Sequential Monte Carlo in the Frequency Domain for Long-Memory Processes

A thesis presented for the MSci Mathematics degree
Supervised by Dr. James Martin

Matthieu Georges Papahagi

'I affirm that this represents my own work, completed independently, without assistance, except where acknowledged and properly referenced.'

June 13, 2022

Department of Mathematics
Imperial College London
4th Year MSci
CID: 01492464
E-mail: matthieu-georges.papahagi18@imperial.ac.uk

Contents

Sequential Monte Carlo in the Frequency Domain for Long-Memory Processes

Matthieu Papahagi

June 13, 2022

Abstract

In the parametric modelling of seasonal time series, it is often advantageous to infer the distributions of model parameters in the frequency domain to accurately identify the frequencies associated with seasonal components. In particular, much attention has been devoted to constructing and inferring time series models for long memory processes, however most research resorts to Markov Chain Monte Carlo (MCMC) techniques for parameter estimation. The present article builds upon prior frequency-domain modelling work, but instead simulates from the posterior distribution of the model parameters using a Sequential Monte Carlo (SMC) sampling scheme. SMC samplers offers several advantages over MCMC simulation, such as improved computational efficiency and higher robustness in the face of multimodality. In the present work, we focus mainly on the methodological aspects of SMC in the frequency domain, which we demonstrate on both a simulated GARMA process and a real dataset.

Keywords: Long Memory, Seasonal Persistence, Fractional Differencing, Gegenbauer, GARMA, Frequency Domain, Whittle Likelihood, Likelihood Annealing, Sequential Monte Carlo.

1 Introduction

1.1 Aims and Contributions of the Paper

Time series exhibiting long-range seasonal dependence structures are ubiquitous in nature – from biology and geosciences to economics and finance. Models for such time series, particularly for those with multiple periodicities, need to incorporate sufficient complexity, while avoiding overparametrisation. Building on the fundamental autoregressive (AR) and moving average (MA) processes, various models have been developed. Preference has been argued both in favour of simpler but higher-order models such as a high-order autoregressive moving average (ARMA) model, and of more parsimonious, but also more complex models such as the autoregressive integrated moving average (ARIMA) model, its fractionally integrated variant, ARFIMA, or the Gegenbauer ARMA (GARMA) model, the latter being the choice adopted in the present work. Having chosen a suitable model, it is convenient to perform inference in the frequency domain, as this can reveal important hidden periodicities. It is thus crucial that this inference be performed accurately in order to forecast future values of the time series. However, both a high model order and a high model complexity render simulation via standard Markov Chain Monte Carlo (MCMC) methods inefficient, the former due to the high dimensionality of the likelihood function and the latter due to the complex functional form of the spectral density function. Recently, Sequential Monte Carlo (SMC) methods have been successfully applied to posterior simulation in the context of Bayesian inference and have demonstrated several key advantages over standard MCMC algorithms.

The main goal of this paper is to develop the core methodology of SMC in the frequency domain and to highlight some important modelling choices behind this methodology. A secondary goal is to apply this methodology to fitting GARMA models to time series data so as to demonstrate the versatility of this model

for data which exhibits both short and long memory. We begin by discussing in Section 2 the basic theory of time series models and the concept of long memory, focusing on the Gegenbauer ARMA (GARMA) process, which we parametrise in terms of its latent components. In Section 3, we develop the framework for Bayesian inference in the frequency domain, discussing the choices for the prior distributions of the model parameters and for the sample spectral estimator. We also define the Whittle-type likelihood as a frequency-domain approximation to the exact likelihood and sketch a derivation of this approximation. In Section 4, we detail the theory SMC, motivating this procedure from first principles. We also discuss the main algorithmic choices one can make when developing an SMC scheme to sample from a general distribution and highlight some advantages over pure MCMC schemes. In Section 5, we detail how one can reconstruct the spectrum and the time-domain representation of a GARMA process from its latent-structure parametrisation. We also outline a general framework for simulating Gaussian processes from their spectra, and introduce a particular method to simulate GARMA processes directly from their functional representations. These simulation methods can then be used to both generate synthetic datasets and to simulate estimated models. In Section 6, we outline the specific settings and hyperparameter choices of a practical SMC algorithm, and demonstrate this algorithm on a simulated GARMA process as well as on a real dataset. Finally, in Section 7, we discuss several possible extensions and improvements, including demodulation, forecasting, and transdimensional SMC samplers for automated model selection. A full inventory of the notation used can be found in Appendix A.1.

1.2 Connections with Prior Work

The theoretical content in Section 2 on long memory, fractional differencing, and the Gegenbauer process follows closely the discussion in Dissanayake et al. [2018]. The general methodology of frequency-domain Bayesian inference in Section 3 for time series is largely based on the work of McCoy and Stephens [2004] and of Huerta and West [1999a] and Huerta and West [1999b], from which the present paper adopts the the GARMA model, the prior specification, and the latent-structure parametrisation. Subsection 3.5 on the Whittle likelihood is mainly based on McCoy et al. [2007], Sykulski et al. [2019], and Hurvich [2002]. The theoretical background on SMC samplers in Section 4, including the algorithm described in Subsection 4.8, is predominantly based on the work of Neal [2001], Chopin [2002], Del Moral et al. [2006], Chopin and Papaspiliopoulos [2020], and Li et al. [2021]. The background material on MCMC in Subsection 4.7 is based on Gilks et al. [1995]. Finally, the method in Subsection 5.2 of simulating a GARMA process from its spectrum is due to Percival [1992].

For modern and extensive reviews of SMC methods, we refer the interested reader to Naesseth et al. [2019], and Chopin and Papaspiliopoulos [2020]. For a recent article with a similar methodology on estimating GARMA processes using MCMC, see Hunt et al. [2022].

2 Time Series Models

2.1 Fundamental Time Series Models – the MA, AR, and ARMA Models

A discrete stochastic process $\{X_t\}$ is a zero-mean q-th order moving average process MA(q) if it can be expressed in the form:

$$X_t = \epsilon_t - \sum_{j=1}^{q} \theta_j \epsilon_{t-j} \iff$$
$$X_t = \left(1 - \sum_{j=1}^{q} \theta_j B^j\right)\epsilon_t := \Theta(B)\epsilon_t$$

where $\theta_q \neq 0$, $\{\epsilon_t\}$ is a zero-mean white-noise process with variance σ_ϵ^2, and B is the backward-shift operator, defined through the action $BX_t = X_{t-1}$. An MA(q) process is always (second-order) stationary, meaning that

it has constant expectation and variance in time ($\mathbb{E}(X_t)$, $Var(X_t) < +\infty$, constant) and its autocovariance function $s_\tau := Cov\{X_t, X_{t+\tau}\} < +\infty$ depends only on the lag τ.

Using similar notation, the p-th order zero-mean autoregressive process AR(p) can be defined through the recursive equation:

$$X_t = \epsilon_t + \sum_{j=1}^{p} \phi_j X_{t-j} \iff$$
$$\epsilon_t = \left(1 - \sum_{j=1}^{p} \phi_j B^j\right) X_t := \Phi(B) X_t$$

where $\phi_p \neq 0$. An AR(p) process is trivially always invertible, meaning that it can be written in autoregressive form. Combining an AR(p) and MA(q) process, one defines the ARMA(p, q) process through:

$$\Phi(B) X_t = \Theta(B) \epsilon_t$$

One can add a d-th order polynomial trend to the zero-mean ARMA process, by filtering this process through an additional $(1 - B)^d$ term to obtain the autoregressive integrated moving average process ARIMA(p, d, q):

$$\Phi(B)(1 - B)^d X_t = \Theta(B) \epsilon_t$$

One can further add n short-term seasonalities at periods $s_i,\ i = 1, \ldots, n$ by filtering the process through additional terms of the form $(1 - B^{s_i})$.

2.2 Spectral Representation

Throughout the following sections we focus on modelling time series in the frequency domain. For this we state the Spectral Representation Theorem:

Theorem 1 (Spectral Representation)**.** Let $\{X_t\}$ be a real-valued, discrete, zero-mean stationary process. Then there exists a complex-valued orthogonal process $\{Z(f)\}$ defined on $[-1/2, 1/2]$ such that:

$$X_t = \int_{-1/2}^{1/2} e^{2\pi i f t} dZ(f), \quad \forall t \in \mathbb{Z}$$

and the process $\{Z(f)\}$ has the following properties:

- $\mathbb{E}\{dZ(f)\} = 0 \quad \forall f \in [-1/2, 1/2]$.
- $\mathbb{E}\{|dZ(f)|^2\} =: dS^I(f) =: S(f)df \quad \forall f \in [-1/2, 1/2]$, where $S^I(f)$ is called the integrated spectrum of $\{X_t\}$ and $S(f)$ is the spectral density.
- $\forall f, f' \in [-1/2, 1/2],\ Cov\{dZ(f'), dZ(f)\} = \mathbb{E}\{dZ^*(f')dZ(f)\} = 0$

In particular, the spectral density function (SDF) represents in the frequency domain the time-domain information encapsulated by the autocovariance function. As high values of the autocovariance correspond to the lags where the process is most correlated with itself, so do peaks in the spectral density correspond to the frequencies with the highest power. Formally, the autocovariance function and the SDF are Fourier Transform (FT) pairs:

$$S_X(f) = \sum_{\tau=-\infty}^{+\infty} s_\tau e^{-2\pi i f \tau} \quad \text{and} \quad s_\tau = \int_{-1/2}^{1/2} S(f) e^{2\pi i f \tau} df$$

Using the fact that the SDF of the zero-mean white-noise process with variance σ_ϵ^2 is σ_ϵ^2, and applying to this white noise process the linear time invariant (LTI) filters corresponding to the AR and MA processes, it can be shown that the SDF of an ARMA(p,q) process is given by:

$$S_X(f) = \sigma_\epsilon^2 \frac{\left|\Theta(e^{-2\pi i f})\right|^2}{\left|\Phi(e^{-2\pi i f})\right|^2} \tag{2.2.1}$$

with the numerator or the denominator being 1 in the case of a pure AR or MA process, respectively. Writing the AR(p) and MA(q) polynomials in terms of their reciprocal roots ξ_{rj}, $r = 1, 2$:

$$\Phi(z) = \prod_{j=1}^{p}(1 - \xi_{1j}z) \quad \text{and} \quad \Theta(z) = \prod_{j=1}^{q}(1 - \xi_{2j}z)$$

one can rewrite the spectrum of the ARMA(p,q) process as:

$$S_Y(f) = \sigma_\epsilon^2 \frac{\prod_{j=1}^{q}\left|1 - \xi_{2j}e^{2\pi i f}\right|^2}{\prod_{j=1}^{p}\left|1 - \xi_{1j}e^{2\pi i f}\right|^2} \tag{2.2.2}$$

One can further separate these reciprocal roots into R_p and R_q real roots and and $2C_p$ and $2C_q$ complex-conjugate roots. Furthermore, one can write these $p = R_p + 2C_p$ reciprocal AR roots and $p = R_q + 2C_q$ reciprocal MA roots in the following modulus-argument representation:

$$\xi_{1j}^R = \alpha_{1j}^R,\ j = 1, \ldots, R_p, \quad \text{and} \quad \left(\xi_{1j}^C, \overline{\xi_{1j}^C}\right) = (\alpha_{1j}^C e^{2\pi i \omega_{1j}^C}, \alpha_{1j}^C e^{-2\pi i \omega_{1j}^C}),\ j = 1, \ldots, C_p \tag{2.2.3}$$

and

$$\xi_{2j}^R = \alpha_{2j}^R,\ j = 1, \ldots, R_q, \quad \text{and} \quad (\xi_{2j}^C, \overline{\xi_{2j}^C}) = (\alpha_{2j}^C e^{2\pi i \omega_{2j}^C}, \alpha_{2j}^C e^{-2\pi i \omega_{2j}^C}),\ j = 1, \ldots, C_q \tag{2.2.4}$$

where

$$\alpha_{1i}^R \in [0,1],\ i = 1, \ldots, R_p, \quad \alpha_{1i}^C \in (0,1],\ i = 1, \ldots, C_p \tag{2.2.5}$$

are the real AR roots and the moduli of the complex conjugate AR reciprocal roots, respectively, and:

$$\omega_{1i}^R = 0,\ i = 1, \ldots, R_p, \quad \omega_{1i}^C \in (0, 1/2),\ i = 1, \ldots, C_p \tag{2.2.6}$$

are the corresponding arguments. One defines similarly α_{2j}^R, $j = 1, ..., R_q$, α_{2j}^C, $j = 1, ..., C_q$, $\omega_{2j}^R = 0$, $j = 1, ..., R_q$, and ω_{2j}^C, $j = 1, \ldots, C_q$ for the MA reciprocal roots. With this notation, the ARMA(p,q) spectrum becomes:

$$S_X(f) = \sigma_\epsilon^2 \frac{\prod_{j=1}^{Rq}\left|1 - \alpha_{2j}^R e^{2\pi i f}\right|^2 \prod_{j=1}^{Cq}\left|(1 - \alpha_{2j}^C e^{2\pi i f - \omega_{2j}^C})(1 - \alpha_{2j}^C e^{2\pi i f + \omega_{2j}^C})\right|^2}{\prod_{j=1}^{Rp}\left|1 - \alpha_{1j}^R e^{2\pi i f}\right|^2 \prod_{j=1}^{Cp}\left|(1 - \alpha_{1j}^C e^{2\pi i f - \omega_{1j}^C})(1 - \alpha_{1j}^C e^{2\pi i f + \omega_{1j}^C})\right|^2} \tag{2.2.7}$$

From this expression it can be deduced (see McCoy and Stephens [2004]) that the complex AR roots induce bounded spectral peaks at frequencies:

$$\tilde{\omega}_{1j} = \frac{1}{2\pi}\cos^{-1}\left(\frac{1 + (\alpha_{1j}^C)^2}{2\alpha_{1j}^C}\cos(2\pi\omega_{1j}^C)\right)$$

where taller peaks occur for values of α_{1j}^C closer to 1, which correspond to roots close to the unit circle, where the AR process becomes nearly non-stationary. Similarly, the complex MA roots induce local minima (valleys) at frequencies:

$$\tilde{\omega}_{2j} = \frac{1}{2\pi}\cos^{-1}\left(\frac{1 + (\alpha_{2j}^C)^2}{2\alpha_{2j}^C}\cos(2\pi\omega_{2j}^C)\right).$$

2.3 Long Memory and Fractional Differencing

As we saw in the previous section, the complex roots of the AR polynomial correspond to bounded peaks in the spectral density and thus identify the frequencies where there is the highest power density, and consequently autocorrelation, in the data. For most processes, including the ARMA model, the autocorrelation decays exponentially as the number of lags increases. This translates to smooth spectral peaks at the corresponding frequencies. However, for some processes, the autocorrelation function is persistent, i.e. it does not decay much even for a large number of lags. In fact, in many naturally occurring situations, the autocorrelation function displays a decaying oscillatory pattern which can last up to very large time lags. This seasonal persistence phenomenon is also known as long memory. Following Dissanayake et al. [2018], we give the following definition of long-memory processes:

Definition 1 (Long memory)**.** A process $\{X_t\}$ is said to have long memory at the frequency $\tilde{\omega}_0 \in [0, \pi]$ if its autocorrelation function $\rho(\tau)$ satisfies the following equation:

$$\lim_{k\to\infty} \frac{\rho(\tau)}{C_\rho k^{2\delta-1}\cos(\tau\tilde{\omega}_0)} = 1$$

where $\delta \in (0, 1/2)$ is called the memory parameter and $C_\rho > 0$ is a constant.

Intuitively, this means that the autocorelation function $\rho(\tau)$ asymptotically behaves like a damped (co)sinusoidal wave. This phenomenon is illustrated in a comparative manner in Figure 2.1 below.

