2023-05-01

# Spatially Adaptive Estimation Of Spectrum

Yi Xie
*University of Texas at El Paso*

SPATIALLY ADAPTIVE ESTIMATION OF SPECTRUM

YI XIE

Doctoral Program in Computational Science

APPROVED:

Ori Rosen, Ph.D., Chair

Elizabeth J. Walsh, Ph.D.

Suneel Babu Chatla, Ph.D.

Sergio D. Cabrera, Ph.D.

Stephen L. Crites, Jr., Ph.D.
Dean of the Graduate School

*to my*

*MOTHER and FATHER*

*with love*

# SPATIALLY ADAPTIVE ESTIMATION OF SPECTRUM

by

YI XIE

DISSERTATION

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

DOCTOR OF PHILOSOPHY

Computational Science Program

THE UNIVERSITY OF TEXAS AT EL PASO

May 2023

# Abstract

A time series may be analyzed either in the time or in the frequency domain. When working in the frequency domain, the main objective is to estimate the underlying spectrum. Various approaches have been proposed to this end, but most are based on smoothing the periodogram using a single smoothing parameter across all Fourier frequencies. Such a global smoothing parameter may result in a biased estimate. To improve the estimation, in this paper, we smooth the log periodogram by placing a dynamic shrinkage prior, such that varying degrees of smoothing may be applied to different regions of the Fourier frequencies, resulting in a less biased estimate of the spectrum.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

A time series is a sequence of data points that occur in successive order over some period of time. This type of data is prevalent in numerous areas. For example, stock prices, rainfall measurements, annual retail sales, and monthly subscribers to Netflix. The Southern Oscillation Index (SOI) is a standardized index based on the observed sea level pressure (SLP) differences between Tahiti and Darwin, Australia. Figure 1.1 displays the Southern Oscillation Index (SOI) for a period of 453 months ranging over the years 1950-1987. This dataset is also analyzed in Rosen et al. (2012).

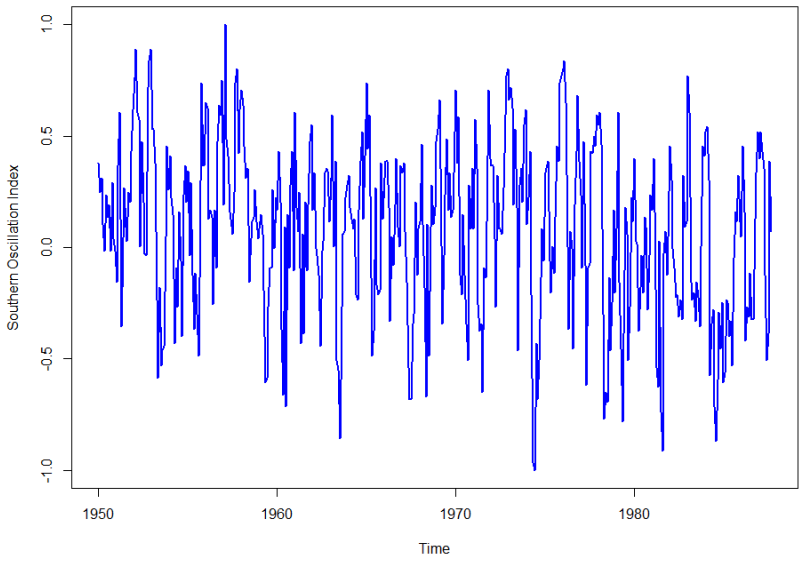


Figure 1.1: Southern Oscillation Index (SOI).

By analyzing a time series, we can identify trends or systemic patterns over time and

predict future values. If a time series is analyzed in the frequency domain, then estimating its spectrum is usually the main objective. Estimating the spectrum helps understand the cyclical nature of the time series. In this dissertation, we propose a new method that can improve the estimation of the spectral density of a time series.
The following definitions are taken from Shumway and Stoffer (2017).

**Definition 1** A 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 a time series $\{x_t,\ t = 0, \pm 1, \pm 2, \ldots\}$ is defined as

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

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

**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, $\mu_t$, 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 $\sigma_w^2$.

**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, such that $\phi_p \neq 0$. We assume that $w_t$ is a Gaussian white noise series with mean zero and variance $\sigma_w^2$.

*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 $\sigma_w^2$.

**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 $\sigma_w^2$.

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

**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

$$\gamma(h) = \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{2\pi i \omega h} f(\omega) d\omega, \quad h = 0, \pm 1, \pm 2, \ldots, \tag{1.1}$$

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 $\omega$.
2. $f(-\omega) = f(\omega)$,i.e., it is an even function.
3. $f(\omega + 1) = f(\omega)$, i.e., It is a periodic function.

In addition, putting $h = 0$ in (1.1) yields

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

which expresses the total variance as the integrated spectral density over all 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 $j = 0, 1, \ldots, n-1$, where the frequencies $w_j = j/n$ are called the **Fourier** or **fundamental frequencies** and $i = \sqrt{-1}$.

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

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

for $j = 0, 1, 2, \ldots, n-1$.

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

**Definition 13** Given data $x_1, \ldots, x_n$, we define the **cosine transform**

$$d_c(\omega_j) = n^{-\frac{1}{2}} \sum_{t=1}^{n} x_t \cos(2\pi\omega_j t)$$

and the **sine transform**

$$d_s(\omega_j) = n^{-\frac{1}{2}} \sum_{t=1}^{n} x_t \sin(2\pi\omega_j t),$$

where $w_j = j/n$, for $j = 0, 1, 2, \ldots, n-1$.

We can see that $d(\omega_j) = d_c(\omega_j) - i d_s(\omega_j)$ and

$$I(\omega_j) = |d(\omega_j)|^2 = d_c^2(\omega_j) + d_s^2(\omega_j).$$

**Definition 14** Let $z = z_1 + iz_2$, where $i = \sqrt{-1}$ and $z_1, z_1 \overset{\text{iid}}{\sim} N(0, \frac{\sigma^2}{2})$. 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 that $z$ has a **complex normal distribution** with mean 0 and variance $\sigma^2$ and denote it as $z \sim CN(0, \sigma^2)$. The pdf of z is given by

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

where $z^*$ is the complex conjugate.

**Definition 15** A random variable $X$ has a **Z distribution** with parameters $a, b, \mu, \sigma$, denoted $X \sim Z(a, b, \mu, \sigma)$, if its pdf is given by

$$f(x) = \frac{1}{\sigma * \text{Beta}(a, b)} \exp\Big\{\frac{(x-\mu)}{\sigma}\Big\}^a \Big[1 + \exp\Big\{\frac{(x-\mu)}{\sigma}\Big\}\Big]^{-(a+b)},$$

where $\text{Beta}(a, b)$ is the beta function with parameters $a$ and $b$.

**Definition 16** A random variable $X$ has a **Pólya-Gamma distribution** with parameters $b > 0$ and $c \in R$, denoted $X \sim PG(b, c)$, if

$$X = \frac{1}{2\pi^2} \sum_{k=1}^{\infty} \frac{g_k}{(k - \frac{1}{2})^2 + \frac{c^2}{4\pi^2}},$$

where the $g_k \sim Ga(b, 1)$ are independent gamma random variables.

# Chapter 2

# Estimation of Spectral Densities

Consider a weakly stationary time series, $x_1, \ldots, x_n$, with mean zero, where we assume $n$ is even. Apply to it the DFT (Definition 11) to obtain

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

for $j = 0, 1, 2, \ldots, n-1$. The periodogram (Definition 12) of this time series at frequency $\omega_j$ is $I(\omega_j) = |d(\omega_j)|^2$.

**Details of the Periodogram**

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

Let $x_1, \ldots, x_n$ be an observed time series of size $n$. For simplicity, we assume that $n$ is odd, then

$$x_t = a_0 + \sum_{j=1}^{m} \Big[ a_j \cos(2\pi\omega_j t + b_j \sin(2\pi\omega_j t)) \Big], \tag{2.1}$$

where $m = (n-1)/2$, $a_0 = (\sum_{j=1}^{n} x_j)/n = \bar{x}$ and

$$a_j = \frac{2}{n} \sum_{t=1}^{n} x_t \cos(2\pi\omega_j t) = \frac{2}{\sqrt{n}} d_c(\omega_j),$$

$$b_j = \frac{2}{n} \sum_{t=1}^{n} x_t \sin(2\pi\omega_j t) = \frac{2}{\sqrt{n}} d_s(\omega_j),$$

is exact for $t = 1, 2, \ldots, n$. For the definition of $d_c(\omega_j)$ and $d_c(\omega_j)$, see Definition 13 in Chapter 1.

From Equation (2.1), we have

$$(x_t - \bar{x}) = \frac{2}{n}\sum_{j=1}^{m}\Big[d_c(\omega_j)\cos(2\pi\omega_j t) + d_s(\omega_j)\sin(2\pi\omega_j t)\Big]. \tag{2.2}$$

Recall that if $j \neq 0$ or $j \neq n/2$,

$$\sum_{t=1}^{n}\cos^2(2\pi\omega_j) = \sum_{t=1}^{n}\sin^2(2\pi\omega_j) = \frac{n}{2},$$

then squaring both sides of the Equation (2.1) and summing results in

$$\sum_{t=1}^{n}(x_t - \bar{x})^2 = 2\sum_{j=1}^{m}\Big[d_c^2(\omega_j) + d_s^2(\omega_j)\Big] = 2\sum_{j=1}^{m} I(\omega_j).$$

Thus, we have turned the sum of squares, $\sum_{t=1}^{n}(x_t - \bar{x})^2$, from the time domain to the frequency domain. Now we can split the sum of squares by the frequency $\omega_j$ and obtain Table 2.1.

Table 2.1: ANOVA Table

| Source | df | Error Sum of Squares | Mean Square Error |
|---|---|---|---|
| $\omega_1$ | 2 | $2I(\omega_1)$ | $I(\omega_1)$ |
| $\omega_2$ | 2 | $2I(\omega_2)$ | $I(\omega_2)$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $\omega_m$ | 2 | $2I(\omega_m)$ | $I(\omega_m)$ |
| Total | $n-1$ | $\sum_{t=1}^{n}(x_t - \bar{x})^2$ | |

If the time series data contain some strong periodic components, then the periodogram values corresponding to those frequencies (or near those frequencies) will be large. Otherwise, the periodograms will be small. This explains why periodograms are important in spectral analysis.

### Traditional Methods

Traditional methods for estimating the spectral density are described in Shumway and Stoffer (2017).

The **Averaged Periodogram**: let $\mathcal{D}$ be a band of $\mathcal{L} << n$ contiguous frequencies centered around frequency $\omega_j = j/n$, i.e.,

$$\mathcal{D} = \Big\{\omega_j + \frac{k}{n} : \ k = 0, \pm 1, \pm 2, \ldots, \pm m\Big\},$$

where $\mathcal{L} = 2m + 1$ is an odd number. The values of the spectral density in $\mathcal{D}$ are

$$f(\omega_j + \frac{k}{n}), \ k = 0, \pm 1, \pm 2, \ldots, \pm m,$$

which are approximately equal to $f(\omega_j)$. Let

$$\bar{f}(\omega_j) = \frac{1}{\mathcal{L}} \sum_{k=-m}^{m} I(\omega_j + \frac{k}{n}),$$

then $E[\bar{f}(\omega_j)] \approx f(\omega_j)$ and $\mathrm{Var}[\bar{f}(\omega_j)] \approx \frac{f^2(\omega_j)}{\mathcal{L}}$. Letting $\mathcal{L} \to \infty$ with $\mathcal{L} << n$, $\mathrm{Var}[\bar{f}(\omega_j)] \to 0$. Key to the success of this method is finding an appropriate value of $\mathcal{L}$. Also, this method may not be able to provide a satisfactory estimate of a peak in the spectral density, since averaging may flatten the peak.

### Whittle Likelihood

For a weakly stationary zero-mean time series $x_1, \ldots, x_n$, where $n$ is even, let $V$ be an $n \times n$ matrix with elements $V_{t,j} = \frac{1}{\sqrt{n}} \exp\Big\{-2\pi i(t-1)\omega_j\Big\}$ for $t = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. Here $i = \sqrt{-1}$. Whittle (1962) proved that for large $n$, the likelihood of $x_1, \ldots, x_n$ can be approximated by

$$P(\boldsymbol{x} \mid f) \ = \ \frac{1}{(2\pi)^{1/2}} \det(R)^{1/2} \exp\Big\{-\frac{1}{2}\boldsymbol{x}^T V R V^* \boldsymbol{x}\Big\}, \tag{2.3}$$

where $\boldsymbol{x} = (x_1, \ldots, x_n)^T$, $R = \mathrm{diag}(\frac{1}{f(\omega_1)}, \ldots, \frac{1}{f(\omega_n)})$ and $V^*$ is the conjugate transpose of $V$. Since $V$ is unitary, i.e., $V^*V = VV^* = I$, it follows that $V^T\boldsymbol{x} = (d(\omega_1), d(\omega_2), \ldots, d(\omega_n))^T$.

We then have $\det(R)^{1/2} = \prod_{j=1}^{n} \frac{1}{f(\omega_j)^{1/2}}$ and $\boldsymbol{x}^T V R V^* \boldsymbol{x} = \sum_{j=1}^{n} \frac{I(\omega_j)}{f(\omega_j)}$. It follows that the likelihood is given by

$$\begin{aligned} P(\boldsymbol{x} \mid f) &= \tfrac{1}{(2\pi)^{1/2}} \det(R)^{1/2} \exp\Big\{-\tfrac{1}{2}\boldsymbol{x}^T V R V^* \boldsymbol{x}\Big\} \\ &= \tfrac{1}{(2\pi)^{1/2}} \prod_{j=1}^{n} \tfrac{1}{f(\omega_j)^{1/2}} \exp\Big\{-\tfrac{1}{2}\tfrac{I(\omega_j)}{f(\omega_j)}\Big\}. \end{aligned} \tag{2.4}$$

One motivation for this approximation is that for large $n$, $d(\omega_j) \overset{\text{ind}}{\sim} \text{CN}(0, f(\omega_j))$ (Definition 14), approximately. This implies

$$g(d(\omega_j)) \propto \frac{1}{f(\omega_j)} \exp\Big\{-\frac{|d(\omega_j)^2|}{f(\omega_j)}\Big\} = \frac{1}{f(\omega_j)} \exp\Big\{-\frac{I(\omega_j)}{f(\omega_j)}\Big\}, \tag{2.5}$$

where $g(d(\omega_j))$ is the pdf of $d(\omega_j)$, and $I(\omega_j)$ is the periodogram. From Formula (2.5) we see that $I(\omega_j) \sim \text{Exponential}\Big(f(\omega_j)\Big)$ approximately, where $\text{Exponential}\Big(f(\omega_j)\Big)$ denotes the exponential distribution with mean $f(\omega_j)$. Let $\epsilon_j = \frac{I(\omega_j)}{f(\omega_j)}$, then $\epsilon_j \sim \text{Exponential}(1)$. It follows that $I(\omega_j) = \epsilon_j f(\omega_j)$. Taking logs of both sides leads to the log-linear model

$$\log(I(\omega_j)) = \log(f(\omega_j)) + \eta_j, \quad \text{for} \quad j = 1, \ldots, n, \tag{2.6}$$

where $\eta_j = \log \epsilon_j \sim \log(\text{Exponential}(1)) = \log(\frac{1}{2}\chi^2_2)$. Model (2.6) was used by Wahba (1980) to estimate the spectral density by smoothing splines.

Contreras-Cristán et al. (2006) found that the Whittle approximation may be unreliable in the non-Gaussian case, even for moderate sample sizes. For small samples, if the autocorrelation of the process is high, Whittle's approximation is not efficient even in the Gaussian case. Sykulski et al. (2019) propose a method to de-bias Whittle's estimates for second-order stationary stochastic processes without increasing the computational cost.

### A Mixture Approximation to the log Exponential Distribution

Carter and Kohn (1997), use a Bayesian approach where the distribution of the error term $\eta_j$ (Equation (2.6)) is approximated by a mixture of five normal distributions with

fixed parameters. The weights, means, and variances of the components are listed in Table 2.2.

Table 2.2: Table of the 5-component Gaussian mixture

| Weight | Mean | Variance |
|---|---|---|
| $p_{5,1} = 0.19$ | $\mu_{5,1} = -2.20$ | $\sigma^2_{5,1} = 1.93$ |
| $p_{5,2} = 0.11$ | $\mu_{5,2} = -0.80$ | $\sigma^2_{5,2} = 1.01$ |
| $p_{5,3} = 0.27$ | $\mu_{5,3} = -0.55$ | $\sigma^2_{5,3} = 0.69$ |
| $p_{5,4} = 0.25$ | $\mu_{5,4} = -0.035$ | $\sigma^2_{5,4} = 0.60$ |
| $p_{5,5} = 0.18$ | $\mu_{5,5} = 0.48$ | $\sigma^2_{5,5} = 0.29$ |

Generally speaking, estimating a spectral density can be done parametrically or nonparametrically. A parametric approach assumes knowledge of the functional form of the spectral density as a function of frequency. If the functional form is correctly specified, this method will be powerful and highly efficient. Our approach in this dissertation is *nonparametric*.

## 2.1 Nonparametric Regression

In nonparametric regression, the form of the relationship between the independent variables and the response variable is estimated based on observed data without assuming a pre-determined regression function such as a linear function. There are various approaches to nonparametric regression. One such approach uses a set of basis functions, whose linear combination estimates the regression function. In this dissertation, we use B-splines.

### B-Splines

There are two requirements in order to define a family of B-spline functions of order $p + 1$ uniquely.

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, $t_1, \ldots, , t_q$.

The $i$th member of a family of B-splines of order 1 is then defined as

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

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

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

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 neighbors.

