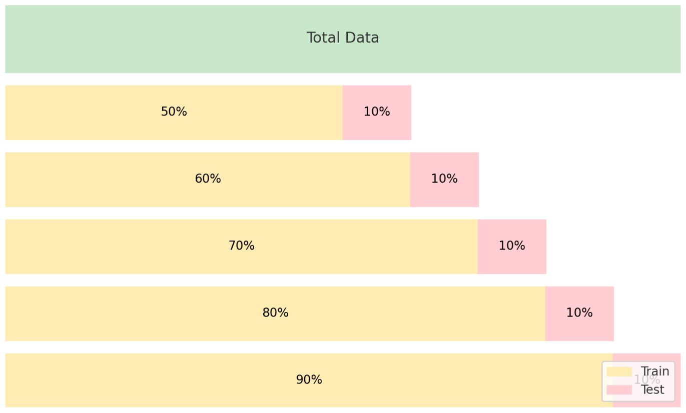

Figure 6.1: Rolling-origin cross-validation

This methodology is supported by the forecasting literature. Tashman [34] presents a detailed review of out-of-sample forecast evaluation methods and recommends rolling-origin validation as a reliable strategy in the presence of time dependence. Bergmeir et al. [5] further demonstrate that while traditional cross-validation can be valid under certain assumptions (e.g., uncorrelated errors), time-series-specific procedures such as rolling-origin cross-validation are generally more robust in practice.

Based on this rolling-origin cross-validation procedure, we find that the optimal regularization parameter λ_{cv} lies within the range $[0.10, 0.11]$. To evaluate the sensitivity of the model to different penalty levels, we also consider values that are two-thirds and three-halves of this range:

$$\lambda = \frac{2}{3} \times \lambda_{cv} \approx 0.06 \quad \text{and} \quad \lambda = \frac{3}{2} \times \lambda_{cv} \approx 0.17.$$

These three values-$\lambda = 0.06$, λ_{cv}, and $\lambda = 0.17$—are used in the simulation study presented in the following section to examine the robustness of LASSO's variable selection and estimation performance across different levels of regularization.

6.4 Simulation

To evaluate the consistency of variable selection by LASSO in high-dimensional time series regression with both stationary and nonstationary predictors, we conduct a series of simulation studies using the model introduced in Equation (6.1). The predictors $X_{i,t}$ are generated to reflect a mixture of stationary ($I(0)$) and nonstationary ($I(1)$) time series, capturing the heterogeneous dynamics often observed in macroeconomic and financial data.

Example 7.1: Independent Predictors

This example assesses the performance of the LASSO estimator in a high-dimensional setting where predictors are mutually independent but exhibit heterogeneous dynamic behavior. Specifically, the covariates consist of a mix of stationary and nonstationary processes, with no cross-sectional correlation.

We simulate data following the model introduced in Equation (6.1). The predictors are generated as follows:

- For $i = 1, \dots, k$: $X_{i,t} \sim \text{AR}(1)$ with $X_{i,t} = \phi_i X_{i,t-1} + u_{i,t}$, where $\phi_i \sim \text{Uniform}(0, 0.9)$ and $u_{i,t} \sim \mathcal{N}(0, 1)$ (stationary).
- For $j = k+1, \dots, k+p$: $X_{j,t} \sim \text{ARIMA}(1, 1, 0)$ with $\Delta X_{j,t} = \phi_j \Delta X_{j,t-1} + v_{j,t}$, where $\phi_j \sim \text{Uniform}(0, 0.9)$ and $v_{j,t} \sim \mathcal{N}(0, 1)$ (nonstationary).
- $\beta_0 = 0$, and the true coefficient vector $\beta \in \mathbb{R}^{k+p}$ is sparse.

We set $k = 500$, $p = 500$, and assign nonzero values to the first 10 coefficients of each group:

- Stationary variables: $\beta = [1.1,\ 1.2,\ 1.1,\ 1.5,\ -1.1,\ 0.3,\ -0.2,\ 0.8,\ -0.6,\ 0.7]$.
- Nonstationary variables: $\beta = [1.1,\ 1.3,\ 1.2,\ 1.1,\ -1.1,\ 0.4,\ -0.2,\ 0.6,\ -0.9,\ 0.3]$.

All other coefficients are set to zero.

We perform simulations for two sample sizes: $N = 200$ and $N = 400$, with 500 replications for each case. The LASSO regularization parameter λ is selected using a rolling-window cross-validation procedure. A separate test set of 50 observations is used for forecasting evaluation.

Evaluation metrics are defined as follows:

- True Positives (TP): Number of truly nonzero coefficients that are correctly identified (i.e., estimated as nonzero).
- False Positives (FP): Number of truly zero coefficients that are incorrectly estimated as nonzero.
- False Negatives (FN): Number of truly nonzero coefficients that are incorrectly estimated as zero.
- True Negatives (TN): Number of truly zero coefficients correctly estimated as zero.
- Mean Squared Error (MSE): Mean squared difference between predicted and actual responses on the test set.

Results.

The simulation outcomes are reported in Table 6.1 and Table 6.2. The results show that as the sample size increases, the LASSO estimator demonstrates improved selection consistency, true positives approach the correct sparsity level while false positives and false negatives decline.

Table 6.1: Simulation results for Example 6.1: Independent predictors with mixed stationarity ($N = 200$). LASSO performance with λ ranging from 0.06 to 0.17 and cross-validated λ_{cv}.