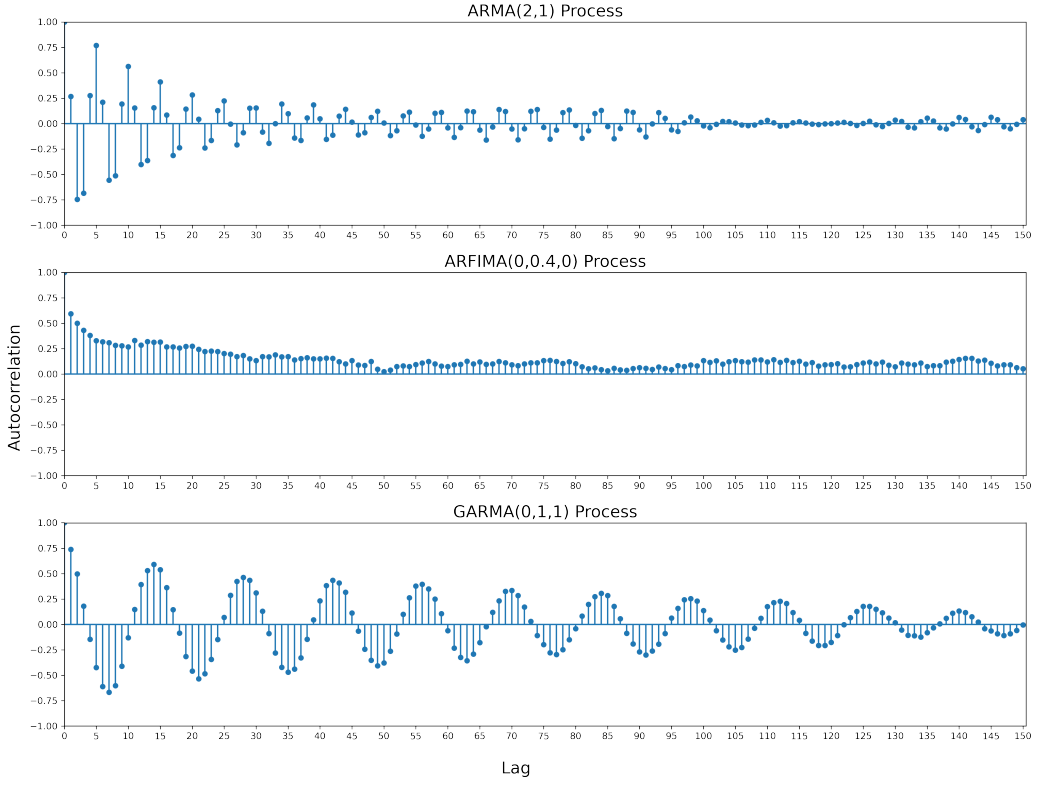


Figure 2.1: From top to bottom: autocorrelation functions up to lag 150 of an ARMA(2,1) process with complex AR and real MA roots, an ARFIMA(0,0.4,0) process, and a GARMA(0,1,1) process with $\lambda = 0.7$, $\delta = 0.4$. All three simulated time series have length 500. Note the damped sinusoidal pattern for the Gegenbauer process versus the irregular structure for the ARMA process and the positive and hyperbolically decaying ARFIMA autocorrelations.

The class of processes $\{X_t\}$ which exhibit this long-memory property and which can be represented in General Linear Process (GLP) form $X_t = \sum_{j=0}^{\infty} \psi_j \epsilon_{t-j}$, $\epsilon_t \overset{i.i.d.}{\sim} N(0, \sigma^2)$ can be obtained by introducing a fractional differencing term into a standard time series model, such as the ARMA model. The concept of fractional differencing was originally introduced by Hosking [1981] as an extension to the ARIMA process, by allowing the power d in the differencing operator $(1-B)^d$ to take any real value $d = \delta$ rather than being limited to integer values. Since the backward-shift operator cannot be formally raised to a non-integer power, the fractional differencing operator is defined in term of a generalised Binomial expansion:

$$(1-B)^{\delta} = \sum_{k=0}^{\infty} \binom{\delta}{k} (-B)^k = 1 - \delta B - \frac{1}{2}\delta(1-\delta)B^2 - \frac{1}{6}\delta(1-\delta)(2-\delta)B^3 - \dots$$

In particular, Hosking [1981] shows that values $\delta \in (0, 1/2)$ give rise to long-memory or persistent processes, while values $\delta \in (-1/2, 0)$ model short-memory or anti-persistent processes (values $\delta > 1/2$ have the additional effect of introducing trend). Hence, the former case is characterised by an autocorrelation function that is positive and decreases monotonically to zero, whereas the latter case has negative autocorrelations at all lags (except 0) which increase monotonically to zero. However, the crucial aspect in both cases is that the autocorrelations decay to zero in absolute value hyperbolically instead of exponentially, as is the case in the ARIMA model with integer differencing d. It is precisely this slower mode of decay that gives rise to the persistence phenomenon.

In the present work, we model seasonal persistence using a modification of the ARFIMA process, known as the Gegenbauer ARMA (GARMA) process due to its connection to Gegenbauer polynomials. Like the ARFIMA model, the GARMA model has the long-memory property, however it also incorporates a seasonal pattern in the autocorrelation sequence (see Figure 2.1). This is done by replacing the differencing operator $(1-B)^{\delta}$ in the ARFIMA model with an operator of the form $(1 - 2\lambda B + B^2)^{\delta}$ for $\lambda \in (-1, 1)$. Since the focus of this article is on persistent processes, we limit our attention to the long-memory case $\delta \in (0, 1/2)$. In the next section we introduce the (one-factor) Gegenbauer process (or GARMA(0,1,0)), which we later expand by adding k such factors to an ARMA process, leading to the general k-factor GARMA model.

2.4 The Gegenbauer Process

Following again the description in Dissanayake et al. [2018], the (one-factor) Gegenbauer process is a long-memory process defined through the equation:

$$(1 - 2\lambda B + B^2)^{\delta} X_t = \epsilon_t,$$

where $\lambda \in (-1, 1)$, $\delta \in (0, 1/2)$, and $\epsilon_t \overset{i.i.d.}{\sim} N(0, \sigma_{\epsilon}^2)$ is a white noise process. The definition above is equivalent to filtering the white noise process through the infinite non-linear filter $(1 - 2\lambda B + B^2)^{-\delta}$. This filter is characterised by the Gegenbauer or ultraspherical polynomials C_j which appear in the polynomial expansion of this filter:

$$(1 - 2\lambda z + z^2)^{-\delta} = \sum_{j=0}^{\infty} C_j^{\delta}(\lambda) z^j, \tag{2.4.1}$$

with

$$C_j^{\delta}(\lambda) = \sum_{k=0}^{\lfloor j/2 \rfloor} (-1)^k \frac{\Gamma(j-k+\delta)}{\Gamma(\delta)\Gamma(j+1)\Gamma(j-2k+1)} (2\lambda)^{j-2k}, \tag{2.4.2}$$

where $\lfloor j/2 \rfloor$ denotes the integer part of $j/2$. The bounds on λ and δ ensure that the complex roots of the polynomial $1 - 2\lambda z + z^2$ lie on the unit circle and that the $C_j^{\delta}(\lambda)$ are square summable, i.e. $\sum_{j=0}^{\infty} (C_j^{\delta}(\lambda))^2 < \infty$, so that the stationarity condition in Gray et al. [1989] is satisfied. As pointed out in the recent paper by Hunt et al. [2022] this operator can be viewed as a generalisation of the simple seasonality term $(1 - B^s)$.

The spectral density of X_t is given by:

$$\begin{aligned} S_X(f) &= \frac{\sigma_\epsilon^2}{\pi}\left\{2|\cos(2\pi f) - \cos(2\pi\tilde{\omega}_0)|\right\}^{-2\delta} \\ &= \frac{\sigma_\epsilon^2}{\pi}\left|4\sin\left(2\pi\frac{f+\tilde{\omega}_0}{2}\right)\sin\left(2\pi\frac{f-\tilde{\omega}_0}{2}\right)\right|^{-2\delta} \\ &\to \frac{\sigma_\epsilon^2}{\pi}\left\{4\pi(f-\tilde{\omega}_0)\sin(2\pi\tilde{\omega}_0)\right\}^{-2\delta} \text{ as } f \to \tilde{\omega}_0. \end{aligned} \tag{2.4.3}$$

Due to the boundedness of the sin function, the stationary Gegenbauer process has one unbounded spectral peak corresponding to the pole located at $\tilde{\omega}_0$, which is called the G-frequency. It can be shown that the process has long memory at this frequency provided that $\delta \in (0, 1/2)$, which here we take as an assumption in the definition. Although here we focus on the frequency-domain analysis, we note that the autocovariance function can be reconstructed through an inverse Fourier Transform of this spectrum – we refer the interested reader to Gray et al. [1989] for the specific form of this.

2.5 The k-Factor GARMA Model

Combining the ARMA and the Gegenbauer process, the GARMA process is built by filtering the Gegenbauer process by the AR polynomial and the white noise by the MA polynomial. This leads to the following characterisation of the (one-factor) GARMA process $\{Y_t\}$:

$$\Phi(B)(1 - 2\lambda B + B^2)^\delta Y_t = \Theta(B)\epsilon_t,$$

where $\Phi(B)$ and $\Theta(B)$ are the AR and MA operators.

Since $\{Y_t\}$ is assumed to be second-order stationary, the conditions of the Wold decomposition theorem require the AR component to be stationary and the MA component to be invertible, namely for both $\Phi(z)$ and $\Theta(z)$ to have all roots outside the unit cicle $|z| \leq 1$, and additionally for the two polynomials to have no common roots. As before, $\delta \in (0, 1/2)$ denotes the memory parameter and $\lambda \in (-1, 1)$.

Woodward et al. [1998] expanded this definition to incorporate k Gegenbauer factors, leading to the general k-factor GARMA process, which here we denote by GARMA(p, k, q). This notation is slightly different from that of other authors, who write the δ parameter instead of k, but we adopt this notation as it is consistent with that for the ARIMA model and because it makes the model order, rather than the value of the memory parameter, explicit. This process is represented as follows:

$$\Phi(B)\prod_{j=1}^{k}(1 - 2\lambda_j B + B^2)^{\delta_j} Y_t = \Theta(B)\epsilon_t \tag{2.5.1}$$

Building upon the SDF of the Gegenbauer process in equation (2.4.3) and that of the ARMA process in equation (2.2.1), Woodward et al. [1998] showed that the spectrum of the k-factor GARMA process $\{Y_t\}$ is:

$$S_Y(f) = \sigma_\epsilon^2 \frac{\left|\Theta(e^{-2\pi i f})\right|^2}{\left|\Phi(e^{-2\pi i f})\right|^2}\prod_{j=1}^{k}\{4[\cos(2\pi f_j) - \lambda_j)]^2\}^{-\delta_j}.$$

Therefore, the SDF of a k-factor GARMA process has k unbounded peaks at frequencies $f_j = \tilde{\omega}_{0j} = \frac{1}{2\pi}\cos^{-1}\lambda_j$, $j \in \{1, \ldots, k\}$ and the autocorrelations decay exponentially at rates δ_j. Using the same notation for the AR(p) and MA(q) polynomials in terms of the modulus-argument decomposition of their complex reciprocal roots $\xi_{rj} = \left(\xi_{rj}^R, \xi_{rj}^C, \overline{\xi_{rj}^C}\right)$, $r = 1, 2$, we rewrite the spectrum of the GARMA(p,k,q) process as:

$$\begin{aligned} S_Y(f) &= \sigma_\epsilon^2 \frac{\prod_{j=1}^{q}\left|1 - \xi_{2j}e^{2\pi i f}\right|^2}{\prod_{j=1}^{p}\left|1 - \xi_{1j}e^{2\pi i f}\right|^2}\frac{1}{\prod_{j=1}^{k}[4\{\cos(2\pi f) - \lambda_j\}^2]^{\delta_j}} \\ &= \sigma_\epsilon^2 \frac{\prod_{j=1}^{Rq}\left|1 - \alpha_{2j}^R e^{2\pi i f}\right|^2 \prod_{j=1}^{Cq}\left|(1 - \alpha_{2j}^C e^{2\pi i f - \omega_{2j}^C})(1 - \alpha_{2j}^C e^{2\pi i f + \omega_{2j}^C})\right|^2}{\prod_{j=1}^{Rp}\left|1 - \alpha_{1j}^R e^{2\pi i f}\right|^2 \prod_{j=1}^{Cp}\left|(1 - \alpha_{1j}^C e^{2\pi i f - \omega_{1j}^C})(1 - \alpha_{1j}^C e^{2\pi i f + \omega_{1j}^C})\right|^2}\frac{1}{\prod_{j=1}^{k}[4\{\cos(2\pi f) - \lambda_j\}^2]^{\delta_j}} \end{aligned} \tag{2.5.2}$$

Based on this expression for the spectrum, and since we assume that $\delta_j \in (0, 1/2)$, $j \in \{1, \ldots, k\}$, the process experiences long memory at each of the G-frequencies f_j. The k peaks induced by these G-frequencies are unbounded and can be clearly distinguished from the bounded AR peaks – see Figure 2.2 below.

For an accessible introduction to k-factor GARMA processes, see Woodward et al. [1998]. For a more extensive review of the GARMA process, see Dissanayake et al. [2018] and Wu and S. [2018]. For a formal description of fractionally differenced processes, see Hosking [1981] and Gray et al. [1989].

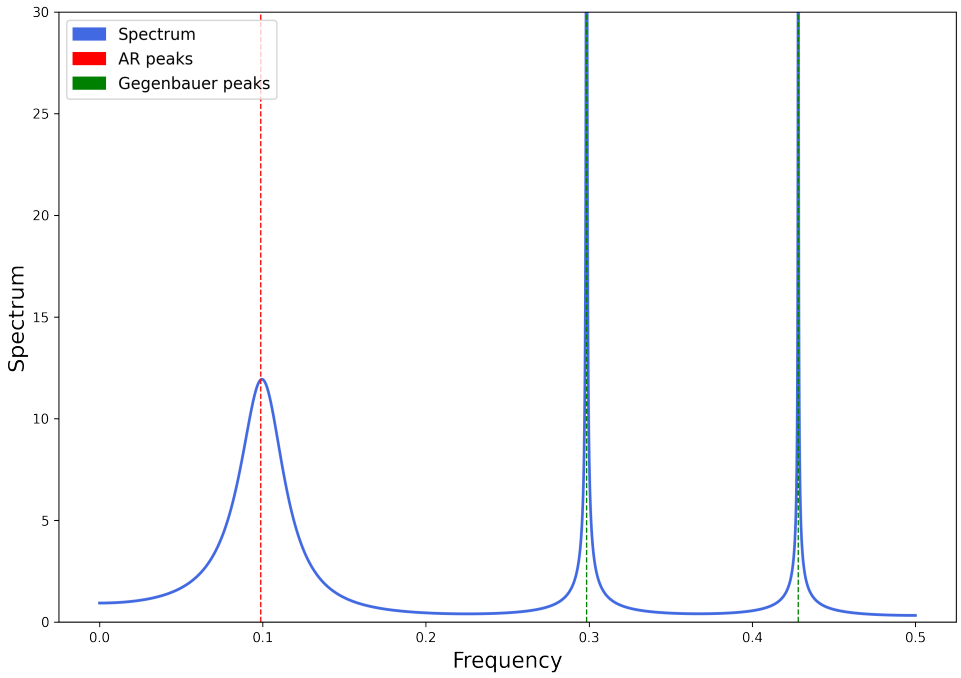


Figure 2.2: Comparison of AR and Gegenbauer peaks for a GARMA(3,2,2) process with two complex conjugate AR roots, two Gegenbauer factors, and two complex conjugate MA roots, plotted on a high-resolution frequency grid. Note the unbounded peaks at the two G-frequencies versus the smooth and bounded AR peak.

3 Frequency-Domain Bayesian Inference

3.1 Motivation

Having introduced in the previous section the time series analysis concepts necessary for this work, we now turn our attention to estimating the parameters of time series models. When investigating time series data, one can perform the analysis either in the time domain or in the frequency domain. The main tool of univariate time-domain analysis is the autocovariance function, which reveals the lags at which a time series is most correlated with itself. However, if our goal is to uncover seasonal patterns in the time series, it is often more revealing to look at the frequency-domain counterpart of the covariance sequence, namely the spectral density, which measures the power over a range of frequencies. High values of the spectrum, and in particular well separated peaks, correspond to the frequencies at which the time series exhibits periodicity. Consider, for instance, the simulated AR(4) process in Figure 3.1 below with two pairs of complex-conjugate roots. While the sample autocorrelation sequence (normalised version of the autocovariance sequence) only displays a subtle dampened oscillatory pattern, the periodogram, the simplest sample estimate of the spectrum, reveals two distinct peaks at frequencies 0.1 and 0.3, which were indeed used to generate the process.

