2018-01-01

A Bayesian Model For Spectral Density Estimation

Yi Xie
University of Texas at El Paso, 739934261@qq.com

A BAYESIAN MODEL FOR SPECTRAL DENSITY ESTIMATION

YI XIE

Master's Program in Mathematical Sciences

APPROVED:

Ori Rosen, Ph.D., Chair

Naijun Sha, Ph.D.

James M. Wood, Ph.D.

Charles Ambler, Ph.D.
Dean of the Graduate School

to my

MOTHER and FATHER

with love

A BAYESIAN MODEL FOR SPECTRAL DENSITY ESTIMATION

by

YI XIE

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Mathematical Sciences

THE UNIVERSITY OF TEXAS AT EL PASO

August 2018

Abstract

When we analyze a stationary time series, one of the questions we often meet is how to estimate its spectral density. Many approaches have been proposed to this end. In this paper we estimate the spectral density of a stationary time series nonparametrically. We fit a nonparametric regression model to the log periodogram and use third-degree B-spline functions as basis functions. Since the the number of basis functions is relatively large, we place priors such as random-walk and regularized horseshoe on the coefficients of the basis functions to avoid over-fitting and smooth the log periodogram.

Table of Contents

List of Figures

Chapter 1

Introduction

The following definitions are taken from Shumway and Stoffer (2017).

Definition 1 A discrete time series is a sequence of data points being recorded at specific times. Usually these time points are equally spaced, in which case the time series is denoted by $\{x_t; t = 0, \pm 1, \pm 2, \ldots\}$.

Definition 2 The **mean function** of time series $\{x_t; t = 0, \pm 1, \pm 2, \ldots\}$ is defined as

$$\mu_{xt} = E(x_t),$$

where E denotes the usual expectation operator. When no confusion exists about which time series we are referring to, we will drop a subscript and write μ_{xt} as μ_t.

Definition 3 The **auto-covariance function** of a time series $\{x_t; t = 0, \pm 1, \pm 2, \ldots\}$ is defined as

$$\gamma_x(s,t) = \mathsf{cov}(x_s, x_t) = E((x_s - \mu_s)(x_t - \mu_t))$$

for all s and t. When no possible confusion exists about which time series we are referring to, we will drop the subscript and write $\gamma_x(s,t)$ as $\gamma(s,t)$.

Definition 4 A **weakly stationary** time series is a finite variance process where

1. the mean value function, μ_t, is constant and does not depend on time t, and

2. the auto-covariance function, $\gamma(s,t)$, depends on s and t only through their difference $|s-t|$.

Since the mean function, $E(x_t) = \mu_t$, of a stationary time series is independent of time t, we will write $\mu_t = \mu$. Also, because the auto-covariance function, $\gamma(s,t)$, of a stationary time series, x_t, depends on s and t only through their difference $|s-t|$, we may simplify the notation. Let $s = t+h$, where h represents the time shift or lag. Then

$$\gamma_x(t+h,t) = \mathsf{cov}(x_{t+h}, x_t) = \mathsf{cov}(x_t, x_0) = \gamma(h,0)$$

because the time difference between times $t+h$ and t is the same as the time difference between times h and 0. Thus, the auto-covariance function of a stationary time series does not depend on the time argument t. Henceforth, for convenience, we will drop the second argument of $\gamma(h,0)$.

Definition 5 The **auto-covariance function of a stationary time series** will be written as

$$\gamma(h) = \mathsf{cov}(x_{t+h}, x_t) = E((x_{t+h}-\mu)(x_t-\mu)).$$

Definition 6 A **strictly stationary** time series is one for which the probabilistic behavior of every collection of values and shifted values

$$\{x_{t_1}, x_{t_2}, \ldots, x_{t_k}\} \quad \textit{and} \quad \{x_{t_1+h}, x_{t_2+h}, \ldots, x_{t_k+h}\}$$

are identical, for all $k = 1, 2, \ldots$, all time points $t_1, t_2, \ldots, t_k$, and all time shifts $h = 0, \pm 1, \pm 2, \ldots$.

Definition 7 A time series $\{x_t; t = 0, \pm 1, \pm 2, \ldots\}$ is **ARMA(p,q)** if it is stationary and

$$x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q}$$

with $\phi_p \neq 0$, $\theta_q \neq 0$, and $\sigma_w^2 > 0$. The parameters p and q are called the autoregressive and the moving average orders, respectively. We assume that w_t is a Gaussian white noise series with mean zero and variance σ_w^2.

Definition 8 An **autoregressive model** of order p, abbreviated **AR(p)**, is of the form

$$x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t,$$

where x_t is stationary, and $\phi_1, \phi_2, \ldots, \phi_p$ are constants $\phi_p \neq 0$. We assume that w_t is a Gaussian white noise series with mean zero and variance σ_w^2.

Example. $x_t = x_{t-1} - 0.9x_{t-2} + w_t$ is an AR(2) model, where w_t is white Gaussian noise with σ_w^2.

Definition 9 The **moving average model** of order q, or **MA(q)**, is defined to be

$$x_t = w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q},$$

where x_t is stationary, and $\phi_1, \phi_2, \ldots, \phi_p$ are constants such that $\phi_p \neq 0$. We assume that w_t is a Gaussian white noise series with mean zero and variance σ_w^2.

Example. $x_t = w_t + \theta w_{t-1}$ is an MA(1) model, where w_t is white Gaussian noise with σ_w^2, $\theta \neq 0$.

Definition 10 If the auto-covariance function, $\gamma(h)$, of a stationary process satisfies

$$\sum_{h=-\infty}^{\infty} |\gamma(h)| < \infty,$$

then it has the representation

$$\gamma(h) = \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{2\pi i \omega h} f(\omega) d\omega \quad h = 0, \pm 1, \pm 2, \ldots,$$

where $f(\omega)$ is the **spectral density**. The latter has the representation

$$f(\omega) = \sum_{h=-\infty}^{\infty} \gamma(h) e^{-2\pi i \omega h} \quad -\frac{1}{2} \leq \omega \leq \frac{1}{2}.$$

Properties of the spectral density function:

1. $f(\omega) \geq 0$ for all ω.

2. $f(-\omega) = f(\omega)$, it is an even function.

3. It is a periodic function, $f(\omega + 1) = f(\omega)$.

In addition, putting $h = 0$ in

$$\gamma(h) = \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{2\pi i\omega h} f(\omega) d\omega \quad h = 0, \pm 1, \pm 2, \ldots$$

yields

$$\gamma(0) = Var(x_t) = \int_{-\frac{1}{2}}^{\frac{1}{2}} f(\omega) d\omega,$$

which expresses the total variance as the integrated spectral density over all of the frequencies.

Definition 11 Given data $x_1, \ldots, x_n$, we define the **discrete Fourier transform (DFT)** to be

$$d(\omega_j) = n^{-\frac{1}{2}} \sum_{t=1}^{n} x_t e^{-2\pi i\omega_j t}$$

for $j = 0, 1, \ldots, M$, where the frequencies $w_j = \frac{j}{n}$ are called the **Fourier** or **fundamental frequencies**, $M = [\frac{n-1}{2}]$ (the largest positive integer no greater than $\frac{n-1}{2}$).

Definition 12 Given data $x_1, \ldots, x_n$, we define the **periodogram** to be

$$I(\omega_j) = |d(\omega_j)|^2$$

for $j = 0, 1, \ldots, M$, $M = [\frac{n-1}{2}]$.

When the sample size n is large, $I(\omega_j) \overset{ind}{\sim} \text{Exponential}(f(\omega_j))$, approximately.

Definition 13 Let $z = z_1 + iz_2$, where $i = \sqrt{-1}$ and $z_1, z_1 \overset{iid}{\sim} N(0, \frac{\sigma^2}{2})$. Then

$$f(z_1, z_2) \propto \frac{1}{\sigma} \exp\Big(-\frac{z_1^2}{\sigma^2}\Big) \times \frac{1}{\sigma} \exp\Big(-\frac{z_2^2}{\sigma^2}\Big) = \frac{1}{\sigma^2} \exp\Big(-\frac{z_1^2 + z_2^2}{\sigma^2}\Big).$$

We say z has a **complex normal distribution** with mean 0 and variance σ^2 and denote it as $z \sim CN(0, \sigma^2)$. The pdf of z is given by

$$f(z) \propto \frac{1}{\sigma^2} \exp(-\frac{z^* z}{\sigma^2}),$$

where z^* is the complex conjugate.

Chapter 2

Estimation of spectral densities

Several approaches have been taken to estimating the spectral density nonparametrically.

When the sample size n is large, $d(\omega_j) \overset{ind}{\sim} CN(0, f(\omega_j))$ approximately. This implies

$$g(d(\omega_j)) \propto \frac{1}{f(\omega_j)} \exp\Big\{-\frac{|d(\omega_j)^2|}{f(\omega_j)}\Big\} = \frac{1}{f(\omega_j)} \exp\Big\{-\frac{I(\omega_j)}{f(\omega_j)}\Big\}, \tag{2.1}$$

where $g(d(\omega_j))$ is the pdf of $d(\omega_j)$ and $I(\omega_j)$ is the peridogram. From (2.1) we see that $I(\omega_j) \sim \text{Expon}\Big(f(\omega_j)\Big)$ approximately, where $\text{Expon}\Big(f(\omega_j)\Big)$ denotes the exponential distribution with mean $f(\omega_j)$. Let $\epsilon_j = \frac{I(\omega_j)}{f(\omega_j)}$, then $\epsilon_j \sim \text{Expon}(1)$. It follows that $I(\omega_j) = \epsilon_j f(\omega_j)$. Taking logs of both sides leads to the log-linear model

$$\log(I(\omega_j)) = \log(f(\omega_j)) + \eta_j, \quad \text{for} \quad j = 1, \ldots, M, \tag{2.2}$$

where $\eta_j = \log \epsilon_j \sim \log(\text{Expon}(1)) = \log(\frac{1}{2}\chi^2_2)$. Model (2.2) was used by Wahba (1980) to estimate the spectral density by smoothing splines.

Whittle-Likelihood

Since $I(\omega_j) \sim \text{Expon}\Big\{f(\omega_j)\Big\}$ approximately, we can write the likelihood as follows:

$$\begin{aligned} L(I|f) &\propto \prod_{m=1}^{M} \tfrac{1}{f(\omega_m)} \exp\Big\{-\tfrac{I(\omega_m)}{f(\omega_m)}\Big\} \\ &= \exp\Big\{-\textstyle\sum_{m=1}^{M}[\log f(\omega_m) + \exp\{\log I(\omega_m) - \log f(\omega_m)\}]\Big\}, \end{aligned} \tag{2.3}$$

which is called the Whittle-likelihood (Whittle, 1962).

Pawitan and O'Sullivan (1994), used the penalized Whittle-likelihood to estimate the spectral density of a stationary time series.

Carter and Kohn (1997), used a Bayesian approach where the error term η_j was approximated by a mixture of normal distributions with fixed values.

Some traditional methods for estimating the spectral density (such as averaged periodogram) are mentioned in Shumway and Stoffer (2017).

2.1 Nonparametric regression

If in a regression analysis we assume there is a predetermined relation between independent variables and a dependent variable. Such regression analysis is called parametric, otherwise it is called a nonparametric. In nonparametric regression we need to estimate the form of the relationship between the independent variables and the dependent variable based on observed data. To this end, we need a set of basis functions, whose linear combination will capture the interesting features of the data. B-splines are an example of a possible type of basis functions.

B-splines

To define a family of B-spline functions of order $p+1$ uniquely, two things are needed:

1. A polynomial of degree p (the order of a B-spline function equals the polynomial degree p plus 1).

2. A non-decreasing sequence of knots, $t_1, \ldots, , t_q$.

Then the ith member of of a family of B-splines of order 1 is defined as

$$B_{i,1}(x) := \begin{cases} 1 & \text{if} \quad t_i \leq x < t_{i+1} \\ 0 & \text{otherwise.} \end{cases}$$

B-splines of higher order k are defined recursively as follows,

$$B_{i,k}(x) := \delta_{i,k} B_{i,k-1}(x) + (1 - \delta_{i+1,k}) B_{i+1,k-1}(x),$$

where

$$\delta_{i,k} := \begin{cases} \frac{x - t_i}{t_{i+k-1} - t_i} & \text{if} \quad t_i \neq t_{i+k-1} \\ 0 & \text{otherwise.} \end{cases}$$

Here are some general properties of a B-spline of order $p + 1$, see Eilers and Marx (1996).

1. It consists of $p + 1$ polynomial pieces, each of degree p.
2. The polynomial pieces join at p inner knots.
3. At the joining points, derivatives up to order $p - 1$ are continuous.
4. The B-spline is positive on a domain spanned by $p + 2$ knots; everywhere else it is zero.
5. Except at the boundaries, it overlaps with $2p$ polynomial pieces of its neighbours.
6. At a given $x, p + 1$ B-splines are non-zero.

With these good properties, B-splines are ideal basis functions for nonparametric modeling.

P-splines

Marx and Eilers (1999) proposed a generalized linear regression model for curve fitting, in which the idea of P-splines is proposed. P-splines consist of a combination of B-splines and a second-order difference penalty placed on the coefficients of these B-splines (to control the smoothness of the fitted curve).

Lang and Brezger (2004) developed a Bayesian version of P-splines and put a random-walk prior (up to a second-order) on the B-spline coefficients. In this paper, the first order random-walk prior is defined as

$$\beta_\rho = \beta_{\rho-1} + u_\rho, \tag{2.4}$$

and the second order random-walk prior is defined as

$$\beta_\rho = 2\beta_{\rho-1} - \beta_{\rho-2} + u_\rho,$$

where the β_ρs are B-spline coefficients, $u_\rho \sim N(0, \tau^2)$, and diffuse priors are placed on β_1 (for a first-order random-walk) or β_1 and β_2 (for a second order random-walk prior). The diffuse (improper) prior is $p(\beta_i) \propto 1$, $i = 1, 2$. ($p(\beta_i)$ means the prior on β_i). The amount of smoothness is controlled by the smoothing parameter τ^2, which is a global smoothing parameter. In other words, the same amount of smoothing is applied to different covariate values (frequency in our case).

Bayesian variable selection

P-splines are one way to prevent over-fitting. Another approach is to start with a relatively large number of basis functions and to allow some of the coefficients to be close to zero. This approach is common in Bayesian variable selection for regression models (George and McCulloch, 1997).

Spike and slab prior

One of the first priors on the regression coefficients used in Bayesian variable selection was the spike and slab prior (George and McCulloch, 1997). It is often written as a two-component mixture of Gaussians

$$\beta_i|\rho_i, c \sim \rho_i N(0, c^2) + (1 - \rho_i)N(0, \epsilon^2), \quad \rho_i \sim Ber(\pi), \tag{2.5}$$

where $\rho_i \sim Ber(\pi)$ means ρ_i has a Bernoulli distribution with probability π that $\rho_i = 1$. The parameter c is called the slab width.

In (2.5), the first term on the right-hand side is called slab. The variance c^2 is relatively large so $N(0, c^2)$ has its support over a wide range of plausible values of β_i. The second component is the spike with $\epsilon^2 << c^2$. If we set $\epsilon = 0$, then the spike is called a Dirac's delta at δ_0.

The horseshoe prior

The setting of the horseshoe prior is as follows.

$$\beta_i|\lambda_i, \tau \sim N(0, \lambda_i^2\tau^2), \quad \lambda_i|\sigma \sim C^+(0, \sigma), \quad \tau|\eta \sim C^+(0, \eta). \tag{2.6}$$

In (2.6), $C^+(0, a)$ means the half-Cauchy distribution with scale parameter a. We can see that the level of shrinkage of β_i is controlled by two parameters, λ_i (the local smoothing parameter) and τ (the global smoothing parameter). Thus, the horseshoe prior has the freedom to shrink globally (via τ) and yet act locally (via λ_i). The global parameter τ pulls all the weights globally towards zero, while the thick half-Cauchy tails for the local scales λ_j allow some of the weights to escape the shrinkage. See Carvalho et al. (2010).

The density function of the horseshoe prior (Figure 2.1) has an infinitely tall spike at the origin and flat, Cauchy-like tails. These two features allow β_is with large values to remain large and force small β_is to shrink to values close to zero. So it can accommodate unknown levels of sparsity by changing the value of τ.

In (2.6), set $\tau = \sigma = 1$ and let $\kappa_i = \frac{1}{1+\lambda_i^2}$, then we obtain

$$E(\beta_i|y_i, \lambda_i^2) = \Big(\frac{\lambda_i^2}{1+\lambda_i^2}\Big)y_i + \Big(\frac{1}{1+\lambda_i^2}\Big)0 = (1-\kappa_i)y_i, \tag{2.7}$$

where y_i is the observed data.