6. At a given $x$, $p + 1$ B-splines are non-zero.

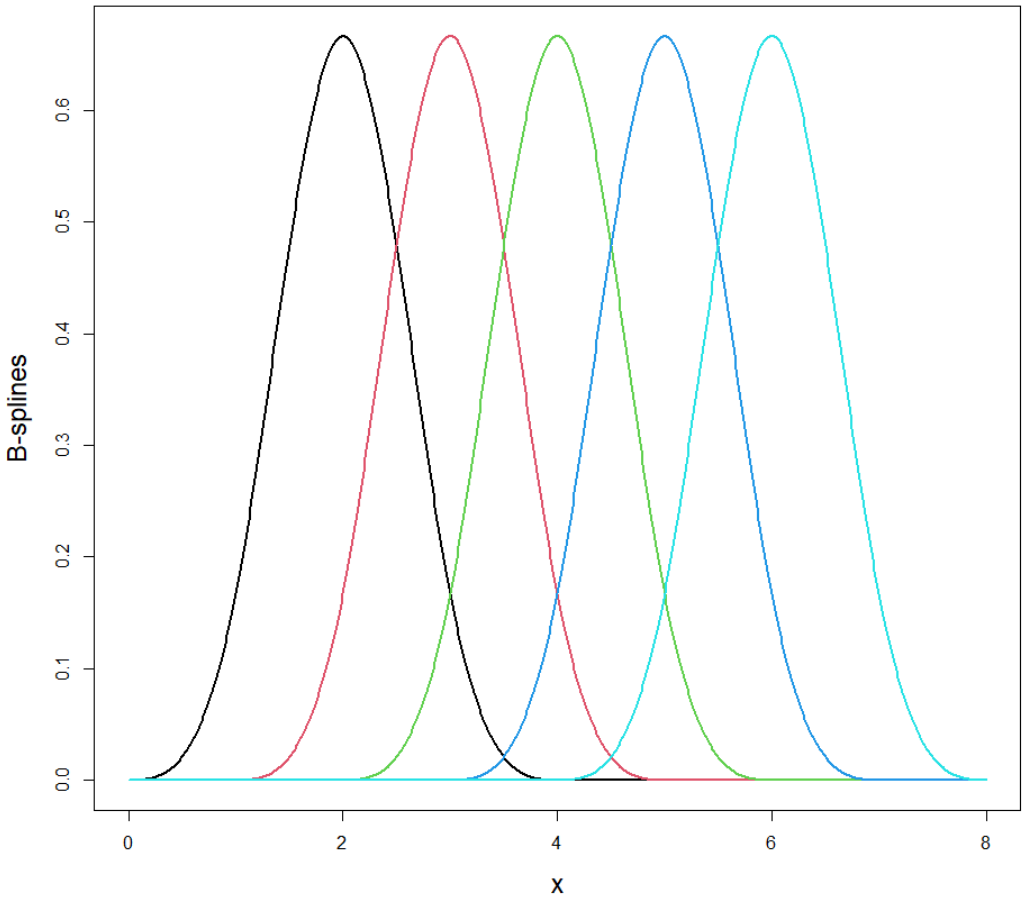


Figure 2.1: B-spline example.

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

Following the definition above and using $(0, 1, 2, \ldots, 7, 8)$ as knots to construct a family of order 4 B-spline functions, Figure 2.1 shows that on the interval [0,1], only $P_{1,4}$ is non-zero, and on the interval [1,2], only $P_{1,4}$ and $P_{2,4}$ are non-zero. But on the intervals [2,3],[3,4], [4,5], [5,6], each interval has three non-zero B-spline functions.

**Periodic B-Splines**

As described in Chapter 1 under Definition 10, spectral densities are even and periodic. Periodicity means that in each period, the curve of the spectral density on the left end of the period should connect smoothly to the right side. Periodicity can be accommodated by periodic B-splines proposed, for example, in Eilers and Marx (2020).

Given a non-decreasing sequence of knots, $t_1, \ldots, t_q$, the interval $[t_1, t_q]$ can be extended by attaching the two endpoints $t_1$ and $t_q$ together and defining $t_1 = t_q$ and $t_{i+q-1} = t_i$ for $i = \pm 1, 2, 3, \ldots$, just like using a paper strip to make a ring. The new family of B-spline functions has the same behavior on the interval $[t_{1+i*T}, t_{1+(i+1)*T}]$ for any $i = \pm 1, 2, \ldots$, where $T$ is the period. The new B-spline functions are "wrapped" at the boundary knots. This can be seen in Figure 2.2, where the knots $(0, 1, 2, \ldots, 7, 8)$ are the same as in Figure 2.1, but now the interval [0,8] is completely covered by the periodic B-splines, i.e., at any point in this interval, there are three non-zero B-spline functions.

### 2.1.1 Regularization of the Basis Function Coefficients

Consider the following model

$$\boldsymbol{Y} = B\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2.7}$$

where $B$ is an $n \times L$ matrix of B-spline basis functions with elements $B_{i,j} = P_{j,4}(\omega_i)$, $P_{j,4}(\omega_i)$ is the $j$-th member of a family of B-splines of order 4 evaluated at frequency $\omega_i$, and $L$ denotes the number of B-spline basis functions. In the sequel, we only consider order 4 B-spline basis functions, and so $P_{j,4}$ will be abbreviated to $P_j$. The vector $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_L)^T$ consists of the coefficients of the B-splines, and $\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2, \ldots, \epsilon_n)^T$ with $\epsilon_i \overset{\text{iid}}{\sim} \text{N}(0, \sigma_\epsilon)$. Merely estimating $\boldsymbol{\beta}$ without imposing any constraints on these coefficients, will usually result in overfitting, see Figure 2.3. Overfitting happens when the model has too many basis functions with non-zero coefficients. In practice, Frequentists apply a penalty to $\boldsymbol{\beta}$, while Bayesians use prior distributions in order to shrink these coefficients towards zero,

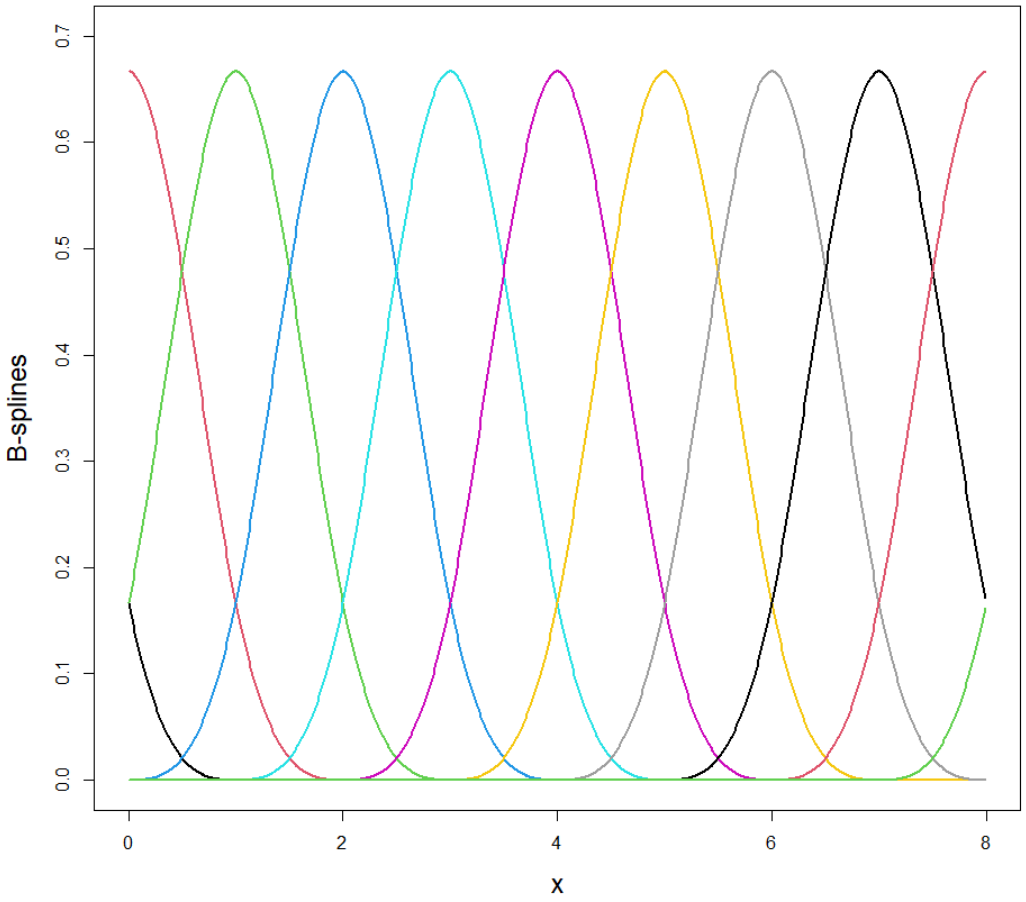


Figure 2.2: Periodic B-spline example.

such that the resulting estimated regression function will provide a good fit to the data.

### P-splines

B-splines are ideal for nonparametric estimation but require some care in deciding on the number and placements of knots. Marx and Eilers (1999) proposed the P-splines approach. 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. It is a flexible tool for smoothing where knots can be placed equally spaced, and as long as the

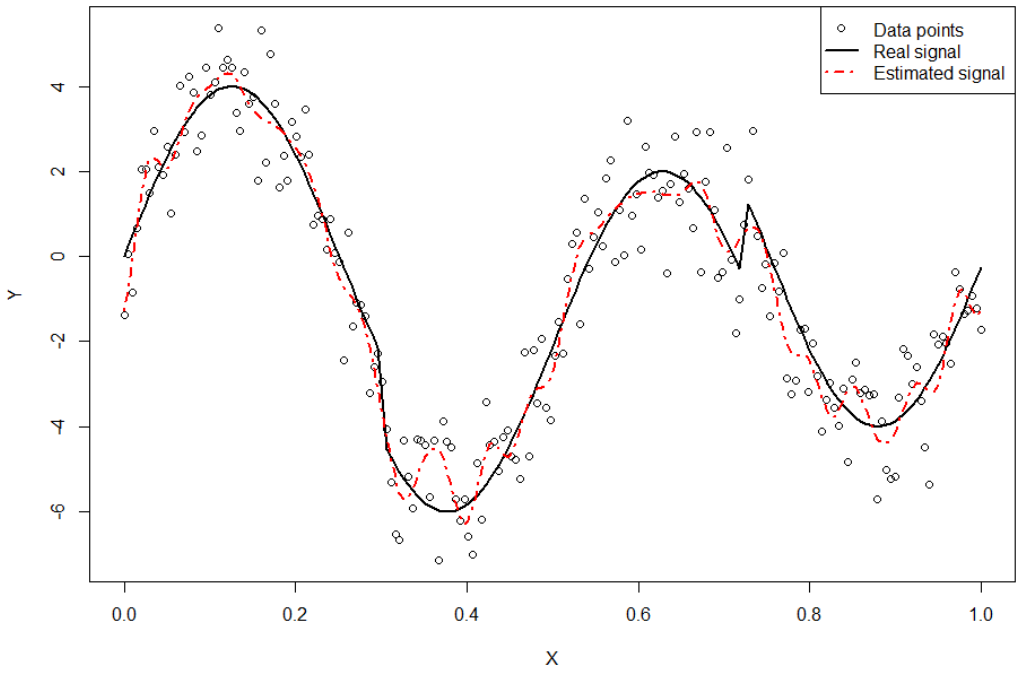


Figure 2.3: Nonparametric regression without constraints on the coefficient vector $\boldsymbol{\beta}$.

number of basis functions is large enough, there is no need to use an "optimal" number of basis functions. The level of smoothness is tuned by the prior on $\boldsymbol{\beta}$.

### 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 equal to zero. This approach is common in Bayesian variable selection for regression models (George and McCulloch, 1997).

Below is a review of some possible priors on $\boldsymbol{\beta}$.

### Random-Walk Prior

Lang and Brezger (2004) developed a Bayesian version of P-splines, in which a random

walk prior is placed on the B-spline coefficients. A first-order random-walk prior is given by $\Delta\beta_\rho = \beta_\rho - \beta_{\rho-1} = \upsilon_\rho$, where $\upsilon_\rho \sim N(0, \tau^2)$, i.e.,

$$\beta_\rho = \beta_{\rho-1} + \upsilon_\rho. \tag{2.8}$$

Similarly, a second-order random-walk prior is obtained by $\Delta^2\beta_\rho = \Delta(\Delta\beta_\rho) = \Delta(\beta_\rho - \beta_{\rho-1}) = \beta\rho - 2\beta_{\rho-1} + \beta_{\rho-2} = \upsilon_\rho$, i.e.,

$$\beta_\rho = 2\beta_{\rho-1} - \beta_{\rho-2} + \upsilon_\rho.$$

Diffuse priors are placed on $\beta_1$ (for a first-order random-walk) or $\beta_1$ and $\beta_2$ (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 $\beta_i$). The amount of smoothness is controlled by the smoothing parameter $\tau^2$, which is a global smoothing parameter. In other words, the same amount of smoothing is applied at different covariate values (frequency in our case).

**Spike and Slab Prior**

One of the earliest priors on 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

$$\beta_i \mid \rho_i, c \sim \rho_i N(0, c^2) + (1 - \rho_i)N(0, \epsilon^2), \quad \rho_i \sim \text{Ber}(\pi), \tag{2.9}$$

where $\rho_i \sim \text{Ber}(\pi)$ means that $\rho_i$ has a Bernoulli distribution with sucess probability $\pi$. The parameter $c$ is called the slab width.

In Equation (2.9), the first term on the right-hand side is called a slab. The variance $c^2$ is relatively large so $N(0, c^2)$ has its support over a wide range of plausible values of $\beta_i$. The second component is the spike with $\epsilon^2 << c^2$. If we set $\epsilon = 0$, then the spike is called the Dirac delta function with

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

**The Horseshoe Prior**

The setting of the horseshoe prior is as follows.

$$\beta_i \mid \lambda_i, \tau \sim N(0, \lambda_i^2 \tau^2), \quad \lambda_i \mid \sigma \sim C^+(0, \sigma), \quad \tau \mid \eta \sim C^+(0, \eta). \tag{2.10}$$

In Equation (2.10), $C^+(0, a)$ denotes the half-Cauchy distribution with scale parameter $a$. We can see that the level of shrinkage of $\beta_i$ is controlled by two parameters, $\lambda_i^2$ (the local smoothing parameter) and $\tau^2$ (the global smoothing parameter). Thus, the horseshoe prior can shrink globally (via $\tau^2$) and yet act locally (via $\lambda_i^2$). The global parameter $\tau^2$ pulls all the weights globally towards zero, while the thick half-Cauchy tails for the local scales $\lambda_j$ allow some of the weights to escape the shrinkage, see Carvalho et al. (2010).

The density function of the horseshoe prior (Figure 2.4) has an infinitely tall spike at the origin and flat, Cauchy-like tails. These two features allow $\beta_i$s with large values to remain large and force small $\beta_i$s to shrink to values close to zero.

In Equation (2.10), set $\tau = \sigma = 1$ and let $\kappa_i = \frac{1}{1+\lambda_i^2}$, then we obtain

$$E(\beta_i \mid y_i, \lambda_i^2) = \Big(\frac{\lambda_i^2}{1+\lambda_i^2}\Big) y_i + \Big(\frac{1}{1+\lambda_i^2}\Big) 0 = (1 - \kappa_i) y_i, \tag{2.11}$$

where $y_i$ is the observed data.

Under the setting $\tau = \sigma = 1$, $\lambda_i \sim C^+(0, 1)$, $\kappa_i = \frac{1}{1+\lambda_i^2} \sim \mathcal{B}(\frac{1}{2}, \frac{1}{2})$, where $\mathcal{B}(a, b)$ denotes the Beta distribution with parameters $a$ and $b$. Figure 2.5 shows the density curve of $\kappa_i$, which looks like a horseshoe. Most of the mass is concentrated at $\kappa_i = 0$ and $\kappa_i = 1$.

In Equation (2.11), if $\kappa_i = 0$, then $E(\beta_i \mid y_i, \lambda_i^2) = y_i$, which means there is no shrinkage. If $\kappa_i = 1$, then $E(\beta_i \mid y_i, \lambda_i^2) = 0$, which means total shrinkage, see Piironen and Vehtari (2017).

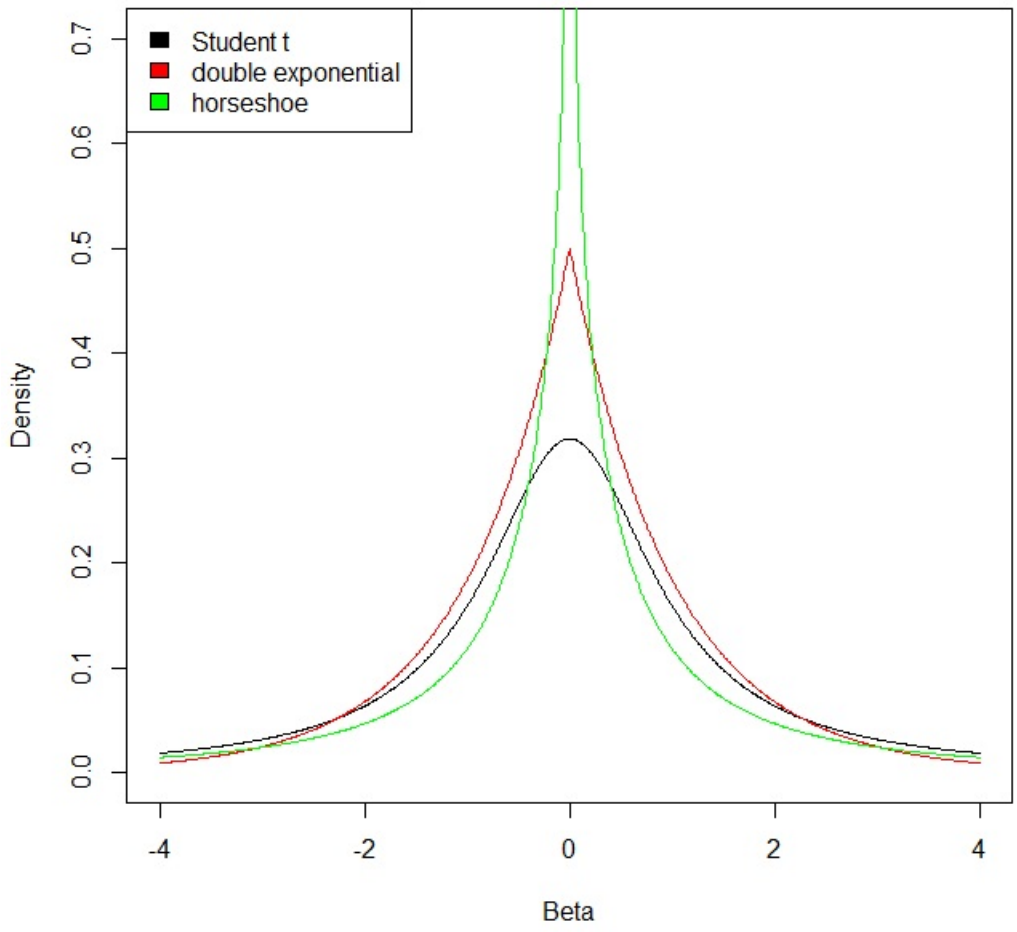


Figure 2.4: Different prior on $\beta_i$.