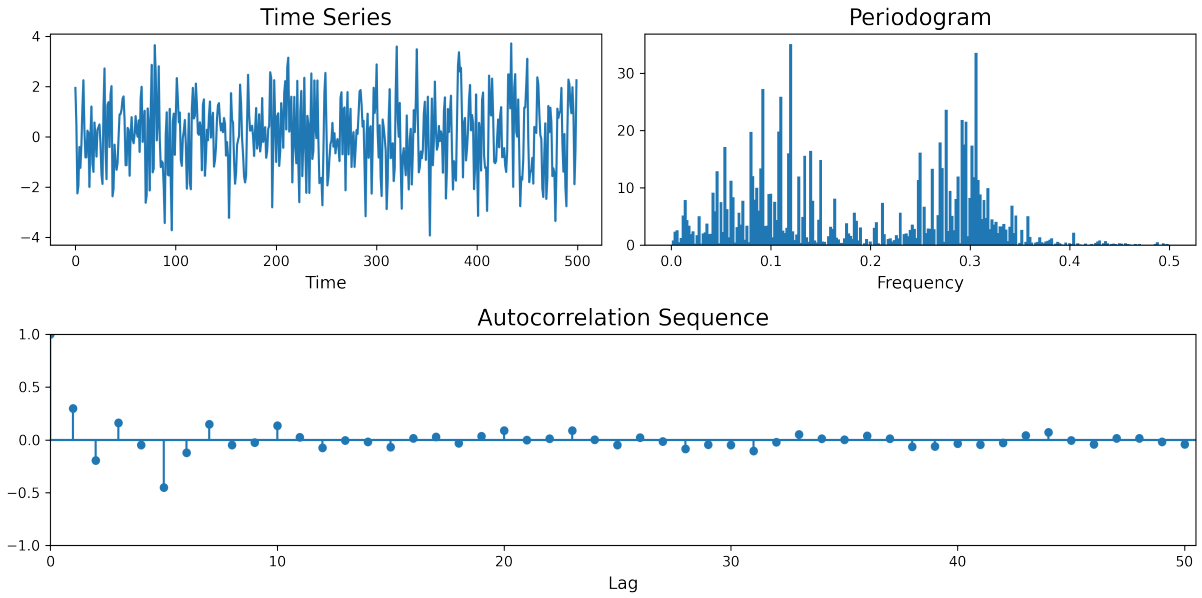


Figure 3.1: AR(4) process with two pairs of complex-conjugate roots

Having outlined the main advantage of frequency-domain analysis, we now justify the appeal of a Bayesian inference approach. Recall that in Bayesian inference, we infer the properties of a parameter $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^n$ by determining its posterior distribution based on a prior distribution, which incorporates prior knowledge about $\boldsymbol{\theta}$, and a likelihood function, which links $\boldsymbol{\theta}$ with the available data y. Lower dimensional components of $\boldsymbol{\theta}$ can be subsequently estimated by marginalising this (joint) posterior. Point estimates can then be obtained by taking an appropriate summary statistic, such as the mean or the median, and estimates of uncertainty can be obtained through so-called credible regions around the point estimate. In order to determine the posterior of $\boldsymbol{\theta}$, one first needs to specify a prior on $\boldsymbol{\theta}$. This prior distribution is usually chosen to be uninformative and it is often convenient to specify it through independent priors on each univariate component of $\boldsymbol{\theta}$. Having specified the prior, one needs to formulate a likelihood for $\boldsymbol{\theta}$ based on the data y, which is often the core of the modelling effort. Finally, one uses Bayes's formula to calculate the posterior based on the prior and the likelihood:

$$p(\boldsymbol{\theta}|y) = \frac{L(\boldsymbol{\theta}|y)p_0(\boldsymbol{\theta})}{\int_\Theta L(\boldsymbol{\theta}|y)p_0(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

where $p(\boldsymbol{\theta}|y)$ is the posterior, $L(\boldsymbol{\theta}|y)$ is the likelihood and $p_0(\boldsymbol{\theta})$ is the prior. The normalising factor in the denominator is a multivariate integral which is often intractable. Depending on the sampling scheme used to simulate from $p(\boldsymbol{\theta}|y)$, calculating this integral can be be circumvented, as is indeed the case in SMC.

To conclude this discussion, we highlight one important advantage of Bayesian inference. Besides its inherent simplicity and fundamental link to Monte Carlo sampling techniques (see Section 4), one well-known advantage of Bayesian inference is its natural measure of uncertainty of estimates. This measure is given by the so-called credible regions for estimates $\hat{\boldsymbol{\theta}}$, which are simply regions in which the parameter falls with a specified probability. This is a more intuitive interpretation than the frequentist analogue of confidence intervals. Used together with the frequency-domain approach, Bayesian inference leads to a versatile tool for time series analysis.

3.2 Model Specification and Parametrisation

Following our discussion of time series models in Section 2, we will employ a GARMA(p, k, q) model due to its flexibility in modelling general time series exhibiting both short seasonality and long-term seasonal persistence. We reparametrise the AR and MA polynomials in terms of their latent structure, that is in terms of the modulus and the argument of their respective reciprocal roots, as in equations (2.2.2)-(2.2.6). This approach is similar to that in Huerta and West [1999b] and McCoy and Stephens [2004]. With the resulting SDF of the GARMA(p, k, q) process given as in (2.5.2) and this latent-structure parametrisation, we define the vector $\boldsymbol{\eta}$ of $m = R_p + 2C_p + R_q + 2C_q + 2k + 1$ model parameters:

$$\begin{aligned}\boldsymbol{\eta} = (&\alpha^R_{11}, \ldots, \alpha^R_{1R_p}, \alpha^C_{11}, \ldots, \alpha^C_{1C_p}, \omega^C_{11}, \ldots, \omega^C_{1C_p},\\ &\alpha^R_{21}, \ldots, \alpha^R_{2R_q}, \alpha^C_{21}, \ldots, \alpha^C_{2C_q}, \omega^C_{21}, \ldots, \omega^C_{2C_q},\\ &\delta_1, \ldots, \delta_k, \lambda_1, \ldots, \lambda_k, \sigma^2_\epsilon)\end{aligned} \tag{3.2.1}$$

Estimating $\boldsymbol{\eta}$ by sampling from its posterior distribution is the goal of this analysis, and as such $\boldsymbol{\eta}$ is the central object in the SMC algorithm in the next section. Note that the above parametrisation has $R_p + R_q + C_p + C_q + 4$ fewer parameters than the parametrisation in McCoy and Stephens [2004]. Firstly, the complex arguments $\omega^R_{11}, \ldots, \omega^R_{1R_p}$ of the R_p real AR roots are all zero, as are those for the R_q real MA roots, i.e. $\omega^R_{21} = 0, \ldots, \omega^R_{2R_q} = 0$. Therefore, these $R_p + R_q$ parameters do not need to be estimated. Secondly, the $C_p + C_q$ moduli and arguments corresponding to the complex conjugate AR and MA roots are condensed into the same parameter, as one only needs to change the sign in the exponent when reconstructing the reciprocal roots. Moreover, since we demean the data as part of the initial pre-processing, unlike in McCoy and Stephens [2004], we do not estimate the mean. Finally, as we adopt a fixed-parameter SMC procedure, we also do not estimate the model parameters p, q, k, as these are specified by the user. For a discussion on how these model order parameters could be estimated, see the discussion on transdimensional SMC in Del Moral et al. [2006] or the Reversible Jump MCMC procedure in McCoy and Stephens [2004], which are briefly discussed in Subsection 7.4.

3.3 Prior Specification

With the parametrisation in (3.2.1), we now proceed to define a sensible prior distribution on $\boldsymbol{\eta}$. For simplicity, we do this componentwise, by specifying independent priors on each component of $\boldsymbol{\eta}$. We use similar uninformative priors to those specified in McCoy and Stephens [2004] and Huerta and West [1999a], namely:

- $\alpha^R_{11}, \ldots, \alpha^R_{1R_p}$, $\alpha^C_{11}, \ldots, \alpha^C_{1C_p}$, $\alpha^R_{21}, \ldots, \alpha^C_{2R_q}$, $\alpha^C_{21}, \ldots, \alpha^C_{2C_q} \overset{i.i.d.}{\sim} Uniform(0, 1)$.
- $\omega^C_{11}, \ldots, \omega^C_{1C_p}$, $\omega^C_{21}, \ldots, \omega^C_{2C_q} \overset{i.i.d.}{\sim} Uniform(0, 1/2)$.
- $\delta_1, \ldots, \delta_k \overset{i.i.d.}{\sim} Uniform(0, 1/2)$.
- $\lambda_1, \ldots, \lambda_k \overset{i.i.d.}{\sim} Uniform(-1, 1)$.
- $\sigma^2_\epsilon \sim InvGamma(1, 1)$, where $InvGamma(1, 1)$ denotes an Inverse Gamma distribution (distribution of the reciprocal of a Gamma random variable) with shape $\alpha = 1$ and scale $\beta = 1$ [1]. These values correspond to a diffuse prior, i.e. one which is not too concentrated around specific values, and which can thus be regarded as uninformative. Additionally, the Inverse Gamma distribution has heavy tails, allowing even for relatively large values to occur with non-negligible probability. The Inverse Gamma distribution is often used in practice due to its convenient property that it is a conjugate prior distribution for the variance of a Gaussian process when this variance is assumed unknown.

[1] The probability density of an Inverse Gamma distribution with shape parameter α and scale parameter β is: $f(x; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}(1/x)^{\alpha+1} \exp(-\beta/x), \; x > 0$.

There are some differences, however between these priors and those in McCoy and Stephens [2004] and Huerta and West [1999a]. In particular, McCoy and Stephens [2004] specify a $Uniform(0, 1/4)$ prior on each δ_j when the corresponding $|\lambda_j| = 1$, and both Huerta and West [1999a] and Huerta and West [1999b] place Dirichlet point masses at the ends of the intervals for the α's and replace the Uniform prior with a Beta prior on these parameters. However, as McCoy and Stephens [2004] notes and our empirical experimentations confirm, the posterior is quite robust to the choice of the prior, and as such we focus on the priors being diffuse and uninformative rather than on their precise shapes. Moreover, the different priors specified for the edge values are mainly relevant formally, since in practice numerical samplers will only ever return boundary values with infinitesimal probability. More details on the distributions above can be found in Appendix A.2.

3.4 Choice of Sample Spectral Estimator

The core idea of our frequency-domain estimation approach is to measure how well the theoretical SDF of a model specified through $\boldsymbol{\eta}$ fits a sample spectral estimate based on the data. In the next subsection we detail a way to quantify this, but first we discuss the available choices for the sample estimator of the spectrum. The simplest such estimator is the periodogram. Let $\{X_t\}_{t=1,\dots,N}$ be a discrete [2] time series of length N with spectrum $S_X(f)$, $f \in [-1/2, 1/2]$. Then one defines the periodogram as the squared norm of the discrete Fourier transform of the data:

$$\hat{S}_p(f) = \frac{1}{N}\left|\sum_{t=1}^{N} X_t e^{-2\pi i f t}\right|^2, \; f \in [-1/2, 1/2]$$

The same Fourier pair relationship that holds between the autocovariance and the spectrum of the time series also holds between the periodogram and the biased autocovariance estimator $\hat{s}_p(\tau)$:

$$\hat{S}_p(f) = \sum_{\tau=-(N-1)}^{(N-1)} \hat{s}_p(\tau) e^{-2\pi i f \tau} \quad \text{and} \quad \hat{s}_p(\tau) = \int_{-1/2}^{1/2} \hat{S}_p(f) e^{2\pi i f \tau} df, \; |\tau| <= N-1$$

where

$$\hat{s}_p(\tau) = \frac{1}{N}\sum_{t=1}^{N-|\tau|} (X_t - \bar{X})(X_{t+|\tau|} - \bar{X}), \; \tau = 0, \pm 1, \dots, \pm(N-1).$$

and $\bar{X}$ is the sample mean. Hence, one can use the periodogram to also estimate the autocovariance sequence, and so again, no information is lost by operating in the frequency domain. While the periodogram has a simple expression, it is an inconsistent estimator of the spectrum, meaning that it does not converge in probability to the true value of the spectrum as N increases. Moreover, it is a well-known result that the periodogram is biased. While the bias does decrease asymptotically to zero, this bias can be significant, especially for shorter time series (see Olhede et al. [2004]). More specifically, it can be shown that:

$$\mathbb{E}\{\hat{S}_p(f)\} = \int_{-1/2}^{1/2} \mathcal{F}(f-\nu) S_X(\nu) d\nu$$

where

$$\mathcal{F}(f) = \left|\sum_{t=1}^{N} \frac{1}{\sqrt{N}} e^{-2\pi i f t}\right|^2 = \frac{\sin^2(N\pi f)}{N \sin^2(\pi f)}$$

denotes Féjer's kernel (for an illustration, see Figure A.2 in Appendix A.3). Thus, the expected value of the periodogram equals the true spectrum convolved with Féjer's kernel. This kernel induces blurring[3], which

[2] This generalises to a discrete sample $X_t = X(t\Delta)$ obtained at a given sampling interval $\Delta > 0$ from a continuous process $X(t)$. In this case, the periodogram is $\hat{S}_p(f) = \frac{\Delta}{N}\left|\sum_{t=1}^{N} X_t e^{-2\pi i f t \Delta}\right|^2$, $f \in [-\frac{1}{2\Delta}, \frac{1}{2\Delta}]$. If the process $X(t)$ is discrete, $\Delta = 1$.

[3] Another type of spectral leakage is aliasing, which arises due to sampling continuous data – however this is not a problem here, as we are operating on discrete data (sampled at $\Delta = 1$).

is a form of spectral leakage, that is the leakage of power from frequencies with high values of the spectrum to other, not necessarily nearby, frequencies – a problem which can affect the entire frequency range. The sinusoidal side-lobe leakage due to Féjer's kernel is caused by the windowing or tapering of the time series using a rectangular window function covering the range $t = 1, \dots, N$ (see Appendix A.3). Finally, we note that the asymptotic unbiasedness of the periodogram is equivalent to Féjer's kernel acting as a Dirac delta function in the limit:

$$\lim_{N \to +\infty} \mathbb{E}\{\hat{S}_p(f)\} = \int_{-1/2}^{1/2} \delta(f - \nu) S_X(\nu) d\nu = S_X(f)$$

by the sifting property of the Dirac delta function. The theory surrounding errors due to spectral leakage is extensive, but the most common way to mitigate this issue is tapering using a window function with a narrower frequency response and defining a tapered (also called direct) spectral estimator as follows:

$$\hat{S}_d(f) = \left| \sum_{t=1}^{N} h_t X_t e^{-2\pi i f t} \right|^2, \ f \in [-1/2, 1/2]$$

where h_t is a window function, which satisfies several properties, such as symmetry about the origin, symmetry of the DFT, and $\sum_{t=1}^{N} h_t^2 = 1$. Clearly, one can see that the periodogram corresponds to the particular case with rectangular taper $h_t = 1/\sqrt{N}$, $t = 1, \dots, N$ and 0 otherwise. One common such taper is the Hann taper – its functional form and an illustration of its sidelobe reduction effect are provided in Appendix A.3. In the next section we also discuss how tapering can be used in conjunction with convolution to reduce the bias in the sample spectral estimate.

3.5 The Whittle Likelihood

Having discussed the issue of sample estimation of the spectrum, we now use this information to construct the likelihood function. Recall that the likelihood function represents the joint probability of the data given a specific value of the vector of model parameters, thus linking the observations with the theoretical model. Consider a zero-mean stationary time series $\boldsymbol{X} = (X_0, \dots, X_{N-1})$ of even length N sampled from the process $\{X_t\}$ and assume a Gaussian model for $\boldsymbol{X}$, specified through $\boldsymbol{\eta}$ via the covariance matrix $\Sigma_{\boldsymbol{\eta}}$. Then, the likelihood has the form of a multivariable normal pdf with mean 0 and covariance $\Sigma_{\boldsymbol{\eta}}$:

$$L(\boldsymbol{\eta}|y) = \frac{1}{(2\pi)^{N/2}} \frac{1}{|\det \Sigma_{\boldsymbol{\eta}}|^{1/2}} \exp\left\{ -\frac{1}{2} \boldsymbol{X}^T \Sigma_{\boldsymbol{\eta}}^{-1} \boldsymbol{X} \right\}$$

where $\boldsymbol{\eta}$ is the vector of parameters as in (3.2.1). Both in frequentist methods such as Maximum Likelihood Estimation (MLE) and in Bayesian Inference, it is often algebraically convenient to work with the loglikelihood $l(\boldsymbol{\eta}|y) = \log L(\boldsymbol{\eta}|y)$. Clearly, maximising the loglikelihood is equivalent to maximising the likelihood, due to the monotonicity of the logarithmic function. In particular, one calculates:

$$l(\boldsymbol{\eta}|y) = -\frac{1}{2} \left[N \log(2\pi) + \log|\det \Sigma_{\boldsymbol{\eta}}| + \boldsymbol{X}^T \Sigma_{\boldsymbol{\eta}}^{-1} \boldsymbol{X} \right].$$

Due to the $O(N^3)$ computational cost of inverting the $N \times N$ matrix $\Sigma_{\boldsymbol{\eta}}$, one may wish to seek an approximation to the loglikelihood which has lower computational complexity. In our case, we are further motivated by the desire to operate entirely in the frequency domain, without having to derive the covariance matrix $\Sigma_{\boldsymbol{\eta}}$ for a model which could potentially be quite complicated. Noting that $\Sigma_{\boldsymbol{\eta}}$ is a Toeplitz matrix (i.e. it has constant diagonals), the cost of inverting it can be reduced to $O(N^2)$ using the Levinson-Durbin recursion algorithm. However, by further noting that, as a covariance matrix, $\Sigma_{\boldsymbol{\eta}}$ is symmetric, it can be shown (see Hurvich [2002]) that for N sufficiently large, the (orthonormal) eigenvectors of $\Sigma_{\boldsymbol{\eta}}$ can be approximated by:

$$v_j = \frac{1}{\sqrt{N}} \exp(-2\pi i f_j t), \ j = 0, \dots, N-1$$

and the corresponding eigenvalues by $\lambda_j = 2\pi S_X(f_j)$, where, as before, $S_X(\cdot)$ is the SDF of $\boldsymbol{X}$, and $f_j = \frac{j}{N-1} - \frac{1}{2},\ j \in \{0, \ldots, N-1\}$ are the so-called Fourier frequencies (Hurvich [2002] uses $\omega_j = 2\pi f_j$ instead). Letting $V = (v_0, \ldots, v_{N-1})$ denote the (unitary) matrix of eigenvectors with v_j the j-th column, V^* its complex conjugate, and Λ the diagonal matrix of eigenvalues with $\Lambda_{jj} = \lambda_j$, one can replace $\Sigma_{\boldsymbol{\eta}}$ by its approximate eigendecomposition $V\Lambda V^*$ in the expression for $l(\boldsymbol{\eta}|y)$ to obtain:

$$l(\boldsymbol{\eta}|Z) \approx -\frac{N}{2}\log(2\pi) - \frac{1}{2N}\sum_{j=0}^{N-1}\left[\log\{S_X(f_j)\} + \frac{Z_j}{S_X(f_j)}\right]$$

where $Z = \left(Z_0 = \hat{S}_p(f_0) \ldots, Z_{N-1} = \hat{S}_p(f_{N-1})\right)$ is the vector of periodogram values evaluated at the Fourier frequencies f_j. Restricting the frequency range to $[0, 1/2]$ and using the symmetry of both the periodogram and the spectrum over the interval $[-1/2, 1/2]$, one defines the approximation:

$$l(\boldsymbol{\eta}|Z) \approx -\frac{N}{2}\log(2\pi) + l^{DW}(\boldsymbol{\eta}|Z)$$

where

$$l_{DW}(\boldsymbol{\eta}|Z) = -\frac{1}{N}\sum_{j=0}^{\lfloor N-1\rfloor/2}\left[\log\{S_X(f_j)\} + \frac{Z_j}{S_X(f_j)}\right] \tag{3.5.1}$$

is the discrete Whittle loglikelihood and $f_j = \frac{j}{N},\ j \in \{0, \ldots, N/2\}$ are the Fourier frequencies over the interval $[0, 1/2]$. We note (see McCoy et al. [2007]) that the expression in (3.5.1) is the Riemann sum discretisation at the Fourier frequencies f_j of the continuous Whittle loglikelihood:

$$l_W(\boldsymbol{\eta}) = -\int_{-1/2}^{1/2}\left[\log\{S_X(f)\} + \frac{\hat{S}_p(f)}{S_X(f)}\right]df$$

This frequency-domain approximation to the loglikelihood is due to Peter Whittle, who introduced it in his 1951 doctoral thesis, and further developed it in Whittle [1953]. Essentially, it can be seen as a measure of spectral divergence between the true spectrum and the sample spectrum estimate (see Rao and Yang [2021]), so that maximising the Whittle likelihood corresponds to minimising this divergence. Besides the already mentioned benefit of circumventing the calculation of the covariance matrix when the modelling is performed in the frequency domain, this approximation has the additional benefit of having the same $O(n \log n)$ computational complexity as the periodogram, provided the latter is calculated using the Fast Fourier Transform (FFT) algorithm.