Under the setting $\tau = \sigma = 1$, $\lambda_i \sim C^+(0, 1)$, $\kappa_i = \frac{1}{1+\lambda_i^2} \sim Beta(\frac{1}{2}, \frac{1}{2})$. Figure 2.2 shows the density curve of κ_i, which looks like a horseshoe. Most of the mass is concentrated at

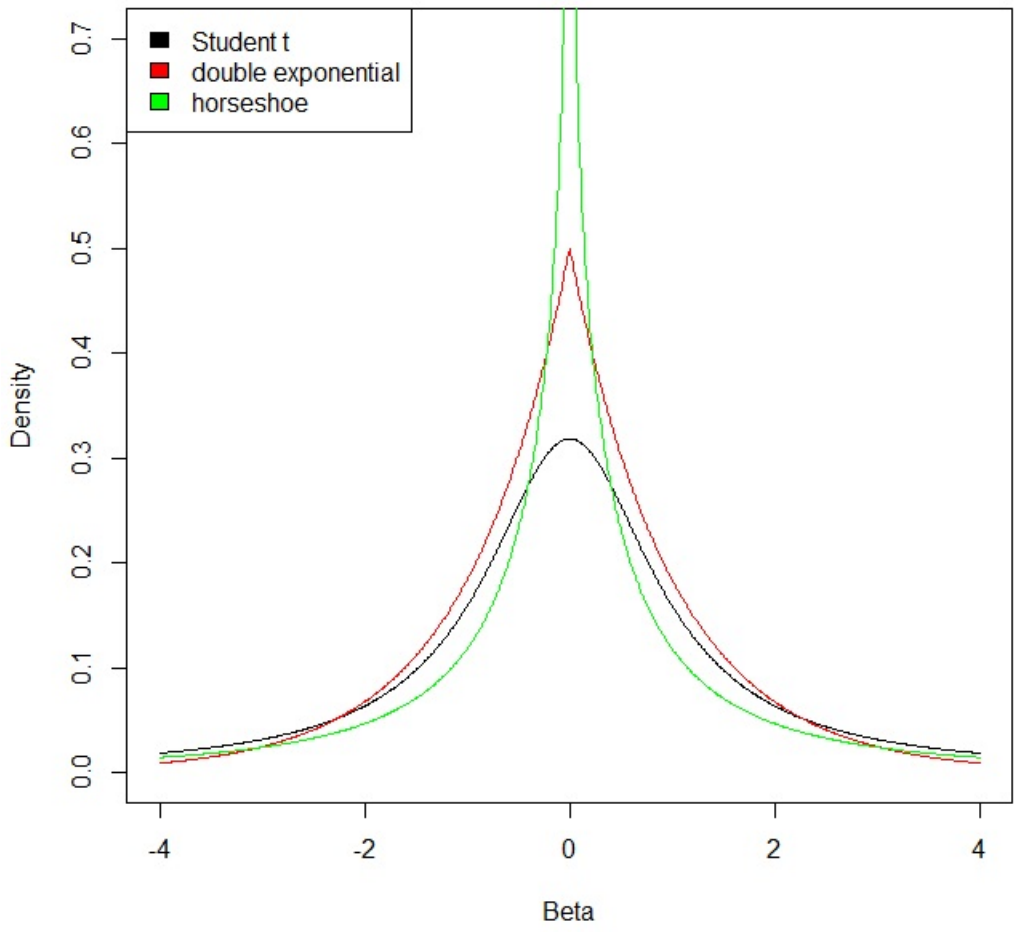

Figure 2.1: horseshoe

$\kappa_i = 0$ and $\kappa_i = 1$.

In (2.7), if $\kappa_i = 0$, then $E(\beta_i|y_i, \lambda_i^2) = y_i$, which means there is no shrinkage. If $\kappa_i = 1$, then $E(\beta_i|y_i, \lambda_i^2) = 0$, which means total shrinkage, see Piironen and Vehtari (2017).

In Bayesian linear regression, we usually assume that regression coefficients β_is are independently normally distributed. In this case, the spike and slab prior can be rewritten as

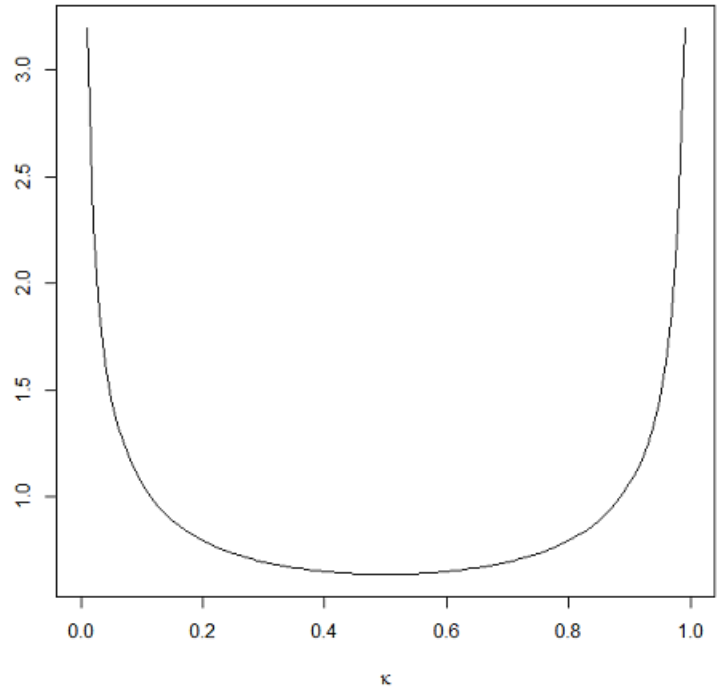

Figure 2.2: Density of the κ_i

$$\beta_i \sim \rho N(0, c^2) + (1 - \rho)\delta_0(\beta_i),$$

where

$$\delta_0(\beta_i) := \begin{cases} 1 & \text{if} \quad \beta_i = 0 \\ 0 & \text{otherwise,} \end{cases}$$

which concentrates all its mass at zero and makes those β_i corresponding to unimportant covariates shrink to zero.

If the value of β_i is not close to zero, then by the slab component, $\beta_i \sim N(0, c^2)$, and

$$E(\beta_i|y_i) = \frac{c^2}{1+c^2} y_i = \Big(1 - \frac{1}{1+c^2}\Big) y_i,$$

so the shrinkage factor is $\kappa_i = \frac{1}{1+c^2}$, which has the same form as $\kappa_i = \frac{1}{1+\lambda_i^2}$. If we set $\lambda_i = c$, then both priors will be the same, which means under this setting, both the horseshoe prior

and the spike and slab prior will assign the same amount of shrinkage to the nonzero β_is. If the value of β_i is close to zero, then by the spike component, it will be shrunk to zero, which means total shrinkage ($\kappa_i = 1$). Thus, the horseshoe prior can closely mimic the spike and slab prior.

The performance of the spike and slab prior mainly depends on the choice of $g(\cdot)$ and ρ. A spike and slab prior is often considered as the 'gold standard' for variable selection. The horseshoe prior often performs better than the spike and slab prior in terms of the mixing of the MCMC (Markov chain Monte Carlo) algorithm. For more details about the Spike and Slab prior and the Horseshoe prior, see Piironen and Vehtari (2017), who also proposed the regularized horseshoe prior as an improvement of the horseshoe prior.

The regularized horseshoe prior

The setting of the regularized horseshoe prior is as follows.

$$\begin{aligned} &\beta_i|\lambda_i, \tau, c \sim N(0, \tilde{\lambda_i}^2\tau^2), \quad \lambda_i \sim C^+(0,1), \quad \tilde{\lambda_i}^2 = \frac{c^2\lambda_i^2}{c^2+\tau^2\lambda_i^2}, \\ &\tau \sim C^+(0, \tau_0), \quad \tau_0 = \frac{p_0}{L-p_0}\frac{\iota}{\sqrt{M}}. \end{aligned} \tag{2.8}$$

In (2.8), p_0 is the number of relevant covariates expected, and L is the total number of coefficients of basis functions. In linear regression, ι is the standard deviation of the error term. In the context of spectral estimation, it is the standard deviation of η_j in (2.1), which is equal to $\frac{\pi}{\sqrt{6}}$.

Compared with the horseshoe prior with $\sigma = 1$, we can see that if β_i is close to 0, then its corresponding local shrinkage parameter λ_i will be small, thus $\tau^2\lambda_i^2 << c^2$. In this case $\tilde{\lambda_i}^2 = \frac{c^2\lambda_i^2}{c^2+\tau^2\lambda_i^2} \approx \lambda_i^2$, which leads to $\beta_i|\lambda_i, \tau, c \sim N(0, \lambda_i^2\tau^2)$, the same as the horseshoe prior.

When β_i has a large value, then its corresponding local shrinkage parameter λ_i will be large, thus $\tau^2\lambda_i^2 >> c^2$. In this case, $\tilde{\lambda_i}^2 = \frac{c^2\lambda_i^2}{c^2+\tau^2\lambda_i^2} \approx \frac{c^2}{\tau^2}$, which leads to $\beta_i|\lambda_i, \tau, c \sim N(0, \frac{c^2}{\tau^2}\tau^2) = N(0, c^2)$. This is identical to the slab term in the spike and slab prior.

Now we can see that, both the horseshoe prior and the regularized horseshoe prior will shrink β_is that are close to 0 in a similar fashion. However, large β_is will be regularized by

the regularized horseshoe prior, but the horseshoe prior will not do any regularization.

Chapter 3

The Proposed Models

The Posterior Distribution

Combining the likelihood with the prior distributions yields the posterior distribution needed for Bayesian inference, i.e.

$$\text{posterior} \propto \text{prior} \times \text{likelihood}. \tag{3.1}$$

In our case, the likelihood is the Whittle-likelihood (2.3), and we let $y_m = \log I(\omega_m)$. Let $\boldsymbol{q}_m = (1, q_{m1}, \ldots, q_{mL})'$, where $q_{m1}, q_{m2}, \ldots, q_{mL}$ are basis functions evaluated at ω_m and let $\boldsymbol{\beta} = (\alpha_0, \beta_1, \ldots, \beta_L)'$ be a vector of unknown coefficients such that $\log f(\omega_m) = \mathbf{q}'_{\mathbf{m}}\boldsymbol{\beta}$. We then rewrite the Whittle-likelihood as

$$\exp\Big\{-\sum_{m=1}^{M}[\boldsymbol{q}'_m\boldsymbol{\beta} + \exp\{y_m - \boldsymbol{q}'_m\boldsymbol{\beta}\}]\Big\}. \tag{3.2}$$

The prior we place on $\boldsymbol{\beta}$ depends on what model we use.

The priors for P-splines

In this case, the posterior distribution is given by

$$P(\beta, \tau|y) \propto \exp\Big\{-\sum_{m=1}^{M}[\boldsymbol{q}'_m\boldsymbol{\beta} + \exp\{y_m - \boldsymbol{q}'_m\boldsymbol{\beta}\}]\Big\} \times P(\boldsymbol{\beta}) \times P(\tau),$$

where

1. The prior on $\boldsymbol{\beta}$ satisfies $P(\boldsymbol{\beta}) = P(\alpha_0) \times P(\beta_1) \times P(\beta_2) \times \cdots \times P(\beta_L)$, where

$$\alpha_0 \sim N(0, 10^2), \beta_1 \sim N(0, \tau^2), \beta_\rho = \beta_{\rho-1} + u_\rho, u_\rho \sim N(0, \tau^2),$$

for $\rho = 2, 3, \ldots, L$.

2. The prior on τ is Half-$t_3(0, 10^3)$, where Half-$t_3(0, 10^3)$ means the half t distribution with degrees of freedom 3, location parameter 0, and scale parameter 10^3. If $\tau \sim$ Half-$t_\nu(\mu, \sigma)$, then its density function is

$$g(\tau) = \frac{2\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\nu\pi\sigma^2}}\Big(1 + \frac{1}{\nu}\frac{\tau^2}{\sigma^2}\Big)^{-\frac{\nu+1}{2}} \quad \text{for } \tau \geq 0.$$

The Horseshoe prior

In this case, the posterior distribution is given by

$$P(\beta, \tau, \lambda|y) \propto \exp\Big\{-\sum_{m=1}^{M}[\boldsymbol{q}_m'\boldsymbol{\beta} + \exp\{y_m - \boldsymbol{q}_m'\boldsymbol{\beta}\}]\Big\} \times P(\boldsymbol{\beta}) \times P(\tau) \times P(\lambda).$$

1. The prior on $\boldsymbol{\beta}$ satisfies $P(\boldsymbol{\beta}) = P(\alpha_0) \times P(\beta_1) \times P(\beta_2) \times \cdots \times P(\beta_L)$, where

$$\alpha_0 \sim N(0, 10^2), \beta_\rho \sim N(0, \lambda_\rho^2\tau^2),$$

for $\rho = 1, 2, \ldots, L$.

2. The prior on λ_ρ is $\lambda_\rho|\sigma \sim C^+(0, \sigma)$. In our case, we let $\sigma = 1$, so $P(\lambda_\rho|\sigma) = \frac{2}{\pi(1+\lambda_\rho^2)}$.

3. The prior on τ is $\tau|\eta \sim C^+(0, \eta)$. In our case, we let $\eta = 1$, so $P(\tau|\eta) = \frac{2}{\pi(1+\tau^2)}$.

The regularized Horseshoe prior

In this case, the posterior distribution is given by

$$P(\beta, \tau, \lambda, c^2, \iota|y) \propto \exp\Big\{-\sum_{m=1}^{M}[\boldsymbol{q}_m'\boldsymbol{\beta} + \exp\{y_m - \boldsymbol{q}_m'\boldsymbol{\beta}\}]\Big\} \times P(\boldsymbol{\beta}) \times P(\tau) \times P(\lambda) \times P(c^2).$$

1. The prior on $\boldsymbol{\beta}$ satisfies $P(\boldsymbol{\beta}) = P(\alpha_0) \times P(\beta_1) \times P(\beta_2) \times \cdots \times P(\beta_L)$, where

$$\alpha_0 \sim N(0, 10^2), \beta_\rho \sim N(0, \tilde{\lambda_\rho}^2 \tau^2),$$

   for $\rho = 1, 2, \ldots, L$.

2. The $\tilde{\lambda_i}^2 = \frac{c^2 \lambda_i^2}{c^2 + \tau^2 \lambda_i^2}$ is a transformed parameter. The prior on λ_ρ is $\lambda_\rho \sim C^+(0, 1)$. The prior on c^2 is Inv-Gamma(15,375), where Inv-Gamma(15,375) means the inverse Gamma distribution with shape parameter 15 and scale parameter 375. It allows c^2 to take larger values, so the amount of shrinkage of large β_is will be small

3. The prior on τ is $\tau \sim C^+(0, \tau_0) = C^+(0, \frac{p_0}{L - p_0} \frac{\iota}{\sqrt{M}})$, where $\iota = \frac{\pi}{\sqrt{6}}$.

Hamiltonian Monte Carlo

The random walk nature of the Metropolis algorithm makes it slow to explore the parameter space and to converge to the target distribution (Gelman et al., 2013).

The Hamiltonian Monte Carlo (HMC) was proposed by Duane et al. (1987), and was first used in statistics by Neal (2011). HMC is based on Hamiltonian dynamics borrowed from physics to reduce the local random walk behaviour of the Metropolis algorithm.

Hamiltonian dynamics use an object's location $\boldsymbol{\beta}$ and momentum $\boldsymbol{\zeta}$ at time t to describe its motion in the system. Each location of the object is associated with potential energy $U(\boldsymbol{\beta})$, and for each momentum there is an associated kinetic energy $K(\boldsymbol{\zeta})$. The sum of these two types of energy is given by $H(\boldsymbol{\beta}, \boldsymbol{\zeta})$.

$$H(\boldsymbol{\beta}, \boldsymbol{\zeta}) = U(\boldsymbol{\beta}) + K(\boldsymbol{\zeta}), \tag{3.3}$$

where the H($\boldsymbol{\beta}$, $\boldsymbol{\zeta}$) is the total energy.

Equation (3.3) leads to the Hamiltonian equations

$$\frac{d\,\boldsymbol{\beta}}{d\,t} = \frac{d\,H}{d\,\boldsymbol{\zeta}} = \frac{d\,K(\boldsymbol{\zeta})}{d\,\boldsymbol{\zeta}},$$

$$\frac{d\,\boldsymbol{\zeta}}{d\,t} = -\frac{d\,H}{d\,\boldsymbol{\beta}} = -\frac{d\,U(\boldsymbol{\beta})}{d\,\boldsymbol{\beta}}.$$

Given $\frac{d\,K(\boldsymbol{\zeta})}{d\,\boldsymbol{\zeta}}$ and $\frac{d\,U(\boldsymbol{\beta})}{d\,\boldsymbol{\beta}}$ and a set of initial values of $\boldsymbol{\zeta}$ and $\boldsymbol{\beta}$, we can use the Hamiltonian equations to predict the location $\boldsymbol{\beta}$ and momentum $\boldsymbol{\zeta}$.

In HMC, we use the vector of unknown coefficients $\boldsymbol{\beta}$ as the location, and for each β_i in $\boldsymbol{\beta}$ we assign a corresponding momentum variable ζ_i. The potential energy $U(\boldsymbol{\beta})$ is the log-posterior density of $\boldsymbol{\beta}$, i.e., $U(\boldsymbol{\beta}) = \log P(\boldsymbol{\beta}|y)$. As for $\boldsymbol{\zeta}$, we assume it has a multivariate normal distribution with independent components, so its variance covariance matrix M is diagonal, i.e., $\zeta_i \sim N(0, M_{i,i})$ where $M_{i,i}$ is the ith diagonal element of M. The kinetic energy is given by $K(\boldsymbol{\zeta}) = \frac{1}{2}\boldsymbol{\beta}^T M^{-1}\boldsymbol{\beta}$.

At the beginning of the HMC iterations, draw random values of $\boldsymbol{\beta}$ and denote them as $\boldsymbol{\beta}^0$, where $\boldsymbol{\beta}^i$ is the value of $\boldsymbol{\beta}$ after the ith HMC iteration.