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

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

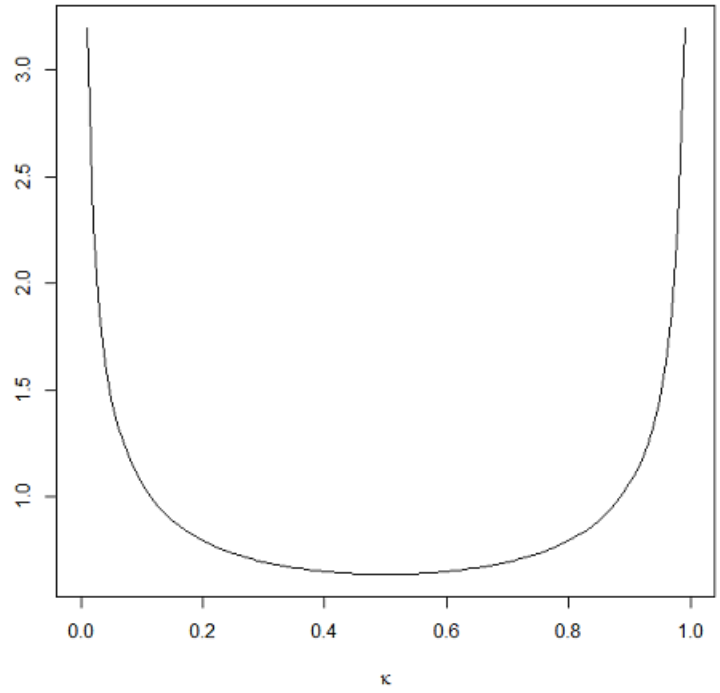


Figure 2.5: The density of $\kappa_i$.

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 $\beta_i$ corresponding to unimportant covariates shrink to zero.

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

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

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 identical, which means under this setting, both the horseshoe prior and the spike and slab prior will assign the same amount of shrinkage to nonzero $\beta_i$s. If the value of $\beta_i$ 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 slab width $c$ and $\rho$. 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.

# Chapter 3

# The Dynamic Shrinkage Prior and Periodic B-splines Basis

To motivate the dynamic shrinkage prior (DSP), proposed by Kowal et al. (2019), consider figures 3.1 and 3.2 which display on their left panels true spatially inhomogeneous signals. These were used by Donoho and Johnstone (1994) as examples of spatially inhomogeneous signals. The right panels display data generated by adding noise to these signals, along with two types of fitted curves. For each signal, there are 128 equally spaced sample points, in the time interval $[0, 1]$. The red lines are based on smoothing splines while the green ones were fit using nonparametric regression with the DSP, implemented in the R package `dsp` (Kowal (2020)). The DSP is explained later in this chapter. As evident from these plots, smoothing splines which are not spatially adaptive, miss some important features of the signals. Nonparametric regression with the DSP does a good job due to its spatial adaptivity. This method allows for different amounts of smoothing in different intervals. This property is advantageous when fitting data from spatially inhomogeneous signals like the ones shown in these plots.

### The Dynamic Shrinkage Prior (DSP)

Priors such as smoothing splines are controlled by a single smoothing parameter which in turn results in a uniform amount of smoothing across the covariate space. Kowal et al. (2019) proposed the dynamic shrinkage prior (DSP), which introduces dependence between local scale parameters. This new prior inherits the desirable shrinkage behavior of popular global–local priors, such as the horseshoe prior (see Section 2.1.1), but has additional

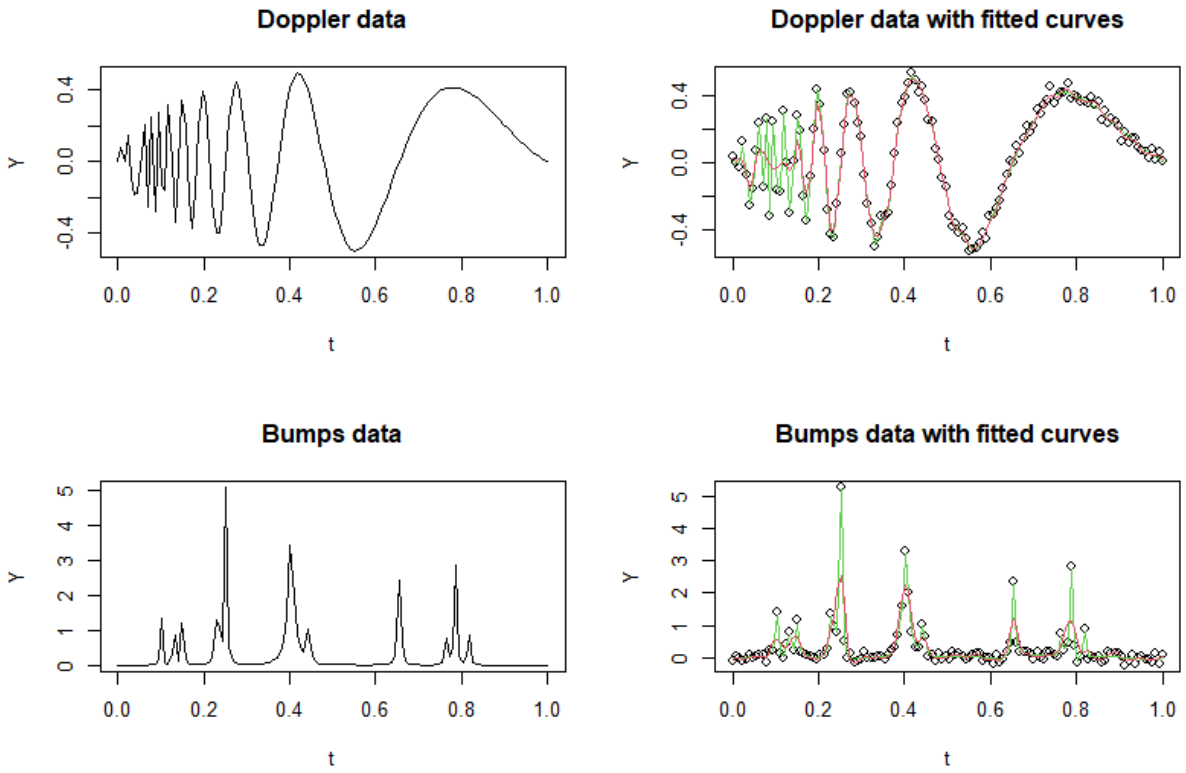


Figure 3.1: Left: Doppler and Bumps signals. Right: Data along with fitted curves. The red lines are based on smoothing splines while the green ones were fit using the nonparametric regression with the DSP.

localized adaptivity, which is important for modeling time series data or regression functions with local features. This prior is placed on the second-order differences of the coefficients of the B-splines, as in Eilers and Marx (2020)

$$\Delta^2 \beta_{i+1} = \upsilon_i, \quad \upsilon_i \mid \tau, \lambda_i \overset{\text{ind}}{\sim} \mathrm{N}(0, \tau^2 \lambda_i^2). \tag{3.1}$$

Just as in the Horseshoe prior, the $\lambda_i^2$'s are the local shrinkage parameters while $\tau^2$ is the global shrinkage parameter. Kowal et al. (2019) then let $h_i = \log(\tau^2 \lambda_i^2)$ and define

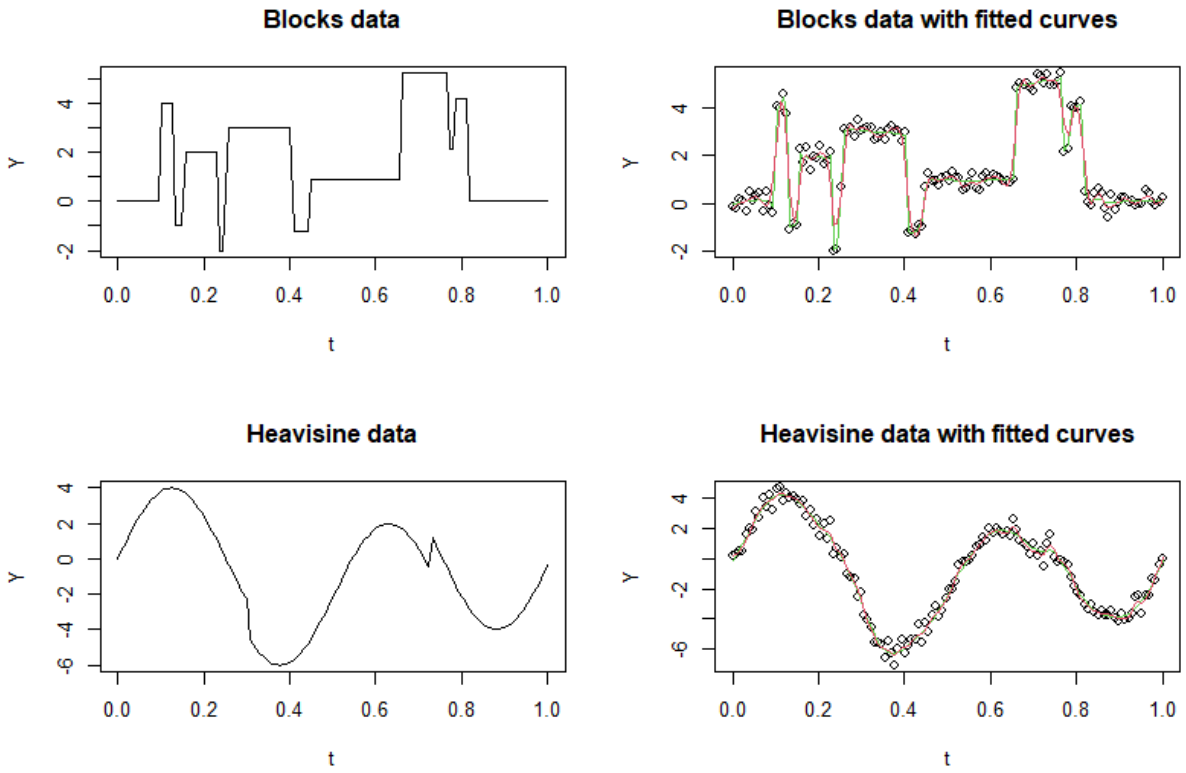


Figure 3.2: Left: Blocks and Heavisine signals. Right: Data along with fitted curves. The red lines are based on smoothing splines while the green ones were fit using the single-component nonparametric regression with the DSP.

$$\begin{aligned} h_{i+1} &= \iota + \phi(h_i - \mu) + \eta_i, \\ \eta_i &\overset{\text{iid}}{\sim} Z(\tfrac{1}{2}, \tfrac{1}{2}, 0, 1), \end{aligned} \tag{3.2}$$

where $\iota = \log(\tau^2)$ and $\phi(h_{i-1} - \iota) + \eta_{i-1} = \log(\lambda_i^2)$. Equation (3.2) induces dependence between $\lambda_i$ and $\lambda_{i+1}$. If $\phi = 0$, then $\log(\lambda_i^2) = \eta_{i-1}$ and $h_i = \iota + \eta_i = \log(\tau^2) + \log(\lambda_i^2)$. Since $\eta_i \overset{\text{iid}}{\sim} Z(\alpha, \beta, 0, 1)$, the $h_i$ are i.i.d, which is the usual global-local prior, i.e., there is no extra spatial adaptivity. When $\phi > 0$, it follows from the first equation of (3.2) that the $\{h_i\}$ follow an AR(1) process, and the dependence between $\lambda_i$ and $\lambda_{i+1}$ is controlled by the AR(1) coefficient $\phi$. Because $\phi$ is positive, the correlation between $\lambda_i$ and $\lambda_{i+1}$ is positive.

The larger $\phi$, the stronger the relation between $\lambda_i$ and $\lambda_{i+1}$. The amount of smoothing in adjacent intervals will thus not change a lot.

The prior on $\tau$ is $\tau \sim C^+(0, \gamma)$, where $\gamma = \frac{\sigma_\epsilon}{\sqrt{M}}$ is constant, and $C^+$ is the half-Cauchy distribution with pdf $P(\tau) = \frac{2}{\pi \cdot \gamma} \cdot \frac{1}{1+(\frac{\tau}{\gamma})^2}$. The evolution equation (3.2) defines $h_{i+1}$ for $i = 1, 2, 3, \ldots, M-1$. Thus when implementing the DSP, one works with $\iota = \log(\tau^2)$ rather than with $\tau^2$. For this reason, we derive below the distribution of $\iota$. Since $\tau \sim C^+(0, 1)$, it follows that the density function of $A = \tau^2$ is

$$\begin{aligned} P(A) &= \frac{2}{\pi \cdot \gamma} \cdot \frac{1}{1 + \frac{A}{\gamma^2}} \cdot \frac{1}{2} \frac{1}{\sqrt{A}} \\ &= \frac{1}{\pi\gamma} \cdot \frac{\gamma^2}{\gamma^2 + A} \cdot \frac{1}{\sqrt{A}} \\ &= \frac{\gamma}{\pi\sqrt{A}(A^2 + \gamma^2)}. \end{aligned}$$

Now, letting $\iota = \log(A) = \log(\tau^2) \implies A = \exp(\iota)$, we obtain

$$P(\iota) = \frac{1}{\pi} \frac{e^{\frac{1}{2}\iota - \log(\gamma)}}{1 + e^{\iota - 2\log(\gamma)}} = \frac{1}{\pi} \cdot \frac{e^{\frac{1}{2}(\iota - 2\log(\gamma))}}{1 + e^{\iota - 2\log(\gamma)}}.$$

By Theorem 1 of Polson et al. (2013), let $p(\xi_\iota)$ denote the pdf of the Pólya-Gamma random variable $\xi_\iota \sim PG(b; 0),\ b > 0$. Then the following integral identity holds for all $a \in \mathbb{R}$.

$$\frac{(e^\psi)^a}{(1 + e^\psi)^b} = 2^{-b} e^{\zeta\psi} \int_0^\infty e^{-\xi_\iota \frac{\psi^2}{2}} p(\xi_\iota) d\xi_\iota,$$

where $\zeta = a - \frac{b}{2}$. Letting $\psi = \iota - 2\log(\gamma)$, we see that

$$\begin{aligned} P(\psi) &= \frac{1}{\pi} \cdot \frac{e^{\frac{1}{2}(\iota - 2\log(\gamma))}}{1 + e^{\iota - 2\log(\gamma)}} = \frac{1}{\pi} \cdot \frac{(e^\psi)^{\frac{1}{2}}}{(1 + e^\psi)^1} \\ &= \frac{1}{2\pi} e^{(\frac{1}{2} - \frac{1}{2})\psi} \int_0^\infty e^{-\xi_\iota \frac{\psi^2}{2}} p(\xi_\iota) d\xi_\iota \\ &= \frac{1}{2\pi} \int_0^\infty e^{-\xi_\iota \frac{\psi^2}{2}} p(\xi_\iota) d\xi_\iota. \end{aligned}$$

Thus, given $\xi_\iota \sim PG(b, 0),\ \psi \mid \xi_\iota \sim \mathrm{N}(0, \xi_\iota^{-1})$. Also, the conditional distribution $\xi_\iota \mid \psi \sim PG(b, \psi)$.

If we let $\psi = \iota - 2\ln(\gamma)$, $a = \frac{1}{2}$ and $b = 1$, then $\zeta = \frac{1}{2} - \frac{1}{2} = 0$. It follows that $P(\iota) = \frac{1}{2\pi}\int_0^\infty e^{-\frac{(\iota - 2\log(\gamma))^2\xi_\iota}{2}} p(\xi_\iota)d\xi_\iota$. Thus $(\iota \mid \xi_\iota, \sigma_\epsilon) \sim \mathrm{N}(2\log(\gamma), \xi_\iota^{-1}) = \mathrm{N}(\log(\frac{\sigma_\epsilon^2}{M}), \xi_\iota^{-1})$ and $\xi_\iota \sim PG(1, 0)$.

Since $\upsilon_i = \Delta^2\beta_{i+1}$ and $\upsilon_i \sim \mathrm{N}(0, \tau^2\lambda_i^2) = \mathrm{N}(0, \exp(h_i))$, we see that $\frac{\upsilon_i}{\exp(\frac{h_i}{2})} \sim \mathrm{N}(0, 1)$ and $\frac{\upsilon_i^2}{\exp(h_i)} \sim \chi_1^2$. Taking the log, we obtain $\log(\upsilon_i^2) - h_i \sim \log(\chi_1^2) \iff \log(\upsilon_i^2) = h_i + \log(\chi_1^2)$. Kastner and Frühwirth-Schnatter (2014) use this expression to write the joint distribution of the $h_i$. The distribution of $\log(\epsilon_i^2)$ can be approximated by the 10-component mixture of normal distributions proposed in Omori et al. (2007). Conditional on the mixture component indicators $s_{10,i}$, $[\log(\epsilon_i^2) \mid s_{10,i}] \sim \mathrm{N}(\mu_{s_{10,i}}, \sigma_{s_{10,i}}^2)$, where $\mu_{10,j}$, $p_{10,j}$ and $\sigma_{10,j}^2, j = 1, \ldots, 10$, are the pre-specified means, weights and variances of the 10-component Gaussian mixture provided in Omori et al. (2007). These values are provided in Table A.1. Thus, $\log(\upsilon_i^2) = h_i + \log(\epsilon_i^2) \mid s_i$ implies $\log(\upsilon_i^2) \sim \mathrm{N}(h_i + \mu_{s_{10,i}}, \sigma_{s_{10,i}}^2)$. In practice, to avoid numerical problems when $\upsilon_i^2$ is too small, we add a small offset $c = 10^{-4}$ to $\upsilon_i^2$, resulting in $\log(\upsilon_i^2 + c) \sim \mathrm{N}(h_i + \mu_{s_{10,i}}, \sigma_{s_{10,i}}^2)$.