While the Whittle likelihood clearly presents some advantages over the exact likelihood, it is only as good an approximation of the likelihood as the sample spectral estimator used to compute it is of the true spectrum. Consequently, the Whittle likelihood is prone to errors due to spectral leakage in the sample estimator. As discussed in the previous subsection, tapering the periodogram is one way to address this problem. However, in a recent paper, Sykulski et al. [2019] show that tapering only results in a moderate reduction in the bias, and suggest a complementary approach, which consists in replacing the true spectrum by the expected value of the periodogram, which, as we saw in the previous subsection, is the true spectrum (under the assumed model) convolved with Féjer's kernel. Recalling the fact introduced earlier that $\lim_{N\to+\infty}\mathbb{E}\{\hat{S}_p(f)\} = S_X(f)$, we can see that replacing $S_X(f)$ by $\mathbb{E}\{\hat{S}_p(f)\}$ is likely to result in a greater reduction of the spectral divergence. In their study on the Matérn process, Sykulski et al. [2019] showed that this modification results in a bias reduction by nearly an order of magnitude. This modification, called the Debiased Whittle likelihood is given by:

$$l^D_{DW}(\boldsymbol{\eta}) = -\frac{1}{N}\sum_{j=0}^{\lfloor N-1\rfloor/2}\left[\log\{\mathbb{E}\{\hat{S}_p(f_j)\}\} + \frac{Z_j}{\mathbb{E}\{\hat{S}_p(f_j)\}}\right]$$

This can be naturally further enhanced through the inclusion of tapering by replacing the periodogram values Z with the tapered estimator values $Z^T = \left(Z_0^T = \hat{S}_d(f_0) \ldots, Z_{N-1}^T = \hat{S}_d(f_{N-1})\right)$ and $\mathbb{E}\{\hat{S}_p(f)\}$ with:

$$\mathbb{E}\{\hat{S}_d(f)\} = \int_{-1/2}^{1/2} \mathcal{H}(f-\nu) S_X(\nu) d\nu$$

where $\mathcal{H}(f) = \left| \sum_{t=1}^{N} h_t e^{-2\pi i f t} \right|^2$, $f \in [-1/2, 1/2]$ is the DFT of the taper h_t.

Finally, Sykulski et al. [2019] also discuss the addition of differencing as a bias reduction technique due to its effect of reducing the dynamic range of the time series and hence the blurring of the spectrum. However this is less relevant to our discussion, since we assume that the data is already stationary. Another way to reduce the bias is via a technique called demodulation (see Olhede et al. [2004] and McCoy et al. [2007]), which we briefly outline in Section 7.3 as a potential further development. Finally, for a discussion of the bias, variance, and asymptotic properties of the Whittle likelihood, we point the interested reader to Rao and Yang [2021], who provide a comprehensive review of these theoretical aspects.

4 Sequential Monte Carlo (SMC)

4.1 Motivation – Sampling from a Distribution

So far we have developed the necessary tools for the frequency-domain Bayesian analysis of long-memory processes using a GARMA model specified through the vector of model parameters $\boldsymbol{\eta}$ from (3.2.1). We now turn our attention to the problem of estimating $\boldsymbol{\eta}$ by sampling from its posterior distribution. For this, we introduce the general framework of SMC sampling.

Consider the problem of sampling from a target distribution with density π. In a Bayesian inference context, the target would be the (joint) posterior distribution $\pi(\boldsymbol{\theta}|y)$ of a parameter $\boldsymbol{\theta}$ based on the data y. In particular, in our situation of a static time series estimation setup considered in the frequency domain, the target is the posterior $\pi(\boldsymbol{\eta}|Z) \propto L(\boldsymbol{\eta}|Z)\pi(\boldsymbol{\eta})$, where Z is the periodogram (or another sample spectral estimate) of the full time series y, $L(\boldsymbol{\eta}|Z) = \exp\{l(\boldsymbol{\eta}|Z)\}$ is the Whittle likelihood from Subsection 3.5, and $\pi(\boldsymbol{\eta})$ is the prior of $\boldsymbol{\eta}$ (note that in what follows, we will continue to use the notation $\pi(\boldsymbol{\eta}|y)$ to highlight the dependency on the data). In theory, this setup could be extended to the situation where the data arrives sequentially (see Chopin [2002]), by considering $Z_{1:t}$, the periodogram calculated based on the values of the time series $y_{1:t}$ up to time t. However, in practice, updating the periodogram (or any other sample spectral estimate) would require a full run through the data, providing no computational advantage to such an approach.

When π is a simple distribution, methods such as Rejection Sampling, Ratio-of-Uniforms, or even Inversion in some very simple cases, are sufficient. However, for general distributions, particularly multivariate and multimodal distributions, one requires more general Monte Carlo Methods. In this section we introduce SMC from first principles, also discussing and contrasting it with pure MCMC.

The key idea of methods such as Importance Sampling, and Sequential Monte Carlo more broadly, is to construct a set of independent samples (also called particles) from simpler auxiliary densities, and then to weigh these samples by a set of importance weights. We will see that constructing these weighted samples poses several technical challenges, and we will explore ways to address these challenges, ultimately reaching the SMC algorithm in Subsection 4.8.

4.2 Importance Sampling (IS)

Importance sampling is a Monte Carlo integration method which extends Perfect (Crude) Monte Carlo integration by sampling from an auxiliary (also called sampling) distribution. Recall that in crude Monte Carlo, we estimate the integral:

$$I(f) = E_\pi[f(X)] = \int f(x)\pi(x)dx$$

by

$$\hat{I}(f) = \frac{1}{N}\sum_{i=1}^{N} f(X_i)$$

where $X_1, ..., X_N$ are N samples drawn from the distribution π with support $\mathcal{X}$. In particular, if $f(X) = \mathbb{I}_A(X)$ is the indicator function for some set A, then $I(f) = P(X \in A)$, leading back to the problem of simulating from π. However, if simulating from π is precisely the goal of our problem, we are naturally operating under the assumption that obtaining samples from π is not easy. Importance sampling attempts to address this by considering an auxiliary distribution μ of the same dimensionality as π and whose support includes that of π, but which is easier to sample from. This leads to the integral being rewritten as:

$$I(f) = E_\pi[f(X)] = \int f(x)\pi(x)dx = \int f(x)\frac{\pi(x)}{\mu(x)}\mu(x)dx =: \int f(x)W(x)\mu(x)dx = E_\mu[W(X)f(X)]$$

where $W(x) \coloneqq \frac{\pi(x)}{\mu(x)}$ is called the importance function. The Monte Carlo estimate of this integral then becomes:

$$\hat{I}_{IS}(f) = \frac{1}{N}\sum_{i=1}^{N} W(X_i)f(X_i)$$

where this time, $X_1, \ldots, X_N$ are N samples drawn from the auxiliary distribution μ. The terms $\tilde{w}^i = W(X_i)$, $i = 1, \ldots, N$ are called the (unnormalised) importance weights. As in the case of the Crude Monte Carlo estimate, it is easy to check that this estimate is unbiased for $I(f)$, provided that π and μ are normalised. When π and μ are known only up to proportionality, this issue can be addressed by normalising the weights so that they sum to 1. The estimate of the integral can be rewritten in terms of the normalised weights $w^i = \frac{\tilde{w}^i}{\sum_{j=1}^{N}\tilde{w}^j}$ as:

$$\hat{I}_{IS}(f) = \sum_{i=1}^{N} w^i f(X_i).$$

While theoretically attractive, this approach presents some challenges. The main one is that the approximation will only be efficient if μ has a similar shape as π (more precisely, if they are close based on a measure of statistical distance, such as cross-entropy or Kullback-Leibler divergence). However, finding such a distribution μ in practice is non-trivial, particularly if π is high-dimensional or has a complicated functional form. Moreover, as pointed out in the introductory chapter of Doucet et al. [2001], IS is inadequate in a recursive setup, since one needs to recompute the weights for the entire data as soon as a new datum is introduced. The Sequential Importance Sampling (SIS) method in the next subsection seeks to address these issues.

4.3 Sequential Importance Sampling (SIS)

The core idea of Sequential Importance Sampling is to sample from a sequence of distributions, where each term in the sequence is sampled from using importance sampling with the previous term as the auxiliary distribution. The first distribution would ideally be easy to sample from, and we would iteratively build our way up to the ultimate target distribution. More precisely, consider a sequence of T distributions $\pi_0, \dots, \pi_T = \pi$, where π is the target distribution and π_0 is a distribution that is easy to sample from – this would be the prior in a Bayesian context. Then, we can apply importance sampling sequentially as follows:

$$\begin{aligned}\pi(x) = \pi_T(x) &= \frac{\pi_T(x)}{\pi_{T-1}(x)}\frac{\pi_{T-1}(x)}{\pi_{T-2}(x)}\dots\frac{\pi_1(x)}{\pi_0(x)}\pi_0(x) \\ &= W_T(x)W_{T-1}(x)\dots W_1(x)\pi_0(x).\end{aligned}$$

Again, in a Monte Carlo estimation framework, we can define the sequence of (unnormalised) importance weights $\tilde{w}_1^i, \dots, \tilde{w}_T^i$ for each sample $i = 1, \dots, N$. This allows us to compute recursively the weights:

$$\tilde{w}_t^i \propto \tilde{w}_{t-1}^i \frac{\pi_t(X_i)}{\pi_{t-1}(X_i)},\ t = 1, \dots, T$$

In particular, in a Bayesian context, where we take $\pi_T = \pi(\boldsymbol{\eta}|y)$ as the posterior (see Chopin [2002]) and $\pi_0 = \pi(\boldsymbol{\eta})$ as the prior, these weights become:

$$\tilde{w}_t^i \propto \tilde{w}_{t-1}^i \frac{\pi_t(\boldsymbol{\eta}_{t-1}^i|y)}{\pi_{t-1}(\boldsymbol{\eta}_{t-1}^i|y)},\ t = 1, \dots, T \tag{4.3.1}$$

Unfortunately, as pointed out in Doucet et al. [2001], the SIS method gives rise to the problem of weight degeneracy, which means that the importance weights $\{\tilde{w}_t^i\}$ become increasingly skewed as t increases, until eventually very few particles have non-zero importance weights. This phenomenon is problematic because progressively fewer particles contribute to the sampling, which is likely to cause the sampling scheme to fail to adequately identify the shape of the target distribution. The next subsection discusses how to address this issue.

4.4 Resampling and The Bootstrap Filter

The Bootstrap filter, as introduced originally in Gordon et al. [1993], attempts to address the problem of weight degeneracy by filtering out the particles with low importance weights. In order to preserve the number of particles, this is done in conjunction with replicating the particles with high importance weights, and thus high information content. This in turn is done by resampling with replacement the N particles according to their (normalised) importance weights, so that particles with high importance weights are split into several particles, while particles with low weights are discarded. A good illustration of this is given in Figure 4.1 below, which is extracted from the introductory chapter of Doucet et al. [2001]. More precisely, considering our previously introduced Bayesian setup, assume that at iteration t of the SIS algorithm we have sampled N particles $\{\boldsymbol{\eta}_t^i\}_{i=1,\dots,N}$ from $\pi_t(\boldsymbol{\eta}_{t-1}^i|y)$, with corresponding unnormalised weights $\{\tilde{w}_t^i\}_{i=1,\dots,N}$. A natural method for resampling these particles according to their importance weights, which is introduced in Gordon et al. [1993] and further described in Chopin [2002], is multinomial selection, where the resampled particles $\{\dot{\boldsymbol{\eta}}_t^i\}_{i=1,\dots,N}$ come from the discrete distribution:

$$P(\dot{\boldsymbol{\eta}}_t^i = \boldsymbol{\eta}_t^i) = w_t^i = \frac{\tilde{w}_t^i}{\sum_{j=1}^N \tilde{w}_t^j}$$

This sample is then relabelled as the original sample $\{\boldsymbol{\eta}_t^i\}_{i=1,\dots,N}$ and the importance weights are reset to $w_t^i = 1/N$, $i = 1, \dots, N$. After this resampling step, the SIS procedure is resumed, continuing to iteration $t+1$. This method is also known as Sequential Importance Resampling (SIR).

Having described the concept of resampling, it is important to quantify the degeneracy in the importance weights in order to have a condition for when one should resample the particles. The standard measure for this is the so-called Effective Sample Size (ESS). For a set of normalised weights $\{w_t^i\}_{i=1,\dots,N}$ at iteration n of SIS, we define the ESS as:

$$\text{ESS} = \frac{1}{\sum_{j=1}^{N}(w_n^j)^2} \tag{4.4.1}$$

The ESS ranges from 0 to the number of particles N, and a commonly used threshold for resampling is when $\text{ESS} < N/2$, although one can choose any threshold $\text{ESS} < \alpha N$ with $\alpha \in (0, 1]$. Intuitively, the ESS describes the degree of correlation in the sample, so that a lower ESS corresponds to particles that have higher correlation, and hence higher intra-class variance (an ESS of 0 reflects perfect correlation, a case in which effectively all particles are identical). Since a key element of SIS is to have (roughly) independent samples, the ESS falling below a certain threshold should prompt the resampling of the particles. Following independent resampling, the ESS reaches the maximum value of N, before starting to decrease again at the next iteration (see Figure 6.2).

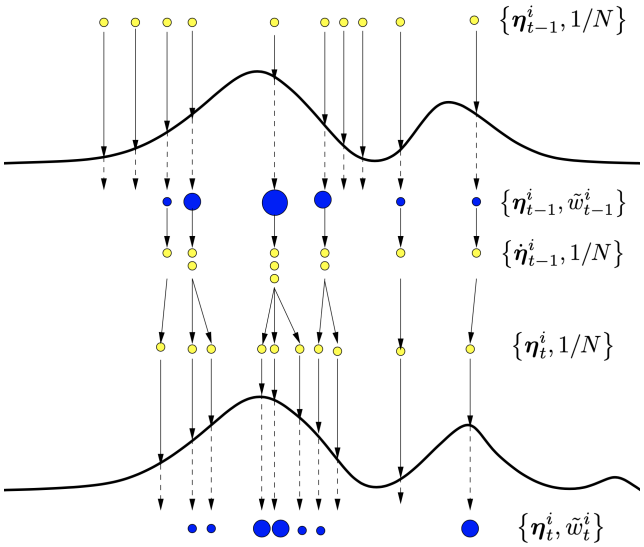


Figure 4.1: Illustration of the resampling step (Figure 1.1 from Doucet et al. [2001])

4.5 Constructing a Sequence of Distributions

Having established the components of the SIR procedure, we now briefly discuss several ways to build a sequence of distributions $\{\pi_t\}_{t=1,\dots,T}$. The technique of building such a sequence is also known as tempering. Some popular construction methods include:

1. Constant sequence. The simplest method would be to set $\pi_t = \pi$, $\forall t = 1, \dots, T$. This choice essentially reduces the IS procedure back to Perfect MC, and is thus likely to be inefficient when used in an SMC algorithm, particularly when the target π is high-dimensional. Moreover, as pointed out in Del Moral et al. [2006], this choice is unlikely to produce accurate results when π has multiple well-separated narrow modes.