	MSE	TP	FP	TN	FN
Settings: $N = 200$, *500 AR(1)* $I(0)$ *rvs, 500 ARIMA(1,1,0)* $I(1)$ *rvs,* $s = 20$					
$\lambda = 0.06$	0.00131	18.80 (0.88)	1.20 (0.88)	883.24 (6.36)	96.76 (6.36)
$\lambda = 0.11$	0.00141	18.42 (1.04)	1.58 (1.04)	917.34 (7.70)	62.66 (7.70)
$\lambda = 0.17$	0.00178	17.75 (0.97)	2.25 (0.97)	937.74 (6.44)	42.26 (6.44)
$\lambda = \lambda_{cv}$	0.00124	18.95 (0.87)	1.05 (0.87)	890.65 (16.01)	89.35 (16.01)

Table 6.2: Simulation results for Example 1: Independent predictors with mixed stationarity ($N = 400$). LASSO performance with λ ranging from 0.06 to 0.17 and cross-validated λ_{cv}.

	MSE	TP	FP	TN	FN
Settings: $N = 400$, *500 AR(1)* $I(0)$ *rvs, 500 ARIMA(1,1,0)* $I(1)$ *rvs,* $s = 20$					
$\lambda = 0.06$	0.00031	19.96 (0.20)	0.04 (0.20)	885.65 (7.91)	94.35 (7.91)
$\lambda = 0.11$	0.00036	19.75 (0.50)	0.25 (0.50)	940.90 (6.24)	39.10 (6.24)
$\lambda = 0.17$	0.00059	19.44 (0.62)	0.56 (0.62)	960.33 (4.79)	19.67 (4.79)
$\lambda = \lambda_{cv}$	0.00029	19.95 (0.26)	0.05 (0.26)	894.09 (18.63)	85.91 (18.63)

Example 7.2: Correlated Predictors

We now consider a more general setting where predictors are correlated. This example examines the performance of the LASSO estimator when covariates exhibit cross-sectional dependence. In contrast to Example 6.1, where predictors were assumed independent, here correlation is introduced separately among stationary and nonstationary series to mimic more complex interdependencies frequently observed in financial and macroeconomic data.

We generate data based on the model specified in Equation (6.1). The predictors are simulated as follows:

- For $i = 1, \dots, k$ (stationary): $X_{i,t} = \phi_i X_{i,t-1} + \varepsilon_{i,t}$, where $\phi_i \sim \text{Uniform}(0, 0.9)$. The error terms $\varepsilon_{i,t} \sim \mathcal{N}(0, 1)$ are jointly Gaussian with correlation structure $\text{Corr}(\varepsilon_{i,t}, \varepsilon_{j,t}) = \rho^{|i-j|}$, where $\rho \in \{0.25, 0.5, 0.75, 0.9\}$.

- For $j = k+1, \dots, k+p$ (nonstationary): $\Delta X_{j,t} = \phi_j \Delta X_{j,t-1} + Z_{j,t}$, where $\phi_j = 0.8$ and $Z_{j,t} \sim \mathcal{N}(0, 1)$ with $\text{Corr}(Z_{i,t}, Z_{j,t}) = \rho^{|i-j|}$, where $\rho \in \{0.25, 0.5, 0.75, 0.9\}$.

- The intercept is set to $\beta_0 = 0$, and the true coefficient vector $\beta \in \mathbb{R}^{k+p}$ is sparse.

We again set $k = 500$, $p = 500$, and assign nonzero coefficients as follows:

- Stationary variables: $\beta = [1.1,\ 1.2,\ 1.1,\ 1.5,\ -1.1,\ 0.3,\ -0.2,\ 0.8,\ -0.6,\ 0.7]$.

- Nonstationary variables: $\beta = [1.1,\ 1.3,\ 1.2,\ 1.1,\ -1.1,\ 0.4,\ -0.2,\ 0.6,\ -0.9,\ 0.3]$.

All remaining coefficients are set to zero.

We consider two sample sizes: $N = 200$ and $N = 400$. For each sample size and each correlation setting $\rho \in \{0.25, 0.5, 0.75, 0.9\}$, we generate 500 independent replications. The regularization parameter is fixed at $\lambda = 0.1$ and the same evaluation metrics are applied. Forecasting accuracy is evaluated using a test set of 50 observations.

Results.

Simulation results are presented in Table 6.3 and Table 6.4, highlighting the deterioration in performance as predictor correlation increases.

Table 6.3: Simulation results for Example 6.2: Correlated predictors with mixed stationarity ($N = 200$). LASSO performance under varying correlation levels ρ with λ fixed at 0.1.

Correlation ρ	**MSE**	**TP**	**FP**	**TN**	**FN**
Settings: $N = 200$, 500 AR(1) stationary, 500 ARIMA(1,1,0) nonstationary, $s = 20$					
$\rho = 0.25$	0.00185	17.93 (1.00)	2.07 (1.00)	911.97 (7.69)	68.03 (7.69)
$\rho = 0.5$	0.00348	16.71 (1.26)	3.29 (1.26)	911.42 (8.21)	68.58 (8.21)
$\rho = 0.75$	0.00942	13.28 (1.71)	6.72 (1.71)	913.75 (7.21)	66.25 (7.21)
$\rho = 0.9$	0.01641	7.50 (2.12)	12.50 (2.12)	927.82 (7.06)	52.18 (7.06)

Table 6.4: Simulation results for Example 6.2: Correlated predictors with mixed stationarity ($N = 400$). LASSO performance under varying correlation levels ρ with λ fixed at 0.1.