A random variable from $\eta_i \sim Z(\frac{1}{2}, \frac{1}{2}, 0, 1)$, can be generated by drawing from $\eta_i \mid \xi_i \sim \mathrm{N}(0, \xi_i^{-1})$ where $\xi_i \sim PG(1, 0)$.

As for $\phi$, Kowal et al. (2019) set $\frac{\phi+1}{2} \sim \mathcal{B}(10, 2)$, which places most of the mass of the density of $\phi$ on (0, 1), so that $\phi$ has a prior mean of 2/3 and a prior mode of 4/5. For $\sigma_\epsilon$, these authors apply Jeffreys' prior, i.e., $p(\sigma_\epsilon) \propto \frac{1}{\sigma_\epsilon}$.

The setting of the DSP is summarized as follows.

$$
\begin{aligned}
\Delta^2\beta_{i+1} &= w_i, \;\; w_i \mid \tau, \lambda_i \overset{\text{ind}}{\sim} \mathrm{N}(0, \tau^2\lambda_i^2), \;\; \text{for } 1 \le i \le M, \\
\tau &\sim C^+(0, \tfrac{\sigma_\epsilon}{\sqrt{M}}), \iota = \log(\tau^2) \implies (\iota \mid \sigma_\epsilon, \xi_\iota^{-1}) \sim \mathrm{N}(\log(\tfrac{\sigma_\epsilon^2}{M}), \xi_\iota^{-1}), \\
&\quad \xi_\iota \sim PG(1, 0), \\
\eta_i \mid \xi_i &\sim \mathrm{N}(0, \xi_i^{-1}), \;\; \xi_i \overset{\text{iid}}{\sim} PG(1, 0), \;\; i = 1, 2, \ldots, M, \\
\tfrac{\phi+1}{2} &\sim \mathcal{B}(10, 2), \;\; p(\sigma_\epsilon) \propto \tfrac{1}{\sigma_\epsilon^2}.
\end{aligned}
\tag{3.3}
$$

As mentioned earlier, in the DSP, the local parameters $\lambda_i$ depend on the AR(1) coefficient $\phi$, and so the shrinkage parameter $\kappa_i$ introduced in Section 2.1.1 also depends on $\phi$.

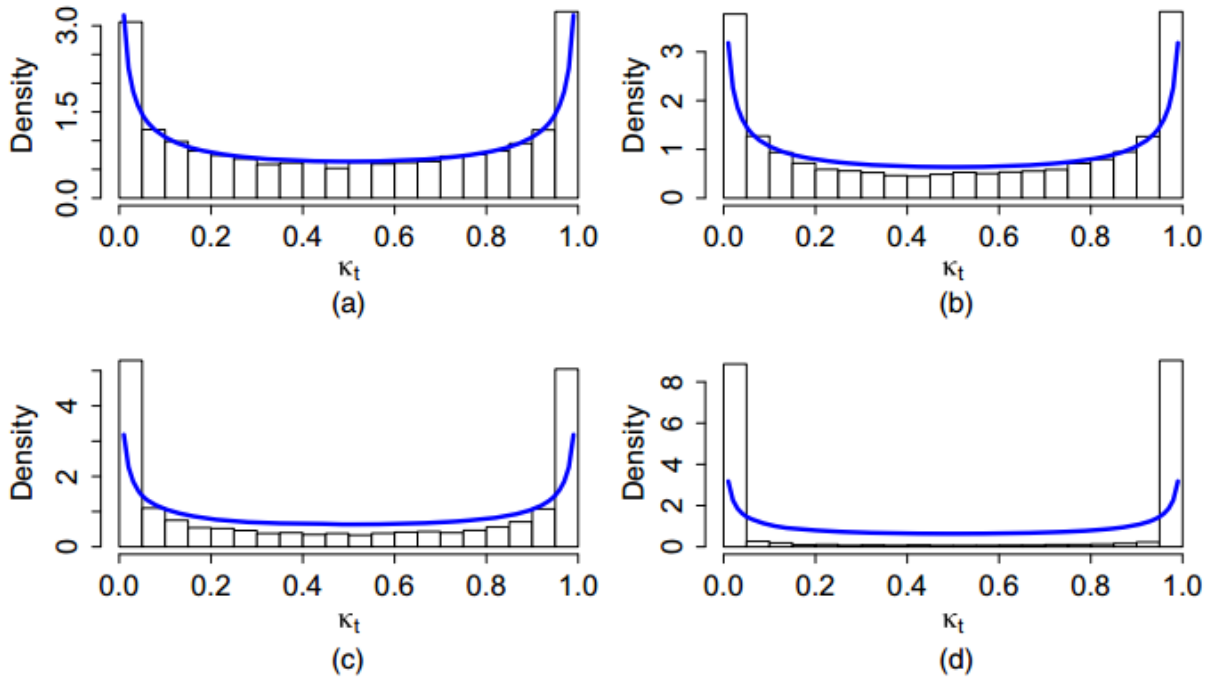


Figure 3.3: Blue line: density of $\kappa_t$ of the horseshoe prior. Histograms: densities of $\kappa_t$ for the DSP. (a) $\phi = 0.25$, (b) $\phi = 0.5$, (c) $\phi = 0.75$, (d) $\phi = 0.99$. The plot is from Kowal et al. (2019).

Figure 3.3 displays simulation-based estimates of the stationary distribution of $\kappa_i$ for various AR(1) coefficients $\phi$. The blue line represents the density of the shrinkage parameter $\kappa_i$ of the horseshoe prior, i.e., $\kappa_i \sim \mathcal{B}(\frac{1}{2}, \frac{1}{2})$. The histograms show the densities of $\kappa_i$ in the DSP for different values of $\phi$: (a) $\phi = 0.25$, (b) $\phi = 0.5$, (c) $\phi = 0.75$, (d) $\phi = 0.99$. We see that when $\phi$ is close to 0, which means weak dependence between $\lambda_i$ and $\lambda_{i+1}$, the DSP and the horseshoe prior are similar. But when $\phi$ is close to 1, compared with the horseshoe prior, the density of $\kappa_i$ for the DSP gives more mass to values near 0 (no shrinkage) and 1 (maximum shrinkage).

### Periodic Basis Functions

For curves with a circular or periodic domain, such as spectral densities, conventional B-splines with the usual prior placed on the second-order differences are not able to provide a periodic fit. To achieve periodic behavior, Eilers and Marx (2020) proposed one way of obtaining periodic smoothing. Specifically, periodic B-splines are used as the basis functions, and instead of the second-order differences of the coefficients, $\Delta^2\beta_\rho = \beta_\rho - 2\beta_{\rho-1} + \beta_{\rho-2}$, the prior is placed on $\beta_\rho - 2\cos(\frac{2\pi}{L})\beta_{\rho-1} + \beta_{\rho-2}$, where $L$ is the number of periodic B-spline basis functions.

### Dynamic Shrinkage for Spectral Estimation (DSSE)

The method we propose in this dissertation is Dynamic Shrinkage for Spectral Estimation (DSSE), which combines the DSP, the mixture of Gaussians approximation to the log(Exp(1)) distribution, and periodic smoothing. Thus, our proposed method is spatially adaptive, it provides a periodic fit, and the computation retains its relative simplicity.

To implement our proposed method, some modifications need to be made to the original setting.

1. Since the spectral density is even and periodic, we usually only estimate it on the positive half of the Fourier frequencies, $[0, \frac{1}{2}]$. However, since we want to apply periodic smoothing, we will estimate the spectral density on the full domain $[-\frac{1}{2}, \frac{1}{2}]$. We let $y_i = \log I(\omega_i)$ for $i = 1, 2, \ldots, n$, where $n$ is the length of the time series.

2. We use periodic B-spines as the basis functions. The number of basis functions is $L = \min(\frac{n}{4}, 40)$, as in Ruppert (2002). This is widely used as a rule of thumb for choosing the number of basis functions in the P-spline literature.

3. Recall Model (2.7), which for convenience is repeated here

$$\boldsymbol{Y} = B\boldsymbol{\beta} + \boldsymbol{\epsilon}.$$

In the context of spectral estimation, the vector $\boldsymbol{Y} = (y_1, \ldots, y_n)^T$ consists of the $\log I(\omega_i)$, and $B$ is the matrix of the basis functions, as described in Section 2.1.1.

As in Carter and Kohn (1997), $\epsilon_i \overset{\text{iid}}{\sim} \log(\frac{\chi_2^2}{2})$. These authors approximate the $\log(\frac{\chi_2^2}{2})$ distribution by the five-component approximation described in Chapter 2. We introduce latent mixture indicators $s_{5,i}$, such that

- $P(s_{5,i} = q) = p_{5,q}$ for $q = 1, 2, \ldots, 5$.
- $s_{5,i} = q \implies \epsilon_i \sim \mathrm{N}(\mu_{5,q}, \sigma_{5,q}^2)$.

4. In the DSP, the prior is placed on the second-order differences of the coefficients of the B-splines, where the difference matrix is

$$D_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -2 & 1 \end{bmatrix}.$$

In our case, to allow periodic smoothing, the difference matrix is changed to

$$D_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 1 & -2\cos(\frac{2\pi}{n}) \\ -2\cos(\frac{2\pi}{n}) & 1 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & -2\cos(\frac{2\pi}{n}) & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -2\cos(\frac{2\pi}{n}) & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -2\cos(\frac{2\pi}{n}) & 1 \end{bmatrix}.$$

## Prior Specification

### The Priors on $\boldsymbol{\beta}$

The prior placed on $\boldsymbol{\beta}$ is

$$\upsilon_1 = \beta_1 - 2\beta_L + \beta_{L-1}; \quad \upsilon_2 = \beta_2 - 2\beta_1 + \beta_L;$$

$$\upsilon_i = \beta_i - 2\beta_{i-1} + \beta_{i-2}, \text{ for } 3 \leq i \leq L,$$

and $\upsilon_i \mid \tau^2, \lambda_i \overset{\text{ind}}{\sim} \mathrm{N}(0, \tau^2\lambda_i^2)$ for $i = 1, 2, \ldots, L$. This can be written in the matrix form as

$$D_2\boldsymbol{\beta} \sim \mathrm{N}_L(\mathbf{0}, \Sigma_\upsilon),$$

where $\mathbf{0}$ is a zero vector of length $L$, and $\Sigma_\upsilon = \mathrm{diag}(\{\tau^2\lambda_i^2\}_{i=1}^L)$, so that

$$P(D_2\boldsymbol{\beta} \mid \Sigma_\upsilon) = (2\pi)^{-\frac{L}{2}} \det(\Sigma_\upsilon)^{-\frac{1}{2}} \exp\Big\{ -\frac{1}{2}(D_2\boldsymbol{\beta})^T \Sigma_\upsilon^{-1} (D_2\boldsymbol{\beta}) \Big\}.$$

### The Prior on $\iota = \log(\tau^2)$

The prior placed on $\tau$ is half Cauchy, i.e., $\tau \sim C^+(0, A_\tau)$, where $A_\tau = \frac{\pi}{\sqrt{6}}$. Thus,

$$P(\tau) = \frac{2}{\pi \cdot A_\tau} \cdot \frac{1}{1 + (\frac{\tau}{A_\tau})^2}.$$

It follows that the density function of $G = \tau^2$ is

$$\begin{aligned} P(G) &= \frac{2}{\pi \cdot A_\tau} \cdot \frac{1}{1 + \frac{G}{A_\tau^2}} \cdot \frac{1}{2} \frac{1}{\sqrt{G}} \\ &= \frac{1}{\pi A_\tau} \cdot \frac{A_\tau^2}{A_\tau^2 + G} \cdot \frac{1}{\sqrt{G}} \\ &= \frac{A_\tau}{\pi} \frac{1}{A_\tau^2 + G} \cdot \frac{1}{\sqrt{G}}. \end{aligned}$$

Let $\iota = \log(G) = \log(\tau^2) \implies G = \exp(\iota)$, the density function of $\iota$ is

$$P(\iota) = \frac{A_\tau}{\pi} \frac{1}{A_\tau^2 + e^\iota} \cdot \frac{1}{e^{\frac{1}{2}\iota}} \cdot e^\iota.$$

We express $\frac{1}{A_\tau^2} = e^{-2\ln(A_\tau)}$ to rewrite the above equation as

$$\begin{aligned} P(\iota) &= \frac{1}{\pi A_\tau} \frac{1}{1 + e^{\iota - 2\ln(A_\tau)}} \cdot e^{\frac{1}{2}\iota} \\ &= \frac{1}{\pi} \cdot \frac{e^{\frac{1}{2}(\iota - 2\log(A_\tau))}}{1 + e^{\iota - 2\log(A_\tau)}}. \end{aligned}$$

From Polson et al. (2013), if we let $p(\xi_\iota)$ denote the density of the random variable $\xi_\iota \sim PG(b, 0),\ b > 0$, then the following integral identity holds for all $a \in R$:

$$\frac{(e^\rho)^a}{(1 + e^\rho)^b} = 2^{-b} e^{k\rho} \int_0^\infty e^{-\xi_\iota \frac{\rho^2}{2}} p(\xi_\iota) d\xi_\iota,$$

where $k = a - \frac{b}{2}$. Letting $\rho = \iota - 2\log(A_\tau)$, we see that

$$\begin{aligned} P(\iota) &= \frac{1}{\pi} \cdot \frac{e^{\frac{1}{2}(\iota - 2\log(A_\tau))}}{1 + e^{\iota - 2\log(A_\tau)}} \\ &= \frac{1}{\pi} \cdot \frac{(e^\rho)^{\frac{1}{2}}}{(1 + e^\rho)^1} \\ &= \frac{1}{2\pi} e^{(\frac{1}{2} - \frac{1}{2})\rho} \int_0^\infty e^{-\xi_\iota \frac{\rho^2}{2}} p(\xi_\iota) d\xi_\iota \\ &= \frac{1}{2\pi} \int_0^\infty e^{-\xi_\iota \frac{\rho^2}{2}} p(\xi_\iota) d\xi_\iota. \end{aligned}$$

Plugging in $\rho = \iota - 2\ln(A_\tau)$, we obtain

$$P(\iota) = \frac{1}{2\pi} \int_0^\infty e^{-\frac{(\iota - 2\log(A_\tau))^2}{2\xi_\iota^{-1}}} p(\xi_\iota) d\xi_\iota.$$

Theorem 1 of Polson et al. (2013) can now be applied to yield $[\iota \mid \xi_\iota^{-1}] \sim \mathrm{N}\big(\log(A_\tau^2), \xi_\iota^{-1}\big)$, where $\xi_\iota \sim PG(1, 0)$.

**The Prior on $h_i = \log(\tau^2 \lambda_i^2)$**

As in the DSP, Equation (3.2) can be rewritten as

$$\eta_i = (h_{i+1} - \iota) - \phi(h_i - \iota).$$

Since $\eta_i \sim Z(\alpha, \beta, 0, 1)$, if $\alpha = \beta = \frac{1}{2}$ we can draw samples $\eta_i \sim Z(\alpha, \beta, 0, 1)$ by $[\eta_i \mid \xi_i] \overset{\text{ind}}{\sim} \text{N}(0, \xi_i^{-1})$ for $[\xi_i] \overset{\text{iid}}{\sim} PG(1, 0)$.

Let $\tilde{\boldsymbol{h}} = (h_1 - \iota, h_2 - \iota, \ldots, h_L - \iota)'$, $\boldsymbol{\eta} = (\eta_0, \eta_1, \ldots, \eta_{L-1})'$ and

$$D_\phi = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ -\phi & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -\phi & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -\phi & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -\phi & 1 \end{bmatrix}_{L \times L}.$$

Then $\boldsymbol{\eta} = D_\phi \tilde{\boldsymbol{h}} \sim \text{N}(0, \Sigma_\xi)$, where $\Sigma_\xi = \text{diag}\{(\xi_i^{-1})_{i=0}^{L-1}\}$.

**The Prior on $\phi$**

The prior placed on $\phi$ is as in the DSP, i.e., $\frac{\phi+1}{2} \sim \mathcal{B}(10, 2)$.

## The Posterior Distribution

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

$$\text{posterior} \propto \text{prior} \times \text{likelihood}. \tag{3.4}$$

**The Conditional Posterior Distribution of $\boldsymbol{\beta}$**

The Augmented likelihood can be expressed as

$$\prod_{i=1}^{N} \Big[ p_{5, s_{5,i}} \cdot \frac{1}{\sqrt{2\pi\sigma^2_{s_{5,i}}}} \exp\Big\{ -\frac{1}{2\sigma^2_{s_{5,i}}} (y_i - \boldsymbol{B}_i\boldsymbol{\beta} - \mu_{s_{5,i}})^2 \Big\} \Big]$$

or

$$P(\boldsymbol{\epsilon} \mid \boldsymbol{s}_5) = (2\pi)^{-N}\det(\Sigma_{\boldsymbol{s}_5})^{-\frac{1}{2}} \exp\Big\{-\frac{1}{2}(\boldsymbol{Y} - B\boldsymbol{\beta} - \boldsymbol{\mu}_{\boldsymbol{s}_5})^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - B\boldsymbol{\beta} - \boldsymbol{\mu}_{\boldsymbol{s}_5})\Big\},$$

where $\boldsymbol{B}_i$ is the $i$-th row of the matrix $B$, and $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_L)^T$ with $\epsilon_i \overset{\text{iid}}{\sim} \log(\frac{\chi^2_2}{2})$. $\boldsymbol{s}_5 = (s_{5,1}, s_{5,2}, \ldots, s_{5,n})^T$, $\boldsymbol{\mu}_{\boldsymbol{s}_5} = (\mu_{s_{5,1}}, \mu_{s_{5,2}}, \ldots, \mu_{s_{5,n}})^T$ and $\Sigma_{\boldsymbol{s}_5} = \text{diag}\{(\sigma_{s^2_{5,i}})_{i=1}^n\}$.