2. Data tempering. This method was originally introduced in Chopin [2002] and further developed in Chopin and Papaspiliopoulos [2020]. Here, the idea is to keep the same distribution π throughout, but to introduce the data sequentially. Considering the Bayesian setup of sampling from the posterior, we define $\pi_t(\theta) = \pi(\theta|y_{1:t})$, $t = 0, \dots, T$ based on the first t data points $y_{1:t} = (y_1, \dots, y_t)$. Again, when $t = 0$, and $t = T$ this reduces to sampling from the prior and the full posterior, respectively.

While this approach could be adapted to static estimation problems such ours by artificially retaining parts of the dataset and then introducing them at specific iterations, this method is most natural in an on-line setting, where the data arrives sequentially (e.g. streaming market prices). Moreover, as mentioned earlier, this approach is not well-suited to estimation in the frequency domain, where the sample spectral estimate needs to be recalculated for every newly introduced datapoint, since each new value y_t changes the correlation structure of the time series $y_{1:t}$. This would therefore introduce an additional computational cost at every iteration, rendering the sampling procedure less efficient.

3. Simulated annealing. This method is used particularly in optimisation applications and consists in defining $\pi_t(x) \propto \pi(x)^{\gamma_t}$, where $\{\gamma_t\}_{t=1,\dots,T}$ is an increasing sequence, where usually $\gamma_T = 1$. This method was introduced in Kirkpatrick et al. [1983] to address the issue of estimates getting stuck in local modes for multimodal targets. However, this choice does not simplify by much the target distribution and certainly does not make use of the prior in a Bayesian context. Moreover, as pointed out in Neal [2001], this method is not guaranteed to identify each mode with the correct probability.

4. Likelihood annealing. This method was originally introduced in Gelman and Meng [1998] and further developed in Neal [2001] and is our method of choice for our problem. The core idea of this method, also known as Annealed Importance Sampling (AIS) is to build a 'bridge' of distributions between the sampling and the target distribution. While there is no general rule for constructing this sequence, in the formulation of Neal [2001] this is set to $\pi_t(x) = \pi_0(x)^{1-\gamma_t}\pi_T(x)^{\gamma_t}$, where the terms $\{\gamma_t\}_{t=0,\dots,T}$ (also individually known as temperatures and collectively as the schedule) satisfy $0 = \gamma_0 \leq \cdots \leq \gamma_T = 1$. Natural choices for this sequence would be a linear or geometric spacing of the interval $[0, 1]$. Note that Neal [2001] uses π_t for the sampling distribution and π_0 for the target, while we use the opposite notation, following the convention in Del Moral et al. [2006]. We detail this method further and the reasons for choosing it for our problem in the next subsection.

4.6 Annealed Importance Sampling (AIS)

Annealed Importance Sampling, also known as likelihood annealing, is introduced in Neal [2001] as a method for dealing with multimodal target distributions. Instead of sampling directly from the target distribution at every iteration of the SIS procedure, as is the case even in methods such as data tempering, in AIS the target is introduced gradually at each iteration. A common such annealing method is to dampen the target by a subunitary exponential factor γ_t at iteration t, and to gradually increase this factor to one. Recall that the main problem with SIS was that after being initially sampled diffusely from the prior, even suboptimal subsequent samples of $\boldsymbol{\eta}$ may result in sudden jumps in the likelihood in the early iterations. This problem is even more pronounced if the parameter space is high dimensional. The annealing of the likelihood mitigates the risk of the SIS algorithm getting stuck in the local modes identified in these early iterations, while ignoring other, potentially more significant, modes.

Another advantage of AIS is that it applies naturally to Bayesian inference. In particular, in our situation, one would define $\pi_0(\boldsymbol{\eta}|y) = \pi(\boldsymbol{\eta})$ as the prior and $\pi_T(\boldsymbol{\eta}|y) = \pi(\boldsymbol{\eta}|y)$ as the posterior for the vector of model parameters $\boldsymbol{\eta}$, giving:

$$\pi_t(\boldsymbol{\eta}|y) = \pi(\boldsymbol{\eta})^{1-\gamma_t}\pi(\boldsymbol{\eta}|y)^{\gamma_t} = \pi(\boldsymbol{\eta})^{1-\gamma_t}[L(\boldsymbol{\eta}|y)\pi(\boldsymbol{\eta})]^{\gamma_t} = \pi(\boldsymbol{\eta})L(\boldsymbol{\eta}|y)^{\gamma_t} \tag{4.6.1}$$

with corresponding sequential importance weights $\left\{w_t^i = \frac{\pi_t(\boldsymbol{\eta}_t^i|y)}{\pi_{t-1}(\boldsymbol{\eta}_t^i|y)}\right\}_{\substack{i=1,\dots,N \\ t=1,\dots,T}}$ and overall importance weights $\{w^i\}_{i=1,\dots,N}$ given by:

$$w^i = \frac{\pi_T(\boldsymbol{\eta}_T^i)}{\pi_0(\boldsymbol{\eta}_1^i)} = \frac{\pi_T(\boldsymbol{\eta}_T^i)}{\pi_{T-1}(\boldsymbol{\eta}_T^i)} \cdots \frac{\pi_1(\boldsymbol{\eta}_1^i)}{\pi_0(\boldsymbol{\eta}_1^i)} = \prod_{t=1}^{T} w_t^i \tag{4.6.2}$$

Because of this simplified expression in a Bayesian setup, this method is also called likelihood annealing, as it builds a 'bridge' between the prior and the posterior, by gradually introducing more of the likelihood.

Finally, as pointed out in Neal [2001], AIS also provides an estimate of the marginal likelihood of the chosen model by approximating the ratio of the normalising constants of the sampling and target distributions. More precisely, let $\mathcal{C}_0 = \int \pi_0(\boldsymbol{\eta}|y)d\boldsymbol{\eta} = \int \pi(\boldsymbol{\eta})d\boldsymbol{\eta}$ and $\mathcal{C}_T = \int \pi_T(\boldsymbol{\eta}|y) = \int \pi(\boldsymbol{\eta}|y)d\boldsymbol{\eta}$ be the normalising constants of the prior and posterior distributions, respectively, where the prior can be unnormalised, but the likelihood must be normalised. Then, for a given model M specified through $\boldsymbol{\eta}$, the ratio $\mathcal{C}_T/\mathcal{C}_0 = P(y|M)$ is the marginal likelihood of the model given the time series data y. Neal [2001] shows that the average of the overall importance weights, $\frac{1}{N}\sum_{i=1}^{N} w^i$ converges to $P(y|M)$ for large values of N. This result provides a method for model comparison, where for two competing models M_1 and M_2 specified through $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ (where $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ can have different dimensions), one approximates the Bayes factor as:

$$BF_{12} = \frac{P(y|M_1)}{P(y|M_2)} \approx \frac{\sum_{i=1}^{N} w^i_{M_1}}{\sum_{i=1}^{N} w^i_{M_2}}$$

However, Neal [2001] cautions that when the likelihood dominates the prior (such as when there is a large amount of data), this estimate may have very large (or even infinite) variance. Choosing a larger number of bridging densities should alleviate this issue to some extent, due to the dampening of the likelihood in more terms (however, results will also depend on the spacing chosen between the exponents γ_t). Moreover, estimating Bayesian factors in this way is computationally costly, as it requires keeping track of and storing a potentially large array of weights over each iteration. Therefore, we resort to an alternative approximation to the marginal likelihood, known as the Bayesian Information Criterion (BIC). The BIC is defined as:

$$\text{BIC} = m\log(N) - 2\log(L_{\max})$$

where m is the number of parameters estimated, N is the sample size, and and $L_{\max}$ is the maximised likelihood. In our case, m is the dimension of $\boldsymbol{\eta}$, that is $m = R_p + 2Cp + Rq + 2Cq + 2k + 1$, N is the length of the data $\{X_t\}_{t=1,\dots,N}$, and the maximised loglikelihood is approximated by the discretised Whittle loglikelihood $l_{DW}(\cdot|Z)$ evaluated at the final estimate $\hat{\boldsymbol{\eta}}$. The advantage of the BIC is that it only requires evaluation of the likelihood once at the final estimate; due to its simplicity and efficiency, the BIC is our method of choice for comparing the models in Section 6.2.2. Finally, in Subsection 4.9 we briefly discuss the variance of estimates and the efficiency of AIS – for a more detailed analysis of these aspects we refer the interested reader to Neal [2001].

4.7 Markov Chain Monte Carlo (MCMC)

Having discussed the problem of constructing a sequence of distributions for the Sequential Importance Resampling (SIR) procedure, we now turn our attention to a different sampling method, known as Markov Chain Monte Carlo (MCMC). This procedure will be integrated in the SMC algorithm in the next section as a complementary part to the resampling step.

4.7.1 Markov Theory

Before introducing MCMC, we state without proof the core definitions and results from the theory of Markov Processes needed for MCMC. The proofs are standard and can be found in most textbooks on Markov Chains and MCMC, such as Robert and Casella [2005], or Liang et al. [2011]. The theoretical discussion below is largely based on Gilks et al. [1995]. We begin by giving the definition of a discrete-time Markov Process (also called Markov Chain).

Definition 2 (Discrete-Time Countable State Space Markov Process)**.** A discrete-time stochastic process $\{X_t\}_{t=0,1,\dots}$ taking values in a countable state space $\mathcal{I}$ is called a Markov Chain if it has the Markov property:

$$P(X_t = i_t|X_0 = i_0, X_1 = i_1, \dots, X_{t-1} = i_{t-1}) = P(X_t = i_t|X_{t-1} = i_{t-1})$$

for states $i_0, i_1, \dots, i_t \in \mathcal{I}$. Additionally, the Markov chain is called time-homogeneous if:

$$P(X_{t+1} = i|X_t = j) = P(X_t = j|X_{t-1} = i),\ \forall t \in \mathbb{N}/\{0\}$$

Intuitively, the Markov property means that the present value of the chain only depends on the previous value, while time-homogeneity means that the dependence or transition structure is constant in time. The transitions between every state i and state j are encapsulated in a transition matrix (also known as transition kernel) $P = (p_{ij})_{i,j\in\mathcal{I}}$ with elements $p_{ij} = P(X_{t+1} = i|X_t = j) = P(X_1 = i|X_0 = j)$, assuming time-homogeneity. Given an initial distribution π_0 with components $\pi_0^{(i)} = P(X_0 = i)$, $i, j \in \mathcal{I}$, we seek the distribution of the chain at time t, which we denote by $\pi_t = \left(\pi_t^{(i)}\right)$, where, as before, $\pi_t^{(i)} = P(X_t = i)$, $i \in \mathcal{I}$. One can show using the Markov property and the Law of Total Probability that for a homogeneous chain:

$$\pi_t = \pi_0 P^t.$$

where $P^t = (p_{ij}(t))_{i,j\in\mathcal{I}}$ is the t-step transition matrix. Thus, knowledge of the initial distribution and of the transition matrix is enough to determine the distribution of the process at any time t. We are now interested in understanding the limiting behaviour of π_t as $t \to \infty$ and how it relates to the concept of a stationary distribution π, which is defined through the equation:

$$\pi = \pi P.$$

For this, we introduce some additional concepts:

Definition 3 (Positive Recurrence, Irreducibility, Aperiodicity, and Ergodicity of Markov Processes)**.** Let $\{X_t\}_{t=0,1,\dots}$ be a discrete-time Markov Chain. Then a state $i \in \mathcal{I}$:

1. Communicates with a state $j \in \mathcal{I}$ if $\exists\, \tau \in \mathbb{N}/\{0\}$ s.t. $p_{ij}(\tau) > 0$ (state j is accessible from state i) and $\exists\, \tau' \in \mathbb{N}/\{0\}$ s.t. $p_{ji}(\tau') > 0$ (j is accessible from i). The collections of states that communicate with one another are known as communicating classes. Classes which do not communicate with other classes are said to be closed, otherwise they are said to be open.

2. Is recurrent if $P(X_t = i|X_0 = i) = 1$ for some $t \in \mathbb{N}/\{0\}$ and positive recurrent, if additionally, $\mathbb{E}[T_i|X_0 = i]$ is finite (and null recurrent if it is not), where $T_i \coloneqq \min\{t \in \mathbb{N} : X_t = i\}$ denotes the first passage time. Otherwise, if $P(X_t = i|X_0 = i) < 1$, the chain is transient. If i is positive recurrent, then $\lim_{t\to+\infty} p_{ji}(t) \neq 0$, $\forall j \in \mathcal{I}$.

3. Is aperiodic if $\gcd\{t : p_{ii}(t) > 0\} = 1$.

4. Is ergodic if it is positive recurrent and aperiodic.

We say that the chain $\{X_t\}$ has any of the properties above if the respective property holds true for all states $i \in \mathcal{I}$. Additionally, the chain is called irreducible if all states communicate, or equivalently, if the state space consist of a single (closed) communicating class.

Without dwelling too much on these properties, we focus on some further results that are relevant to our discussion of MCMC.

Lemma 1 (Uniqueness of the Stationary Distribution and Limiting Distribution)**.** Let $\{X_t\}_{t=0,1,\dots}$ be an irreducible Markov chain. Then:

1. If the state space is finite or $\{X_t\}$ is positive recurrent, there exists a unique stationary distribution π.

2. If additionally, $\{X_t\}$ is aperiodic, and hence ergodic, $\lim_{t\to+\infty} \pi_t^{(i)} = \pi^{(i)}$, $\forall i \in \mathcal{I}$, i.e. the limiting (equilibrium) distribution is the stationary distribution.

Therefore, provided that we can generate a long enough ergodic Markov Chain with a pre-specified stationary distribution π, we can approximate π by the limiting distribution of the chain. Indeed, this is the core idea of MCMC. It is important to note that, unlike in previous methods such as Crude Monte Carlo or SIS, the values of a Markov process are, by nature, not independent. Nonetheless, the dependence in the

sample obtained via MCMC decreases as the chain progresses, and as such a common technique to make the sample closer to independent is to discard a pre-defined number of initial samples, known as the burn-in. Importantly, the question remains how to simulate an ergodic Markov chain with stationary distribution π. For this, we introduce an additional property that a Markov chain can have:

Definition 4 (Reversibility and Detailed Balance)**.** Let $\{X_t\}_{t=0,1,\dots}$ be a discrete-time Markov Chain with stationary distribution π. Then we say that $\{X_t\}$ is reversible if it satisfies the detailed balance condition:

$$\pi^{(j)} p_{ji} = \pi^{(i)} p_{ij}$$

for any two states $i, j \in \mathcal{I}$.

It can be shown that reversibility is a sufficient condition for the existence of a stationary distribution. Moreover, the detailed balance equation provides a way to construct reversible Markov Chains with given stationary distributions – this is the idea of the Metropolis-Hastings kernel, discussed in the next subsection.

4.7.2 The Metropolis-Hastings and Gibbs Sampling Algorithms

One of the earliest and most popular ways of constructing MCMC kernels with a pre-specified invariant distribution is the Metropolis-Hastings (MH) algorithm, originally introduced by Nicholas Metropolis in 1953 and further generalised by Wilfred Hastings in 1970 to include non-symmetric proposals. After initialising the chain $\{X_t\}$ at X_0, the algorithm proceeds iteratively for a pre-defined number of iterations, at each iteration t sampling a candidate value $\dot{X}_t$ from an arbitrary (ideally, easy to sample from) proposal distribution $q(\cdot|X_t)$ (satisfying some technical regularity conditions) and calculating the acceptance ratio:

$$\rho(X_t, \dot{X}_t) = \min\left\{1, \frac{\pi(\dot{X}_t) q(X_t|\dot{X}_t)}{\pi(X_t) q(\dot{X}_t|X_t)}\right\}$$

If the proposal is symmetric, the terms $q(X_t|\dot{X}_t)$ and $q(\dot{X}_t|X_t)$ cancel out and the algorithm reduces to the original Metropolis algorithm. To decide whether to accept or reject the candidate $\dot{X}_t$, one samples a standard uniform random variable $u \sim Uniform(0,1)$ and accepts the candidate if $u \leq \rho(X_t, \dot{X}_t)$, in which case one sets the next value of the chain at $X_{t+1} = \dot{X}_t$. Otherwise, if $u > \rho(X_t, \dot{X}_t)$, one rejects the candidate and sets the next value of the chain at the current value, i.e. $X_{t+1} = X_t$. Essentially, this acceptance-rejection step can be seen as acting like an indicator function within the transition kernel for the chain. A non-rigourous argument for why the Metropolis-Hastings algorithm converges to the invariant distribution uses the assumption of reversibility and can be found on page 7 of the introductory chapter to Gilks et al. [1995]; we omit this here for brevity. The Metropolis-Hastings algorithm is sketched in the context of our problem in the next subsection to generate a chain of model parameters $\{\boldsymbol{\eta}_t\}_{t=0,1,\dots}$ with stationary distribution equal to the posterior $\pi(\boldsymbol{\eta}|y)$.

Finally, a note on the proposal: while any density $q(\cdot)$ satisfying the regularity conditions (which we omit here) works in theory, the rate of convergence depends on the choice of both the functional form of the proposal and of the parameters of the proposal. In particular, considering the example of a Truncated Normal proposal, as used in the next section, the variance of this distribution impacts the mixing time of the chain, a measure of the speed of convergence. Loosely speaking, this is the time that it takes for the Markov chain to become range-bound within the support of π. Both a variance that is too high and one that is too low can cause the chain to mix slowly, the former by making the chain overshoot too frequently before eventually converging, and the latter by evolving the chain too slowly. As there is no general way to address this, especially if π is high-dimensional, we rely on experimentation.