Correlation ρ	**MSE**	**TP**	**FP**	**TN**	**FN**
Settings: $N = 400$, 500 AR(1) stationary, 500 ARIMA(1,1,0) nonstationary, $s = 20$					
$\rho = 0.25$	0.00032	19.86 (0.35)	0.14 (0.35)	932.20 (6.96)	47.80 (6.96)
$\rho = 0.5$	0.00042	19.73 (0.51)	0.27 (0.51)	928.71 (7.71)	51.29 (7.71)
$\rho = 0.75$	0.00110	18.60 (1.02)	1.40 (1.02)	922.81 (8.65)	57.19 (8.65)
$\rho = 0.9$	0.00521	15.65 (1.53)	4.35 (1.53)	921.19 (9.06)	58.81 (9.06)

The simulation results show that LASSO can work quite well when predictors are independent, even if they include a mix of stationary and nonstationary time series. In Example 6.1, as the sample size increases from $N = 200$ to $N = 400$, LASSO does a better job at correctly identifying the important variables, while keeping both the number of missed signals and extra false selections low. However, in Example 6.2, when predictors are correlated, the selection performance drops as the correlation becomes stronger. With higher correlation levels, LASSO tends to pick more irrelevant variables and miss more true ones, and the prediction error also becomes larger.

These results suggest that while LASSO is fairly reliable in settings with independent predictors, its performance becomes more unstable when variables are highly correlated—a common situation in economic and financial data. This highlights the need to be cautious and possibly consider additional methods or adjustments when applying LASSO in real-world time series problems.

CHAPTER 7: Discussion

In this dissertation, we explored the modeling and inference in time-varying and high-dimensional time series settings, with a particular focus on financial applications. We began by studying varying-coefficient stochastic diffusion models and the nonparametric estimation procedures for both the drift and diffusion components. Theoretical properties of the estimators were established, and simulation results demonstrated their flexibility and strong empirical performance when applied to U.S. interest rate data.

We then applied a deep learning-based regression framework to model dependencies across multiple financial time series. Our analysis shows that deep learning methods are capable of improving prediction and capturing nonlinear interactions.

Finally, we studied the consistency of LASSO for variable selection in high-dimensional time series regressions with mixed stationary and nonstationary predictors. Through extensive simulations, we assessed how persistence, dependence, and dimensionality affect selection accuracy. The results provide useful guidance for applying LASSO in practical time series contexts.

Future research directions include extending the nonparametric estimation framework to multivariate stochastic systems and developing inferential tools such as confidence bands for time-varying parameters. In the context of deep learning, incorporating uncertainty quantification and exploring hybrid models that integrate statistical structure with neural networks could be the furure working direction.

REFERENCES

[1] Yacine Aît-Sahalia. Nonparametric pricing of interest rate derivative securities. *Econometrica*, 64(3):527–560, 1996.

[2] Yacine Aît-Sahalia. Maximum likelihood estimation of discretely sampled diffusions: A closed-form approximation approach. *Econometrica*, 70(1):223–262, 2002.

[3] Federico M. Bandi and Peter C. B. Phillips. Fully nonparametric estimation of scalar diffusion models. *Econometrica*, 71(1):241–283, 2003.

[4] Sumanta Basu and George Michailidis. Regularized estimation in sparse high-dimensional time series models. *The Annals of Statistics*, 43(4):1535–1567, 2015.

[5] Christoph Bergmeir, Rob J. Hyndman, and Bonsoo Koo. A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120:70–83, 2018.

[6] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[7] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[8] John Y. Campbell and Motohiro Yogo. Efficient tests of stock return predictability. *Journal of Financial Economics*, 81(1):27–60, 2006. doi: 10.1016/j.jfineco.2005.05.008.

[9] K. C. Chan, G. Andrew Karolyi, Francis A. Longstaff, and Anthony B. Sanders. An empirical comparison of alternative models of the short-term interest rate. *The Journal of Finance*, 47(3):1209–1227, 1992.

[10] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*, pages 1724–1734, 2014.

[11] Jianqing Fan and Irene Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman & Hall/CRC, 1996.

[12] Jianqing Fan and Qiwei Yao. *Nonlinear Time Series: Nonparametric and Parametric Methods*. Springer, 2003.

[13] Jianqing Fan and Chunming Zhang. A reexamination of diffusion estimators with applications to financial model validation. *Journal of the American Statistical Association*, 98(464):118–134, 2003.

[14] Jianqing Fan, Jiancheng Jiang, Chunming Zhang, and Zhenwei Zhou. Time-dependent diffusion models for term structure dynamics. *Statistica Sinica*, 13 (4):965–992, 2003.

[15] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[16] Danielle Florens-Zmirou. On estimating the diffusion coefficient from discrete observations. *Journal of Applied Probability*, 30(4):790–804, 1993.

[17] Dean P. Foster and Daniel B. Nelson. Continuous-record asymptotics for rolling sample variance estimators. *Econometrica*, 64(1):139–174, 1996.