The conditional posterior distribution is proportional to

$$\begin{aligned}
& \exp\Big\{-\frac{1}{2}(\boldsymbol{Y} - B\boldsymbol{\beta} - \boldsymbol{\mu}_{\boldsymbol{s}_5})^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - B\boldsymbol{\beta} - \boldsymbol{\mu}_{\boldsymbol{s}_5})\Big\} \times \exp\Big\{-\frac{1}{2}(D_2\boldsymbol{\beta})^T\Sigma_\upsilon^{-1}(D_2\boldsymbol{\beta})\Big\} \\
\propto\ & \exp\Big\{-\frac{1}{2}[\boldsymbol{\beta}^T B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B\boldsymbol{\beta} - \boldsymbol{\beta}^T B^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}) \\
& -(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5})^T(\Sigma_{\boldsymbol{s}_5})^{-1}B\boldsymbol{\beta} + \boldsymbol{\beta}^T D_2^T\Sigma_\upsilon^{-1}D_2\boldsymbol{\beta}]\Big\} \\
=\ & \exp\Big\{-\frac{1}{2}[\boldsymbol{\beta}^T(B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)\boldsymbol{\beta}] \\
& -\boldsymbol{\beta}^T B^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}) - (\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5})^T(\Sigma_{\boldsymbol{s}_5})^{-1}B\boldsymbol{\beta}]\Big\} \\
\propto\ & \exp\Big\{-\frac{1}{2}[(\boldsymbol{\beta} - (B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)^{-1}B^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}))^T \\
& (B^T\Sigma_{\boldsymbol{s}_5}^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)(\boldsymbol{\beta} - (B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)^{-1}B\Sigma_{\boldsymbol{s}_5}^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}))]\Big\}.
\end{aligned}$$

From the above derivation,

$$\boldsymbol{\beta} \sim \text{N}_n\Big((B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)^{-1}\boldsymbol{X}^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}), (B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)^{-1}\Big). \tag{3.5}$$

Let $(B^T(\Sigma_{\boldsymbol{s}_5})^{-1}B + D_2^T\Sigma_\upsilon^{-1}D_2)^{-1} = Q_\beta^{-1}$ and $B^T(\Sigma_{\boldsymbol{s}_5})^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}_{\boldsymbol{s}_5}) = l_\beta$. Equation (3.5) can now be rewritten as

$$\boldsymbol{\beta} \sim \text{N}_n(Q_\beta^{-1}\boldsymbol{l}_\beta, Q_\beta^{-1}).$$

The vector $\boldsymbol{\beta}$ is sampled as follows.

1. Compute the Cholesky decomposition $Q_\beta = LL^T$, where $L$ is lower triangular.
2. solve $L\boldsymbol{\alpha} = \boldsymbol{l}_\beta$ for $\boldsymbol{\alpha}$.
3. Solve $L^T\boldsymbol{\beta} = \boldsymbol{\alpha} + \boldsymbol{e}$, where $\boldsymbol{e} \sim \text{N}(\boldsymbol{0}, I)$.

**The Conditional Posterior Distribution of $\iota = \log(\tau^2)$**

The prior placed on $\iota = \log(\tau^2)$ is $[\iota \mid \xi_\iota^{-1}] \sim \mathrm{N}(\log(A_\tau^2), \xi_\iota^{-1})$ with $\xi_\iota \sim PG(1,0)$. Therefore

$$\begin{aligned} P(\iota \mid A_\tau, \xi_\iota^{-1}) &= \frac{1}{\sqrt{2\pi\xi_\iota^{-1}}} \exp\Big\{-\frac{1}{2}\frac{(\iota - \log(A_\tau^2))^2}{\xi_\iota^{-1}}\Big\} \\ &\propto \exp\Big\{-\frac{1}{2}\frac{(\iota - \log(A_\tau^2))^2}{\xi_\iota^{-1}}\Big\}. \end{aligned}$$

From

$$h_{i+1} = \iota + \phi(h_i - \iota) + \eta_i, \;\; \eta_i \overset{\text{iid}}{\sim} Z(\alpha, \beta, 0, 1) \;\text{ for } i = 0, 1, \ldots, L-1,$$

.

and $h_1$

$$\iota + \eta_0 = h_1 \sim \mathrm{N}(\iota, \xi_0^{-1}), \xi_0 \sim PG(1,0)$$

we have

$$\begin{aligned} P(h_1 \mid \iota, \xi_0) &= \frac{1}{\sqrt{2\pi\xi_0^{-1}}} \exp\Big\{-\frac{(h_1 - \iota)^2}{2\xi_0^{-1}}\Big\} \\ &\propto \exp\Big\{-\frac{(h_1 - \iota)^2}{2\xi_0^{-1}}\Big\}. \end{aligned}$$

For $h_i$, $i = 1, 2, \ldots, L-1$, we have $h_{i+1} = \iota + \phi(h_i - \iota) + \eta_i$, $\eta_i \overset{\text{iid}}{\sim} Z(\alpha, \beta, 0, 1)$ implies that $h_{i+1} = \phi h_i + (1-\phi)\iota + \eta_i$, so $h_{i+1} \sim \mathrm{N}(\phi h_i + (1-\phi)\iota, \xi_i^{-1})$, $\xi_i \sim PG(1,0)$ for $i = 1, 2, \ldots, L-1$. Then

$$\begin{aligned} p(h_{i+1} \mid \iota, \phi, \xi_i) &= \frac{1}{\sqrt{2\pi\xi_i^{-1}}} \exp\Big\{-\frac{(h_{i+1} - \phi h_i - (1-\phi)\iota)^2}{2\xi_i^{-1}}\Big\} \\ &\propto \exp\Big\{-\frac{(h_{i+1} - \phi h_i - (1-\phi)\iota)^2}{2\xi_i^{-1}}\Big\} \quad \text{for } i = 1, 2, \ldots, L-1. \end{aligned}$$

The likelihood is proportional to

$$\exp\Big\{-\frac{(h_1-\iota)^2}{2\xi_0^{-1}}\Big\}\times\prod_{i=1}^{L-1}\exp\Big\{-\frac{(h_{i+1}-\phi h_i-(1-\phi)\iota)^2}{2\xi_i^{-1}}\Big\}.$$

We now compute the conditional posterior distribution of $\iota$ as follows

$$\begin{aligned}
&\exp\Big\{-\frac{1}{2}\frac{(\iota-\log(A_\tau^2))^2}{\xi_\iota^{-1}}\Big\}\times\exp\Big\{-\frac{(h_1-\iota)^2}{2\xi_0^{-1}}\Big\}\times\prod_{i=1}^{L-1}\exp\Big\{-\frac{(h_{i+1}-\phi h_i-(1-\phi)\iota)^2}{2\xi_i^{-1}}\Big\}\\
\propto\ &\exp\Big\{-\frac{1}{2}[(\xi_\iota+\xi_0+(1-\phi)^2\sum_{i=1}^{L-1}\xi_i)\iota^2-2(\log(A_\tau^2)\xi_\iota+h_1\xi_0\\
&+(1-\phi)\sum_{i=1}^{L-1}(h_{i+1}-\phi h_i)\xi_i)]\Big\},
\end{aligned}$$

which implies that $\iota$ has a normal distribution with mean

$$\xi_\iota+\xi_0+(1-\phi)^2\sum_{i=1}^{L-1}\xi_i)^{-1}(\log(A_\tau^2)\xi_\iota+h_1\xi_0+(1-\phi)\sum_{i=1}^{L-1}(h_{i+1}-\phi h_i)\xi_i$$

and variance

$$\xi_\iota+\xi_0+(1-\phi)^2\sum_{i=1}^{L-1}\xi_i)^{-1}.$$

**The Conditional Posterior Distribution of** $h_i=\log(\tau^2\lambda_i^2)$

The prior placed on $h_i$ is $D_\phi\tilde{\boldsymbol{h}}\sim\mathrm{N}(0,\Sigma_\xi)$, where $\Sigma_\xi=\mathrm{diag}\{(\xi_i^{-1})_{i=1}^L\}$, and $\tilde{\boldsymbol{h}}=(h_1-\iota,h_2-\iota,\ldots,h_L-\iota)^T$, $\boldsymbol{\eta}=(\eta_0,\eta_1,\ldots,\eta_{L-1})^T$.

It follows that

$$P(D_\phi\mid\Sigma_\xi)=(2\pi)^{-\frac{L}{2}}\det(\Sigma_\xi)^{-\frac{1}{2}}\exp\Big\{-\frac{1}{2}(D_\phi\tilde{\boldsymbol{h}})^T\Sigma_\xi^{-1}(D_\phi\tilde{\boldsymbol{h}})\Big\}. \tag{3.6}$$

Denoting $\Sigma_3^{-1}=D_\phi^T\Sigma_\xi^{-1}D_\phi$, Equation (3.6) can be rewritten as

$$P(D_\phi\mid\Sigma_\xi)=(2\pi)^{-\frac{L}{2}}\det(\Sigma_\xi)^{-\frac{1}{2}}\exp\Big\{-\frac{1}{2}\tilde{\boldsymbol{h}}^T\Sigma_3^{-1}\tilde{\boldsymbol{h}}\Big\}.$$

As mentioned in Chapter 2, $\upsilon_i \sim \mathrm{N}(0, \tau^2\lambda_i^2) = \mathrm{N}(0, \exp(h_i))$, from which $\upsilon_i/\exp(\frac{h_i}{2}) \sim \mathrm{N}(0,1)$ and $\upsilon_i^2/\exp(h_i) \sim \chi_1^2$. Taking logs, we obtain $\log(\upsilon_i^2) - h_i \sim \log(\chi_1^2) \iff \log(\upsilon_i^2) = h_i + \log(\chi_1^2)$. Omori et al. (2007) approximate this $\chi_1^2$ distribution with a ten-component mixture of normal distributions, whose pre-specified means, weights and variances are given in Table A.1. We introduce another indicator variable $s_{10,i}$, where

- for fixed $i$, $s_{10,i}$ is a random variable with pmf $P(s_{10,i} = q) = p_{10,q}$ for $q = 1, 2, \ldots, 10$.
- if $s_{10,i} = q \implies (\log(\upsilon_i^2) - h_i) \sim \mathrm{N}(\mu_{10,q}, \sigma_{10,q}^2)$.

Let $\tilde{y_i} = \log(\upsilon_i^2)$, $\boldsymbol{s}_{10} = (s_{10,1}, s_{10,2}, \ldots, s_{10,L})^T$, and $\tilde{\boldsymbol{y}} = (\tilde{y_1}, \tilde{y_2}, \ldots, \tilde{y_L})^T$. Then from $\tilde{y_i} \mid (s_{10,i} = q) \overset{\text{ind}}{\sim} \mathrm{N}(h_i + \mu_{10,q}, \sigma_{10,q}^2)$ we have

$$\tilde{\boldsymbol{y}} \mid \boldsymbol{s}_{10} \sim \mathrm{N}(\boldsymbol{\mu}_{\boldsymbol{s}_{10}} + \tilde{\boldsymbol{h}} + \tilde{\boldsymbol{\iota}}, \Sigma_{\boldsymbol{s}_{10}}),$$

where $\boldsymbol{\mu}_{\boldsymbol{s}_{10}} = (\mu_{s_{10,1}}, \mu_{s_{10,2}}, \ldots, \mu_{s_{10,L}})^T$, $\tilde{\boldsymbol{\iota}} = (\iota, \iota, \iota, \ldots, \iota)^T$, which is an $L \times 1$ vector. The matrix $\Sigma_{\boldsymbol{s}_{10}} = \mathrm{diag}\{(\sigma_{s_{10,i}}^2)_{i=1}^L\}$. We denote $\tilde{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{s}_{10}} - \tilde{\boldsymbol{\iota}}$ by $\tilde{\boldsymbol{c}}$, which leads to

$$P(\tilde{\boldsymbol{y}} \mid \mu_{s_{10,1}}, \Sigma_{\boldsymbol{s}_{10}} = (2\pi)^{-\frac{L}{2}} \det(\Sigma_{\boldsymbol{s}_{10}})^{-\frac{1}{2}} \exp\Big\{-\frac{1}{2}(-\tilde{\boldsymbol{h}} + \tilde{\boldsymbol{c}})^T \Sigma_{\boldsymbol{s}_{10}}^{-1}(-\tilde{\boldsymbol{h}} + \tilde{\boldsymbol{c}})\Big\}.$$

The conditional posterior is

$$\begin{aligned}
& (2\pi)^{-\frac{L}{2}} \det(\Sigma_\xi)^{-\frac{1}{2}} \exp\Big\{-\frac{1}{2}(D_\phi \tilde{\boldsymbol{h}})^T \Sigma_\xi^{-1}(D_\phi \tilde{\boldsymbol{h}})\Big\} \\
& \times (2\pi)^{-\frac{L}{2}} \det(\Sigma_{\boldsymbol{s}_{10}})^{-\frac{1}{2}} \exp\Big\{-\frac{1}{2}(\tilde{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{s}_{10}} - \tilde{\boldsymbol{h}} - \tilde{\boldsymbol{\iota}})^T \Sigma_{\boldsymbol{s}_{10}}^{-1}(\tilde{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{s}_{10}} - \tilde{\boldsymbol{h}} - \tilde{\boldsymbol{\iota}})\Big\} \\
\propto\ & \exp\Big\{-\frac{1}{2}\tilde{\boldsymbol{h}}^T \Sigma_3^{-1} \tilde{\boldsymbol{h}}\Big\} \times \exp\Big\{-\frac{1}{2}(-\tilde{\boldsymbol{h}} + \tilde{\boldsymbol{c}})^T \Sigma_{\boldsymbol{s}_{10}}^{-1}(-\tilde{\boldsymbol{h}} + \tilde{\boldsymbol{c}})\Big\} \\
\propto\ & \exp\Big\{-\frac{1}{2}[\tilde{\boldsymbol{h}}^T \Sigma_3^{-1} \tilde{\boldsymbol{h}} + \tilde{\boldsymbol{h}}^T \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{h}} - \tilde{\boldsymbol{h}}^T \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{c}} - \tilde{\boldsymbol{c}}^T \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{h}}]\Big\} \\
=\ & \exp\Big\{-\frac{1}{2}[\tilde{\boldsymbol{h}}^T (\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1}) \tilde{\boldsymbol{h}} - \tilde{\boldsymbol{h}}^T \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{c}} - \tilde{\boldsymbol{c}}^T \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{h}}]\Big\} \\
\propto\ & \exp\Big\{-\frac{1}{2}\big[\big(\tilde{\boldsymbol{h}} - (\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1})^{-1} \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{c}}\big)^T (\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1}) \big(\tilde{\boldsymbol{h}} - (\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1}) \Sigma_s^{-1} \tilde{\boldsymbol{c}}\big)\big]\Big\}.
\end{aligned}$$

It follows that the conditional posterior distribution of $\tilde{\boldsymbol{h}}$ is $\mathrm{N}\Big((\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1})^{-1} \Sigma_{\boldsymbol{s}_{10}}^{-1} \tilde{\boldsymbol{c}}, (\Sigma_3^{-1} + \Sigma_{\boldsymbol{s}_{10}}^{-1})^{-1}\Big)$ or $\mathrm{N}\Big((D_\phi^T \Sigma_\xi^{-1} D_\phi + \Sigma_{\boldsymbol{s}_{10}}^{-1}) \Sigma_{\boldsymbol{s}_{10}}^{-1} (\tilde{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{s}_{10}} - \tilde{\boldsymbol{\iota}}), (D_\phi^T \Sigma_\xi^{-1} D_\phi + \Sigma_{\boldsymbol{s}_{10}}^{-1})^{-1}\Big)$.

**The Conditional Posterior Distribution of $\phi$**

The prior placed on $\phi$ is $\frac{\phi+1}{2} \sim \mathcal{B}\big(10, 2\big)$, from which

$$p\Big(\frac{1+\phi}{2}\Big) = \frac{\big(\frac{1+\phi}{2}\big)^9\big(1-\frac{1+\phi}{2}\big)^1}{B(10,2)} = \frac{\big(\frac{1+\phi}{2}\big)^9\big(1-\frac{1+\phi}{2}\big)^1}{\frac{1}{110}} = 110\Big(\frac{1+\phi}{2}\Big)^9\Big(\frac{1-\phi}{2}\Big).$$

It follows that

$$p(\phi) = \frac{110}{2}\Big(\frac{1+\phi}{2}\Big)^9\Big(\frac{1-\phi}{2}\Big) \propto (1+\phi)^9(1-\phi).$$

From

$$h_{i+1} = \iota + \phi(h_i - \iota) + \eta_i, \;\; \eta_i \overset{\text{iid}}{\sim} Z(\alpha, \beta, 0, 1) \;\text{ for } i = 0, 1, \ldots, L-1,$$

which implies that $h_{i+1} = \phi h_i + (1-\phi)\iota + \eta_i$, we see that $h_{i+1} \sim \mathrm{N}(\phi h_i + (1-\phi)\iota, \xi_i^{-1})$ and $\xi_i \sim PG(1,0)$ for $i = 0, 1, \ldots, L-1$. Thus,

$$p(h_{i+1} \mid \iota, \phi, \xi_i) = \frac{1}{\sqrt{2\pi\xi_i^{-1}}}\exp\Big\{-\frac{\big(h_{i+1} - \phi h_i - (1-\phi)\iota\big)^2}{2\xi_i^{-1}}\Big\},$$

from which the likelihood is

$$\begin{aligned}
&\prod_{i=1}^{L-1}\frac{1}{\sqrt{2\pi\xi_i^{-1}}}\exp\Big\{-\frac{\big(h_{i+1} - \phi h_i - (1-\phi)\iota\big)^2}{2\xi_i^{-1}}\Big\} \\
\propto\;& \exp\Big\{-\frac{1}{2}\sum_{i=1}^{L-1}[(h_{i+1}-\iota) - \phi(h_i - \iota)]^2\xi_i\Big\} \\
\propto\;& \exp\Big\{-\frac{1}{2}[\phi^2\sum_{i=1}^{L-1}(h_i-\iota)^2\xi_i - 2\phi\sum_{i=1}^{L-1}(h_{i+1}-\iota)(h_i-\iota)\xi_i]\Big\}.
\end{aligned}$$

The conditional posterior distribution is proportional to

$$(1+\phi)^9(1-\phi) \times \exp\Big\{-\frac{1}{2}[\phi^2\sum_{i=0}^{L-1}(h_i-\iota)^2\xi_i - 2\phi\sum_{i=1}^{L-1}(h_{i+1}-\iota)(h_i-\iota)\xi_i]\Big\},$$

where

$$\exp\Big\{-\frac{1}{2}[\phi^2\sum_{i=0}^{L-1}(h_i-\iota)^2\xi_i-2\phi\sum_{i=1}^{L-1}(h_{i+1}-\iota)(h_i-\iota)\xi_i]\Big\}$$

is the kernel of the normal distribution, $\mathrm{N}\Big(\frac{\sum_{i=1}^{N-1}(h_{i+1}-\mu)(h_i-\mu)\xi_i}{\sum_{i=1}^{N-1}(h_i-\mu)^2\xi_i},\frac{1}{\sum_{i=1}^{N-1}(h_i-\mu)^2\xi_i}\Big)$.