Then in the ith HMC iteration ($i = 1, 2, \ldots$)

1. Get current values for this iteration. Let $\boldsymbol{\beta} = \boldsymbol{\beta}^{i-1}$, draw a random sample of $\boldsymbol{\zeta}$ from its posterior distribution, $\boldsymbol{\zeta} \sim N(0, M)$ and denote it as $\boldsymbol{\zeta}^0$, let $\boldsymbol{\zeta} = \boldsymbol{\zeta}^0$.

2. Propose a new candidate for the next position. Update $\boldsymbol{\beta}$ and $\boldsymbol{\zeta}$ by R 'leapfrog steps', each scaled by a factor ϵ. In each leapfrog step, we do

   (a) $\boldsymbol{\zeta} = \boldsymbol{\zeta} - \frac{1}{2}\epsilon\frac{d\,\log P(\boldsymbol{\beta}|y)}{d\,\boldsymbol{\beta}}$. Use $\frac{d\,\log P(\boldsymbol{\beta}|y)}{d\,\boldsymbol{\beta}}$ to update $\boldsymbol{\zeta}$ for half a step.

   (b) $\boldsymbol{\beta} = \boldsymbol{\beta} + \epsilon M^{-1}\boldsymbol{\zeta}$. Use ζ and M^{-1} to update $\boldsymbol{\beta}$ for a whole step.

   (c) $\boldsymbol{\zeta} = \boldsymbol{\zeta} - \frac{1}{2}\epsilon\frac{d\,\log P(\boldsymbol{\beta}|y)}{d\,\boldsymbol{\beta}}$. Use $\frac{d\,\log P(\boldsymbol{\beta}|y)}{d\,\boldsymbol{\beta}}$ to update $\boldsymbol{\zeta}$ for another half step.

   After (a), (b) and (c), we have updated both β and ζ by a whole step. Steps (a), (b) and (c) together are called a leapfrog step. At the end of R leapfrog steps, the values of $\boldsymbol{\beta}$ and $\boldsymbol{\zeta}$ are denoted by $\boldsymbol{\beta}^*$ and $\boldsymbol{\zeta}^*$.

3. Compute the acceptance ratio r.

$$r = \frac{P(\boldsymbol{\beta}^*|y)P(\boldsymbol{\zeta}^*)}{P(\boldsymbol{\beta}^{i-1}|y)P(\boldsymbol{\zeta}^0)}.$$

4. Update

$$\boldsymbol{\beta}_i := \begin{cases} \boldsymbol{\beta}^* & \text{with probability min(r,1)} \\ \boldsymbol{\beta}^{i-1} & \text{otherwise.} \end{cases}$$

The main difference between the Metropolis-Hastings method and Hamiltonian Monte Carlo is how they propose the candidate for the next iteration. In the Metropolis-Hastings algorithm, the proposal distribution only depends on $\boldsymbol{\beta}$, but in the Hamiltonian Monte Carlo, the proposal distribution is the joint distribution $P(\boldsymbol{\beta}|y)P(\boldsymbol{\zeta})$, which depends also on $\boldsymbol{\zeta}$. Beside that, the Hamiltonian Monte Carlo also uses the gradient of $\log P(\boldsymbol{\beta}|y)$, so compared with the Metropolis-Hastings algorithm, each single iteration of Hamiltonian Monte Carlo will be more costly but with a higher acceptance rate which allows Hamiltonian Monte Carlo to move faster and reach convergence earlier.

Chapter 4

Simulation Results

Stan and Rstan

Stan is a platform for statistical modeling and high-performance statistical computation. Its name is short for Sampling Through Adaptive Neighborhoods. It has the Hamiltonian Monte Carlo algorithm as a built-in function, so when we fit a Bayesian model in Stan, we can apply HMC directly without tedious work of programming. It can calculate the gradients and set the tuning parameters automatically. Stan has the No-U-Turn sampler (NUTS) too, avoiding the need to set the step size ϵ and the number of leapfrog steps R, which may also speed up convergence.

Stan has interfaces to many popular computing environments like R, Python, Matlab, etc. Rstan allows one to conveniently fit Stan models from R and access the output. Another useful tool is the 'ShinyStan', which can provide interactive visual summaries and advanced posterior analysis of Rstan output.

4.1 Simulation study

To evaluate the performance of the models, we drew ten time series of length 500 each from the following AR(2) model

$$x_t = x_{t-1} - 0.9x_{t-2} + w_t.$$

Stan runs 4 chains by default, each has 4000 iterations, and the first 2000 iterations in each chain are used as burn-in.

4.1.1 P-spline model

We first report the results of fitting the first-order random-walk prior with 3rd degree B-spline basis functions. The number of knots is 30 (these knots are equally spaced, from 0 to 0.5), so the number of basis functions is 32. Under this setting, in (3.2), $M = \frac{[500-1]}{2} = 249$, $L = 32$, $q_{ml} = B_{l,4}(\omega_m)$ for $l = 1, 2, \ldots, 32$ and $\alpha_0 \sim N(0, 10^2)$. The priors on β_ρ are the first-order random-walk (2.4), where $u_p \sim N(0, \tau^2)$ and the prior on τ is $\tau \sim t_3(0, 1000)$, which is the student t distribution with 3 degrees of freedom, location parameter 0, and scale parameter 10^3.

In Figure 4.1, the blue lines are the theoretical log spectral densities and the red ones are their estimates. On the log scale, we can see that the estimated log spectral densities are very close to the theoretical ones, which suggests that the model successfully captures the shapes of the log spectral densities.

In Figure 4.2, on the original scale (exponentiated log scale), the difference between the theoretical spectral densities and the estimated ones is amplified, so we can see the differences between them more clearly. In all the plots the red peaks are below the blue peaks, which implies that this model may not have the ability to capture the peak part of the spectral density.

4.1.2 Horseshoe prior model

In this section, we report the results of fitting the model with the horseshoe prior and the 3rd degree B-spline basis functions. This time the prior on the coefficients of the basis functions is the horseshoe (2.6), with $\eta = \sigma = 1$.

In the output of this model, there is a warning saying there were divergent transitions after the burn-in iterations, which means the results may not be reliable. For this reason,

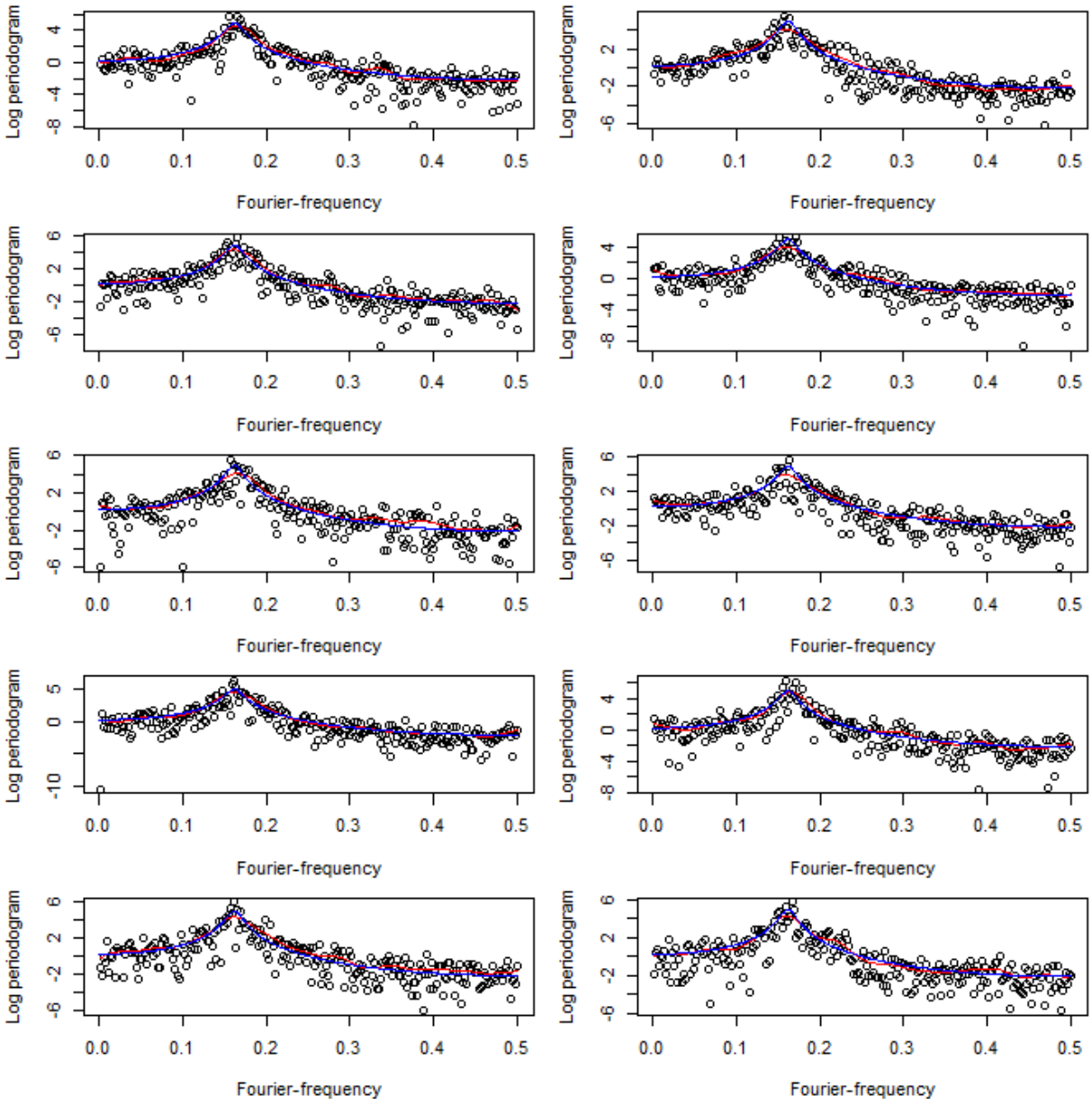

Figure 4.1: The theoretical log spectral densities and their estimates based on the P-spline model.

we do not include in this paper results based on the horseshoe prior. Instead, we report results based on the regularized horseshoe prior.

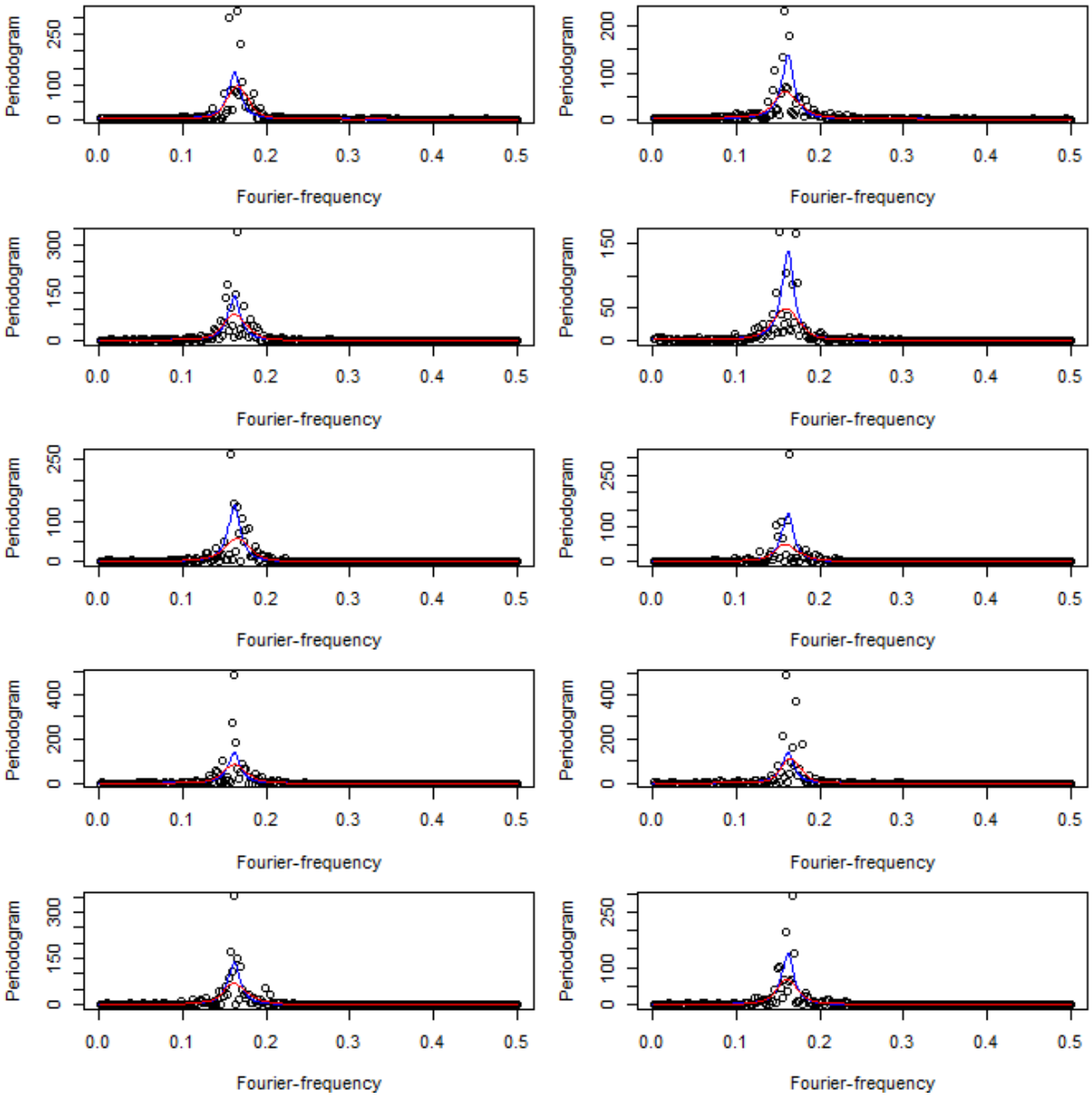

Figure 4.2: The theoretical spectral densities and their estimates based on the P-spline model.

4.1.3 Regularized horseshoe prior model

This section reports results based on the regularized horseshoe prior (2.8), with $p_0 = 20$, $\iota = \frac{\pi}{\sqrt{6}}$ and $c^2 \sim$ Inv-Gamma(15,375).

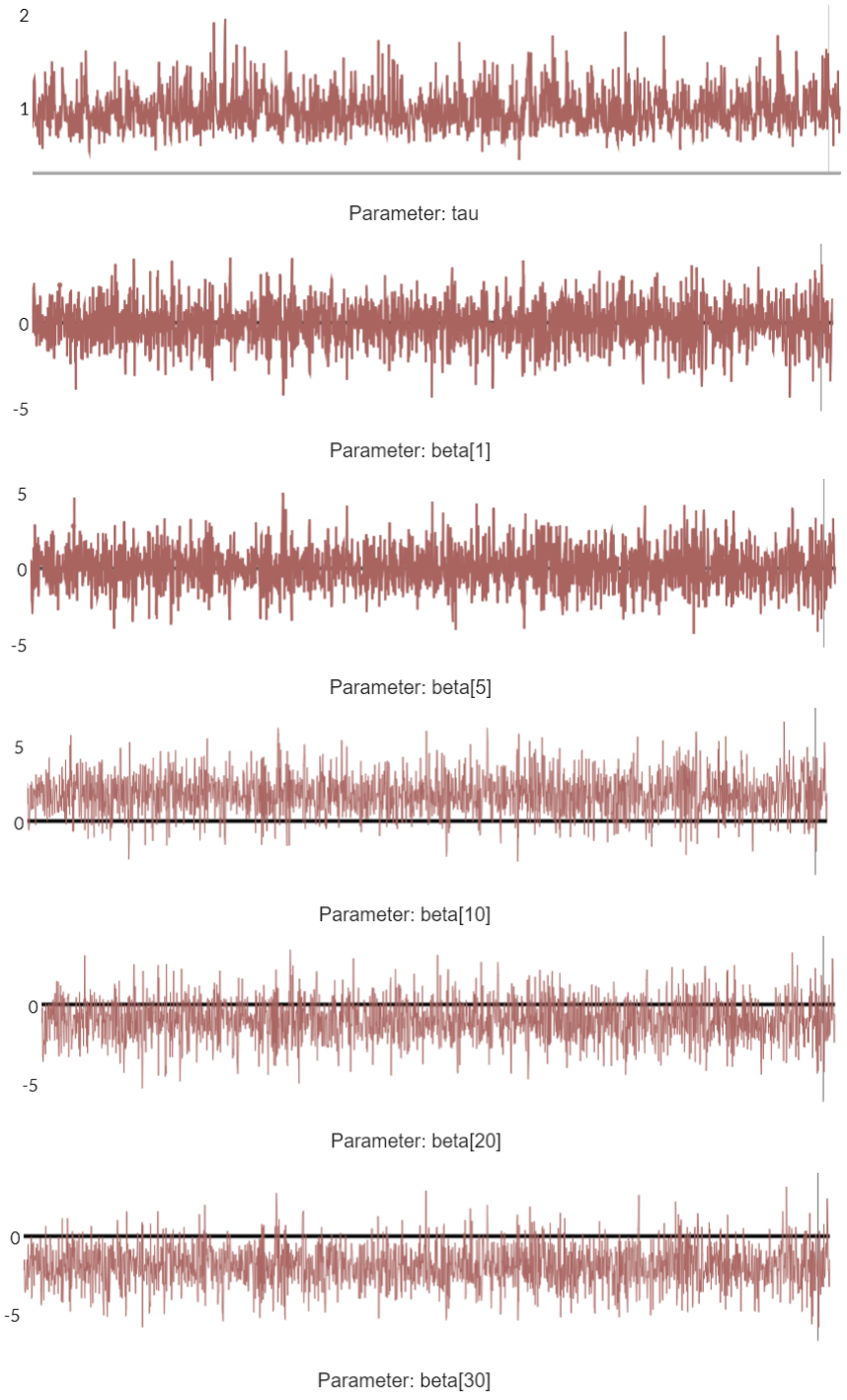

Figure 4.3: The global smoothing parameter τ and some of the βs for the P-spline model.