[18] Zhonghao Fu, Yongmiao Hong, Liangjun Su, and Xia Wang. Specification tests for time-varying coefficient models. *Journal of Econometrics*, 235(2):720–744, 2023.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[20] Lars Peter Hansen and José A. Scheinkman. Back to the future: Generating moment implications for continuous-time markov processes. *Econometrica*, 63 (4):767–804, 1995.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[22] Shaoxin Hong, Jiancheng Jiang, Xuejun Jiang, and Zhijie Xiao. Unifying inference for semiparametric regression. *Econometrics Journal*, 24(3):482–501, 2021. doi: 10.1093/ectj/utab005.

[23] Wolfgang Härdle and Thomas M. Stoker. Investigating smooth multiple regression by the method of average derivatives. *Journal of the American Statistical Association*, 84(408):986–995, 1989.

[24] Donggyu Kim and Minchul Shin. High-dimensional time-varying coefficient estimation. Working paper, 2024.

[25] Tae-Yong Kim, Heung-Yeol Kim, and Dong-Hyun Oh. Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data. *PLOS ONE*, 14(2):e0212320, 2019.

[26] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st International ACM SIGIR Conference*, pages 95–104, 2018.

[27] Yicong Lin, Mingxuan Song, and Bernhard van der Sluis. Bootstrap inference for linear time-varying coefficient models in locally stationary time series. *Journal of Computational and Graphical Statistics*, 34(2):654–667, 2025.

[28] Andrew W. Lo. Statistical tests of contingent-claims asset-pricing models: A new methodology. *Journal of Financial Economics*, 17:143–173, 1986.

[29] Marcelo C Medeiros, Gabriel F. R Vasconcelos, Álvaro Veiga, and Eduardo Zilberman. Forecasting inflation in a data-rich environment: The benefits of machine learning methods. *Journal of Business & Economic Statistics*, 39(1): 98–119, 2021.

[30] Ziwei Mei and Zhentao Shi. On lasso for high dimensional predictive regression. *Journal of Econometrics*, 242(2), 2024.

[31] Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

[32] Song Song and Peter J. Bickel. Large vector autoregressions. *arXiv preprint arXiv:1106.3915*, 2011.

[33] Richard Stanton. A nonparametric model of term structure dynamics and the market price of interest rate risk. *The Journal of Finance*, 52(5):1973–2002, 1997.

[34] Leonard J. Tashman. Out-of-sample tests of forecasting accuracy: An analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000.

[35] Jialin Tian. Forecasting of share prices based on hybrid model of cnn and lstm: A multi-factor approach. *In Proceedings of the 1st International Conference on E-commerce and Artificial Intelligence (ICDEBA)*, pages 222–228, 2024. doi: 10.5220/0013213700004568.

[36] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones,

Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[38] M.P. Wand and M.C. Jones. *Kernel Smoothing.* Chapman and Hall, London, 1995.

[39] Yifan Zeng, Qi Zhang, and Wei Wang. A cnn-gru hybrid model for financial time series prediction. *Expert Systems with Applications*, 208:118165, 2023. doi: 10.1016/j.eswa.2022.118165.

[40] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

APPENDIX A: Regularity Conditions

A1. The functions $\alpha_0(t),\ \alpha_1(t),\ \beta_0(t),\ \beta_1(t)$ are each C^3 in a neighborhood of t_0, with all derivatives up to order 3 are bounded:

$$\sup_{|t-t_0|\le\varepsilon} \left|\alpha_i^{(k)}(t)\right| < \infty, \qquad \sup_{|t-t_0|\le\varepsilon} \left|\beta_j^{(k)}(t)\right| < \infty, \quad k = 0,1,2,3.$$

A2. The sequence $\{(X_{t_i}, \varepsilon_{t_i})\}$ is $\alpha-$mixing with rate $\alpha(k) = O(k^{-a})$ for some $a > 2$. Moreover, for some $\delta > 0$,

$$\sup_i E\left[|X_{t_i}|^{4+\delta}\right] < \infty, \quad \sup_i E\left[|\varepsilon_{t_i}|^{2+\delta}\right] < \infty.$$

A3. The observation times t_i have a continuous density $f_T(t)$ near t_0 satisfying

$$0 < \inf_{|t-t_0|\le\varepsilon} f_T(t) \le \sup_{|t-t_0|\le\varepsilon} f_T(t) < \infty.$$

A4. K is a bounded symmetric density supported on $[-1, 1]$ with

$$\int u\,K(u)\,du = 0, \quad \mu_2 = \int u^2 K(u)\,du < \infty, \quad R(K) = \int K(u)^2\,du < \infty.$$

The bandwidth $h = h_n$ and time-step Δ satisfy as $n \to \infty$:

$$n\,h \ \to\ \infty, \quad n\,h^3 \ \to\ 0, \quad n\,h^5 \ \to\ 0, \quad \sqrt{n\,h}\,\Delta \ \to\ 0.$$

A5. The matrix

$$I_X(t_0) \ =\ f_T(t_0)\,E\left[X_{t_0} X_{t_0}^\top\right]$$

is positive-definite.

A6. The drift estimator $\widehat{\mu}(t, x)$ used to form

$$\widehat{E}_{t_i} = \frac{Y_{t_i} - \widehat{\mu}(t_i, X_{t_i})\,\Delta}{\sqrt{\Delta}}$$

satisfies

$$\sup_{t,x} \big|\widehat{\mu}(t, x) - \mu(t, x)\big| \;=\; o_P(h).$$

A7. $\inf_t \beta_0(t) > 0$.