# Chapter 4

# Simulation Study

## Simulated Data

The setup of our simulation study follows that of Maturana-Russel and Meyer (2021). We generate 300 autoregressive time series of orders 1, 2, and 4 with unit variance Gaussian innovations of lengths $n = 128, 256$, and 512.

For the AR(1) process, we fix the first-order autocorrelation $at\phi = 0.9$. For this process, the log spectral density has no peak or abrupt change as shown in Figure 4.1.

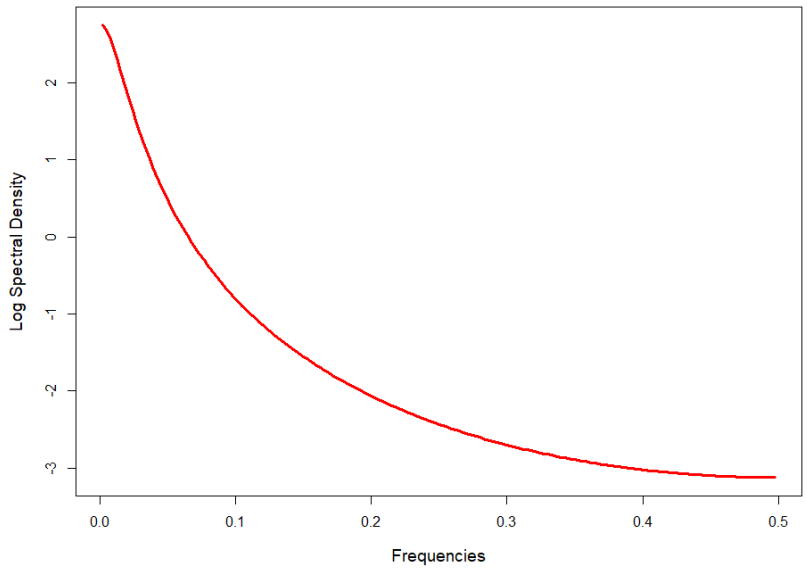


Figure 4.1: Log spectral density of an AR(1) process.

For the AR(2) process, we let $\phi_1 = 0.9$ and $\phi_2 = -0.9$. The corresponding log spectral density has a single peak, see Figure 4.2.

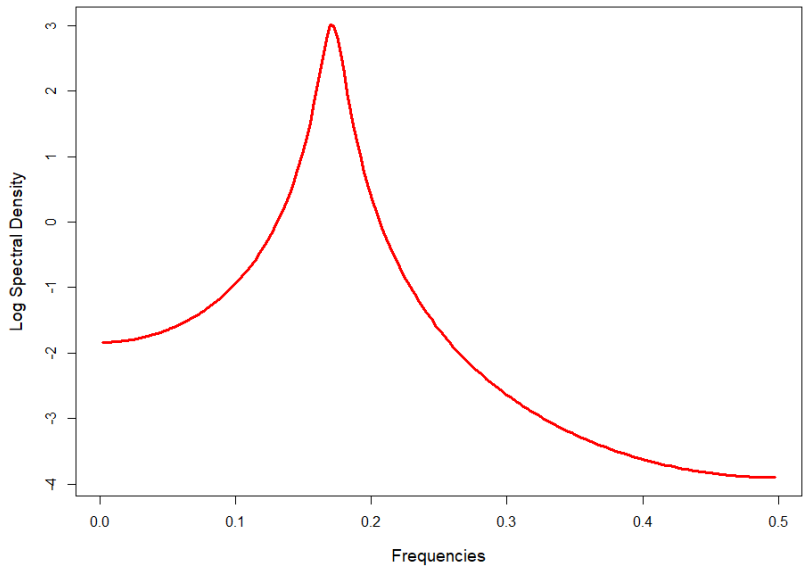


Figure 4.2: Log spectral density of an AR(2) process.

For the AR(4) process, $\phi_1 = 0.9, \phi_2 = -0.9, \phi_3 = 0.9$ and $\phi_4 = -0.9$. The log-spectral density corresponding to this process has two abrupt peaks, which makes the estimation of this spectrum more challenging. Figure 4.3 depicts the log spectrum of this process.

As in Maturana-Russel and Meyer (2021), the accuracy of our estimates is evaluated using the integrated absolute error (IAE) or $L_1$ error. Specifically,

$$\text{IAE} = ||\hat{f} - f||_1 = 2\pi \int_0^{\frac{1}{2}} |\hat{f}(\omega) - f(\omega)| \mathrm{d}\omega, \tag{4.1}$$

where $f(\omega)$ is the spectral density evaluated at frequency $\omega$, and $\hat{f}(\omega)$ is its estimate.

To fit our model, we run our proposed MCMC procedure for a total of 10000 iterations, the first 2000 of which are used for warm-up and then discarded.

To fit the model proposed by Maturana-Russel and Meyer (2021), we use these authors' R package, `psplinePsd`.

In the simulation setting of Maturana-Russel and Meyer (2021) the total number of iterations is 100000. The first 20000 iterations are the pilot run. The warm-up period for

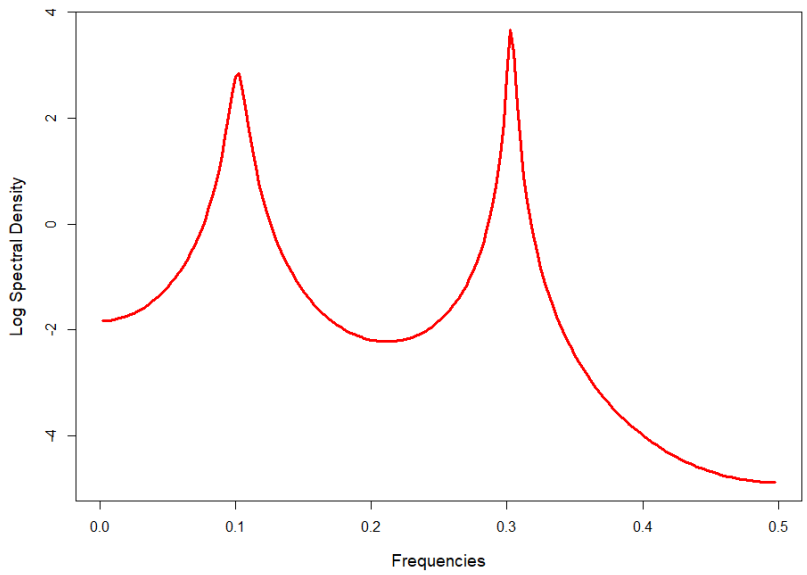


Figure 4.3: Log spectral density of an AR(4) process.

this pilot run is 5000 iterations with a thinning factor of 10. The rest 80000 iterations are called the final sample, which contains a warm-up period of 5000 iterations and a thinning factor of 10. This results in 7500 samples used for posterior inferences. In order to make this setting comparable to ours, we made the following changes: the total number of iterations is set to 10000, the number of iterations for the pilot run is 2000, with 500 iterations as warm-up, and the thinning factor is changed to 1. The number of iterations for the final sample is 8000, with 500 iterations as warm-up and thinning factor of 1. Thus, 7500 samples are used for posterior inferences.

We draw 300 realizations from each combination of process and length and fit our proposed model, as well as the method of Maturana-Russel and Meyer (2021). The IAE is computed for each realization. Tables 4.1, 4.2 and 4.3 display the median IAE we obtain from DSSE, P-spline, and the median IAE shown in Maturana-Russel and Meyer (2021). In the tables, row DSSE shows the medians of the IAES for each time series based on fitting our proposed model. Row P-spline displays the medians of the IAES for each time

series based on fitting the P-spline model with equidistant knots and penalty order equal to 2. Row P-spline in MRM shows the medians of the IAES for each time series shown in Maturana-Russel and Meyer (2021) with equidistant knots and penalty order equal to 2. In Maturana-Russel and Meyer (2021) no result is reported for AR(2) time series. For this reason, Table 4.2 does not include a line for P-spline in MRM.

Table 4.1: Results for AR(1) time series

| | $n = 128$ | $n = 256$ | $n = 512$ |
|---|---|---|---|
| DSSE | 0.8589954 | 0.6894644 | 0.5315180 |
| P-spline | 0.7656965 | 0.6130747 | 0.6416707 |
| P-spline in MRM | 0.6980000 | 0.6090000 | 0.6290000 |

Table 4.2: Results for AR(2) time series

| | $n = 128$ | $n = 256$ | $n = 512$ |
|---|---|---|---|
| DSSE | 1.796634 | 1.344991 | 1.0139690 |
| P-spline | 2.073324 | 1.6904200 | 1.3731590 |

Table 4.3: Results for AR(4) time series

| | $n = 128$ | $n = 256$ | $n = 512$ |
|---|---|---|---|
| DSSE | 3.022248 | 2.254752 | 2.0105980 |
| P-spline | 3.172541 | 2.594320 | 2.4018260 |
| P-spline in MRM | 3.149000 | 2.566000 | 2.3870000 |

Tables 4.1, 4.2, 4.3 display the median of the IAE based on our method as well as the P-spline model for different time series. Figures 4.4, 4.5 and 4.6 display the distribution of the IAE. It can be seen that, for the AR(2) and AR(4) processes, which are the more complicated cases, our model exhibits better results. For the AR(1) process, time series

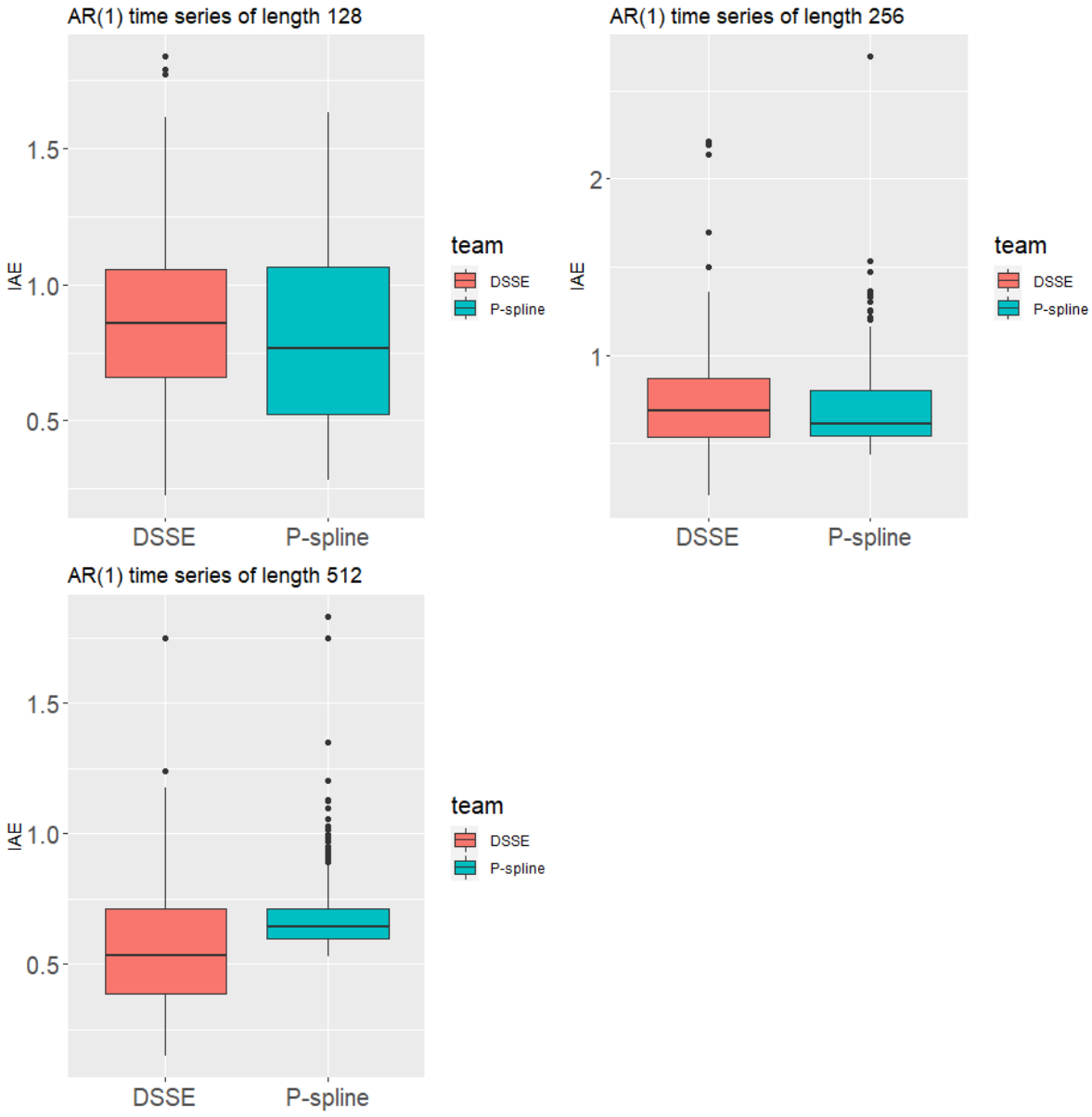


Figure 4.4: Side by side box plots of IAE for the AR(1).

lengths 128 and 256, our model does not perform as well as the P-spline model, but the performance of our model improves with the length of the time series, and outperforms the P-spline model for the length of 512.

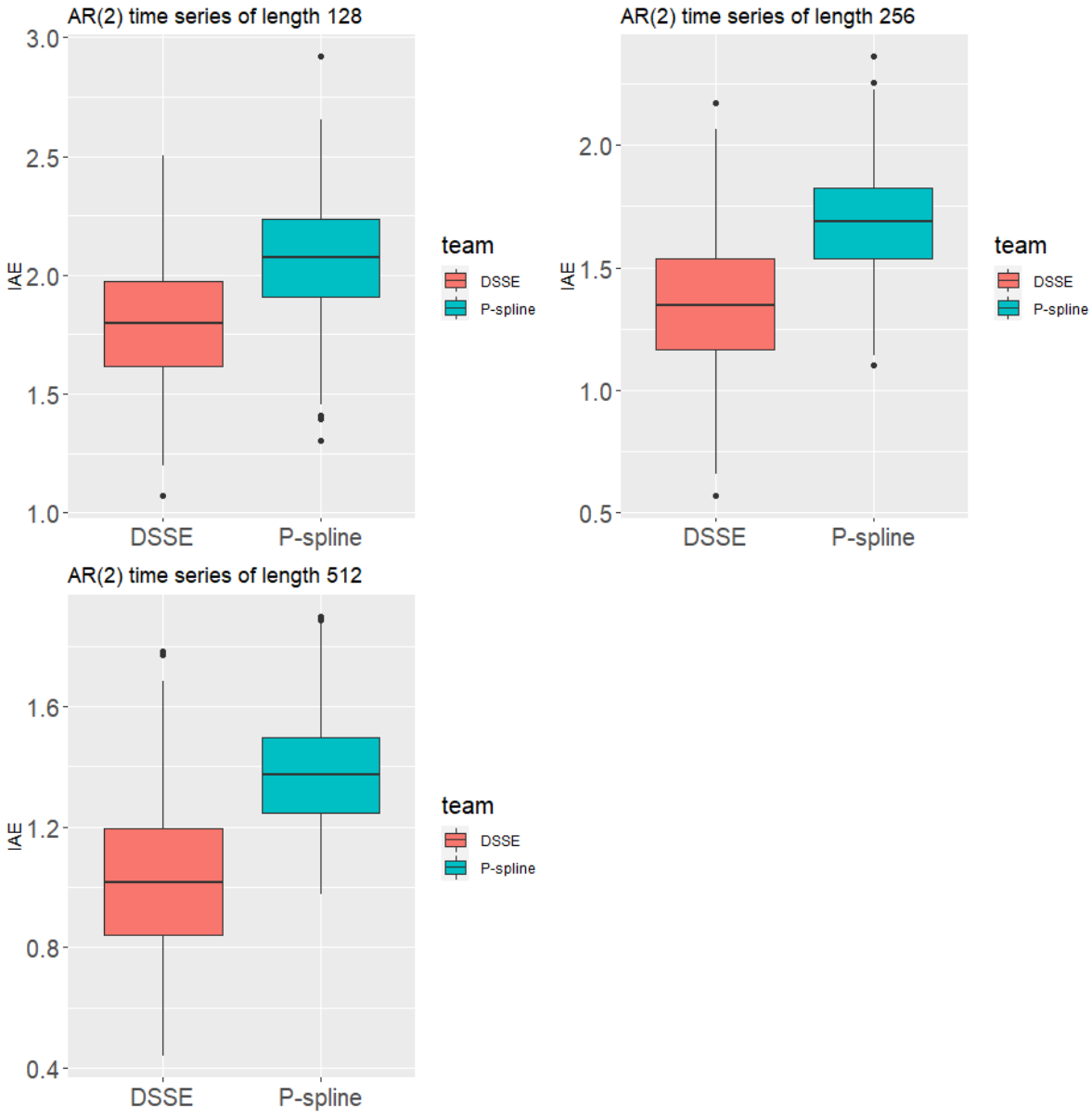


Figure 4.5: Side by side box plots of IAE for the AR(2).