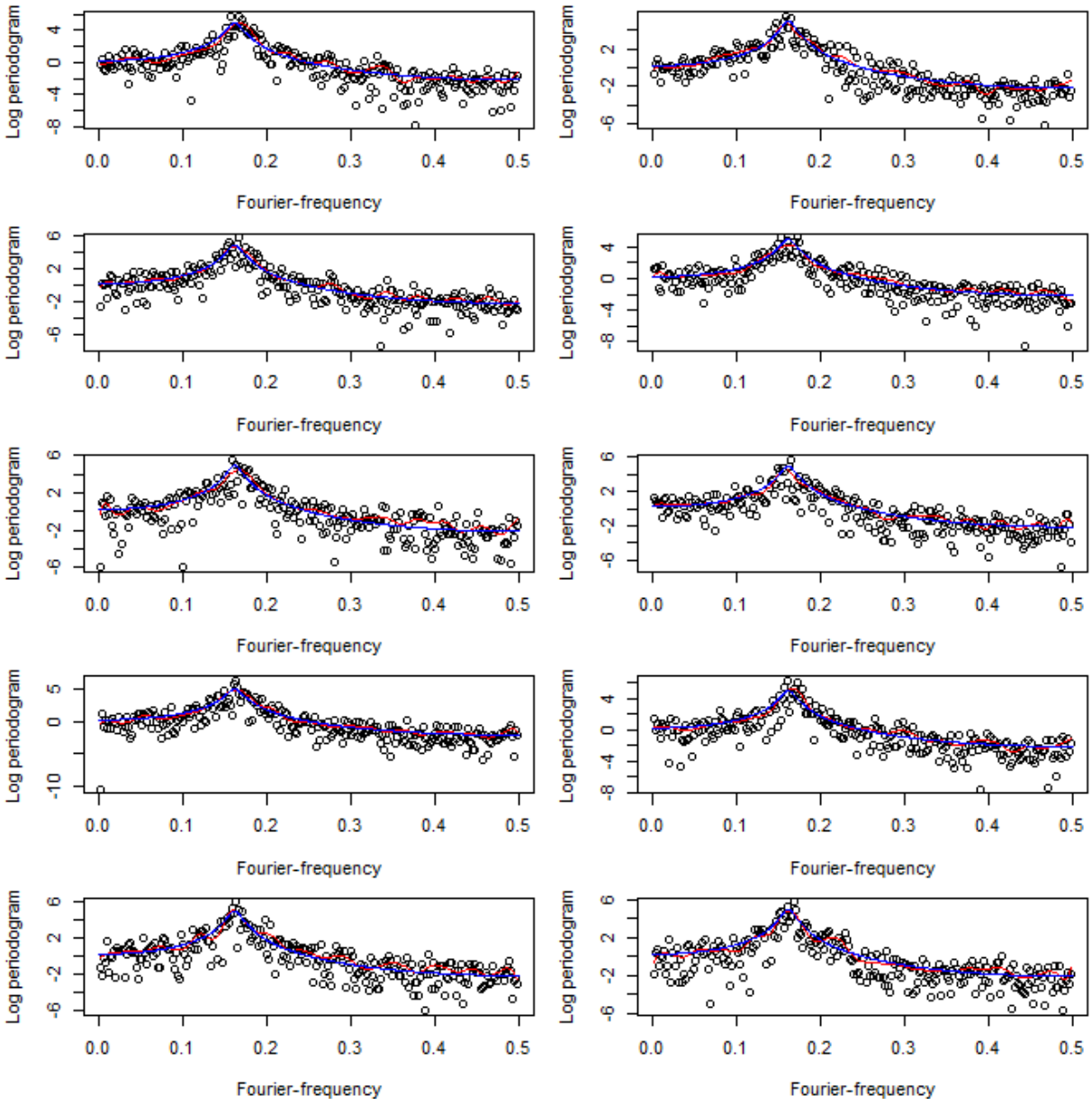

Figure 4.4: The theoretical log spectral density and its estimates based on the regularized horseshoe prior model.

In Figure 4.4, the blue lines are the theoretical log spectral densities and the red ones are the estimated log spectral densities. On the log scale, we can see that estimated log spectral densities are very close to the theoretical ones, which shows that the regularized

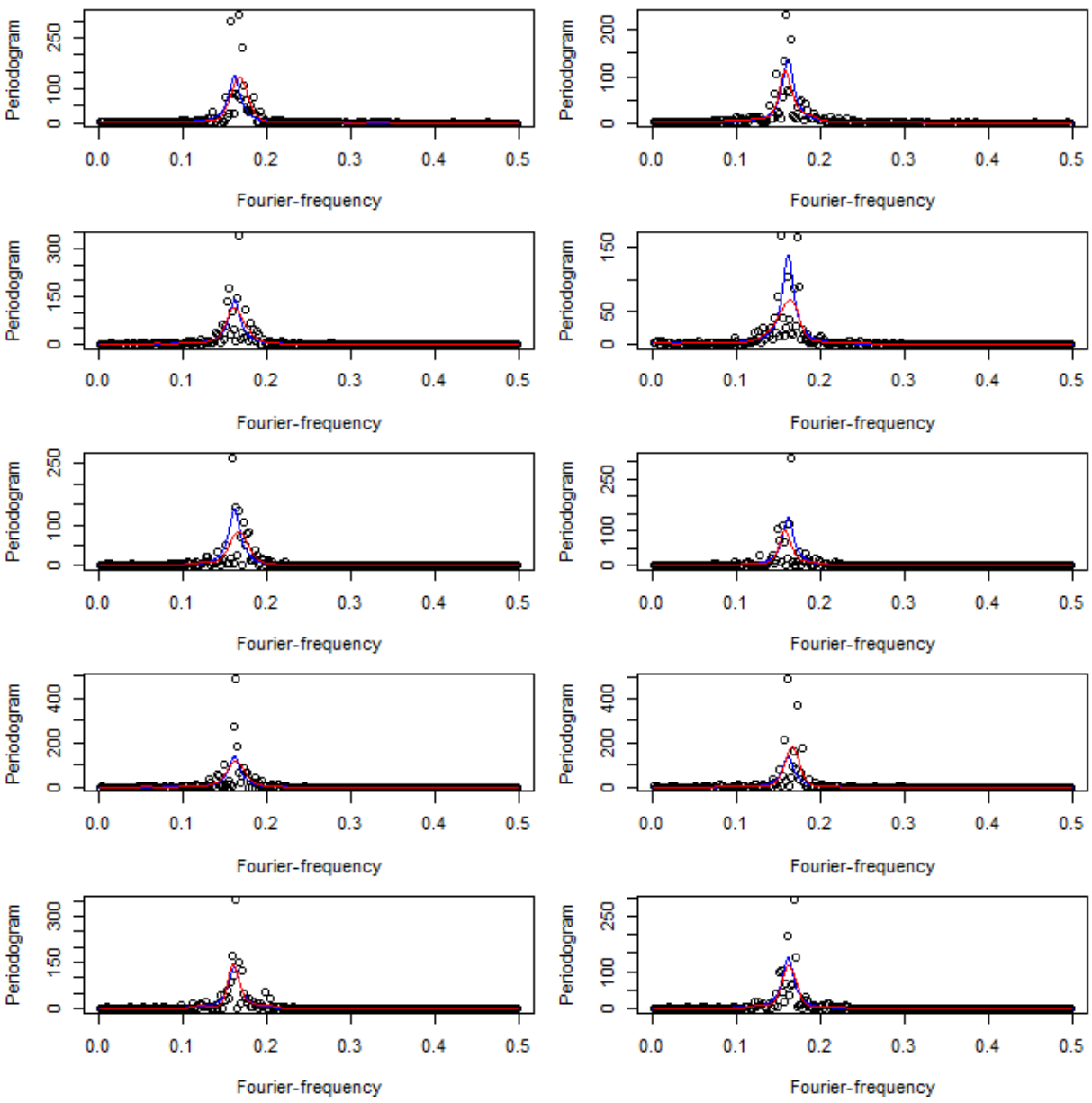

Figure 4.5: The theoretical spectral density and its estimates based on the regularized horseshoe prior model.

horseshoe prior model did a good job.

In Figure 4.5, on the original scale (exponentiated log scale), this time the red peaks are closer to the blue peaks compared to the results based on the P-spline model. Some of

the red peaks are above the blue peaks, which shows that the regularized horseshoe prior model has the capacity to capture the peak part of the spectral density.

4.2 Assessing Convergence

4.2.1 P-spline model

Figure 4.3 shows some of the parameter values of the P-spline model after the burn-in iterations. As we have mentioned before, HMC reaches convergence quite fast and we can see that all of these parameters have reached convergence early on after the burn-in iterations.

Another two useful indices are $\hat{R}$ and n_{eff}, $\hat{R}$ is a potential scale reduction factor on split chains, whereas n_{eff} is the effective number of simulation draws. These two indices can help assess convergence. For more details about these two indices, see chapters 11.4 and 11.5 of Gelman et al. (2013).

As mentioned before, Stan runs 4 chains by default, and $\hat{R}$ measures the consistency of all 4 Markov chains. At convergence, $\hat{R} = 1$, but in practice, we require $\hat{R}$ for each parameter in the model to be smaller than 1.1, otherwise we conclude that convergence has not been reached.

For each parameter, n_{eff} is a crude measure of the effective sample size. Since simulation inference from correlated draws is generally less precise than from the same number of independent draws, we need a large enough number of effective simulation draws to assure the accuracy of our estimation. A good check for such issues is the number of effective samples per iteration; in our model the total number of iterations after burn-in is 8000, so if $\frac{n_{eff}}{8000} < 0.001$, it means we do not have a large enough number of effective simulation draws.

Figure 4.6 shows $\hat{R}$ and n_{eff} for each parameter in the model. We can see that for β_i, $i = 1, 2, \ldots, 33$, the n_{eff}s are all equal to 8000 and the $\hat{R}$s are 1 which suggests that the simulation has reached convergence.

4.2.2 Regularized horseshoe prior model

Figures 4.7 and 4.8 display trace plots for some of the parameters of the regularized horseshoe model following the burn-in iterations. We can see once again that all these parameters have reached convergence shortly after the burn-in iterations.

Figure 4.9 shows the $\hat{R}$ and n_{eff} of each parameter in the regularized horseshoe model. All of the $\hat{R}$s are 1 which implies that all of them have reached convergence. The n_{eff} are all greater than 4000, $\frac{n_{eff}}{8000} > 0.5 >> 0.001$. Therefore, the number of effective samples per iteration is large enough to indicate the results are reliable.

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
beta_raw[1]	-0.09	0.01	1.32	-2.68	-0.96	-0.10	0.76	2.58	8000	1
beta_raw[2]	0.02	0.01	0.99	-1.91	-0.65	0.02	0.69	1.99	8000	1
beta_raw[3]	-0.05	0.01	0.86	-1.74	-0.62	-0.04	0.53	1.62	8000	1
beta_raw[4]	0.29	0.01	0.71	-1.12	-0.19	0.28	0.77	1.67	8000	1
beta_raw[5]	0.49	0.01	0.67	-0.81	0.03	0.48	0.94	1.78	8000	1
beta_raw[6]	-0.24	0.01	0.64	-1.51	-0.66	-0.24	0.19	1.02	8000	1
beta_raw[7]	-0.41	0.01	0.63	-1.64	-0.84	-0.41	0.01	0.82	8000	1
beta_raw[8]	0.52	0.01	0.64	-0.71	0.09	0.53	0.94	1.79	8000	1
beta_raw[9]	0.57	0.01	0.64	-0.68	0.14	0.57	1.00	1.86	8000	1
beta_raw[10]	0.31	0.01	0.62	-0.89	-0.09	0.30	0.72	1.55	8000	1
beta_raw[11]	0.94	0.01	0.69	-0.35	0.46	0.93	1.39	2.34	8000	1
beta_raw[12]	1.92	0.01	0.65	0.68	1.48	1.90	2.35	3.26	8000	1
beta_raw[13]	0.41	0.01	0.61	-0.81	0.00	0.41	0.83	1.63	8000	1
beta_raw[14]	-1.92	0.01	0.64	-3.20	-2.36	-1.92	-1.48	-0.69	8000	1
beta_raw[15]	-1.36	0.01	0.67	-2.70	-1.81	-1.35	-0.91	-0.12	8000	1
beta_raw[16]	-0.23	0.01	0.69	-1.62	-0.68	-0.22	0.25	1.09	8000	1
beta_raw[17]	-0.91	0.01	0.63	-2.16	-1.33	-0.90	-0.49	0.32	8000	1
beta_raw[18]	-0.29	0.01	0.64	-1.59	-0.70	-0.28	0.13	0.96	8000	1
beta_raw[19]	0.12	0.01	0.63	-1.16	-0.29	0.13	0.54	1.34	8000	1
beta_raw[20]	-1.22	0.01	0.61	-2.46	-1.62	-1.22	-0.81	-0.02	8000	1
beta_raw[21]	-0.14	0.01	0.62	-1.40	-0.55	-0.15	0.28	1.08	8000	1
beta_raw[22]	0.72	0.01	0.62	-0.52	0.32	0.72	1.13	1.94	8000	1
beta_raw[23]	-0.24	0.01	0.63	-1.50	-0.66	-0.24	0.18	0.99	8000	1
beta_raw[24]	-1.32	0.01	0.64	-2.58	-1.75	-1.31	-0.89	-0.09	8000	1
beta_raw[25]	0.06	0.01	0.66	-1.29	-0.37	0.07	0.51	1.33	8000	1
beta_raw[26]	-0.03	0.01	0.61	-1.24	-0.44	-0.04	0.38	1.18	8000	1
beta_raw[27]	0.26	0.01	0.60	-0.94	-0.14	0.26	0.68	1.43	8000	1
beta_raw[28]	-0.19	0.01	0.66	-1.47	-0.64	-0.20	0.25	1.10	8000	1
beta_raw[29]	-0.27	0.01	0.62	-1.48	-0.69	-0.27	0.15	0.96	8000	1
beta_raw[30]	0.28	0.01	0.63	-0.95	-0.14	0.28	0.70	1.50	8000	1
beta_raw[31]	-0.29	0.01	0.65	-1.54	-0.73	-0.30	0.15	1.00	8000	1
beta_raw[32]	0.12	0.01	0.71	-1.27	-0.35	0.12	0.60	1.50	8000	1
beta_raw[33]	-0.06	0.01	0.79	-1.58	-0.61	-0.06	0.48	1.49	8000	1
tau	1.07	0.00	0.21	0.73	0.92	1.05	1.19	1.56	3190	1
beta[1]	-0.09	0.01	1.32	-2.68	-0.96	-0.10	0.76	2.58	8000	1
beta[2]	0.02	0.01	1.08	-2.10	-0.67	0.02	0.73	2.19	8000	1
beta[3]	-0.03	0.02	1.43	-2.90	-0.95	-0.02	0.89	2.77	8000	1
beta[4]	0.26	0.02	1.41	-2.55	-0.66	0.28	1.18	3.02	8000	1
beta[5]	0.78	0.02	1.41	-2.02	-0.14	0.78	1.69	3.58	8000	1
beta[6]	0.53	0.02	1.39	-2.25	-0.35	0.53	1.44	3.22	8000	1
beta[7]	0.08	0.02	1.40	-2.74	-0.82	0.12	1.00	2.79	8000	1
beta[8]	0.64	0.02	1.41	-2.20	-0.29	0.66	1.57	3.41	8000	1
beta[9]	1.23	0.02	1.38	-1.55	0.35	1.24	2.14	3.94	8000	1
beta[10]	1.55	0.02	1.39	-1.27	0.63	1.57	2.46	4.27	8000	1
beta[11]	2.51	0.02	1.39	-0.31	1.61	2.52	3.41	5.23	8000	1
beta[12]	4.51	0.02	1.37	1.74	3.63	4.51	5.39	7.23	8000	1
beta[13]	4.95	0.02	1.40	2.18	4.04	4.94	5.85	7.73	8000	1
beta[14]	2.93	0.02	1.39	0.08	2.02	2.95	3.86	5.64	8000	1
beta[15]	1.51	0.02	1.41	-1.42	0.60	1.55	2.45	4.24	8000	1
beta[16]	1.31	0.02	1.38	-1.42	0.42	1.32	2.22	4.06	8000	1
beta[17]	0.35	0.02	1.40	-2.47	-0.56	0.39	1.27	3.07	8000	1
beta[18]	0.05	0.02	1.40	-2.78	-0.87	0.07	0.98	2.90	8000	1
beta[19]	0.20	0.02	1.39	-2.54	-0.70	0.18	1.13	2.93	8000	1
beta[20]	-1.09	0.02	1.38	-3.89	-1.97	-1.07	-0.18	1.58	8000	1
beta[21]	-1.24	0.02	1.40	-4.11	-2.14	-1.20	-0.31	1.43	8000	1
beta[22]	-0.47	0.02	1.39	-3.26	-1.36	-0.47	0.43	2.26	8000	1
beta[23]	-0.72	0.02	1.40	-3.49	-1.61	-0.73	0.20	2.05	8000	1
beta[24]	-2.12	0.02	1.40	-4.97	-3.02	-2.09	-1.16	0.56	8000	1
beta[25]	-2.03	0.02	1.38	-4.76	-2.92	-2.02	-1.14	0.73	8000	1
beta[26]	-2.07	0.02	1.40	-4.93	-2.97	-2.07	-1.14	0.63	8000	1
beta[27]	-1.79	0.02	1.40	-4.55	-2.70	-1.78	-0.89	0.93	8000	1
beta[28]	-1.99	0.02	1.39	-4.74	-2.90	-1.98	-1.09	0.72	8000	1
beta[29]	-2.28	0.02	1.40	-5.13	-3.20	-2.26	-1.37	0.42	8000	1
beta[30]	-1.98	0.02	1.38	-4.78	-2.88	-1.98	-1.08	0.74	8000	1
beta[31]	-2.29	0.02	1.42	-5.15	-3.20	-2.29	-1.35	0.50	8000	1
beta[32]	-2.16	0.02	1.44	-4.98	-3.10	-2.15	-1.21	0.67	8000	1
beta[33]	-2.23	0.02	1.47	-5.11	-3.17	-2.23	-1.26	0.63	8000	1
lp__	-260.76	0.13	5.41	-272.26	-264.24	-260.46	-256.97	-251.11	1830	1

Figure 4.6: Convergence measures $\hat{R}$ and n_{eff} for the parameters of the P-spline model.