Another popular algorithm is the Gibbs sampler, which allows for efficient simulation from multivariate densities. The Gibbs sampler is a particular case of the single-component Metropolis-Hastings, which consists in updating the components of an m-dimensional multivariate chain $\boldsymbol{X}_t = \left(X_t^{(k)}\right)_{k=1,\dots,m}$ independently instead of updating all of $\boldsymbol{X}_t$ simultaneously, or en-bloc. In particular, the idea of this algorithm, due

to Stuart and Donald Geman, is to update each component of $\boldsymbol{X}_t$ sequentially, by defining the proposal distribution for each component to be equal to the corresponding full conditional distribution. As such, after initialising the chain $\boldsymbol{X}_t$ at $\boldsymbol{X}_0$, the algorithm again proceeds iteratively for a pre-defined number of iterations, at iteration t sampling a value $\boldsymbol{X}_t = \left(X_t^{(k)}\right)_{k=1,\dots,m}$ based on the sample $\boldsymbol{X}_{t-1} = \left(X_{t-1}^{(k)}\right)_{k=1,\dots,m}$ from the previous iteration. This is done componentwise, as follows:

$$
\begin{aligned}
X_t^{(1)} &\sim f(x_1 | X_{t-1}^{(2)}, \dots, X_{t-1}^{(m)}) \\
X_t^{(2)} &\sim f(x_2 | X_t^{(1)}, X_{t-1}^{(3)}, \dots, X_{t-1}^{(m)}) \\
&\vdots \\
X_t^{(k)} &\sim f(x_k | X_t^{(1)}, \dots, X_t^{(k-1)}, X_{t-1}^{(k+1)}, \dots, X_{t-1}^{(m)}) \\
&\vdots \\
X_t^{(m)} &\sim f(x_m | X_t^{(1)}, \dots, X_t^{(m-1)})
\end{aligned}
$$

where each of the m univariate sampling steps can be performed, for instance, using the Metropolis-Hastings algorithm. Indeed, this is the Metropolis-within-Gibbs algorithm described in detail in the next subsection (Algorithm 3). This algorithm, together with the Metropolis-Hastings algorithm (Algorithm 2) are provided in the next section in the context of our problem.

4.8 Sequential Monte Carlo Sampling (SMC)

Sequential Monte Carlo (also known as the Particle Filter) was originally developed in the late 1990s by Pierre Del Moral and also by Jun Liu and Rong Chen in the context of stochastic filtering in state-space models. SMC was subsequently adapted as a sampling mechanism for general distributions, most notably in Del Moral et al. [2006]. SMC builds further on the SIR procedure discussed in Subsection 4.4 by evolving the particles according to a Markov transition kernel. One can note that, by making this addition, both SIR and MCMC can be seen as particular cases of SMC. The introduction of a Markov kernel addresses the problem of weight degeneracy in a more profound way than the simple resampling procedure in SIR. This is because, as pointed out in Chopin [2002], resampling merely reduces the computational load of updating the particles with low weights by discarding these particles and replacing them with copies of the particles with higher weights. While this increases the ESS, it does not increase the overall information gained from sampling, and also has the side effect of introducing correlations between previously independent particles. By contrast, generating new particles through a Markov kernel increases the information content of the sample, since with this modification the particles with small weights are replaced by entirely new particles, which may provide additional information about the shape of the distribution at their respective location.

The MCMC step is thus akin to the 'rejuvenation' concept introduced in Gilks and Berzuini [2001]. When done at the t-th iteration of SIS, this step consists in moving each particle $\boldsymbol{\eta}_t^i$ away from the previous estimate based on a sensible Markov kernel $K(\cdot, \cdot)$ with stationary distribution π_t. This step is generally performed as part of the wider resampling step as in Li et al. [2021] or in Gilks and Berzuini [2001], but can also be performed at each iteration of the SIS or AIS routine (i.e. for each bridging density), as in Del Moral et al. [2006]. When done in the resampling step, the updates at iteration t are of the form $\boldsymbol{\eta}_t^i = K(\boldsymbol{\eta}_t^i, \cdot)$, $i = 1, \dots, n$, where $\{\boldsymbol{\eta}_t^i\}_{i=1,\dots,N}$ are the resampled particles. When done at every iteration t of the wider procedure, one updates $\boldsymbol{\eta}_t^i = K(\boldsymbol{\eta}_{t-1}^i, \cdot)$, $i = 1, \dots, N$ based on the estimates from the previous iteration. Since the resampling step only happens occasionally, namely when the ESS falls below a pre-defined threshold, performing the MCMC step within the resampling condition leads to fewer runs of the Markov kernel, and hence to a faster execution time, but also potentially to a poorer exploration of the sample space. However, this can be compensated by running a larger number of MCMC updates per resampling cycle. The generic algorithm below (Algorithm 1) is a based on algorithm 3.1 in Li et al. [2021], algorithm 5.1 in Petris et al. [2009], and algorithm 3 in Cappé et al. [2007]:

Algorithm 1 General SMC Algorithm

Initialise the vector of parameter estimates and weights

Sample $\boldsymbol{\eta}_0^i \overset{iid}{\sim} \pi_0$, $i = 1, \ldots, N$ ▷ Sample N particles independently from the prior

Initialise the weights $w_0^i = 1/N$, $i = 1, \ldots, N$ ▷ Initialise the normalised weights as uniform

for $t = 1, \ldots, T$: ▷ Iterate T times

  for $i = 1, \ldots, N$: ▷ Repeat for each of the N particles

    Reweight: $\tilde{w}_t^i = w_{t-1}^i \frac{\pi_t(\boldsymbol{\eta}_{t-1}|y)}{\pi_{t-1}(\boldsymbol{\eta}_{t-1}|y)}$ ▷ Update the weight as in IS

    Set: $\boldsymbol{\eta}_t^i = \boldsymbol{\eta}_{t-1}^i$ ▷ Carry the particle forward to the next iteration

  end for

  Normalise the weights: $w_t^i = \frac{\tilde{w}_t^i}{\sum_{j=1}^N \tilde{w}_t^j}$, $i = 1, \ldots, N$ ▷ Rescale the weights to sum to 1

  Compute The Effective Sample Size: ESS $= \frac{1}{\sum_{j=1}^N (w_t^j)^2}$

  if ESS $< \alpha N$ or $t = T$, with $\alpha \in (0, 1]$: ▷ Resample for low ESS and on the last iteration

    for $i = 1, \ldots, N$:

      Resample the particles: $\dot{\boldsymbol{\eta}}^i \sim P(\dot{\boldsymbol{\eta}}^i = \boldsymbol{\eta}_t^i) = w_t^i$ ▷ Multinomial selection

      Relabel the sample: $\boldsymbol{\eta}_t^i = \dot{\boldsymbol{\eta}}^i$

      Set: $w_t^i = 1/N$ ▷ Reinitialise the weights

      for $r = 1, \ldots, R_t$:

        Evolve the particles using a Markov kernel K with invariant distribution π_t: $\boldsymbol{\eta}_t^i = K(\boldsymbol{\eta}_t^i, \cdot)$

      end for

    end for

  end if

end for

This algorithm is most similar to that in Li et al. [2021], where the particles are evolved using an MCMC kernel within the resampling step for a separately defined number of iterations R_t. While the number of MCMC iterations R_t can be different for each iteration t of the SMC algorithm, and thus could be chosen adaptively, for simplicity we will use the same $R_t = R$ for each resampling step. Additionally, we perform one more resampling and MCMC step on the last iteration. This ensures that the sample has been weighted according to the latest importance weights, and increases the likelihood that the final particles are unique (i.e. that no copies are present).

We considered two options for the MCMC kernel – en-bloc Metropolis-Hastings (Algorithm 2) and Metropolis-within-Gibbs (Algorithm 3). While both algorithms use a Metropolis-Hastings acceptance-rejection step, the way in which candidates are proposed is different. For the i-th particle, at iteration $r \in \{1, \ldots, R_t\}$, Algorithm 2 proposes a candidate $\dot{\boldsymbol{\eta}}^i$ for the entire vector of parameters, which it then compares to the previous estimate $\boldsymbol{\eta}_{r-1}^i$. This is done by sampling from the joint proposal distribution, which here we assume to be a product of independent Truncated Normal proposals (see Appendix A.3), whose locations are the previous estimates. On the other hand, for the same particle i and MCMC iteration r, Algorithm 3 cycles through the m components of the m-dimensional vector $\boldsymbol{\eta}^i$, proposing a candidate $\dot{\eta}_{(k)}^i$ for the k-th component, $k \in \{1, \ldots, m\}$, conditional on the vector $\boldsymbol{\eta}_{(*)}^i$ of current components. $\boldsymbol{\eta}_{(*)}^i$ is

comprised of all the previous components (up to $k-1$ inclusive) estimated at the current iteration, as well as the k-th component and above estimated at the previous iteration. This candidate is then substituted at its corresponding position in $\boldsymbol{\eta}^i_{(*)}$ to generate an m-dimensional candidate $\dot{\boldsymbol{\eta}}^i_{(*)}$, which is then compared with $\boldsymbol{\eta}^i_{(*)}$. Thus in this second algorithm, mR_t MCMC iterations are performed in total, and as such R_t should be chosen smaller to account for this.

Algorithm 2 Metropolis-Hastings Step (Joint Proposals, En-Bloc Updates)

$\boldsymbol{\eta}^i_{r=0} = \boldsymbol{\eta}^i_t$ ▷ Initialise the chain at the latest estimate

for $r = 1, \ldots, R_t$: ▷ Iterate R_t times

Sample $\dot{\boldsymbol{\eta}}^i$ from the joint proposal distribution $q(\dot{\boldsymbol{\eta}}^i | \boldsymbol{\eta}^i_{r-1})$

Compute ▷ Compute the acceptance ratio

$\rho(\dot{\boldsymbol{\eta}}^i, \boldsymbol{\eta}^i_{r-1}) = \min\left\{1, \frac{\pi_t(\dot{\boldsymbol{\eta}}^i|y)\ q(\boldsymbol{\eta}^i_{r-1}|\dot{\boldsymbol{\eta}}^i)}{\pi_t(\boldsymbol{\eta}^i_{r-1}|y)\ q(\dot{\boldsymbol{\eta}}^i|\boldsymbol{\eta}^i_{r-1})}\right\}$

Sample $u \sim \text{Uniform}(0,1)$ ▷ Accept or reject the proposed value

if $\rho(\dot{\boldsymbol{\eta}}^i, \boldsymbol{\eta}^i_{r-1}) \geq u$:

$\boldsymbol{\eta}^i_r = \dot{\boldsymbol{\eta}}^i$ ▷ Update the particle

else:

$\boldsymbol{\eta}^i_r = \boldsymbol{\eta}^i_{r-1}$ ▷ Leave the particle unchanged

end if

end for

Algorithm 3 Metropolis-within-Gibbs Step (Independent Proposals, Sequential Updates)

$\boldsymbol{\eta}^i_{r=0} = \boldsymbol{\eta}^i_t$ ▷ Initialise the chain at the latest estimate

for $r = 1, \ldots, R_t$: ▷ Iterate R_t times

for $k = 1, \ldots, m$: ▷ Iterate over the m components of $\boldsymbol{\eta}$

Set $\boldsymbol{\eta}^i_{(*)} = (\eta^i_{(1)r}, \ldots, \eta^i_{(k-1)r}, \eta^i_{(k)r-1}, \eta^i_{(k+1)r-1}, \ldots, \eta^i_{(m)r-1})$ ▷ Vector of current components

Sample $\dot{\eta}^i_{(k)}$ from the k-th proposal distribution $q(\dot{\eta}^i_{(k)} | \boldsymbol{\eta}^i_{(*)})$

Set $\dot{\boldsymbol{\eta}}^i_{(*)} = (\eta^i_{(1)r}, \ldots, \eta^i_{(k-1)r}, \dot{\eta}^i_{(k)}, \eta^i_{(k+1)r-1}, \ldots, \eta^i_{(m)r-1})$ ▷ Vector of proposed components

Compute ▷ Compute the acceptance ratio

$\rho(\dot{\boldsymbol{\eta}}^i_{(*)}, \boldsymbol{\eta}^i_{(*)}) = \min\left\{1, \frac{\pi_t(\dot{\boldsymbol{\eta}}^i_{(*)}|y)\ q(\eta^i_{(k)r-1}|\dot{\boldsymbol{\eta}}^i_{(*)})}{\pi_t(\boldsymbol{\eta}^i_{(*)}|y)\ q(\dot{\eta}^i_{(k)}|\boldsymbol{\eta}^i_{(*)})}\right\}$

Sample $u \sim \text{Uniform}(0,1)$ ▷ Accept or reject the proposed value

if $\rho(\dot{\boldsymbol{\eta}}^i_{(*)}, \boldsymbol{\eta}^i_{(*)}) \geq u$:

$\eta^i_{(k)r} = \dot{\eta}^i_{(k)}$ ▷ Update the k-th component of the particle

else:

$\eta^i_{(k)r} = \eta^i_{(k)r-1}$ ▷ Leave the k-th component of the particle unchanged

end if

end for

end for

Note that the above algorithms are relatively general. In practice, we would often choose a symmetric and independent proposal for each sample (i.e. identity covariance matrix w.r.t. $q(\cdot|\cdot)$), so that, in the case of the Metropolis-Hasting algorithm, the terms $q(\boldsymbol{\eta}^i_{r-1}|\dot{\boldsymbol{\eta}}^i)$ and $q(\dot{\boldsymbol{\eta}}^i|\boldsymbol{\eta}^i_{r-1})$ would cancel out, and similarly for the corresponding terms in the Metropolis-within-Gibbs scheme.

In the practical SMC algorithm implemented in the code in Appendix C, we are using the Metropolis-within-Gibbs algorithm with independent Truncated Normal proposals for each component of the parameter vector $\boldsymbol{\eta}$. Both this choice and the Metropolis-Hastings algorithm ensure that our kernel has π_t as its invariant distribution for each iteration t. However, the Metropolis-within-Gibbs scheme is preferable, as it has a higher acceptance rate, and is thus more efficient. This is because, by updating the parameters sequentially instead of en-bloc, it reduces the probability of the less influential components of $\boldsymbol{\eta}$ getting 'carried away' to random values whenever a candidate is accepted on the grounds that the candidate contained accurate estimates of the more influential components. Here, by an influential component we mean that determining the parameter corresponding to that component influences significantly the shape of the estimated spectrum, and hence the likelihood.

Finally, we note (see Chopin and Papaspiliopoulos [2020]) that there is no general rule for choosing the optimal number R of MCMC iterations and, more generally, that designing Markov kernels within SMC, which can reliably deal with multi-modal targets (particularly in high dimensions) remains an open problem. For a discussion on optimal Markov kernels, see Del Moral et al. [2006]. For a comprehensive introduction to SMC, see Chopin and Papaspiliopoulos [2020]. For a recent review of SMC methods and a discussion on designing efficient proposal distributions and sequences of intermediate densities using machine learning techniques, see Naesseth et al. [2019].

4.9 Variance and Convergence of SMC

Having established a full framework for SMC sampling, we now briefly discuss the variance and convergence of the SMC procedure with likelihood annealing. First, we consider the variance of importance sampling estimates in general. Recall the general problem we introduced in Subsection 4.2 of estimating the expectation $I(f) = E_\pi[f(X)]$, where $X \sim \pi$. Again, if one is interested in probabilities, one takes f to be an indicator function. We saw that both the Crude Monte Carlo estimate $\hat{I}(f) = \frac{1}{N}\sum_{i=1}^N f(X_i)$ and the Importance Sampling estimate $\hat{I}_{IS}(f) = \sum_{i=1}^N w^i f(X_i)$ are unbiased for $I(f)$. However, their variances are different. Unbiasedness implies that the variances of these two estimators are given by:

$$Var(\hat{I}(f)) = \frac{1}{N}\int [f(x) - I(f)]^2 \pi(x)dx \quad \text{and} \quad Var(\hat{I}_{IS}(f)) = \frac{1}{N}\int [W(x)f(x) - I(f)]^2 \pi(x)dx$$

The variance of the Importance Sampling estimator is highly dependent on the weight function $W(x)$, however for an appropriately chosen approximating distribution (or sequence of distributions in the case of SIS) it is generally smaller than that of the Crude Monte Carlo estimator. Geweke [1989] shows that $Var(\hat{I}_{IS}(f))$ can be approximated from the sample as:

$$\widehat{Var}(\hat{I}_{IS}(f)) \approx \frac{\sum_{i=1}^N [w^i(f(X_i) - \hat{I}_{IS}(f))]^2}{\left(\sum_{i=1}^N w^i\right)^2} = \text{ESS} \cdot \sum_{i=1}^N [w^i(f(X_i) - \hat{I}_{IS}(f))]^2$$

using the notation in Subsection 4.2 and the definition of the ESS from equation (4.4.1). In the case of AIS, one would take $w^i = \prod_{t=1}^T w_t^i$, the overall importance weight of particle i as defined in equation (4.6.2). It is important to note that this approximation for the vaiance treats the sample $X_1, \ldots, X_n$ as independent. However, in the case of SMC, $X_1, \ldots, X_n$ are dependent samples generated from a Markov Chain. Neal [2001] points out that this may lead to overstating the ESS, and consequently the variance, so that one would also need to take into account the autocorrelation in the sample. Besides this complication, it is clear that the variability of the weighs is a crucial factor in the variability of the estimate.