## Application: Ethernet Traffic Data

The Ethernet traffic data set is available in the `longmemo` R package. It contains a time series of length $n = 4000$, which records the number of packages passing through a local area network (LAN) at the Bellcore Morristown Research and Engineering Center (MRE)

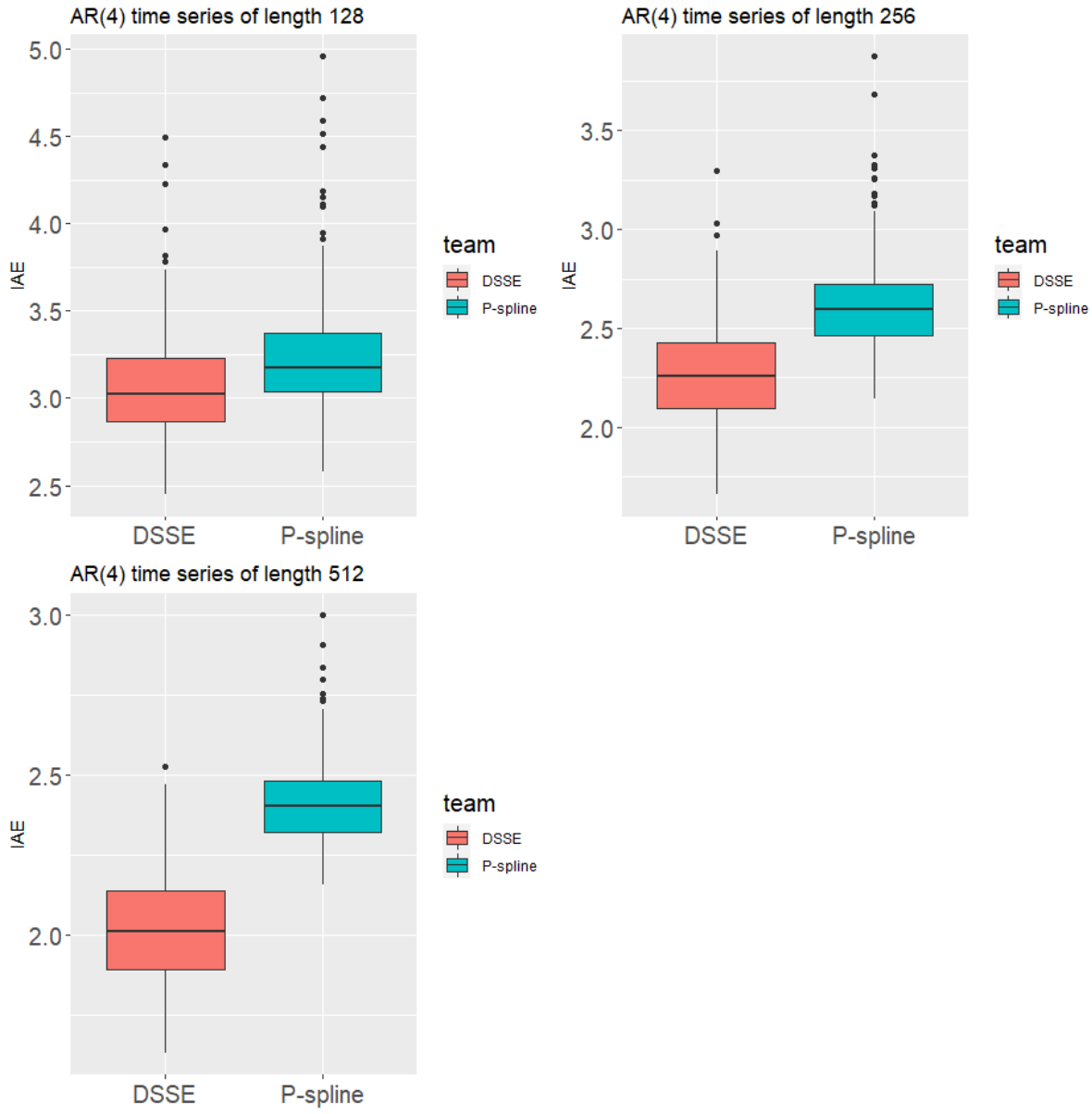


Figure 4.6: Side by side box plots of IAE for the AR(4).

per time unit. This dataset is used in Leland et al. (1993) and Chopin et al. (2013). As in Chopin et al. (2013), we divide all data values by 1000. Figure 4.7 shows the estimated posterior median of the log-spectral density (black line), 90% pointwise credible intervals (blue dashed lines), as well as the log-periodogram (gray line). The estimated log-spectral

density based on the DSSE has peaks at the frequencies $\omega$ =0, $\omega$ =0.21, and $\omega$ =0.4.

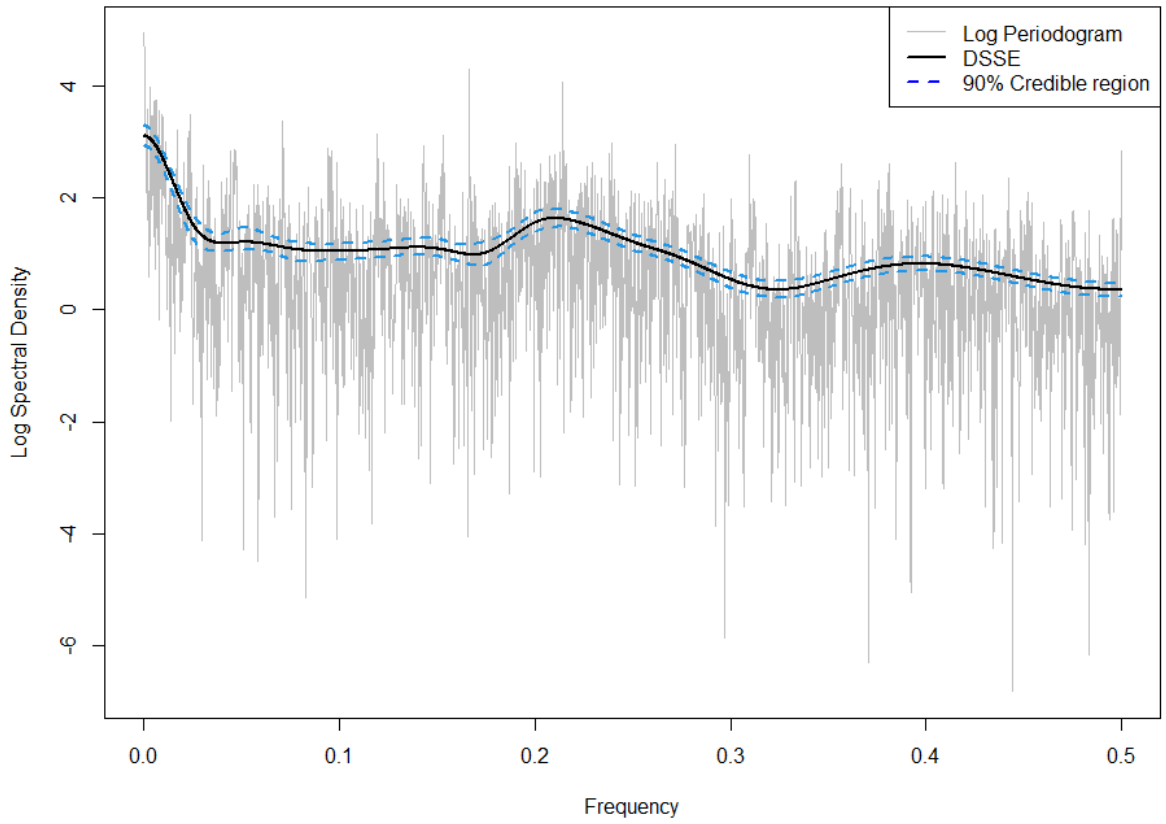


Figure 4.7: Log periodogram for the Ethernet Data along with the fitted log spectral density and 90% pointwise credible intervals.

# 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.

Chopin, N., Rousseau, J., and Liseo, B. (2013), "Computational Aspects of Bayesian Spectral Density Estimation," *Journal of Computational and Graphical Statistics*, 22, 533–557.

Contreras-Cristán, A., Gutiérrez-Peña, E., and Walker, S. G. (2006), "A Note on Whittle's Likelihood," *Communications in Statistics - Simulation and Computation*, 35, 857–875.

Donoho, D. L. and Johnstone, I. M. (1994), "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, 81, 425–455.

Eilers, P. H. and Marx, B. D. (2020), *Practical smoothing: The joys of p-splines*, Cambridge University Press, [Online; accessed 2022-09-13].

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

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

Kastner, G. and Frühwirth-Schnatter, S. (2014), "Ancillarity-sufficiency interweaving strategy (ASIS) for boosting MCMC estimation of stochastic volatility models," *Comput. Statist. Data Anal.*, 76, 408–423.

Kowal, D. R. (2020), *dsp: Dynamic Shrinkage Processes*, r package version 0.1.0 — For new features, see the 'Changelog' file (in the package source).

Kowal, D. R., Matteson, D. S., and Ruppert, D. (2019), "Dynamic shrinkage processes," *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 81, 781–804.

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

Leland, W. E., Taqqu, M. S., Willinger, W., and Wilson, D. V. (1993), "On the Self-Similar Nature of Ethernet Traffic," *SIGCOMM Comput. Commun. Rev.*, 23, 183–193.

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.

Maturana-Russel, P. and Meyer, R. (2021), "Bayesian spectral density estimation using P-splines with quantile-based knot placement," *Computational Statistics*, 36, 2055–2077.

Omori, Y., Chib, S., Shephard, N., and Nakajima, J. (2007), "Stochastic volatility with leverage: fast and efficient likelihood inference," *J. Econometrics*, 140, 425–449.

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

Polson, N. G., Scott, J. G., and Windle, J. (2013), "Bayesian inference for logistic models using Pólya-Gamma latent variables," *J. Amer. Statist. Assoc.*, 108, 1339–1349.

Rosen, O., Wood, S., and Stoffer, D. S. (2012), "AdaptSPEC: Adaptive Spectral Estimation for Nonstationary Time Series," *Journal of the American Statistical Association*, 107, 1575–1589.

Ruppert, D. (2002), "Selecting the Number of Knots for Penalized Splines," *Journal of Computational and Graphical Statistics*, 11, 735–757.

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.

Sykulski, A. M., Olhede, S. C., Guillaumin, A. P., Lilly, J. M., and Early, J. J. (2019), "The debiased Whittle likelihood," *Biometrika*, 106, 251–266.

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

# Appendix

## A.1 Mixture Component Indicators $\{s_{10,i}\}$

In Omori et al. (2007) , the $\mathrm{P}(s_{10,i} = k)$, $\mu_{10,k}$ and $\sigma^2_{10,k}$ for $k = 1, 2, 3, \ldots, 10$ are shown in the following table.

Table A.1: Table of the 10-component Gaussian mixture

| $k$ | $\mathrm{P}(s_{10,i} = k)$ | $\mu_{10,k}$ | $\sigma^2_{10,k}$ |
|---|---|---|---|
| 1 | 0.00609 | 1.92677 | 0.11265 |
| 2 | 0.04775 | 1.34744 | 0.17788 |
| 3 | 0.13057 | 0.73504 | 0.26768 |
| 4 | 0.20674 | 0.02266 | 0.40611 |
| 5 | 0.22715 | -0.85173 | 0.62699 |
| 6 | 0.18842 | -1.97278 | 0.98583 |
| 7 | 0.12047 | -3.46788 | 1.57469 |
| 8 | 0.05591 | -5.55246 | 2.54498 |
| 9 | 0.01575 | -8.68384 | 4.16591 |
| 10 | 0.00115 | -14.65000 | 7.33342 |