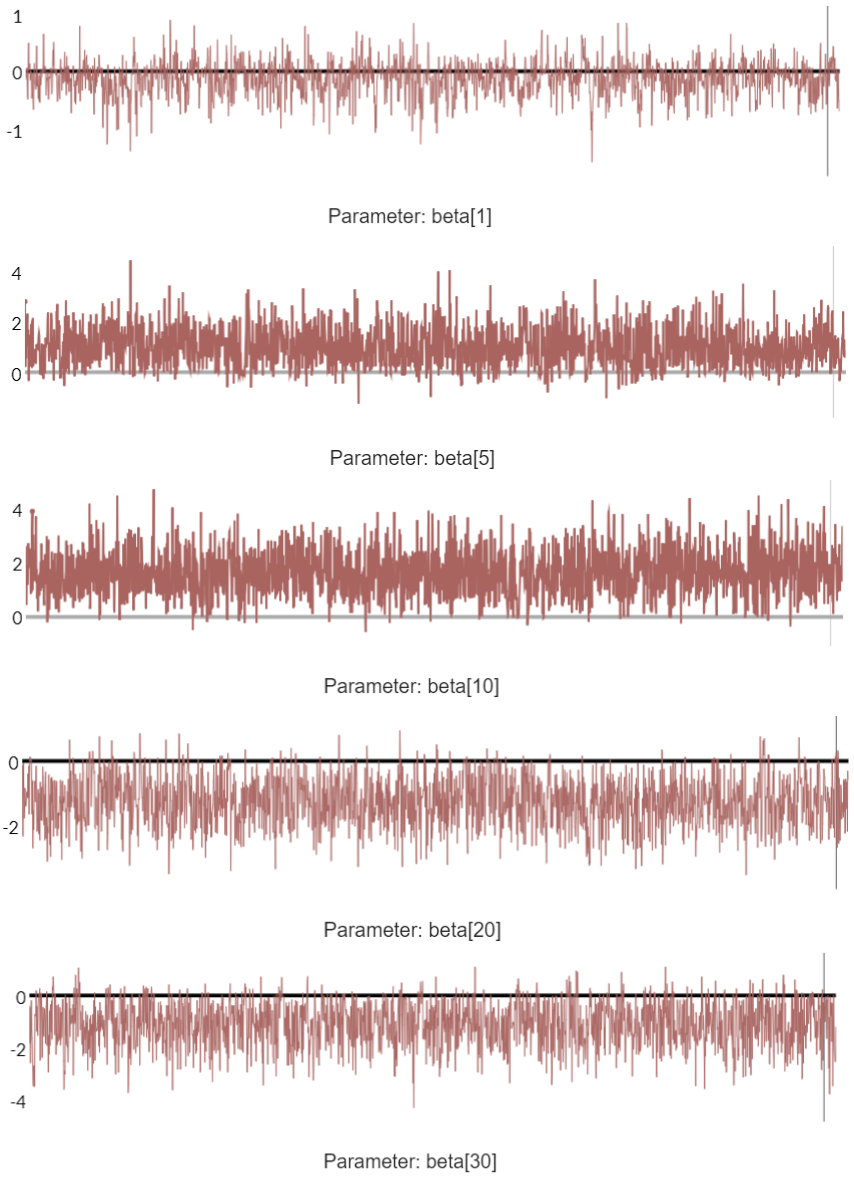

Figure 4.7: Trace plots of some of the βs for the regularized horseshoe model.

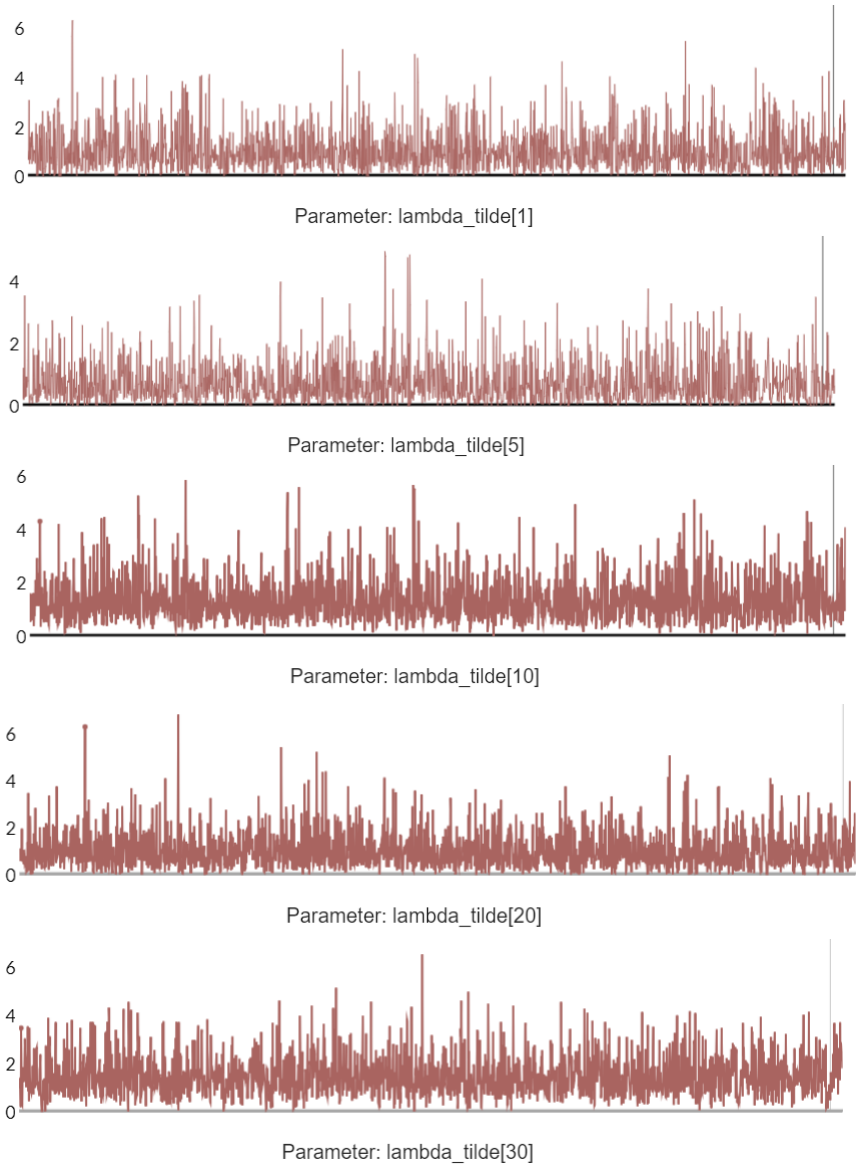

Figure 4.8: Trace plots for some of the $\tilde{\lambda}$s corresponding to βs of the regularized horseshoe model.

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
tau	2.33	0.02	1.14	0.97	1.60	2.07	2.78	5.13	3207	1
c2	24.00	0.07	6.38	14.60	19.52	22.92	27.34	39.38	8000	1
lambda_tilde[1]	1.03	0.01	0.87	0.03	0.36	0.83	1.47	3.27	8000	1
lambda_tilde[2]	0.72	0.01	0.69	0.02	0.22	0.51	1.01	2.52	8000	1
lambda_tilde[3]	0.72	0.01	0.71	0.02	0.22	0.50	1.00	2.68	8000	1
lambda_tilde[4]	0.97	0.01	0.79	0.05	0.41	0.78	1.33	3.03	8000	1
lambda_tilde[5]	0.71	0.01	0.68	0.02	0.22	0.51	1.00	2.54	8000	1
lambda_tilde[6]	0.67	0.01	0.67	0.02	0.20	0.46	0.93	2.45	8000	1
lambda_tilde[7]	0.85	0.01	0.75	0.03	0.32	0.65	1.17	2.75	8000	1
lambda_tilde[8]	1.06	0.01	0.80	0.07	0.50	0.88	1.42	3.06	6635	1
lambda_tilde[9]	1.21	0.01	0.83	0.13	0.61	1.01	1.62	3.30	8000	1
lambda_tilde[10]	1.35	0.01	0.86	0.25	0.74	1.15	1.73	3.55	6224	1
lambda_tilde[11]	1.92	0.01	0.96	0.61	1.24	1.73	2.41	4.26	5247	1
lambda_tilde[12]	1.97	0.01	0.98	0.65	1.27	1.76	2.46	4.41	5329	1
lambda_tilde[13]	1.51	0.01	0.85	0.39	0.91	1.33	1.92	3.68	6470	1
lambda_tilde[14]	0.91	0.01	0.77	0.04	0.35	0.71	1.26	2.89	8000	1
lambda_tilde[15]	1.29	0.01	0.84	0.20	0.69	1.11	1.68	3.40	5595	1
lambda_tilde[16]	0.65	0.01	0.66	0.02	0.19	0.46	0.90	2.45	8000	1
lambda_tilde[17]	0.66	0.01	0.64	0.02	0.21	0.47	0.91	2.40	8000	1
lambda_tilde[18]	0.75	0.01	0.68	0.03	0.26	0.56	1.05	2.54	8000	1
lambda_tilde[19]	1.03	0.01	0.76	0.08	0.49	0.86	1.39	2.93	8000	1
lambda_tilde[20]	1.04	0.01	0.80	0.07	0.49	0.85	1.40	3.05	8000	1
lambda_tilde[21]	0.75	0.01	0.70	0.02	0.24	0.54	1.04	2.62	8000	1
lambda_tilde[22]	0.67	0.01	0.67	0.02	0.20	0.48	0.93	2.43	8000	1
lambda_tilde[23]	1.59	0.01	0.88	0.44	0.95	1.41	2.01	3.79	5652	1
lambda_tilde[24]	1.05	0.01	0.80	0.07	0.48	0.86	1.44	3.11	8000	1
lambda_tilde[25]	1.41	0.01	0.86	0.31	0.80	1.23	1.81	3.60	6549	1
lambda_tilde[26]	1.18	0.01	0.84	0.10	0.58	1.00	1.57	3.35	8000	1
lambda_tilde[27]	1.12	0.01	0.81	0.08	0.54	0.93	1.51	3.08	8000	1
lambda_tilde[28]	1.57	0.01	0.91	0.41	0.92	1.36	2.00	3.81	5739	1
lambda_tilde[29]	0.96	0.01	0.77	0.05	0.40	0.79	1.31	2.89	6996	1
lambda_tilde[30]	1.53	0.01	0.92	0.27	0.88	1.35	2.00	3.76	6227	1
lambda_tilde[31]	1.16	0.01	0.87	0.07	0.53	0.98	1.58	3.35	8000	1
lambda_tilde[32]	1.01	0.01	0.78	0.06	0.45	0.84	1.38	3.02	6509	1
beta[1]	-0.13	0.01	0.32	-0.76	-0.33	-0.13	0.07	0.49	3205	1
beta[2]	-0.03	0.00	0.08	-0.23	-0.06	-0.01	0.01	0.14	3915	1
beta[3]	-0.05	0.01	0.77	-1.61	-0.47	-0.03	0.33	1.61	8000	1
beta[4]	0.20	0.01	0.71	-1.16	-0.17	0.10	0.58	1.83	8000	1
beta[5]	1.03	0.01	0.80	-0.25	0.41	0.99	1.59	2.67	5447	1
beta[6]	0.39	0.01	0.64	-0.70	-0.03	0.26	0.77	1.85	8000	1
beta[7]	-0.02	0.01	0.62	-1.32	-0.36	-0.01	0.29	1.33	6505	1
beta[8]	0.75	0.01	0.79	-0.49	0.12	0.63	1.26	2.49	5881	1
beta[9]	1.23	0.01	0.82	-0.15	0.64	1.20	1.79	2.90	5894	1
beta[10]	1.67	0.01	0.90	0.01	1.02	1.64	2.26	3.50	6591	1
beta[11]	2.10	0.01	0.87	0.40	1.52	2.09	2.68	3.87	6570	1
beta[12]	4.90	0.01	0.80	3.39	4.35	4.88	5.42	6.56	6226	1
beta[13]	5.35	0.01	0.82	3.79	4.79	5.34	5.89	7.01	8000	1
beta[14]	2.71	0.01	0.80	1.14	2.17	2.70	3.24	4.30	6052	1
beta[15]	0.86	0.01	0.80	-0.40	0.23	0.78	1.38	2.60	8000	1
beta[16]	1.89	0.01	0.84	0.22	1.32	1.90	2.45	3.56	5414	1
beta[17]	0.04	0.01	0.57	-1.08	-0.27	0.01	0.34	1.30	8000	1
beta[18]	0.01	0.01	0.62	-1.27	-0.32	0.00	0.33	1.36	8000	1
beta[19]	0.53	0.01	0.64	-0.51	0.04	0.43	0.93	1.95	5997	1
beta[20]	-1.23	0.01	0.79	-2.78	-1.78	-1.22	-0.65	0.14	5051	1
beta[21]	-1.22	0.01	0.80	-2.77	-1.79	-1.22	-0.63	0.18	6039	1
beta[22]	-0.42	0.01	0.71	-1.95	-0.84	-0.31	0.03	0.87	6143	1
beta[23]	0.07	0.01	0.64	-1.21	-0.28	0.02	0.39	1.52	8000	1
beta[24]	-3.10	0.01	0.88	-4.76	-3.69	-3.12	-2.53	-1.27	8000	1
beta[25]	-1.22	0.01	0.81	-2.82	-1.79	-1.21	-0.62	0.19	5898	1
beta[26]	-2.30	0.01	0.86	-3.95	-2.89	-2.33	-1.73	-0.50	6328	1
beta[27]	-1.62	0.01	0.96	-3.45	-2.30	-1.64	-0.92	0.14	6130	1
beta[28]	-1.48	0.01	0.94	-3.29	-2.14	-1.49	-0.78	0.15	5498	1
beta[29]	-2.86	0.01	0.86	-4.46	-3.46	-2.89	-2.31	-1.10	5388	1
beta[30]	-1.04	0.01	0.89	-2.88	-1.66	-0.98	-0.32	0.38	5168	1
beta[31]	-2.94	0.02	1.31	-5.31	-3.87	-3.03	-2.07	-0.15	6158	1
beta[32]	-1.69	0.02	1.31	-4.33	-2.63	-1.63	-0.65	0.43	6621	1
beta[33]	-1.18	0.01	0.92	-2.87	-1.87	-1.20	-0.45	0.47	4041	1

Figure 4.9: Convergence measures $\hat{R}$ and n_{eff} for the parameters of the regularized horseshoe prior model.

Chapter 5

Application to Real Data

The time series analyzed in this chapter is from the DataMarket website. It is named 'Monthly milk production: pounds per cow, Jan 62 to Dec 75'. This time series recorded the monthly milk production from January 1962 to December 1975 so that its length is 168.

Figure 5.1 displays the original time series. Since this time series is not stationary we use the stl function in R to decompose it. Figure 5.2 shows that the original time series has been split into three parts: seasonal, trend and remainder. The remainder part is the stationary time series we analyze next.

We first apply the discrete Fourier transform to the time series and obtain the log periodogram. Next we apply our models to estimate the spectral density corresponding to the time series. We apply both the P-spline and the regularized horseshoe models, with the settings described in the last chapter except that now the length of the time series is 168.

Figure 5.3 displays the log periodogram and the estimated log spectral densities on the left panel. The right panel shows the corresponding exponentiated quantities. The blue lines denote estimates based on the regularized horseshoe prior model while the red lines are estimated based on the P-spline model. We can see that the red lines are smoother than the blue lines but the blue lines captured the peak better.