A8. Let $\psi_i(\theta)$ be the local score of the pseudo-likelihood at θ_0. There exists $\varepsilon > 0$ such that

$$\sup_i E\big\|\psi_i(\beta_0(t_0), \beta_1(t_0))\big\|^{2+\varepsilon} < \infty.$$

APPENDIX B: Proof of Theorem in Section 4.2

Proof of Theorem 4.1

Recall that the local constant estimator of $\alpha(t_0)$ is given by

$$\widehat{\alpha}(t_0) = S_n^{-1} R_n,$$

where

$$S_n = \sum_{i=1}^{n} K\left(\frac{t_i - t_0}{h}\right) X_i X_i^\top, \quad R_n = \sum_{i=1}^{n} K\left(\frac{t_i - t_0}{h}\right) X_i \frac{Y_{t_i}}{\Delta}.$$

Define the centered and scaled error as

$$\delta_n := \sqrt{nh}\left[\widehat{\alpha}(t_0) - \alpha(t_0) - h^2 B(t_0)\right], \tag{B.1}$$

where $B(t_0) = \frac{\mu_2}{2 f_T(t_0)} I_X(t_0)^{-1} \alpha''(t_0)$.

Since $S_n \widehat{\alpha}(t_0) = R_n$, it follows that

$$S_n \left[\alpha(t_0) + h^2 B(t_0) + \frac{\delta_n}{\sqrt{nh}}\right] = R_n.$$

Rewriting the above equation we get,

$$\frac{S_n}{nh}\delta_n = \frac{R_n - S_n \alpha(t_0) - S_n h^2 B(t_0)}{\sqrt{nh}}. \tag{B.2}$$

We now analyze the right-hand side. First, we expand R_n using the Euler–Maruyama approximation:

$$\frac{Y_{t_i}}{\Delta} = \alpha_0(t_i) + \alpha_1(t_i) X_{t_i} + \sigma(t_i, X_{t_i}) \Delta^{-1/2} \varepsilon_{t_i} + \frac{r_{t_i}}{\Delta}.$$

Substituting into R_n, we obtain

$$R_n = A_n + M_n + R_n^{(r)},$$

where

$$\begin{aligned}
A_n &:= \sum_{i=1}^{n} K_i X_i X_i^\top \alpha(t_i), \\
M_n &:= \sum_{i=1}^{n} K_i X_i \sigma(t_i, X_{t_i}) \Delta^{-1/2} \varepsilon_{t_i}, \\
R_n^{(r)} &:= \sum_{i=1}^{n} K_i X_i \frac{r_{t_i}}{\Delta}, \quad K_i := K\left(\frac{t_i - t_0}{h}\right).
\end{aligned}$$

We assume the remainder term satisfies

$$\frac{R_n^{(r)}}{\sqrt{nh}} = o_P(1),$$

which holds since $r_{t_i} = O(\Delta^2)$ and $\sum K_i = O(nh)$.

We now expand A_n. Using the second-order Taylor expansion of $\alpha(t_i)$ around t_0 is,

$$\alpha(t_i) = \alpha(t_0) + h u_i \alpha'(t_0) + \tfrac{1}{2} h^2 u_i^2 \alpha''(t_0) + O(h^3 |u_i|^3), \quad u_i := \frac{t_i - t_0}{h}.$$

Thus,

$$\begin{aligned}
A_n = \alpha(t_0) \sum_{i=1}^{n} K_i X_i X_i^\top + h\, \alpha'(t_0) \sum_{i=1}^{n} u_i K_i X_i X_i^\top \qquad \text{(B.3)} \\
+ \tfrac{1}{2} h^2\, \alpha''(t_0) \sum_{i=1}^{n} u_i^2 K_i X_i X_i^\top + \sum_{i=1}^{n} O(h^3 |u_i|^3) K_i X_i X_i^\top. \qquad \text{(B.4)}
\end{aligned}$$

Under regularity conditions A4, the following approximations hold:

$$\sum_{i=1}^{n} K_i X_i X_i^\top = nh\, f_T(t_0)\, I_X(t_0) + o_P(nh), \tag{B.5}$$

$$\sum_{i=1}^{n} u_i K_i X_i X_i^\top = o_P(nh), \tag{B.6}$$

$$\sum_{i=1}^{n} u_i^2 K_i X_i X_i^\top = nh\, \mu_2\, f_T(t_0)\, I_X(t_0) + o_P(nh), \tag{B.7}$$

$$\sum_{i=1}^{n} O(h^3|u_i|^3) K_i X_i X_i^\top = o_P(nh^3). \tag{B.8}$$

Now we will show the above approximation:

Recall that

$$K_i = K_h(t_i - t_0) = \frac{1}{h}\, K\Big(\frac{t_i - t_0}{h}\Big), \quad u_i = \frac{t_i - t_0}{h},$$

and $K_i = 0$ whenever $|u_i| > 1$.