From the formulation of AIS, and adapting to our problem, we have that for the i-th particle $\boldsymbol{\eta}^i$:

$$\begin{aligned}\log(w^i) &= \log(\pi_T(\boldsymbol{\eta}_T^i)) - \log(\pi_0(\boldsymbol{\eta}_1^i)) \\ &= \sum_{t=1}^{T}[\log(\pi_t(\boldsymbol{\eta}_t^i)) - \log(\pi_{t-1}(\boldsymbol{\eta}_t^i))] \\ &= \sum_{t=1}^{T}(\gamma_t - \gamma_{t-1})[\log(\pi_T(\boldsymbol{\eta}_t^i)) - \log(\pi_0(\boldsymbol{\eta}_t^i))]\end{aligned}$$

When the temperatures $\gamma_1, \ldots, \gamma_T$ are spaced evenly (linear schedule), the expression above simplifies to:

$$\log(w^i) = \frac{1}{T}\sum_{t=1}^{T}[\log(\pi_T(\boldsymbol{\eta}_t^i)) - \log(\pi_0(\boldsymbol{\eta}_t^i))]$$

Assuming that the states $\boldsymbol{\eta}_t^i$, $t = 1, \ldots, T$ are independent and that $\log(\pi_T(\boldsymbol{\eta}_t^i)) - \log(\pi_0(\boldsymbol{\eta}_t^i))$ has finite variance for all t, Neal [2001] argues that the Central Limit Theorem (CLT) implies that $\log(w^i)$ is asymptotically Normal with mean 0 and variance σ_i^2/T, for some non-negative constant σ_i^2. Thus the weights are asymptotically Lognormal and their variance decreases at the rate of $\exp(1/T)$. Neal [2001] shows that this result also holds when the schedule is not linear, provided that the terms $\gamma_t - \gamma_{t-1}$ decay at an appropriate rate for all t. Finally, Neal [1996] shows that the optimal schedule in terms of minimising the variance of the weights is the geometric spacing, i.e. a linear spacing of the log-temperatures. This asymptotic result for the variance of the weights can be used to obtain a measure of convergence, since the time needed to obtain estimates with a pre-specified variance σ^2 will be proportional to $T\exp(\sigma^2/T)$. One can in practice monitor this convergence by tracking the ESS throughout the iterations $1, \ldots, T$. For instance, under a linear schedule, the ESS should decrease more slowly after each resampling, and indeed this pattern can be seen in Figure 6.2 for the simulated data.

In the discussion above we focused on the variance of AIS estimates. Del Moral et al. [2006] notes that in general, the variance of SMC estimates cannot be established to be smaller than the variance of the SIS (and AIS in particular) estimates with the same number of iterations and the same sequence of densities. The relation between the two variances depends on whether resampling is necessary. While any resampling introduces some additional variance, if successive bridging densities are very different (e.g. if we are using geometric spacing for a small number of iterations), resampling can have a net positive effect on the variance. For a more in-depth discussion of this, we refer the interested reader to section 3.4 in Del Moral et al. [2006]. For a detailed discussion of variance estimation in SMC, see Lee and Whiteley [2018] and for an in-depth theoretical review of convergence, see Crisan and Doucet [2007].

4.10 Advantages of SMC over MCMC

We conclude the section on SMC by highlighting some advantages of SMC samplers over MLE and standard MCMC procedures, such as the Metropolis-Hastings algorithm or the Gibbs sampler. When constructing a time series model that aims to capture many features in the data, one often needs to estimate a relatively large number of parameters. The standard frequentist approach is Maximum Likelihood Estimation (MLE). While exact MLE does not require a complex sampling algorithm like SMC or MCMC and is simply a brute-force numerical optimisation, it has some shortcomings. The main shortcoming of MLE is that it only produces point estimates, whereas Bayesian methods produce distributions for each parameter, from which measures of uncertainty, such as credible intervals, can be easily computed. In contrast, the asymptotic theory of MLE for non-Gaussian models is technically challenging, especially if one takes into account the correlation structure for high-order models. Therefore, if one wishes to estimate the model parameters in a Bayesian framework, one has to sample from a potentially high-dimensional and/or multi-modal posterior. This task often renders simple simulation techniques such as Rejection Sampling or Crude Monte Carlo infeasible, prompting the need for more sophisticated Monte Carlo techniques. Standard MCMC methods

are a popular tool to achieve this goal, however these methods have some drawbacks, as we outline below. While to some extent, SMC is also affected by the same problems (since it uses a MCMC kernel), the tempering effect of operating on a sequence of distributions and the resampling step help SMC overcome some of these obstacles:

- For multi-modal target distributions, MCMC samplers may get stuck in local modes for a long time, rendering the procedure inefficient, while providing no visibility into how many iterations are necessary to ensure convergence to the stationary distribution. In contrast, the tempering effect of operating on a sequence of distributions (whether through data or likelihood annealing) reduces the chances of this occurring. Although the Importance Sampling component of SMC tends to concentrate the particles around local modes, thus failing to detect other, perhaps more significant modes, the resampling and reweighting steps help mitigate this tendency.

- SMC has the convenient property that when applied to an unbiased likelihood, it produces unbiased posterior estimates (see Li et al. [2021]). While the standard Whittle likelihood, as employed here is generally biased, one can see that using SMC in conjunction with the debiased Whitlle likelihood of Sykulski et al. [2019] should lead to a good approximation of the posterior.

- Unlike MCMC, SMC can be parallelised, which is particularly useful for high-dimensional sample spaces, when a large number of parameters have to be estimated. It is true that unless the proposals are independent, the MCMC kernel is not parallelisable, however, this part is only run occasionally, when the ESS falls below the threshold, and the SIS procedure is entirely parallelisable. Thus, SMC can result in computational improvements. We provide some more details in Subsection 7.2.

While SMC clearly provides some improvements over MCMC procedures, it is also prone to some of the same problems, the main one being choosing an optimal proposal distribution, that is a proposal which balances good exploration of the sample space with a high acceptance rate, and thus a high efficiency. Even with a simple proposal, such as a Normal or Truncated Normal distribution, it may be computationally expensive to tune the variance of this distribution. The main trade-off is that a low variance will cause the model to have a high acceptance rate, but will lead to a poor exploration of the sample space, with the converse being true for a higher value.

Due to the aforementioned qualities, SMC is becoming an increasingly popular tool for purposes outside its original use in filtering, such as sampling from general distributions, having been more recently used in the context of time series analysis in papers such as Li et al. [2021] (although not in the frequency domain).

5 Simulating in the Time Domain

Given a vector of model parameters $\boldsymbol{\eta}$ as in (3.2.1) one can straightforwardly calculate the resulting spectral shape $S_Y(\cdot)$ of the GARMA process $\{Y_t\}$ using (2.5.2). Whether $\boldsymbol{\eta}$ is pre-specified for synthetic data generation or estimated through SMC as in Subsection 4.8, it is desirable to translate this quantity into the time domain to simulate the process Y_t with the corresponding spectrum. In this section we suggest two approaches for achieving this.

The first approach consists in simulating the process directly from the values of the spectrum. Percival [1992] discusses four variants of this approach for Gaussian processes: three require first calculating the autocovariance sequence through an inverse Fourier transform, but the fourth, an approximate frequency-domain method, does not. All four methods discussed in Percival [1992] require the simulated process to be Gaussian and stationary. While we are working under the assumption of stationarity, unlike the ARMA process, the general k-factor GARMA process is not Gaussian. However, it can be shown to be asymptotically Gaussian, (see Dissanayake et al. [2018], Hunt et al. [2022]), rendering these methods a valid approximation.

Our second approach to simulation consists in generating a Gaussian white noise process with variance σ_ϵ^2, and then successively applying the MA, AR, and Gegenbauer digital filters, in this order. While applying the AR and MA LTI filters is straightforward, due to the fractional differencing, correctly constructing

the Gegenbauer filter presents some technical challenges. The approach suggested relies on the ideas in Dissanayake et al. [2018]. For this second simulation approach, it is clear that we first need to recover the functional form of the k-factor GARMA process given in equation (2.5.1). While estimates for the parameters $(\delta_1, \ldots, \delta_k)$ and $(\lambda_1, \ldots, \lambda_k)$ in the k-factor Gegenbauer component are readily available from the output of the SMC sampler, we need to recover the time-domain coefficients $(\phi_1, \ldots, \phi_p)$ and $(\theta_1, \ldots, \theta_q)$ of the AR and MA polynomials, respectively, from the estimates of the moduli and arguments. We briefly discuss how this is done below.

5.1 Reconstructing the AR and MA Polynomials

Given $\boldsymbol{\eta}$ as above, we write the AR and MA reciprocal roots $\xi_1 = \left(\xi_1^R, \xi_1^C, \overline{\xi_1^C}\right)$ and $\xi_2 = \left(\xi_2^R, \xi_2^C, \overline{\xi_2^C}\right)$ as in (2.2.3) and (2.2.4) and reconstruct the AR and MA polynomials as:

$$\Phi(z) = \left(1 - \sum_{j=1}^{p} \phi_j z^j\right) = \prod_{j=1}^{p}(1 - \xi_{1j} z) = \left((-1)^p \prod_{j=1}^{p} \xi_{1j}\right) \prod_{j=1}^{p} \left(z - \frac{1}{\xi_{1j}}\right)$$

and

$$\Theta(z) = \left(1 - \sum_{j=1}^{q} \hat{\theta}_j z^j\right) = \prod_{j=1}^{q}(1 - \xi_{2j} z) = \left((-1)^q \prod_{j=1}^{q} \xi_{2j}\right) \prod_{j=1}^{q} \left(z - \frac{1}{\xi_{2j}}\right)$$

In order to reconstruct the polynomials $\Phi(z)$ and $\Theta(z)$ from their (reciprocal) roots, one can use Viète's relations. Recall that for a given polynomial

$$P(z) = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 = a_n (z - r_n)(z - r_{n-1}) \ldots (z - r_1)$$

with roots $r_1, r_2, \ldots, r_n$, Viète's relations are:

$$\begin{cases} r_1 + r_2 + \cdots + r_{n-1} + r_n = -\dfrac{a_{n-1}}{a_n} \\ (r_1 r_2 + r_1 r_3 + \cdots + r_1 r_n) + (r_2 r_3 + r_2 r_4 + \cdots + r_2 r_n) + \cdots + r_{n-1} r_n = \dfrac{a_{n-2}}{a_n} \\ \quad \vdots \\ r_1 r_2 \ldots r_n = (-1)^n \dfrac{a_0}{a_n}. \end{cases}$$

Therefore, to obtain $(\phi_1, \ldots, \phi_p)$ and $(\theta_1, \ldots, \theta_q)$ we use Viète's relations with the polynomials $\Phi(z)$ and $\Theta(z)$ and the leading factors $a_{1n} = \left((-1)^p \prod_{j=1}^{p} \xi_{1j}\right)$, and $a_{2n} = \left((-1)^q \prod_{j=1}^{q} \xi_{2j}\right)$, respectively.

5.2 Simulating a GARMA Process from its Spectrum

We now outline an approximate frequency-domain method for simulating any Gaussian process from its spectrum. The present treatment of this method is due to Percival [1992]. Since we have been working in the frequency domain throughout the inference process, we favour this method over the other three methods in Percival [1992], as it does not require the calculation of the autocovariance sequence.

Suppose we want to obtain a sample of length N from a Gaussian process $\{Y_t\}$ with SDF $S_Y(\cdot)$, which is continuous over the interval $[-1/2, 1/2]$. Let $f_j = \frac{j}{N}$, $j = 0, \ldots, N-1$ define the grid of Fourier frequencies. Assume that we have simulated N i.i.d. standard Normal random variables $\{W_t\}_{t=0,\ldots,N-1}$ (i.e. a Gaussian white noise process of length N with zero mean and unit variance). Then, for $j = 0, \ldots, N-1$ define:

$$U_j = \begin{cases} \sqrt{S_Y(0)} W_0, & j = 0; \\ \sqrt{\frac{1}{2} S_Y(f_j)} (W_{2j-1} + i W_{2j}), & j = 1, \ldots, \frac{N}{2} - 1; \\ \sqrt{S_Y(1/2)} W_{N-1}, & j = \frac{N}{2}; \\ U_{N-j}^*, & j = \frac{N}{2} + 1, \ldots, N-1; \end{cases}$$

where U^*_{N-j} denotes the complex conjugate of U_{N-j}. We then define the process $\{V_t\}$ with:

$$V_t = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} U_j e^{-i2\pi f_j t}, \quad t = 0, \ldots, N-1.$$

Due to $\{V_t\}$'s construction as the DFT of the complex-valued process $\{U_j\}$, we note that $\{V_t\}$ is a real-valued process. Moreover, $\{V_t\}$ is a zero-mean Gaussian process, since it is a linear combination of values of the zero-mean Gaussian processes $\{W_t\}$. Moreover, it can be shown that $\{V_t\}$ has autocovariance sequence:

$$s_V(\tau) = \frac{1}{N} \sum_{j=0}^{N-1} S(f_j) e^{i2\pi f_j t}.$$

Percival [1992] points out that $\{V_t\}$ is a harmonic process, which implies that both $\{V_t\}$ and $s_V(\cdot)$ are periodic – i.e. V_t is characterised by an integrated spectrum with steps at the Fourier frequencies f_j instead of a continuous SDF. The treatment in Percival [1992] is more general, using a white noise process of even length $M \geq N$ to simulate a larger number of values of the process $\{V_t\}$ and then selecting only the first N. In particular, Percival [1992] recommends that M be a power of two in order to allow for a more efficient calculation of the process $\{V_t\}$ using the Fast Fourier Transform (FFT) algorithm. The reason for choosing $M \geq N$ is to more closely match the statistical properties of the simulated process $\{V_t\}$ and the target process $\{Y_t\}$, the former of which has a periodic autocovariance sequence, whereas the latter's autocovariances decay to 0. However, in our situation, we are limited by the length of our data, since one of our applications is to simulate based on a vector of estimates of the spectrum $\hat{S}_Y(\cdot)$ of length $N/2+1$ ($f_j = j/N, j = 0, \ldots, N/2$), where N is the length of the original data y. The upshot of using $M = N$ is that the periodogram of $\{V_t\}$ will be in agreement with the estimated spectrum, with the flip side being that the periodogram is itself a biased representation of the spectrum of $\{V_t\}$. Finally, note that if we wanted to simulate a process with variance σ^2_ϵ instead of 1, it would suffice to change the variance of $\{W_t\}$ to σ^2_ϵ.

5.3 Simulating a GARMA Process from its Functional Form

An alternative simulation method is to simulate a GARMA process from its time-domain representation. This approach is a direct digital filtering method and has the advantage that it does not require the assumption of Gaussianity of $\{Y_t\}$. Following the steps in Subsection 5.1, suppose that we have obtained the AR and MA coefficients $(\phi_1, \ldots, \phi_p)$ and $(\theta_1, \ldots, \theta_q)$ (and hence the corresponding polynomials $\Phi(\cdot)$ and and $\Theta(\cdot)$), as well as the Gegenbauer parameters $(\delta_1, \ldots, \delta_k)$ and $(\lambda_1, \ldots, \lambda_k)$. We then aim to simulate a process $\{Y_t\}$ satisfying the GARMA functional representation with these parameters:

$$\Phi(B) \prod_{j=1}^{k} (1 - 2\lambda_j B + B^2)^{\delta_j} Y_t = \Theta(B)\epsilon_t$$

Let $G_j(B) \coloneqq (1 - 2\lambda_j B + B^2)^{\delta_j}$. Then we can write the above process as:

$$\Phi(B) \prod_{j=1}^{k} G_j(B) Y_t = \Theta(B)\epsilon_t$$

$$Y_t = \left(\prod_{j=1}^{k} G_j^{-1}(B) \right) \Phi^{-1}(B)\Theta(B)\epsilon_t$$

By construction, the AR polynomial $\Phi(\cdot)$ is invertible, due to the constraints on its reciprocal roots through the bounds on the arguments and moduli of these roots. In order to simulate the process $\{Y_t\}$ through this direct filtering approach, we require the Wold decomposition theorem, which we state below. This theorem is a general result applying to isometries on Hilbert spaces, but in the context of time series analysis, it states that any stationary stochastic process can be represented as the sum of a deterministic component and a (potentially infinite) MA process.

Theorem 2 (Wold Decomposition)**.** Let $\{Y_t\}$ be a second-order stationary process. Then Y_t can be written as:

$$Y_t = \eta_t + \sum_{j=0}^{\infty} b_j \epsilon_{t-j}$$

where $\{\epsilon_t\}$ is a white noise process (not necessarily Gaussian), $(b_j)_{j=0,\dots}$ is the (possibly infinite) vector of moving average coefficients, and $\{\eta_t\}$ is a deterministic sequence.