For the P-spline model, Figure 5.4 shows trace plots of the global smoothing parameter

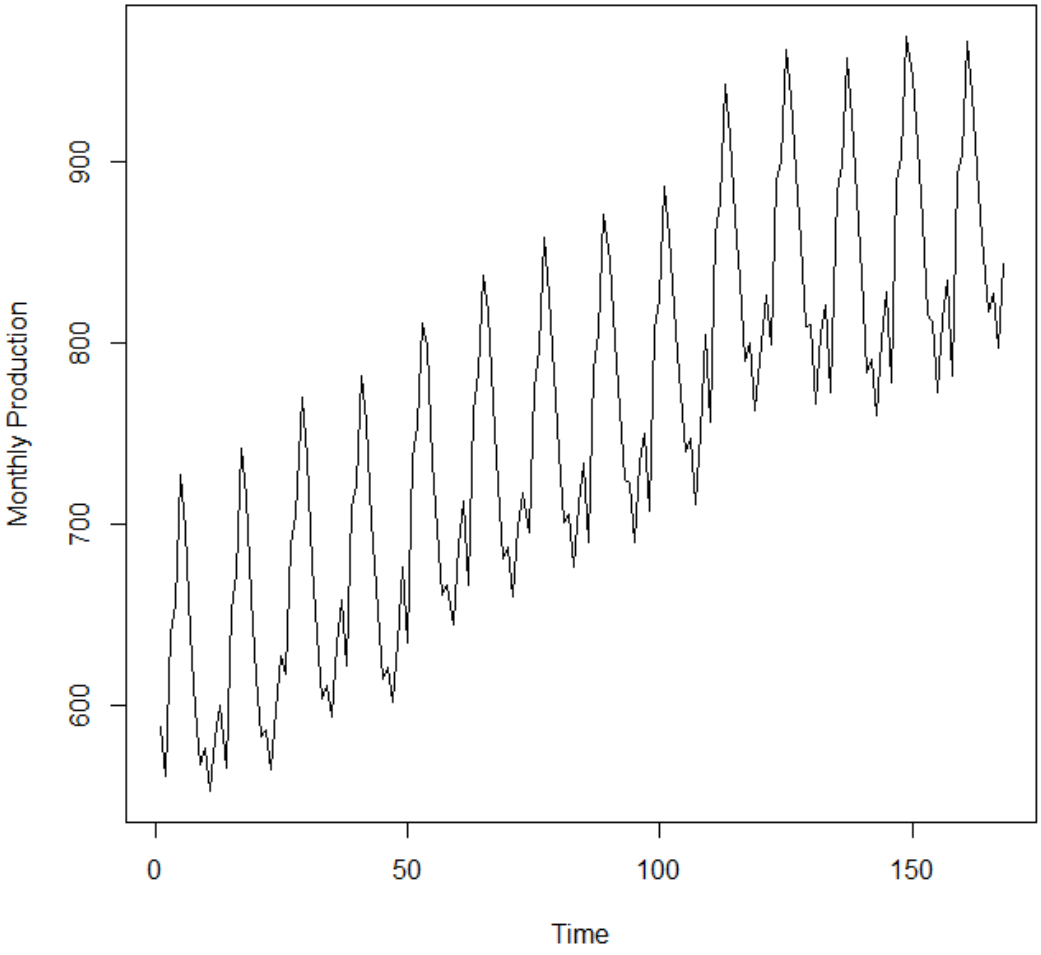

Figure 5.1: Monthly milk production: pounds per cow, Jan 62 to Dec 75.

τ and some of the βs from the iterations after burn-in. Figure 5.5 shows the convergence measures $\hat{R}$ and n_{eff} for the parameters of the P-spline model. Both plots indicate that all the parameters have reached convergence.

For the regularized horseshoe model, figures 5.6 and 5.7 show trace plots for some of the βs and the corresponding $\tilde{\lambda}$s from the iterations after burn-in. Figure 5.8 displays the convergence measures $\hat{R}$ and n_{eff} for the parameters of the regularized horseshoe model.

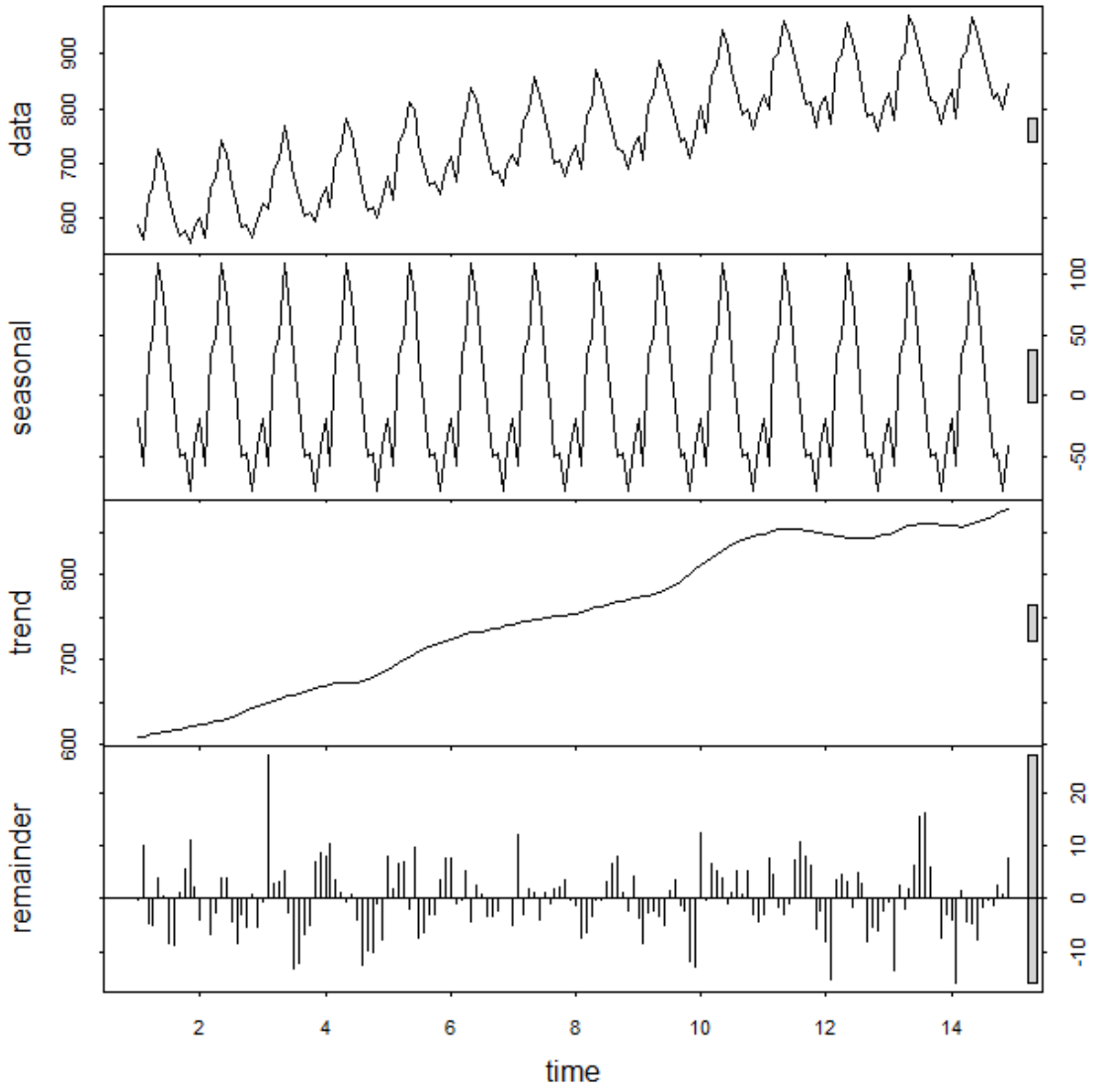

Figure 5.2: Decomposition of the time series into seasonal, trend and remainder parts.

These two plots suggest that all the parameters in the model have reached convergence.

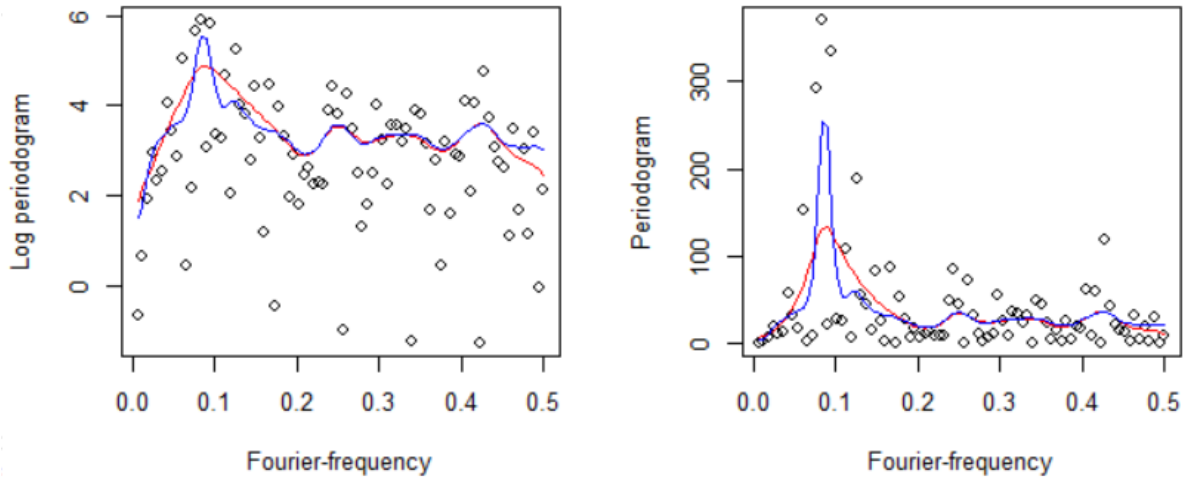

Figure 5.3: Left panel: the log periodogram and estimated log spectral density. Right panel: the periodogram and estimated spectral density. Blue lines denote the regularized horseshoe prior fit, while red lines denote the random walk prior fit.

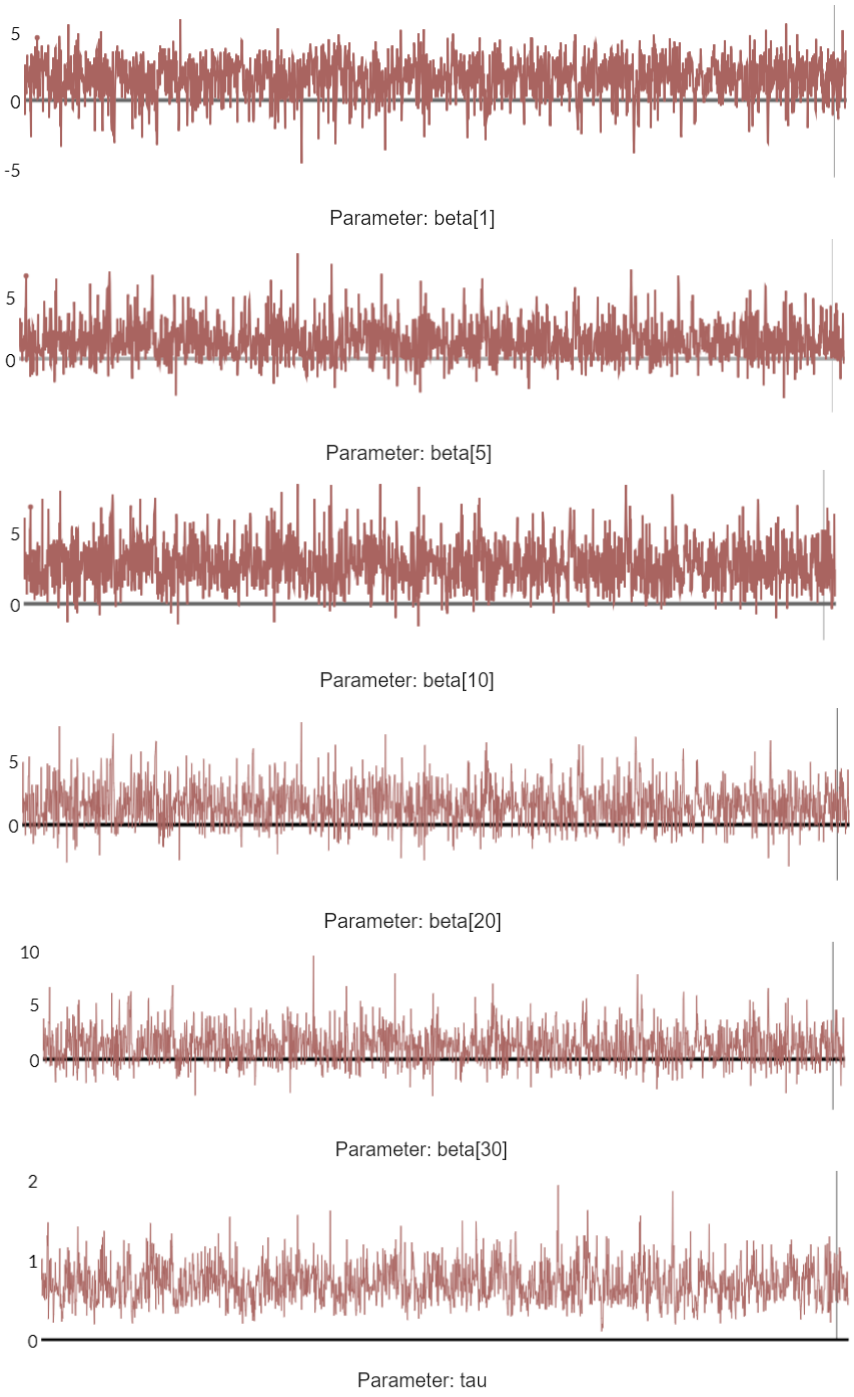

Figure 5.4: Trace plots of the global smoothing parameter τ and some of the βs for the P-spline model fitted to the real data.

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
beta_raw[1]	1.67	0.02	1.45	-1.47	0.77	1.75	2.68	4.26	3731	1
beta_raw[2]	0.01	0.01	1.00	-1.94	-0.66	0.02	0.68	1.98	8000	1
beta_raw[3]	0.21	0.01	0.99	-1.73	-0.46	0.21	0.87	2.18	8000	1
beta_raw[4]	0.80	0.01	0.98	-1.15	0.13	0.80	1.48	2.68	8000	1
beta_raw[5]	1.08	0.01	0.89	-0.68	0.49	1.09	1.67	2.83	8000	1
beta_raw[6]	1.10	0.01	0.84	-0.51	0.52	1.09	1.66	2.79	8000	1
beta_raw[7]	1.00	0.01	0.82	-0.59	0.44	0.99	1.55	2.63	8000	1
beta_raw[8]	0.69	0.01	0.79	-0.87	0.16	0.69	1.21	2.23	8000	1
beta_raw[9]	-0.47	0.01	0.79	-2.03	-1.00	-0.47	0.05	1.10	8000	1
beta_raw[10]	-0.42	0.01	0.78	-1.99	-0.94	-0.41	0.11	1.13	8000	1
beta_raw[11]	-0.50	0.01	0.83	-2.16	-1.06	-0.49	0.07	1.07	8000	1
beta_raw[12]	-0.50	0.01	0.80	-2.09	-1.04	-0.50	0.06	1.03	8000	1
beta_raw[13]	-0.39	0.01	0.81	-1.99	-0.93	-0.38	0.15	1.16	8000	1
beta_raw[14]	-0.62	0.01	0.81	-2.22	-1.19	-0.63	-0.07	0.96	8000	1
beta_raw[15]	-0.41	0.01	0.80	-1.95	-0.95	-0.39	0.13	1.15	8000	1
beta_raw[16]	0.25	0.01	0.82	-1.33	-0.30	0.26	0.81	1.86	8000	1
beta_raw[17]	0.76	0.01	0.80	-0.81	0.22	0.78	1.30	2.31	8000	1
beta_raw[18]	0.01	0.01	0.78	-1.53	-0.51	0.01	0.53	1.55	8000	1
beta_raw[19]	-0.53	0.01	0.84	-2.17	-1.10	-0.52	0.03	1.15	8000	1
beta_raw[20]	0.02	0.01	0.80	-1.56	-0.52	0.02	0.56	1.58	8000	1
beta_raw[21]	0.18	0.01	0.82	-1.44	-0.36	0.19	0.74	1.76	8000	1
beta_raw[22]	0.02	0.01	0.79	-1.52	-0.52	0.02	0.55	1.56	8000	1
beta_raw[23]	-0.02	0.01	0.79	-1.58	-0.55	-0.02	0.51	1.53	8000	1
beta_raw[24]	-0.39	0.01	0.82	-2.00	-0.95	-0.41	0.16	1.21	8000	1
beta_raw[25]	-0.20	0.01	0.82	-1.82	-0.74	-0.21	0.35	1.42	8000	1
beta_raw[26]	0.41	0.01	0.79	-1.13	-0.13	0.42	0.96	1.96	8000	1
beta_raw[27]	0.52	0.01	0.80	-1.04	-0.02	0.52	1.06	2.06	8000	1
beta_raw[28]	0.10	0.01	0.79	-1.45	-0.43	0.10	0.65	1.63	8000	1
beta_raw[29]	-0.73	0.01	0.80	-2.30	-1.28	-0.72	-0.19	0.85	8000	1
beta_raw[30]	-0.43	0.01	0.79	-1.99	-0.96	-0.42	0.12	1.11	8000	1
beta_raw[31]	-0.20	0.01	0.82	-1.82	-0.75	-0.21	0.35	1.42	8000	1
beta_raw[32]	-0.21	0.01	0.89	-1.97	-0.82	-0.21	0.40	1.49	8000	1
beta_raw[33]	-0.20	0.01	0.88	-1.87	-0.79	-0.21	0.38	1.55	8000	1
tau	0.70	0.00	0.23	0.33	0.54	0.67	0.84	1.23	2651	1
beta[1]	1.67	0.02	1.45	-1.47	0.77	1.75	2.68	4.26	3731	1
beta[2]	0.01	0.01	0.74	-1.47	-0.43	0.01	0.44	1.53	8000	1
beta[3]	0.17	0.01	1.04	-1.89	-0.46	0.14	0.76	2.38	6725	1
beta[4]	0.75	0.02	1.28	-1.57	-0.07	0.66	1.51	3.63	4222	1
beta[5]	1.50	0.02	1.46	-1.06	0.52	1.38	2.36	4.72	4151	1
beta[6]	2.23	0.02	1.52	-0.39	1.18	2.12	3.18	5.56	3953	1
beta[7]	2.89	0.03	1.56	0.16	1.78	2.79	3.87	6.26	3876	1
beta[8]	3.36	0.03	1.61	0.54	2.23	3.25	4.36	6.86	3611	1
beta[9]	3.02	0.02	1.53	0.29	1.95	2.92	3.98	6.36	3881	1
beta[10]	2.75	0.02	1.52	0.03	1.69	2.65	3.71	5.99	3975	1
beta[11]	2.42	0.02	1.53	-0.28	1.35	2.34	3.36	5.75	4107	1
beta[12]	2.10	0.02	1.52	-0.62	1.04	2.00	3.05	5.37	4079	1
beta[13]	1.86	0.02	1.52	-0.87	0.80	1.77	2.81	5.11	4029	1
beta[14]	1.44	0.02	1.49	-1.25	0.40	1.36	2.36	4.59	4351	1
beta[15]	1.15	0.02	1.47	-1.57	0.16	1.07	2.05	4.28	4588	1
beta[16]	1.33	0.02	1.47	-1.37	0.32	1.23	2.27	4.49	4396	1
beta[17]	1.89	0.03	1.56	-0.81	0.78	1.77	2.87	5.23	3712	1
beta[18]	1.89	0.03	1.57	-0.85	0.79	1.77	2.87	5.25	3835	1
beta[19]	1.50	0.02	1.49	-1.20	0.47	1.41	2.43	4.71	4346	1
beta[20]	1.53	0.02	1.51	-1.17	0.49	1.42	2.46	4.79	4048	1
beta[21]	1.67	0.02	1.54	-1.07	0.61	1.56	2.61	5.05	3985	1
beta[22]	1.68	0.02	1.53	-1.02	0.61	1.57	2.62	5.02	4047	1
beta[23]	1.68	0.02	1.54	-1.05	0.59	1.57	2.60	5.05	3938	1
beta[24]	1.39	0.02	1.51	-1.36	0.35	1.31	2.33	4.67	4166	1
beta[25]	1.24	0.02	1.48	-1.45	0.22	1.15	2.15	4.42	4459	1
beta[26]	1.54	0.02	1.51	-1.13	0.49	1.45	2.49	4.82	4099	1
beta[27]	1.91	0.03	1.56	-0.82	0.81	1.81	2.88	5.31	3768	1
beta[28]	1.99	0.03	1.57	-0.77	0.90	1.89	2.94	5.37	3526	1
beta[29]	1.47	0.02	1.50	-1.19	0.43	1.38	2.41	4.71	4074	1
beta[30]	1.20	0.02	1.50	-1.48	0.18	1.12	2.10	4.40	4149	1
beta[31]	1.08	0.02	1.52	-1.62	0.02	0.98	2.00	4.41	3954	1
beta[32]	0.94	0.02	1.55	-1.85	-0.15	0.82	1.91	4.25	4360	1
beta[33]	0.80	0.02	1.59	-2.04	-0.28	0.69	1.78	4.18	5199	1