Since $t_i = t_0 + i\Delta$ with $\Delta = T/n$ and the sampling density $f_T(t) = 1/\Delta$ is continuous near t_0, a Riemann-sum argument shows that for any smooth matrix-valued function $G(t)$,

$$\sum_{i=1}^{n} G(t_i) \;=\; f_T(t_0) \int_{t_0-h}^{t_0+h} G(t)\, dt \;+\; o(n).$$

Apply this with

$$G(t) = \frac{1}{h}\, K\Big(\tfrac{t-t_0}{h}\Big) \big[X_t X_t^\top\big].$$

Then the integral

$$\int_{t_0-h}^{t_0+h} \frac{1}{h}\, K\Big(\tfrac{t-t_0}{h}\Big) \big[X_t X_t^\top\big]\, dt \xrightarrow{u=(t-t_0)/h} \int_{-1}^{1} K(u) \big[X_{t_0+uh} X_{t_0+uh}^\top\big]\, du,$$

and as $h \to 0$ the integrand converges uniformly to $[X_{t_0} X_{t_0}^\top]$. Hence

$$\frac{1}{nh} \sum_{i=1}^{n} K_i\, X_i X_i^\top \xrightarrow{P} I_X(t_0), \quad \text{so} \quad \sum_{i=1}^{n} K_i\, X_i X_i^\top = n\, h\, I_X(t_0) + o_P(nh).$$

Following exactly the same steps give

$$\frac{1}{nh} \sum_{i=1}^{n} u_i\, K_i\, X_i X_i^\top \xrightarrow{P} [X_{t_0} X_{t_0}^\top] \int_{-1}^{1} u\, K(u)\, du = 0,$$

since $\int uK(u)\, du = 0$. Thus

$$\sum_{i=1}^{n} u_i\, K_i\, X_i X_i^\top = o_P(nh).$$

Similarly,

$$\frac{1}{nh} \sum_{i=1}^{n} u_i^2\, K_i\, X_i X_i^\top \xrightarrow{P} [X_{t_0} X_{t_0}^\top] \int_{-1}^{1} u^2 K(u)\, du = \frac{\mu_2}{f_T(t_0)}\, I_X(t_0),$$

so

$$\sum_{i=1}^{n} u_i^2\, K_i\, X_i X_i^\top = n\, h\, \mu_2\, f_T(t_0)\, I_X(t_0) + o_P(nh).$$

For the last term:

$$\sum_{i=1}^{n} O\big(h^3 |u_i|^3\big)\, K_i\, X_i X_i^\top$$

Noticing that each summand in it has an factor h^3, so by the same Riemann-sum argument

$$\sum_{i=1}^{n} O\big(h^3 |u_i|^3\big)\, K_i\, X_i X_i^\top = O_P(nh^4) = o_P(nh^3).$$

Combining the above, we find

$$A_n - S_n \alpha(t_0) = \tfrac{1}{2} n h^3 \mu_2 f_T(t_0) I_X(t_0) \alpha''(t_0) + o_P(nh^3), \tag{B.9}$$

$$= n h^3 f_T(t_0) I_X(t_0) B(t_0) + o_P(nh^3). \tag{B.10}$$

Hence,

$$A_n - S_n\alpha(t_0) - S_n h^2 B(t_0) = o_P(nh^{5/2}),$$

and so

$$\frac{A_n - S_n\alpha(t_0) - S_n h^2 B(t_0)}{\sqrt{nh}} = o_P(1).$$

It remains to study the stochastic term M_n. Note that $\mathbb{E}[M_n] = 0$, and its variance is

$$\mathrm{Var}(M_n) = \sum_{i=1}^{n} K_i^2\, \mathbb{E}[\|X_i\|^2\sigma^2(t_i, X_{t_i})\Delta^{-1}] = nh\, R(K)\frac{1}{\Delta}\mathbb{E}[\|X_{t_0}\|^2\sigma^2(t_0, X_{t_0})] + o(nh).$$

Under mixing and moment conditions A2, a central limit theorem yields

$$\frac{M_n}{\sqrt{nh}} \xrightarrow{d} \mathcal{N}\left(0,\ R(K)\,\frac{1}{\Delta}\,\mathbb{E}[\|X_{t_0}\|^2\sigma^2(t_0, X_{t_0})]\right). \tag{B.11}$$

Combining everything into (B.2), using Slutsky's theorem, and the fact that $S_n/(nh) \to f_T(t_0)I_X(t_0)$, we obtain:

$$\delta_n = \sqrt{nh}\left[\widehat{\alpha}(t_0) - \alpha(t_0) - h^2B(t_0)\right] \xrightarrow{d} \mathcal{N}(0, \Sigma(t_0)),$$

where

$$\Sigma(t_0) = I_X(t_0)^{-1}\left[R(K)\, f_T(t_0)\,\mathbb{E}\left(X_{t_0}X_{t_0}^{\top}\frac{\sigma^2(t_0, X_{t_0})}{\Delta}\right)\right] I_X(t_0)^{-1}.$$

Proof of Theorem 4.2

We fix a target time point t_0 and consider the local pseudo-log-likelihood function based on discretely observed data from a time-varying diffusion process. The localized criterion is given by

$$\ell_n(\theta) = \sum_{i=1}^{n} K_h(t_i - t_0)\,\ell_i(\theta),$$

where each contribution to the likelihood is

$$\ell_i(\theta) = -\frac{1}{2}\left\{\log\left[\beta_0^2 X_{t_i}^{2\beta_1}\right] + \frac{\widehat{E}_{t_i}^2}{\beta_0^2 X_{t_i}^{2\beta_1}}\right\},$$

and the standardized residual is defined as

$$\widehat{E}_{t_i} = \frac{X_{t_{i+1}} - X_{t_i} - \widehat{\mu}(t_i, X_{t_i})\Delta}{\sqrt{\Delta}}.$$