## A.2 R Codes

```
## library needed

library(beyondWhittle);  # For psd_arma() function
library(bsplinePsd);
library(psplinePsd);  ## for gibbs_pspline
library(JOPS);  ##for cdiff
library(BayesBD);  ##for uni.slice
library(BayesLogit);  ## for rpg function
library(Matrix); ## for bandSparse

##########################
#### user-defined functions

### calculate the IAE

calIAE = function(estimatedpsd,truepsd){

  selestimatedpsd = estimatedpsd[-c(1,length(estimatedpsd))];

  len1 = length(selestimatedpsd);

  len2 = length(truepsd);

  if (len1 == len2) {
    IAE = sum(abs(selestimatedpsd - truepsd))*pi/(len2+1);
  }
  else{
    stop("the length is not the same");
```

```
  };
  IAE;
};

## calculate the logperiodogram for my method

callogperiodogram = function(timeseriesdata,N){

  n <- length(data);

  pdgrm.mid = (abs(stats::fft(data))^2/(2 * pi * n))[1:N];

  pdgrm.mid2 = pdgrm.mid[-1];

  logperiodogram = log(c(rev(pdgrm.mid2),pdgrm.mid2));

  logperiodogram;

};

## sample the index of mixture of components

ncind = function(y,mu,sig,q){
  sample(1:length(q),
         size = 1,
         prob = q*dnorm(y,mu,sig))
}
```

```
samplefivecompoindicators = function(Y,beta_times_tX){

  Y = as.matrix(Y);
  beta_times_tX = as.matrix(beta_times_tX);
  # make sure they are in matrix form
  n = nrow(Y); p = ncol(Y);

  # Carter & Kohn 5-component mixture:
  m_st  = c(-2.20,-0.8,-0.55,-0.035,0.48);
  v_st2 = c(1.93,1.01,0.69,0.60,0.29);
  q     = c(0.19,0.11,0.27,0.25,0.18);

  # Sample the mixture components
  z = sapply(Y - beta_times_tX, ncind, m_st, sqrt(v_st2), q);

  # Subset mean and variances to the sampled mixture components;
  ##(n x p) matrices
  m_st_all = matrix(m_st[z], nr=n);

  v_st2_all = matrix(v_st2[z], nr=n);

  # Return the (uncentered) log-vols
  list(fiveindicators = z, fivemus = m_st_all,
  fivevariance = v_st2_all);
}

initDHSparameters = function(omega){
```

```
# "Local" number of time points
omega = as.matrix(omega)
n = nrow(omega); p = ncol(omega)

# Initialize the log-volatilities:
ht = log(omega^2 + 0.0001)

## Initialize the AR(1) model to obtain unconditional mean
## and AR(1) coefficient
arCoefs = apply(ht, 2, function(x){
  params = try(arima(x, c(1,0,0))$coef, silent = TRUE);
  if(class(params) == "try-error")
  params = c(0.8, mean(x)/(1 - 0.8))
  params
})
# the mu in evolution equation
dhs_mean = arCoefs[2,];
# the phi in evolution equation
dhs_phi = arCoefs[1,]; dhs_mean0 = mean(dhs_mean)

# Initialize the SD of log-vol innovations
# simply using the expectation:
sigma_eta_t = matrix(pi, nr = n-1, nc = p)
sigma_eta_0 = rep(pi, p) # Initial value

# Evolution error SD:
sigma_wt = exp(ht/2)
```

```
  list(sigma_wt = sigma_wt, ht = ht, dhs_mean = dhs_mean,
  dhs_phi = dhs_phi, sigma_eta_t = sigma_eta_t,
  sigma_eta_0 = sigma_eta_0, dhs_mean0 = dhs_mean0)
}


samplehtandindicators =
function(omega, ht, mu, phi, sigma_eta_t, sigma_eta_0){

  # Compute dimensions:
  omega = as.matrix(omega);
  ht = as.matrix(ht); # Just to be sure (T x p)
  n = nrow(ht); p = ncol(ht);

  # Omori, Chib, Shephard, Nakajima (2007) 10-component mixture:
  m_st  = c(1.92677, 1.34744, 0.73504, 0.02266, -0.85173,
  -1.97278, -3.46788, -5.55246, -8.68384, -14.65000);
  v_st2 = c(0.11265, 0.17788, 0.26768, 0.40611,  0.62699,
  0.98583,  1.57469,  2.54498,  4.16591,   7.33342);
  q     = c(0.00609, 0.04775, 0.13057, 0.20674,  0.22715,
  0.18842,  0.12047,  0.05591,  0.01575,   0.00115);

  # Add an offset: common for all times,
  # but distinct for each j=1,...,p
  yoffset = tcrossprod(rep(1,n),
       apply(as.matrix(omega), 2,
       function(x) any(x^2 < 10^-16)*max(10^-8, mad(x)/10^6)));
```

```
# This is the response in our DLM, log(y^2)
ytilda = log(omega^2 + yoffset);

# Sample the mixture components
z = sapply(ytilda-ht, ncind, m_st, sqrt(v_st2), q);

# Subset mean and variances to the
# sampled mixture components; (n x p) matrices
m_st_all = matrix(m_st[z], nr=n);
v_st2_all = matrix(v_st2[z], nr=n);

# Joint AWOL sampler for j=1,...,p:

# Constant (but j-specific) mean
mu_all = tcrossprod(rep(1,n), mu);

# Constant (but j-specific) AR(1) coef
phi_all = tcrossprod(rep(1,n), phi);

# Linear term:
linht = matrix((ytilda - m_st_all - mu_all)/v_st2_all);

# Evolution precision matrix (n x p)
evol_prec_mat = matrix(0, nr = n, nc = p);
evol_prec_mat[1,] = 1/sigma_eta_0^2;
evol_prec_mat[-1,] = 1/sigma_eta_t^2;

# Lagged version, with zeros as appropriate (needed below)
```

```
  evol_prec_lag_mat = matrix(0, nr = n, nc = p);
  evol_prec_lag_mat[1:(n-1),] = evol_prec_mat[-1,];

  # Diagonal of quadratic term:
  Q_diag = matrix(1/v_st2_all +
  evol_prec_mat + phi_all^2*evol_prec_lag_mat);

  # Off-diagonal of quadratic term:
  Q_off = matrix(-phi_all*evol_prec_lag_mat)[-(n*p)];

  # Quadratic term:
  QHt_Matrix = bandSparse(n*p,
  k = c(0,1), diag = list(Q_diag, Q_off), symm = TRUE)

  # Cholesky:
  chQht_Matrix = Matrix::chol(QHt_Matrix)

  # Sample the log-vols:
  htp1 = mu_all +
  matrix(Matrix::solve(chQht_Matrix,
  Matrix::solve(Matrix::t(chQht_Matrix),
  linht) + rnorm(length(linht))), nr = n);

  htp1tilde = htp1 - mu_all;

  # Return the (uncentered) log-vols
  list(indicators = z, htp1tilde = htp1tilde, htp1 = htp1);
}
```

```
samplephi = function(h_tilda, phi, sigma_eta_t, prior_dhs_phi){

  # Compute dimensions:
  n = nrow(h_tilda); p = ncol(h_tilda)

  # Loop over the j=1:p
  for(j in 1:p){

    # Compute "regression" terms for dhs_phi_j:
    y_ar = h_tilda[-1,j]/sigma_eta_t[,j]
    # Standardized "response"
    x_ar = h_tilda[-n,j]/sigma_eta_t[,j]
    # Standardized "predictor"

    # Using Beta distribution:
    # Check to make sure the prior params make sense
    if(length(prior_dhs_phi) != 2)
    stop('prior_dhs_phi must be a numeric vector of length 2')

    dhs_phi01 = (phi[j] + 1)/2;
    # ~ Beta(prior_dhs_phi[1], prior_dhs_phi[2])

    # Slice sampler when using Beta prior:
    dhs_phi01 = uni.slice(dhs_phi01, g = function(x){
      -0.5*sum((y_ar - (2*x - 1)*x_ar)^2) +
        dbeta(x, shape1 = prior_dhs_phi[1],
        shape2 = prior_dhs_phi[2], log = TRUE)
```

```
    }, lower = 0, upper = 1)[1]#},
    lower = 0.005, upper = 0.995)[1] #

    phi[j] = 2*dhs_phi01 - 1;

  }
  phi;
}

sampleMu = function(h, mu, phi, sigma_eta_t,
sigma_eta_0, log_scale = 0){

  # Compute "local" dimensions:
  h = as.matrix(h);

  n = nrow(h); p = ncol(h)

  # Sample the precision term(s)
  dhs_mean_prec_j = rpg(num = p, h = 1,
  z = mu - log_scale)  ##xi_mu in my code

  # Now, form the "y" and "x" terms in the (auto)regression
  y_mu = (h[-1,] - tcrossprod(rep(1,n-1),
  phi)*h[-n,])/sigma_eta_t;
  x_mu = tcrossprod(rep(1,n-1), 1 - phi)/sigma_eta_t

  # Include the initial sd?
  y_mu = rbind(h[1,]/sigma_eta_0, y_mu);
```

```
  x_mu = rbind(1/sigma_eta_0, x_mu)

  # Posterior SD and mean:
  postSD = 1/sqrt(colSums(x_mu^2) + dhs_mean_prec_j)
  postMean = (colSums(x_mu*y_mu) +
  log_scale*dhs_mean_prec_j)*postSD^2
  dhs_mean = rnorm(n = p, mean = postMean, sd = postSD)

  list(dhs_mean = dhs_mean, dhs_mean_prec_j = dhs_mean_prec_j)
}

createbandsparse = function(diagvector,Kconstant){

  Numofdim = length(diagvector);

  repdiag = c(diagvector,diagvector);

  outputmartrix = bandSparse(Numofdim,
  k = c(0,1,2,(Numofdim-2),(Numofdim-1)),
        diag = list(4*(diagvector)*Kconstant^2 +
        repdiag[2:(Numofdim+1)] +
        repdiag[Numofdim:(2*Numofdim-1)],
        -2*Kconstant*(repdiag[1:(Numofdim-1)]
        + repdiag[2:Numofdim]),
         repdiag[2:(Numofdim-1)],
         repdiag[Numofdim:(Numofdim+1)],
         -2*Kconstant*(diagvector[1] + diagvector[Numofdim])),
          symm = TRUE);
```

```
  outputmartrix
}

## global setting

nsave = 20;

nstart = 1000;

ndatasets = 300;  ## run 5 time series datasets

nlength = 512;  ## length of the time series dataset

## niter is the number of iterations.

niter = 10000;

## warmup is the number of warmup iterations.

warmup = 0.25*niter;

thin = 1;

# prior_dhs_phi = c(10,2);

alphaPlusBeta = 1;

###half cauchy c+(0,A)
```

```
A = 1;

### dimension of beta

P = 1;

## number of knots

# num_knots = max(20, min(ceiling(N/4), 150));

## generate the true psd which will be used for calculate the IAE

freq = 2 * pi / nlength *
(1:(nlength / 2 + 1) - 1)[-c(1, nlength / 2 + 1)];
# Remove first and last frequency

psd.true = psd_arma(freq, ar = c(0.9),
ma = numeric(0), sigma2 = 1);  # True PSD

## create matrices to store the results from
## the gibbs_pspline function

estilogspd = matrix(data = NA, nrow = ndatasets,
ncol = nlength / 2); ## row for each data set

estimedianlogspd = matrix(data = NA, nrow = ndatasets,
ncol = nlength / 2); ## row for each data set
```

```
thinedestilogspd = matrix(data = NA, nrow = ndatasets,
ncol = nlength / 2); ## row for each data set

thinedestimedianlogspd = matrix(data = NA,
nrow = ndatasets, ncol = nlength / 2); ## row for each data set

estilogspdupper = matrix(data = NA, nrow = ndatasets,
ncol = nlength / 2); ## row for each data set

estilogspdlower = matrix(data = NA, nrow = ndatasets,
ncol = nlength / 2); ## row for each data set

IAEVectormymethod = rep(NA,ndatasets);

thinedIAEVectormymethod = rep(NA,ndatasets);

IAEVectormymethodmedian = rep(NA,ndatasets);

thinedIAEVectormymethodmedian = rep(NA,ndatasets);

for (iter in 1:ndatasets) {

  set.seed(nstart + iter);

  ##creat pdf file to store the plots

  if (iter%%nsave == 1) {
```

```
  pdf(paste0("C:/Users/Yi Xie/Dropbox/yiexie/
  Samplingscheme/2022/Sep 10 2022/Plots/Plots ",
  nstart + iter - 1000,
  " to ", nstart - 1000 + nsave + iter -1,
  "my method AR 1 length 512.pdf"));
  par(mfrow = c(2,1));
}

## generate the random seed and the time series dataset

data = arima.sim(nlength, model = list(ar = c(0.9)));
data = data - mean(data);

################################# My method

Y = callogperiodogram(data,nlength/2+1);

num_knots = min(round(nlength/4),40);

N = nlength;

################ Matrices and vectors to store values.

## beta is a N by niter matrix to store
## the betas in each iteration, the ith column is the value of
## the beta in ith iteration.

beta = matrix(data = rep(NA,niter*num_knots),
```

```
ncol = niter, nrow = num_knots);

## phi is a vector to store the phi in each iteration

phi = c(rep(NA,niter));

## sigma_omegas is a matrix to store the sigma_omegas
## in each iteration, the ith column is the value of
## the sigma_omegas in ith iteration, corresponding
## to the evol_sigma_t2 in DSP package.

sigma_omegas = matrix(data = rep(NA,niter*num_knots),
ncol = niter, nrow = num_knots);

## indicators is a N by niter matrix to store
## the indicators in each iteration, the ith
## column is the value of the indicators in ith iteration.

indicators = matrix(data = rep(NA,niter*num_knots),
ncol = niter, nrow = num_knots);

## fivecompoindicators is a N by niter matrix to store
## the five component indicators in each iteration, the
## ith column is the value of the indicators in ith iteration.

fivecompoindicators = matrix(data = rep(NA,niter*N),
ncol = niter, nrow = N);
```

```
## mu is a vector to store the mu in each iteration

mu = c(rep(NA,niter));

## h is a N by niter matrix to store the h in each
## iteration, the ith column is the value of
## the h in ith iteration.

h = matrix(data = rep(NA,niter*num_knots),
ncol = niter, nrow = num_knots);

## mega is a N by niter matrix to store the omega
## in each iteration, the ith column is the value
## of the omega in ith iteration.

omega = matrix(data = rep(NA,niter*num_knots),
ncol = niter, nrow = num_knots);

## beta%*%t(X) vector beta_times_tX

beta_times_tX = matrix(data =c(rep(NA,niter*N)),
ncol = niter, nrow = N);

## beta%*%t(X) vector beta_times_tX

beta_times_tX_plus_fivemean = matrix(data =c(rep(NA,niter*N)),
ncol = niter, nrow = N);
```

```
####################################

x = seq(from = -0.5, to = 0.5, length.out = N);

X = cbase(x, -0.5, 0.5, num_knots, 3);

DM = cdiff(num_knots);

sigma_epsilon = sd(Y, na.rm=TRUE);

tDD = t(DM)%*%DM;

Xty = crossprod(X,Y);

n = length(h[,1]);

##### The initial guess for beta,
## phi, sigma_epsilons etc.

########## initial guess for beta

beta[,1] = rnorm(n = num_knots,
mean = 0,sd = sigma_epsilon);

beta_times_tX[,1] = as.numeric(X%*%beta[,1]);

fivemixture = samplefivecompoindicators(Y,beta_times_tX[,1]);
```

```
fivecompoindicators[,1] = fivemixture$fiveindicators;

###############

tXinv = t(X)%*%diag(x = 1/as.vector(fivemixture$fivevariance));

omega[,1] = DM%*%beta[,1];

evolParams = initDHSparameters(omega[,1]);

Qbeta = tXinv%*%X +
t(DM)%*%diag(1/as.vector(evolParams$sigma_wt^2))%*%DM;

# Qbeta = tXinv%*%X +
createbandsparse(diagvector =
1/as.vector(evolParams$sigma_wt^2),
Kconstant = cos(2*pi/num_knots));

lbeta = tXinv%*%(Y-fivemixture$fivemus);

beta_times_tX_plus_fivemean[,1] =
beta_times_tX[,1] + fivemixture$fivemus;

start_time <- Sys.time()

for (i in 2:niter) {
```

```
##update beta

L = chol(Qbeta);

meanbeta = solve(L,solve(t(L),lbeta));

beta[,i] = as.vector(meanbeta +
solve(t(L),rnorm(num_knots,0,1)));

beta_times_tX[,i] = as.numeric(X%*%beta[,i]);

## update the zi

fivemixture = samplefivecompoindicators(Y,beta_times_tX[,i]);

fivecompoindicators[,i] = fivemixture$fiveindicators;

beta_times_tX_plus_fivemean[,i] =
beta_times_tX[,i] + fivemixture$fivemus;

omega[,i] = DM%*%beta[,i];

### sample ht and h_tilda

ht_htilda_indicator =
samplehtandindicators(omega = omega[,i],
ht = evolParams$ht,
mu = evolParams$dhs_mean, phi = evolParams$dhs_phi,
```

```
sigma_eta_t = evolParams$sigma_eta_t,
sigma_eta_0 = evolParams$sigma_eta_0);

h[,i] = evolParams$ht = ht_htilda_indicator$htp1;

indicators[,i] = ht_htilda_indicator$indicators;

h_tilde = as.matrix(ht_htilda_indicator$htp1tilde);

## sample phi
if(!all(evolParams$dhs_phi == 0)
&& !all(evolParams$dhs_phi == 1)){

  phi[i] = evolParams$dhs_phi =
  samplephi(h_tilda = h_tilde,
  phi = evolParams$dhs_phi,
  sigma_eta_t = evolParams$sigma_eta_t,
  prior_dhs_phi = c(10,2));
}

eta_t = h_tilde[-1,] - tcrossprod(rep(1,n-1),
evolParams$dhs_phi)*h_tilde[-n, ];  # Residuals
evolParams$sigma_eta_t = matrix(1/sqrt(rpg(num = (n-1)*P,
h = alphaPlusBeta, z = eta_t)), nc = P); # Sample
evolParams$sigma_eta_0 = 1/sqrt(rpg(num = P,
h = 1, z = h_tilde[1,]));     # Sample the inital

muSample = sampleMu(h = h[,i], mu = evolParams$dhs_mean,
```

```
                        phi = evolParams$dhs_phi,
                        sigma_eta_t = evolParams$sigma_eta_t,
                        sigma_eta_0 = evolParams$sigma_eta_0,
                        log_scale = log((pi/sqrt(6*1))^2));

  mu[i] = evolParams$dhs_mean = evolParams$dhs_mean0
  = muSample$dhs_mean;

  sigma_omegas[,i] = evolParams$sigma_wt = exp(h[,i]/2);

  ### prepare for sampling next beta

  tXinv = t(X)%*%diag(x
  = 1/as.vector(fivemixture$fivevariance));

  Qbeta = tXinv%*%X +
  createbandsparse(diagvector =
  1/as.vector(evolParams$sigma_wt^2),
  Kconstant = cos(2*pi/num_knots));
  lbeta = tXinv%*%(Y-fivemixture$fivemus);

}

### calculate the posterior means and thinned posterior means

rowmeans = rowMeans(beta_times_tX[,warmup:niter])
[1:(nlength/2)];
```

```
rowmeans = rev(rowmeans);

keep <- seq(from = warmup + 1, to = niter, by = thin);

thinedrowmeans = rowMeans(beta_times_tX[,keep])[1:(nlength/2)];

thinedrowmeans = rev(thinedrowmeans);

### calculate the posterior medians and
## thined posterior medians

rowmedians = apply(beta_times_tX[,warmup:niter],
MARGIN = 1, median)[1:(nlength/2)];

rowmedians = rev(rowmedians);

thinedrowmedians = apply(beta_times_tX[,keep],
MARGIN = 1,median)[1:(nlength/2)];

thinedrowmedians = rev(thinedrowmedians);

#######################################

percentiles = apply(X = beta_times_tX[,warmup:niter],
MARGIN = 1,FUN = quantile,
probs = c(0.05,0.95))[,1:(nlength/2)];

percentiles[1,] = rev(percentiles[1,]);
```

```
percentiles[2,] = rev(percentiles[2,]);

plot(freq,Y[nlength/2+(1:(nlength/2-1))],
type = "l", col = "grey", xlab = "Frequency",
     ylab = "log PSD",
     main = paste0("pspline of dataset ", iter));

lines(freq,rowmeans[-nlength/2],col = 'black',lwd = 2);

lines(freq,percentiles[1,-nlength/2],
lwd = 2, lty = 2, col = 4);

lines(freq,percentiles[2,-nlength/2],
lwd = 2, lty = 2, col = 4);

lines(freq, log(psd.true), col = 2,
lty = 3, lwd = 2);  # Overlay true PSD

legend("topright", legend = c("periodogram",
      "posterior mean", "90% credible region",
      "theoretical logspd"),
       col = c("grey", "black", "blue","red"),
       lwd = c(1, 2, 2, 2), lty = c(1, 1, 2, 3));

######## calc and store IAE with posterior mean

IAEmymethod = calIAE(c(0,exp(rowmeans)),psd.true);
```

```
IAEVectormymethod[iter] = IAEmymethod;

thinedIAEmymethod = calIAE(c(0,exp(thinedrowmeans)),psd.true);

thinedIAEVectormymethod[iter] = thinedIAEmymethod;

######## calc and store IAE with posterior median

IAEmymethodmedian = calIAE(c(0,exp(rowmedians)),psd.true);

IAEVectormymethodmedian[iter] = IAEmymethodmedian;

thinedIAEmymethodmedian = calIAE(c(0,
exp(thinedrowmedians)),psd.true);

thinedIAEVectormymethodmedian[iter] = thinedIAEmymethodmedian;

################################

estilogspd[iter,] = rowmeans;

thinedestilogspd[iter,] = thinedrowmeans;

estimedianlogspd[iter,] = rowmedians;

thinedestimedianlogspd[iter,] = thinedrowmedians;
```

```
  estilogspdupper[iter,] = percentiles[2,];

  estilogspdlower[iter,] = percentiles[1,];

  ## close the pdf file

  if (iter%%nsave == 0) {
    dev.off()
  }
  print(iter);
}

end_time <- Sys.time()

runningtime = end_time - start_time


write.csv(estilogspd,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/estilogspd ",
          nstart + 1 - 1000, " to ",
          nstart + 1 - 1000 + ndatasets - 1,
          "AR 1 length 512.csv"), row.names = FALSE);


write.csv(thinedestilogspd,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
```

```
          Samplingscheme/2022/Sep 10 2022/
          saved results/thinedestilogspd ",
          nstart + 1 - 1000, " to ",
          nstart + 1 - 1000 + ndatasets - 1,
          "AR 1 length 512.csv"),
          row.names = FALSE);

write.csv(estimedianlogspd,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/estimedianlogspd ",
          nstart + 1 - 1000, " to ",
          nstart + 1 - 1000 + ndatasets - 1,
          "AR 1 length 512.csv"),
          row.names = FALSE);


write.csv(thinedestimedianlogspd,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/thinedestimedianlogspd ",
          nstart + 1 - 1000, " to ",
          nstart + 1 - 1000 + ndatasets - 1,
          "AR 1 length 512.csv"),
          row.names = FALSE);


write.csv(IAEVectormymethod,
```

```
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/IAEVectormymethod ",
          nstart + 1 - 1000, " to ",
          nstart + 1 - 1000 + ndatasets - 1,
          " AR 1 length 512.csv"),
          row.names = FALSE);

write.csv(thinedIAEVectormymethod,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/thinedIAEVectormymethod ",
                 nstart + 1 - 1000, " to ",
                 nstart + 1 - 1000 + ndatasets - 1,
                 " AR 1 length 512.csv"),
                 row.names = FALSE)

write.csv(IAEVectormymethodmedian,
          paste0("C:/Users/Yi Xie/Dropbox/yiexie/
          Samplingscheme/2022/Sep 10 2022/
          saved results/IAEVectormymethodmedian ",
                 nstart + 1 - 1000,
                 " to ", nstart + 1 - 1000 + ndatasets - 1,
                 " AR 1 length 512.csv"),
                 row.names = FALSE)


write.csv(thinedIAEVectormymethodmedian,
```

```
paste0("C:/Users/Yi Xie/Dropbox/yiexie/
Samplingscheme/2022/Sep 10 2022/
saved results/thinedIAEVectormymethodmedian ",
       nstart + 1 - 1000,
       " to ", nstart + 1 - 1000 + ndatasets - 1,
       " AR 1 length 512.csv"),
       row.names = FALSE)
```

# 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's degree in science (Mathematics and Applied Mathematics). After that, he furthered his study at Changsha University of Science and Technology and received a master's degree in 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 from his job and went to The University of Texas at El Paso. While pursuing a master's degree in Statistics, he worked as a Teaching and Research Assistant. Two years later in the summer of 2018, he started his Ph.D. program in computational science at UTEP.

Contact Information: yxie3@miners.utep.edu