Figure 5.5: Convergence measures $\hat{R}$ and n_{eff} of the parameters for the P-spline model.

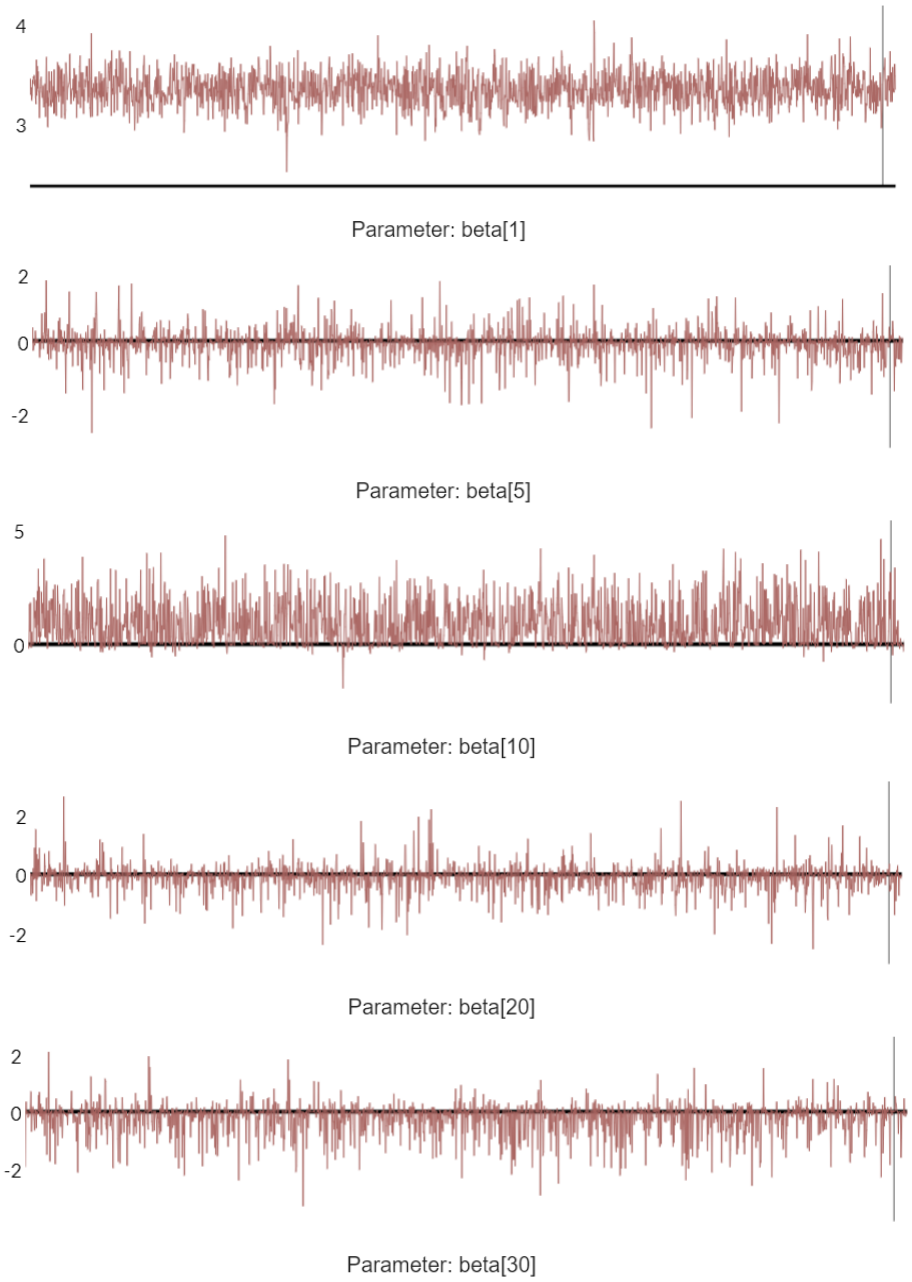

Figure 5.6: Trace plots for some of the βs of the regularized horseshoe model.

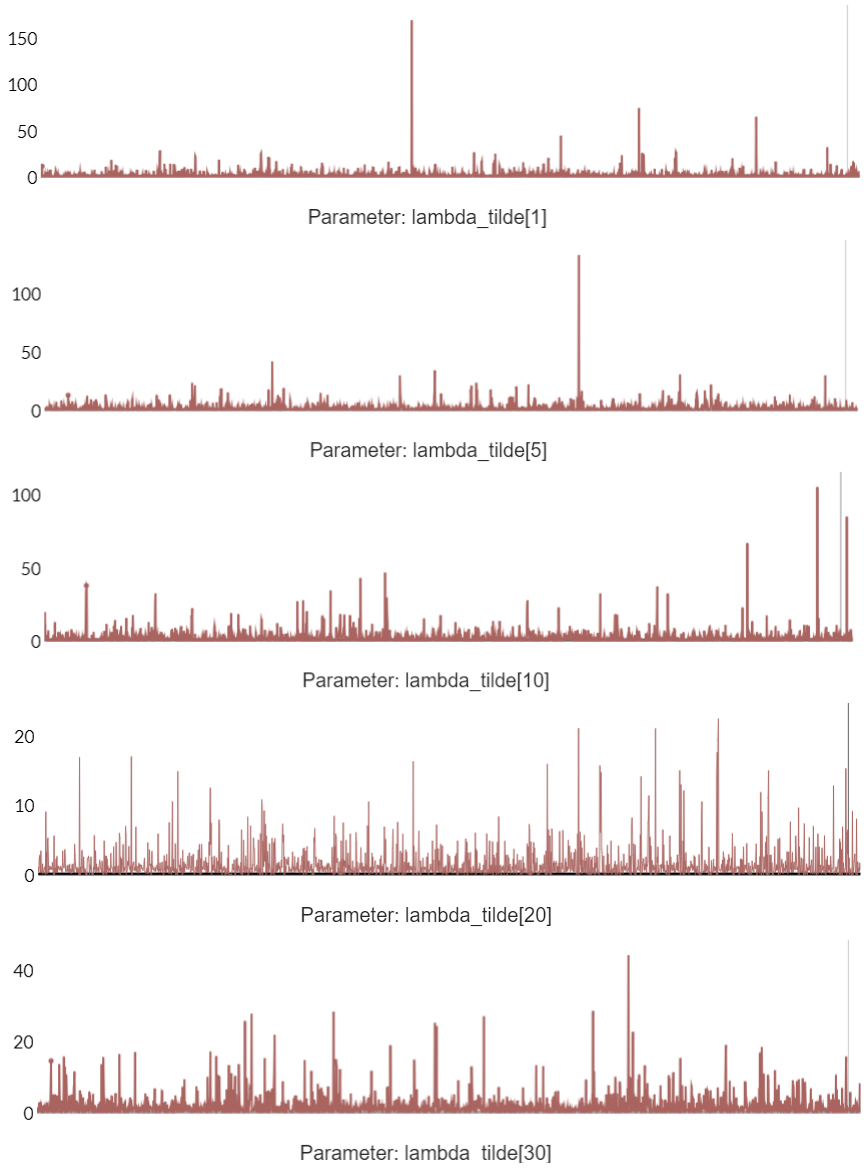

Figure 5.7: Trace plots for some of the $\tilde{\lambda}$s corresponding to βs for the regularized horseshoe model fitted to the real data.

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
tau	0.31	0.00	0.20	0.06	0.17	0.27	0.40	0.83	3448	1
c2	26.61	0.08	7.35	15.91	21.45	25.43	30.49	44.16	8000	1
lambda_tilde[1]	2.08	0.06	3.82	0.04	0.39	0.93	2.11	12.21	4480	1
lambda_tilde[2]	9.30	0.24	13.82	0.08	1.23	5.34	12.35	41.51	3363	1
lambda_tilde[3]	2.27	0.06	4.00	0.05	0.41	1.02	2.41	12.61	4974	1
lambda_tilde[4]	1.54	0.04	3.13	0.03	0.36	0.81	1.70	7.55	7023	1
lambda_tilde[5]	2.05	0.04	3.52	0.04	0.43	1.00	2.24	10.80	6428	1
lambda_tilde[6]	2.02	0.05	3.75	0.04	0.39	0.92	2.10	11.02	5297	1
lambda_tilde[7]	15.34	0.41	24.33	2.03	5.96	10.31	17.76	58.26	3527	1
lambda_tilde[8]	1.89	0.05	3.80	0.03	0.36	0.87	1.95	10.50	5233	1
lambda_tilde[9]	4.56	0.10	7.35	0.06	0.83	2.24	5.34	22.51	5278	1
lambda_tilde[10]	2.38	0.06	4.51	0.04	0.45	1.04	2.41	13.51	4825	1
lambda_tilde[11]	1.69	0.04	3.13	0.03	0.37	0.84	1.80	8.90	5314	1
lambda_tilde[12]	1.55	0.04	2.67	0.03	0.35	0.83	1.74	7.77	5411	1
lambda_tilde[13]	1.68	0.04	2.92	0.03	0.38	0.86	1.81	8.31	5162	1
lambda_tilde[14]	2.61	0.07	4.43	0.05	0.49	1.18	2.88	13.81	4477	1
lambda_tilde[15]	2.02	0.06	3.54	0.04	0.41	0.96	2.21	10.99	4147	1
lambda_tilde[16]	1.78	0.04	2.90	0.04	0.38	0.89	1.94	9.95	5289	1
lambda_tilde[17]	1.69	0.03	2.66	0.04	0.38	0.89	1.87	8.86	5875	1
lambda_tilde[18]	1.93	0.04	3.37	0.03	0.40	0.95	2.07	10.46	5967	1
lambda_tilde[19]	1.59	0.04	2.84	0.04	0.37	0.82	1.74	8.00	5525	1
lambda_tilde[20]	1.61	0.04	2.81	0.03	0.34	0.82	1.71	8.86	6197	1
lambda_tilde[21]	1.53	0.03	2.42	0.03	0.35	0.81	1.72	7.76	6102	1
lambda_tilde[22]	1.51	0.03	2.49	0.04	0.36	0.81	1.64	7.83	5624	1
lambda_tilde[23]	1.70	0.04	2.71	0.04	0.39	0.88	1.88	8.79	5958	1
lambda_tilde[24]	2.35	0.06	4.15	0.04	0.45	1.07	2.46	12.86	4647	1
lambda_tilde[25]	1.64	0.04	3.11	0.04	0.37	0.83	1.73	8.46	5602	1
lambda_tilde[26]	1.67	0.04	2.88	0.04	0.37	0.86	1.84	8.48	5466	1
lambda_tilde[27]	2.02	0.04	3.46	0.04	0.41	1.01	2.21	10.63	5936	1
lambda_tilde[28]	1.77	0.04	3.10	0.04	0.37	0.87	1.92	9.38	5283	1
lambda_tilde[29]	1.99	0.05	3.42	0.04	0.40	0.95	2.17	10.67	5444	1
lambda_tilde[30]	2.03	0.05	3.42	0.04	0.41	0.97	2.18	10.94	5545	1
lambda_tilde[31]	2.02	0.05	3.76	0.04	0.41	0.94	2.08	11.37	5022	1
lambda_tilde[32]	1.98	0.05	3.44	0.04	0.40	0.96	2.15	10.68	5557	1
beta[1]	3.35	0.00	0.17	3.00	3.24	3.35	3.46	3.69	6184	1
beta[2]	0.20	0.00	0.31	0.00	0.03	0.08	0.20	1.31	5053	1
beta[3]	-2.66	0.06	2.53	-7.23	-4.95	-2.38	-0.06	0.32	2007	1
beta[4]	-0.26	0.01	0.70	-2.37	-0.35	-0.04	0.04	0.67	5937	1
beta[5]	-0.04	0.00	0.41	-1.02	-0.14	0.00	0.09	0.80	8000	1
beta[6]	0.29	0.01	0.59	-0.38	-0.02	0.07	0.43	2.01	7060	1
beta[7]	0.21	0.01	0.60	-0.57	-0.04	0.04	0.29	1.97	7198	1
beta[8]	3.33	0.01	0.98	1.47	2.67	3.30	3.97	5.33	8000	1
beta[9]	0.11	0.01	0.53	-0.76	-0.07	0.02	0.19	1.60	6486	1
beta[10]	0.89	0.01	0.95	-0.17	0.06	0.63	1.48	3.12	5059	1
beta[11]	0.32	0.01	0.66	-0.46	-0.02	0.07	0.47	2.22	6145	1
beta[12]	0.12	0.01	0.45	-0.61	-0.06	0.02	0.20	1.34	8000	1
beta[13]	0.10	0.00	0.41	-0.55	-0.06	0.01	0.18	1.22	8000	1
beta[14]	-0.08	0.00	0.43	-1.20	-0.18	-0.01	0.07	0.71	8000	1
beta[15]	-0.41	0.01	0.72	-2.36	-0.66	-0.10	0.01	0.42	5878	1
beta[16]	-0.25	0.01	0.61	-2.00	-0.36	-0.04	0.03	0.54	5868	1
beta[17]	0.21	0.01	0.50	-0.41	-0.03	0.04	0.31	1.62	6758	1
beta[18]	0.19	0.01	0.52	-0.49	-0.04	0.03	0.27	1.70	7508	1
beta[19]	-0.21	0.01	0.57	-1.85	-0.31	-0.03	0.04	0.57	6962	1
beta[20]	-0.07	0.00	0.39	-1.07	-0.17	-0.01	0.07	0.68	8000	1
beta[21]	0.03	0.00	0.41	-0.80	-0.10	0.00	0.12	1.01	8000	1
beta[22]	0.00	0.00	0.39	-0.85	-0.11	0.00	0.11	0.87	8000	1
beta[23]	0.06	0.00	0.39	-0.66	-0.08	0.01	0.14	1.01	8000	1
beta[24]	-0.12	0.01	0.47	-1.36	-0.22	-0.02	0.06	0.67	8000	1
beta[25]	-0.33	0.01	0.64	-2.11	-0.50	-0.08	0.02	0.45	6692	1
beta[26]	-0.03	0.00	0.41	-1.00	-0.14	0.00	0.09	0.80	8000	1
beta[27]	0.13	0.01	0.45	-0.54	-0.05	0.02	0.20	1.36	8000	1
beta[28]	0.28	0.01	0.54	-0.34	-0.01	0.08	0.43	1.77	8000	1
beta[29]	-0.14	0.01	0.48	-1.50	-0.23	-0.02	0.05	0.64	8000	1
beta[30]	-0.20	0.01	0.51	-1.61	-0.32	-0.04	0.03	0.56	8000	1
beta[31]	-0.21	0.01	0.61	-1.96	-0.31	-0.03	0.05	0.67	8000	1
beta[32]	-0.18	0.01	0.62	-1.96	-0.26	-0.02	0.06	0.75	8000	1
beta[33]	-0.20	0.01	0.56	-1.74	-0.33	-0.03	0.04	0.66	8000	1

Figure 5.8: Convergence measures $\hat{R}$ and n_{eff} for the parameters of the regularized horseshoe model fitted to the real data.