Let $\theta_0 = \theta(t_0)$ denote the true parameter vector at time t_0, and define $a_n = (nh)^{-1/2}$, $\delta = \sqrt{nh}(\theta - \theta_0)$. We aim to derive the asymptotic distribution of the local estimator $\widehat{\theta}(t_0)$ by analyzing the normalized objective function

$$I_n(\delta) = \sum_{i=1}^{n} K_h(t_i - t_0)\left\{\ell_i(\theta_0 + a_n\delta) - \ell_i(\theta_0)\right\}.$$

Using the smoothness condition in Assumption A1, we perform a second-order Taylor expansion of $\ell_i(\theta_0 + a_n\delta)$ around θ_0. Let

$$\psi_i(\theta) = \begin{pmatrix} \frac{\partial}{\partial\beta_0}\ell_i(\theta) \\ \\ \frac{\partial}{\partial\beta_1}\ell_i(\theta) \end{pmatrix}, \quad h_i(\theta) = -\begin{pmatrix} \frac{\partial^2}{\partial\beta_0^2}\ell_i(\theta) & \frac{\partial^2}{\partial\beta_0\,\partial\beta_1}\ell_i(\theta) \\ \\ \frac{\partial^2}{\partial\beta_1\,\partial\beta_0}\ell_i(\theta) & \frac{\partial^2}{\partial\beta_1^2}\ell_i(\theta) \end{pmatrix}.$$

denote the score and observed information, respectively. Then

$$\ell_i(\theta_0 + a_n\delta) = \ell_i(\theta_0) + a_n\psi_i(\theta_0)^\top\delta - \frac{1}{2}a_n^2\delta^\top h_i(\theta_0)\delta + R_{i,n}(\delta), \tag{B.12}$$

Due to the condition $nh \to \infty, \quad nh^5 \to 0, \quad \Delta = O(h)$ in Assumption A4, which together ensure that $\sum_{i=1}^{n} K_h R_{i,n}(\delta) = O_P(\|\delta\|^3 (nh)^{-1/2}) = o_P(1)$.

Substituting B.12 into $I_n(\delta)$, we obtain

$$I_n(\delta) = a_n\delta^\top S_n - \frac{1}{2}a_n^2\delta^\top T_n\delta + o_P(1), \tag{B.13}$$

where

$$S_n = \sum_{i=1}^{n} K_h(t_i - t_0)\psi_i(\theta_0), \quad T_n = \sum_{i=1}^{n} K_h(t_i - t_0)h_i(\theta_0).$$

Then we want to study the asymptotic behavior of the normalized hessian.

Define the scaled matrix

$$C_n = \frac{T_n}{nh}.$$

We show that $C_n \xrightarrow{P} I(t_0)$, where $I(t_0)$ denotes the Fisher information matrix at time t_0. Specifically: For the expectation,

$$\mathbb{E}[C_n] = \frac{1}{nh}\sum_{i=1}^{n} K_h(t_i - t_0)\mathbb{E}[h_i(\theta_0) \mid t_i].$$

Under Assumption A4 and writing $m(t) = \mathbb{E}[h_i(\theta_0) \mid t_i = t]$, the Riemann sum approximation yields

$$E\left[C_n\right] = \frac{1}{n}\sum_{|u_i|\le 1} K\left(u_i\right) m\left(t_0 + hu_i\right)\frac{1}{h\Delta} \xrightarrow{n\to\infty} \frac{m\left(t_0\right)}{\Delta}\int_{-1}^{1} K(u)du = f_T\left(t_0\right) m\left(t_0\right) = I\left(t_0\right), \tag{B.14}$$

where $f_T(t_0)$ is the sampling density at t_0.

For the variance, using the α-mixing property (Assumption A2) and bounded moments,

$$Var\left(C_n\right) = \frac{1}{(nh)^2}\sum_{i,j} K_h\left(t_i - t_0\right) K_h\left(t_j - t_0\right) Cov\left(h_i, h_j\right) = O\left(\frac{1}{nh}\right)$$

Thus $C_n \xrightarrow{P} I\left(t_0\right)$.

We now derive the asymptotic distribution of S_n. First, we evaluate its mean:

$$\mathbb{E}[S_n] = \sum_{i=1}^{n} K_h(t_i - t_0)\,\mathbb{E}[\psi_i(\theta_0) \mid t_i] = \frac{1}{2}\,n\,h^3\,\mu_2\,f_T(t_0)\,\partial_t^2\big(\mathbb{E}[\psi_i(\theta_0) \mid t_i]\big)\Big|_{t_i=t_0} + o(nh^3), \tag{B.15}$$

using a second-order expansion in t_i and the symmetry $\int uK(u)du = 0$.

Write the conditional mean of the score as a second-order Taylor expansion about t_0:

$$m(t_i) \;=\; \big[\psi_i(\theta_0) \mid t_i\big] \;=\; m(t_0) + m'(t_0)\,(t_i - t_0) + \tfrac{1}{2}\,m''(t_0)\,(t_i - t_0)^2 + O\big(|t_i - t_0|^3\big), \tag{B.16}$$

where we note that $m(t_0) = 0$ because the true score has zero mean.