References

Carter, C. K. and Kohn, R. (1997), "Semiparametric Bayesian Inference for Time Series with Mixed Spectra," *Journal of the Royal Statistical Society. Series B (Methodological)*, 59, 255–268.

Carvalho, C. M., Polson, N. G., and Scott, J. G. (2010), "The horseshoe estimator for sparse signals," *Biometrika*, 97, 465–480.

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. (1987), "Hybrid Monte Carlo," *Physics Letters B*, 195, 216 – 222.

Eilers, P. H. C. and Marx, B. D. (1996), "Flexible Smoothing with B-splines and Penalties," *Statistical Science*, 11, 89–102.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., and Vehtari, A. (2013), *Bayesian Data Analysis*, Taylor & Francis Ltd.

George, E. I. and McCulloch, R. E. (1997), "Approaches for Bayesian variable selection," *Statistica Sinica*, 7, 339–373.

Lang, S. and Brezger, A. (2004), "Bayesian P-Splines," *Journal of Computational and Graphical Statistics*, 13, 183–212.

Marx, B. D. and Eilers, P. H. C. (1999), "Generalized Linear Regression on Sampled Signals and Curves: A P-Spline Approach," *Technometrics*, 41, 1–13.

Neal, R. M. (2011), "MCMC using Hamiltonian dynamics," in *Handbook of Markov chain Monte Carlo*, CRC Press, Boca Raton, FL, Chapman & Hall/CRC Handb. Mod. Stat. Methods, pp. 113–162.

Pawitan, Y. and O'Sullivan, F. (1994), "Nonparametric spectral density estimation using penalized Whittle likelihood," *J. Amer. Statist. Assoc.*, 89, 600–610.

Piironen, J. and Vehtari, A. (2017), "Sparsity information and regularization in the horseshoe and other shrinkage priors," *Electron. J. Statist.*, 11, 5018–5051.

Shumway, R. H. and Stoffer, D. S. (2017), *Time series analysis and its applications*, Springer Texts in Statistics, Springer, Cham, 4th ed., with R examples.

Wahba, G. (1980), "Automatic Smoothing of the Log Periodogram," *Journal of the American Statistical Association*, 75, 122–132.

Whittle, P. (1962), "Gaussian estimation in stationary time series," *Bull. Inst. Internat. Statist.*, 39, 105–129.

Appendix A

R and Stan code

A.1 Generate the AR(2) time series data

```
t = 550  ## length of the time series

N = 10  ## number of the time series

M = floor((t-50-1)/2) ##calculate the M

dataset = matrix(nrow = N, ncol = M)

w = matrix(nrow = N, ncol = t)

TS = matrix(nrow = N, ncol = t - 50)

P = matrix(nrow = N, ncol = t - 50)

Logperiodogram = matrix(nrow = N, ncol = M)
```

```
par(mfrow=c(2,2))

for( i in 1:N )
{
  w[i,]= rnorm(t,0,1);

  TS[i,] = filter(w[i,],filter=c(1,-0.9),
  method ="recursive")[-(1:50)]

  plot.ts(TS[i,])

  P[i,] = (abs(fft(TS[i,]))^2)/(t-50)

  Logperiodogram[i,]=log(P[i,1:M])
}

write.csv(Logperiodogram, file="E:/DropBox/Dropbox/yiexie
/10 time series data/Logperiodogram.csv")
write.csv(TS, file="E:/DropBox/Dropbox/yiexie/10 time series data
/TS.csv")
```

A.2 The P-spline model

```
library("splines")
```

```
library("rstan")

library("astsa")

library("shinystan")

set.seed(1000)

Logperiodogram = read.csv("E:/DropBox/Dropbox/yiexie
/10 time series data random walk/Logperiodogram.csv")

M = length(Logperiodogram[1,]) - 1

Logperiodogram = Logperiodogram[,2:(M+1)]

TS = read.csv("E:/DropBox/Dropbox/yiexie
/10 time series data random walk/TS.csv")

lTS = length(TS)

TS = TS[,2:lTS]

t = lTS - 1 + 50

N = length(TS[,1])

FF=((1:M)/(t-50))   ########   frequency  as X
```

```
num_knots <- 30 # true number of knots

spline_degree <- 3

num_basis <- num_knots + spline_degree - 1

gg = seq(0,0.5,length.out = num_knots)

gg = gg[-c(1,30)]    ##generate the internal breakpoints that
define the spline

B <- bs(FF,knots = gg, degree=spline_degree,
 Boundary.knots = c(0,0.5),
 intercept = TRUE) # creating the B-splines bases

X = cbind(rep(1,M),B)

betahat = matrix(nrow = N,ncol = num_basis+1 )

yhat = matrix(nrow = N,ncol = M)

for (i in 1:N)
{
  stan.fit <- stan(file = '1st.stan',
                   data = list(n = length(FF),
                               p = num_basis+1,
                               y = as.vector(t(Logperiodogram[i,])),
```

```
                                    x = X,
                                    sigma = 10,
                                    A1 = 1000,
                                    nu1 = 3),
                                    control = list(adapt_delta = 0.99,
                                    max_treedepth = 15),
                                    chains = 4 ,iter = 4000,
                                    warmup = 2000)

betahat[i,] = get_posterior_mean(stan.fit,pars="beta")[,5]
   ## there are 4 chains, so the fifth column is the mean of all the
   four chains.
yhat[i,] = X%*%betahat[i,]
print(stan.fit)
}

ff = arma.spec(ar=c(1,-0.9),log = "yes")

m1 = ff$freq

m2 = log(ff$spec)

par(mfrow=c(1,2))

##par(mfrow=c(5,2),mar = c(4,4,1,1))

for (i in 1:N)
{
```

```
  plot(FF,Logperiodogram[i,],ylab="Log periodogram",
  xlab="Fourier-frequency")

  lines(FF,yhat[i,],col = "red")

  lines(m1,m2,col = "blue")
}

par(mfrow=c(1,2))

##par(mfrow=c(5,2),mar = c(4,4,1,1))

for (i in 1:N)
{
  plot(FF,exp(Logperiodogram[i,]),ylab="Periodogram",
  xlab="Fourier-frequency")

  lines(m1,exp(m2),type = 'l', col = "blue")

  lines(FF,exp(yhat[i,]), col = "red")
}
```

A.3 The Regularized horseshoe prior model

```
library("splines")
```

```
library("rstan")

library("astsa")

library("shinystan")

set.seed(1000)

Logperiodogram = read.csv("E:/DropBox/Dropbox/yiexie
/10 time series data horseshoe prior/Logperiodogram.csv")

M = length(Logperiodogram[1,]) - 1

Logperiodogram = Logperiodogram[,2:M]

TS = read.csv("E:/DropBox/Dropbox/yiexie
/10 time series data horseshoe prior/TS.csv")

lTS = length(TS)

TS = TS[,2:lTS]

t = lTS - 1 + 50

N = length(TS[,1])

##N = 1
```

```
FF=((1:(M-1))/(t-50))   ########   frequency  as X

num_knots <- 30 # true number of knots

spline_degree <- 3

num_basis <- num_knots + spline_degree - 1

knots <- unname(quantile(FF,probs=seq(from=0, to=1,
length.out = num_knots)))

B <- bs(FF,df=num_basis, degree=spline_degree, intercept = TRUE)
# creating the B-splines base

X = cbind(rep(1,M-1),B)

betahat = matrix(nrow = N,ncol = num_basis+1)

yhat = matrix(nrow = N,ncol = M-1)

for (i in 1:N)
{
  Finnish.fit <- stan(file = 'Finnish horseshoe update.stan',
                  data = list(n = length(FF),
                              p = num_basis+1,
                              y = as.vector(t(Logperiodogram[i,])),
                              x = X,
                              sigma = pi/sqrt(6)),
```

```
                                control = list(adapt_delta = 0.99,
                                max_treedepth = 12), chains = 4 ,
                                iter = 4000,warmup = 2000)

  betahat[i,] = get_posterior_mean(Finnish.fit,pars="beta")[,5]
  ## there are 4 chains, so the fifth column is the mean of all
  the four chains.
  yhat[i,] = X%*%betahat[i,]
  print(Finnish.fit)
}

ff = arma.spec(ar=c(1,-0.9),log = "yes")

m1 = ff$freq

m2 = log(ff$spec)

par(mfrow=c(1,2))

##par(mfrow=c(5,2),mar = c(4,4,1,1))

for (i in 1:N)
{
  plot(FF,Logperiodogram[i,],ylab="Log periodogram",
  xlab="Fourier-frequency")

  lines(FF,yhat[i,],col = "red")
```

```
  lines(m1,m2,col = "blue")
}

par(mfrow=c(1,2))

##par(mfrow=c(5,2),mar = c(4,4,1,1))

for (i in 1:N)
{
  plot(FF,exp(Logperiodogram[i,]),ylab="Periodogram",
  xlab="Fourier-frequency")

  lines(m1,exp(m2),type = 'l', col = "blue")

  lines(FF,exp(yhat[i,]), col = "red")
}
```

A.4 Process the real time series data

```
origindata = read.csv("E:/DropBox/Dropbox/yiexie/Finnish horseshoe
/milk.csv")

plot.ts(origindata$Month,origindata$production,type = "l",
xlab = "Time",ylab="Monthly Production")

milk = ts(origindata$production,frequency = 12)
```

```
fit = stl(milk,s.window = "period")

plot(fit)

t = length(fit$time.series[,1])

TS = matrix(nrow = 1,ncol = t)

TS =fit$time.series[,3]

TS

plot.ts(TS)

P = matrix(nrow = 1,ncol = t)

P = (abs(fft(TS))^2)/(t)

Logperiodogram = matrix(nrow = 1,ncol = 0.5*t)

Logperiodogram = log(P[1:(0.5*t)])

qq = (1:(0.5*t))/t

qq

plot.ts(qq,Logperiodogram)
```

```
write.csv(Logperiodogram, file="E:/DropBox/Dropbox/yiexie
/Finnish horseshoe/Logperiodogram.csv")
```

A.5 Apply the P-spline model and the Regularized horseshoe prior model to a real time series

```
library("splines")

library("rstan")

library("astsa")

library("shinystan")

set.seed(1000)

Logperiodogram = read.csv("E:/DropBox/Dropbox/yiexie/
Finnish horseshoe/Logperiodogram.csv")

M = length(Logperiodogram$x)

Logperiodogram = Logperiodogram$x

FF=(1:M)/(2*M)   ########   frequency  as X
```

```
num_knots <- 30 # true number of knots

spline_degree <- 3

num_basis <- num_knots + spline_degree - 1

gg = seq(0,0.5,length.out = num_knots)

gg = gg[-c(1,30)]    ##generate the internal breakpoints
that define the spline

B <- bs(FF,knots = gg, degree=spline_degree,
Boundary.knots = c(0,0.5),
 intercept = TRUE) # creating the B-splines bases

X = cbind(rep(1,M),B)

betahat = matrix(nrow = 2,ncol = num_basis+1 )

yhat = matrix(nrow = 2,ncol = M)

stan.fit <- stan(file = '1st.stan',
                  data = list(n = length(FF),
                              p = num_basis+1,
                              y = as.vector(t(Logperiodogram)),
                              x = X,
                              sigma = 10,
                              A1 = 1000,
```

```
                                nu1 = 3),
                                 control = list(adapt_delta = 0.99,
                                 max_treedepth = 15), chains = 4 ,
                                 iter = 4000,warmup = 2000)

betahat[1,] = get_posterior_mean(stan.fit,pars="beta")[,5]
## there are
 4 chains, so the fifth column is the mean of
 all the four chains.

yhat[1,] = X%*%betahat[1,]

print(stan.fit)

  Finnish.fit <- stan(file = 'Finnish horseshoe update.stan',
                   data = list(n = length(FF),
                               p = num_basis+1,
                               y = as.vector(t(Logperiodogram)),
                               x = X,
                               sigma = pi/sqrt(6)),
                                control = list(adapt_delta = 0.99,
                                 max_treedepth = 12),
                                 chains = 4 ,iter = 4000,
                                 warmup = 2000)

  betahat[2,] = get_posterior_mean(Finnish.fit,pars="beta")[,5]
  ## there are 4 chains, so the fifth column is the mean of all
  the four chains.
```

```
  yhat[2,] = X%*%betahat[2,]

  print(Finnish.fit)

  par(mfrow=c(1,2))

  plot(FF,Logperiodogram,ylab="Log periodogram",
  xlab="Fourier-frequency")

  lines(FF,yhat[1,],col = "red")

  lines(FF,yhat[2,],col = "blue")

  plot(FF,exp(Logperiodogram),ylab="Periodogram",
  xlab="Fourier-frequency")

  lines(FF,exp(yhat[1,]),col = "red")

  lines(FF,exp(yhat[2,]),col = "blue")
```

A.6 Stan code for the P-spline model

```
functions {

  real te(vector beta, matrix x, vector y) {
```

```
    return -sum(x*beta+exp(y-x*beta));
    //  the log  wittle likelihood
  }
}

data {

  int<lower=0> n;       //  half the length of the time series
  (round to the ceiling)

  int<lower=0> p;       //  equal to number of basis functions + 1

  matrix[n,p] x;        //  x matrix n X p matrix

  vector[n] y;          //  respones variable vector

  real<lower=0> sigma;  //  the sd of alpha 1 and alpha 2

  real<lower=0> A1;      //  scale parameter for tau

  real<lower=0> nu1;     //  df of the t distribution for tau
}

parameters {
  vector[p] beta_raw;      // coefficients for the basis functions

  real<lower=0> tau;      // one parameter of the  random walk prior
```

```
}

transformed parameters {

  vector[p] beta;

  for (i in 1:1){
    beta[i] = beta_raw[i];
  }

  for (i in 2:2){
    beta[i] = beta_raw[i]*tau;
  }

  for (i in 3:p)
    beta[i] = beta[i-1] + beta_raw[i]*tau;
}

model {
  tau ~ student_t(nu1,0,A1);

  for (i in 1:1){
    beta_raw[i] ~ normal(0, sigma);
  }

  for (i in 2:p){
    beta_raw[i] ~ normal(0, 1);
  }
```

```
  target += te(beta, x, y);
}
```

A.7 Generate the AR(2) time series

```
functions {
  real te(vector beta, matrix x, vector y) {
    return -sum(x*beta+exp(y-x*beta));
    //the log Whittle likelihood
  }
}

data {
  int<lower=1> n; // Number of data
  int<lower=1> p; // Number of covariates
  matrix[n, p] x;
  vector[n] y;
  real<lower=1> sigma;
}

// slab_scale = 5, slab_df = 25 -> 8 divergences

transformed data {
  real m0 = 20;         // Expected number of large slopes
  real slab_scale = 5;  // Scale for large slopes
  real slab_scale2 = square(slab_scale);
```

```
  real slab_df = 30;    // Effective degrees of freedom for large
  slopes
  real half_slab_df = 0.5 * slab_df;
}

parameters {
  vector[p-1] beta_tilde;
  vector<lower=0>[p-1] lambda;
  real<lower=0> c2_tilde;
  real<lower=0> tau_tilde;
}

transformed parameters {

  real tau0 = (m0 / (p - m0)) * (sigma/ sqrt(1 * n));
  real tau = tau0 * tau_tilde; // tau ~ cauchy(0, tau0)

  // c2 ~ inv_gamma(half_slab_df, half_slab_df * slab_scale2)
  real c2 = slab_scale2 * c2_tilde;

  vector[p-1] lambda_tilde =
      sqrt( c2 * square(lambda) ./ (c2 + square(tau) *
      square(lambda)) );

  vector[p] beta;
  {
    // Implies that marginally beta ~ student_t(slab_df, 0,
     slab_scale)
```

```
    // beta ~ normal(0, tau * lambda_tilde)
    for (i in 1:1){
      beta[i] = 10*beta_tilde[i];
      }
    for (i in 2:p){
      beta[i] = tau * lambda_tilde[i-1]* beta_tilde[i-1];
    }
  }
}

model {
  beta_tilde ~ normal(0, 1);
  lambda ~ cauchy(0, 1);
  tau_tilde ~ cauchy(0, 1);
  c2_tilde ~ inv_gamma(half_slab_df, half_slab_df);

  target += te(beta, x, y);
}
```

Curriculum Vitae

Yi Xie was born on July 18, 1988, as the first and only son of Heping Xie and Qi Gao. In 2006, he went to the Central South University in China and four years later received a bachelor degree of science (Mathematics and Applied Mathematics). After that he furthered his study at Changsha University of Science and Technology and received a masters degree of science (stochastic processes, Markov processes) in 2013. Then he went back to his hometown, a small city in Hunan province, and became a math teacher at a university. In 2016, he decided to resign his job and went to The University of Texas at El Paso. While pursuing a masters degree in Statistics, he worked as a Teaching and Research Assistant.

Contact Information: yxie3@miners.utep.edu