Substituting B.16 into the expected local sum gives

$$E[S_n] = \sum_{i=1}^{n} K_h(t_i - t_0)\,m(t_i) = \sum_{i=1}^{n} K_h(t_i - t_0)\Big[m(t_0) + m'(t_0)\,(t_i - t_0) + \tfrac{1}{2}\,m''(t_0)\,(t_i - t_0)^2 + O\big(|t_i - t_0|^3\big)\Big] \tag{B.17}$$

Because $m(t_0) = 0$ and $\int u\,K(u)\,du = 0$, both the constant and linear $(t_i - t_0)$ terms vanish under the symmetric kernel. Hence only the quadratic term contributes to leading order:

$$E[S_n] = \tfrac{1}{2}\,m''(t_0)\sum_{i=1}^{n} K_h(t_i - t_0)\,(t_i - t_0)^2 \;+\; o(nh^3).$$

Finally, use the usual Riemann-sum approximation for the kernel moment:

$$\sum_{i=1}^{n} K_h(t_i - t_0)\,(t_i - t_0)^2 = \sum_{i=1}^{n} \frac{1}{h}\,K\Big(\tfrac{t_i - t_0}{h}\Big)\,h^2\,u_i^2 \;\approx\; n\,h^3\,f_T(t_0)\int u^2 K(u)\,du = n\,h^3\,\mu_2\,f_T(t_0).$$

It follows that

$$E[S_n] = \frac{1}{2}\,n\,h^3\,\mu_2\,f_T(t_0)\,m''(t_0) \;+\; o(nh^3), \quad m''(t_0) = \frac{d^2}{dt^2}[\psi_i(\theta_0) \mid t_i]\Big|_{t_i=t_0}. \tag{B.18}$$

That's how we derive B.15.

Now, using α-mixing CLT (A2) and finite $(2+\delta)$ moments of ψ_i, it follows that

$$\begin{aligned} Var(S_n) &= \sum_{i,j} K_h(t_i - t_0)K_h(t_j - t_0)\,\mathrm{Cov}(\psi_i, \psi_j) \\ &= (nh) f_T(t_0) \int K(u)^2\, du \cdot \mathbb{E}\left[\psi_i(\theta_0)\psi_i(\theta_0)^\top \mid t_i = t_0\right] + o(nh), \end{aligned}$$

Hence by the CLT for α-mixing triangular arrays,

$$\frac{S_n}{\sqrt{nh}} \xrightarrow{d} \mathcal{N}(0, V(t_0)), \tag{B.19}$$

with

$$V(t_0) = f_T(t_0) \int K(u)^2 du \cdot \mathbb{E}\left[\psi_i(\theta_0)\psi_i(\theta_0)^\top \mid t_i = t_0\right].$$

Now since $\widehat{\delta}_n = \arg\max_\delta I_n(\delta)$, setting $\frac{\partial I_n(\delta)}{\partial \delta} = 0$ yields

$$0 = a_n S_n - a_n^2 T_n \widehat{\delta}_n + o_P(1),$$

which implies

$$\widehat{\delta}_n = C_n^{-1} \frac{S_n}{\sqrt{nh}} + o_P(1).$$

Expanding C_n^{-1} using a Neumann series gives

$$C_n^{-1} = I^{-1}(t_0) - hI^{-1}FI^{-1} + o_P(h),$$

where F is a suitable bias term. Then

$$\widehat{\delta}_n = I^{-1}(t_0)\frac{S_n}{\sqrt{nh}} - hI^{-1}FI^{-1}\frac{S_n}{\sqrt{nh}} + o_P(1).$$

Finally we decompose $S_n = (S_n - \mathbb{E}[S_n]) + \mathbb{E}[S_n]$. Then, for the variance component, by equation (B.19),

$$\frac{S_n - \mathbb{E}[S_n]}{\sqrt{nh}} \xrightarrow{d} \mathcal{N}(0, V(t_0)),$$

so

$$\frac{1}{\sqrt{nh}} I^{-1}(t_0)(S_n - \mathbb{E}[S_n]) \xrightarrow{d} \mathcal{N}(0, I^{-1}VI^{-1}).$$

Now for the bias component, using (B.15) and the definition

$$m(t) = \big[\psi_i(\theta_0) \mid t_i = t\big], \quad m''(t_0) = \frac{d^2}{dt^2} m(t)\Big|_{t=t_0},$$

we have

$$E[S_n] = \tfrac{1}{2}\, n\, h^3\, \mu_2\, f_T(t_0)\, m''(t_0) + o\big(nh^3\big),$$

and hence

$$\frac{1}{\sqrt{nh}}\, I_\theta(t_0)^{-1}\, E[S_n] = h^2\, I_\theta(t_0)^{-1}\, \frac{\mu_2}{2\, f_T(t_0)}\, m''(t_0) + o\big(h^2\big) = h^2\, B_\theta(t_0) + o\big(h^2\big),$$

where

$$B_\theta(t_0) = \frac{\mu_2}{2\, f_T(t_0)}\, I_\theta(t_0)^{-1}\, m''(t_0).$$

Combining both, we arrive at the final expansion:

$$\widehat{\theta}(t_0) - \theta_0 = h^2 B(t_0) + \frac{1}{\sqrt{nh}} Z_n + o_P\left(h^2 + \frac{1}{\sqrt{nh}}\right),$$

where $Z_n \xrightarrow{d} \mathcal{N}(0, I^{-1}(t_0)V(t_0)I^{-1}(t_0))$. This completes the proof.