We can therefore use the Wold representation to obtain a truncation of length N of this infinite MA process. Since we are operating on detrended data and since we are modelling the seasonality through the GARMA parametrisation, we assume that $\eta_t = 0$, $\forall t$. Therefore, we seek to obtain the coefficients $(b_j)_{j=0,\dots,N-1}$. However, there are two challenges with this approach. The first problem is that representing an AR process in MA form is not trivial and as such it is computationally much more efficient to represent an ARMA process in AR(∞) form than it is to represent it in MA(∞) form. Nonetheless, the MA representation allows for a better understanding of the probability structure of the process, not least due to the simpler expression for the autocovariance sequence. The second problem is posed by the k Gegenbauer factors. Recall from equation (2.4.1) that we can represent the inverse of the operator $G(z) = (1 - 2\lambda B + B^2)^{\delta}$ as:

$$G^{-1}(z)(1 - 2\lambda z + z^2)^{-\delta} = \sum_{j=0}^{\infty} C_j^{\delta}(\lambda) z^j,$$

and thus the (one-factor) Gegenbauer process as $\sum_{j=0}^{\infty} C_j^{\delta}(\lambda)\epsilon_{t-j}$, with the ultraspherical polynomials C_j as in equation (2.4.2) – for more details see Dissanayake et al. [2018]. Gray et al. [1989] extended the formula above to show that the k-factor process can be represented as $\prod_{i=1}^{k} \left(\sum_{j=0}^{\infty} C_j^{\delta_i}(\lambda)\epsilon_{t-j} \right)$, however Woodward et al. [1998] highlights that due to the long-memory property of the Gegenbauer process, obtaining a sufficiently accurate truncation is computationally expensive, and suggests an alternative approximate method. Nonetheless, we can generate a GARMA(p, k, q) process by filtering an ARMA(p, q) process through the filter above. In turn, we generate the ARMA(p, q) process by filtering the MA(q) process through the reciprocal of the AR polynomial $\Phi(\cdot)$, which could be obtained through polynomial long division. Finally, filtering the white noise to obtain the MA(q) process is straightforward, as the coefficients are readily available.

We conclude this section by arguing that, while this direct filtering method is theoretically elegant and light on assumptions (namely, requiring only the stationarity of the AR component with no assumption on the Gaussianity of the target process), its inefficiency prompts us to choose the approximate frequency-domain method in Subsection 5.2 as the more practical method of simulation.

6 Spectral Estimation using SMC – Application to Data

6.1 Procedure Outline

So far the discussion of performing SMC in the frequency domain has been largely theoretical with some discussion on specific practical modelling assumptions and algorithmic choices. We now proceed to demonstrate the full methodology on a simulated time series, as well as on a real dataset. First, we review the steps:

1. Demeaning and stationarising the data. This is because we do not estimate the mean of the process and because the GARMA model assumes stationarity. For demeaning, we would simply calculate the sample mean of the data and subtract it from each observation. For stationarisation, we would test the time series for stationarity, e.g. using the Augmented Dickey-Fuller (ADF) test, and difference it until this test is passed at a satisfactory level (typically 5% or 1%).

2. Computing the Fourier frequencies and the periodogram or tapered periodogram at these frequencies.

3. Specifying the model order (R_p, C_p, R_q, C_q, k), namely is the number of components in each of the five groups of parameters, for a total of $m = R_p + 2C_p + R_q + 2Cq + 2k + 1$ parameters to estimate.

4. Specifying the settings for the SMC sampler:

   (i) the number of particles N.

   (ii) the number of bridging densities (number of SMC iterations) T.

   (iii) the number of MCMC updates R at each resampling.

   (iv) the MCMC proposal variance expressed as a common fraction of the length of each parameter's range, except for σ_ϵ^2, for which an absolute value is specified, due to the infinite range of this parameter. A brief discussion on optimising these parameters is provided in Subsection 7.2.

5. Running the SMC Sampler and obtaining estimates of the posterior distributions for each parameter. It is important to note here, as Huerta and West [1999b] observe, that when there are multiple components of the same type (AR, MA, or Gegenbauer), the parameters within each of the five groups are not identified – one cannot, for instance, distinguish two Gegenbauer memory parameters from each other. This is because the SDF is invariant under any permutation of these parameters. Hence, in our implementation (see Appendix C), we have devised a way to sort the parameter estimates based on their mean to provide identification when comparing them with their true values for simulated data.

6. Choosing a point estimate for each parameter based on a suitable summary statistic and computing the estimated spectrum using equation (2.5.2). Generally, we would choose the mode as the summary statistic, since this gives the maximum a posteriori (MAP) estimate, which can be loosely seen as the Bayesian analogue of the maximum likelihood estimate (MLE). However, the best choice of summary statistic depends on the shapes of the marginal posteriors for each parameter, and in some cases the mean or median may give a better estimate of the true spectrum. Additionally, where more than one mode is present, one should ideally try all combinations of modes for each parameter to find the optimal estimate of the spectrum – this could be determined, for instance, by comparing the BIC values for each combination.

7. Reconstructing the functional form of the estimated AR and MA polynomials using the formulas in Subsection 5.1 to obtain the time-domain representation of the estimated GARMA process.

Following this sequence of steps, for both the simulated GARMA process in Subsection 6.2.1 and the real dataset in Subsection 6.2.2, we output the following:

1. A plot of the time series and of its sample autocorrelation sequence and periodogram.

2. A plot of the evolution of the ESS throughout the SMC procedure, highlighting the iterations where resampling occurred, as well as the threshold for resampling (this is an improved version of the plots in Jasra et al. [2007].)

3. Kernel density estimate (KDE) plots of the estimated posterior and the empirical prior distributions of each parameter (here, calculated with a bandwidth of 0.25).

4. Boxplots of the posterior distributions of each parameter (filtered particles), as well as of the corresponding priors (initial particles).

5. A plot of the estimated SDF, with the locations of the peaks highlighted, and overlayed on top of the sample spectral estimate and the true spectrum, in the case of simulated data.

6. A table with the reconstructed AR and MA coefficients and polynomials (see Appendix B.1).

Additionally, for the real dataset in Section 6.4.2 we also output:

7. The loglikelihood, BIC, and the computing time for each fitted model.

6.2 Demonstration on Data

6.2.1 A Simulated GARMA(3,2,2) Process

We demonstrate the steps outlined above on a GARMA(3,2,2) process of length $N = 500$, which was simulated using the method in Section 5.2 from a standard Gaussian white noise process (i.e. $\sigma_\epsilon^2 = 1$). This process was designed to have three AR roots (one real and two complex conjugate), two MA complex conjugate roots, and two Gegenbauer factors, for a total of 10 parameters to be estimated. Moreover, the values for the parameters were chosen such that the process displays three separate peaks at distinct frequencies – one AR and two Gegenbauer. The true spectrum of this process was plotted in Figure 2.2 to illustrate the difference between the smooth AR and the unbounded Gegenbauer peaks.

For our computation, we used a tapered periodogram with Hann taper as the spectral estimator. We ran the SMC sampler with 1000 particles for 100 iterations with linear likelihood annealing, multinomial resampling, and 5 MCMC iterations per parameter (50 in total) at each resampling, for a total runtime of 9 hours and 44 minutes on a 2.3 GHz Quad-Core Intel Core i5 machine. There were 13 resampling occurrences – 12 when the ESS fell below 500 (50% threshold) and one on the final iteration. The Truncated Normal proposal variance was set at 40% of each parameter's respective range, except for σ_ϵ^2, for which it was set at 1. Finally, since in this subsection we chose to focus on spectrum identification rather than on model selection (which we demonstrate in the next subsection), we assumed the model order $(R_p, C_p, R_q, C_q, k) = (1, 1, 0, 1, 2)$ to be specified correctly. The results of the computation are plotted below; a table with the main summary statistics for the estimated parameters, as well as the estimated AR and MA polynomials can be found in Appendix B.1.

We note from Figures 6.3 and 6.4 below that the most influential parameters, which define the locations of the peaks, namely the AR complex modulus α_{11}^C, the corresponding argument ω_{11}^C, and the Gegenbauer parameters λ_1 and λ_2, are identified with high confidence. As expected, there is some bias in these estimates. This bias is almost unnoticeable for the Gegenbauer peaks, partly due to the fact that these peaks are very well separated, and thus well detected by the sampler, despite Olhede et al. [2004] noting that the bias should be especially large near the G-frequencies, where the periodogram is no longer asymptotically unbiased. The bias for the AR peak is more pronounced; however this may be an issue of convergence, since, as can be seen in the ESS plot in Figure 6.2, the procedure was interrupted before convergence, as it could not be known a priori how many iterations were needed to achieve this. Finally, note that while the Gegenbauer peaks appear bounded in Figure 6.5, they are not, and this is simply an artefact of plotting the SDF on a Fourier grid of limited resolution (these peaks can be seen to be unbounded in Figure 2.2).

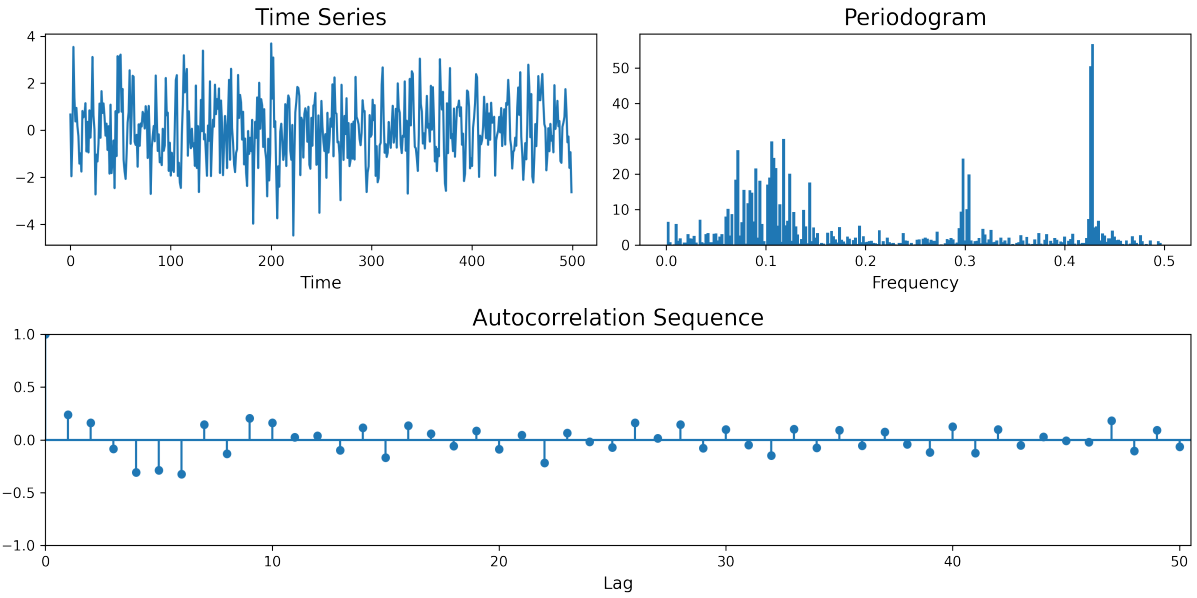


Figure 6.1: Plot of the simulated GARMA(3,2,2) realisation and of its periodogram and autocorrelation sequence.

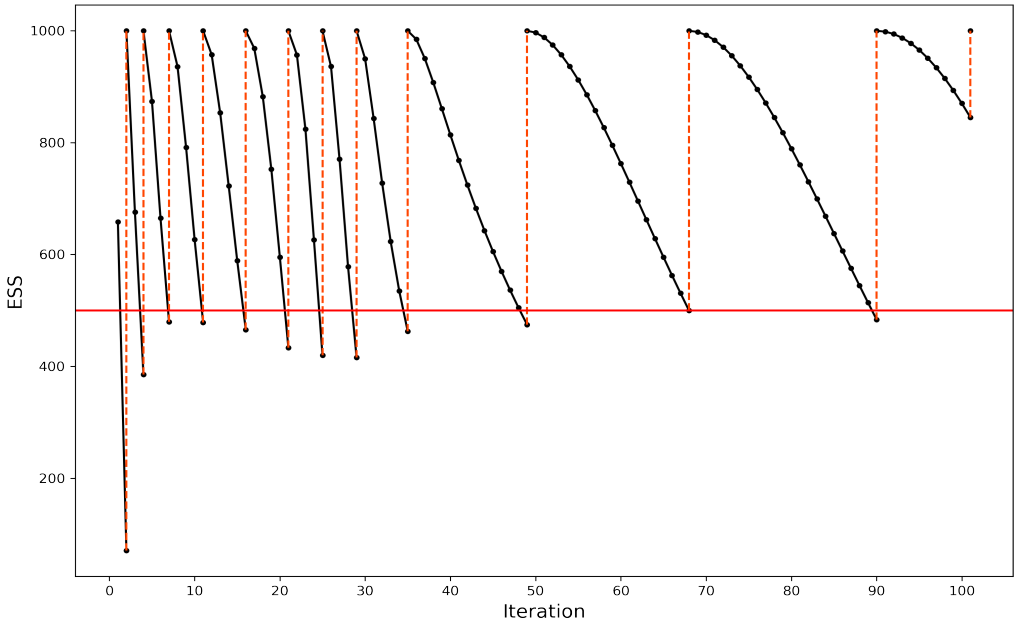


Figure 6.2: ESS plot for the 100 iterations of SMC, with highlighted resampling iterations (red dashed) and resampling threshold (solid red). Note the declining rate at which resampling occurs as the particles converge to the target.

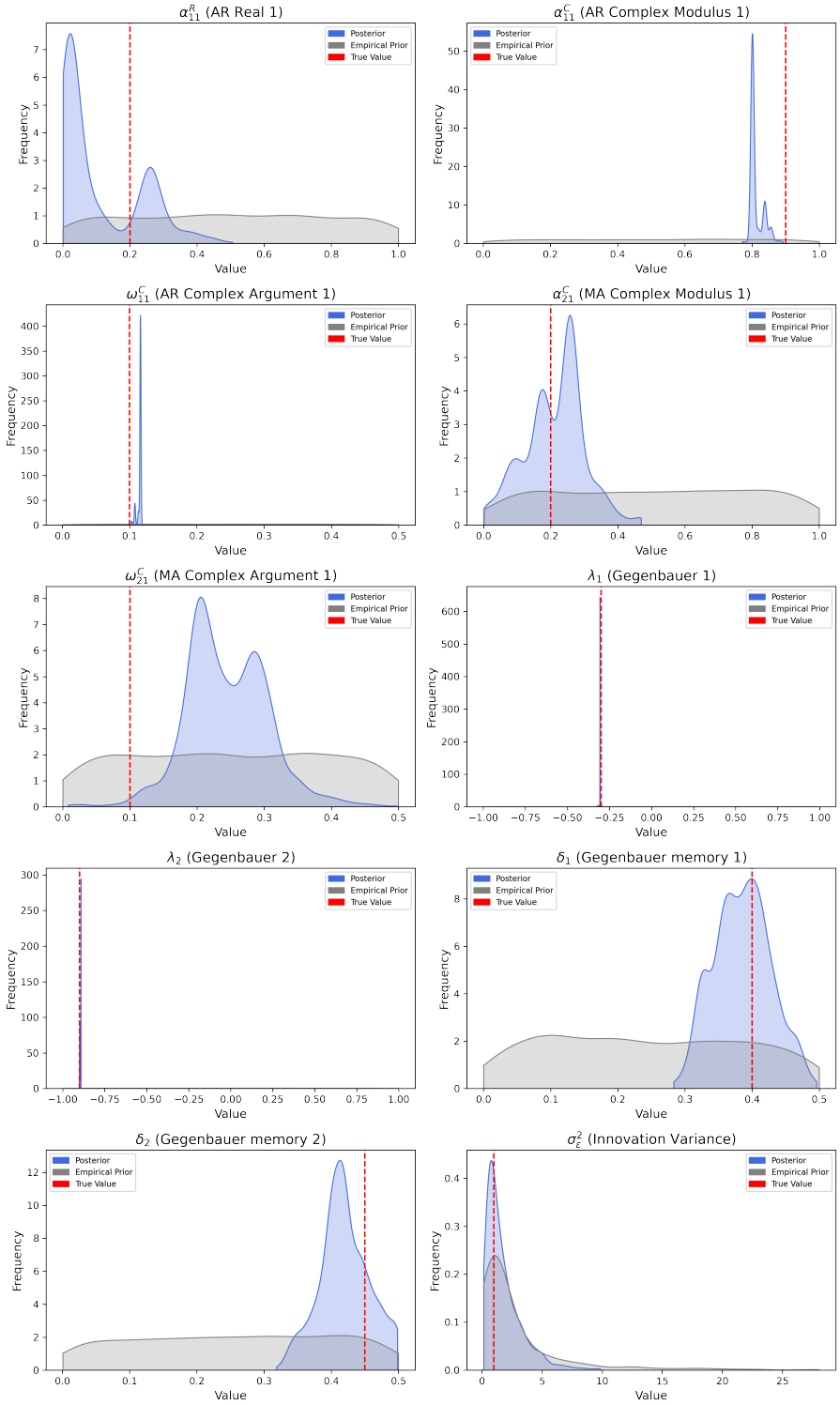


Figure 6.3: KDE plots (bandwidth 0.25) of the posterior samples (blue), with empirical priors (grey), and true parameter values (red).

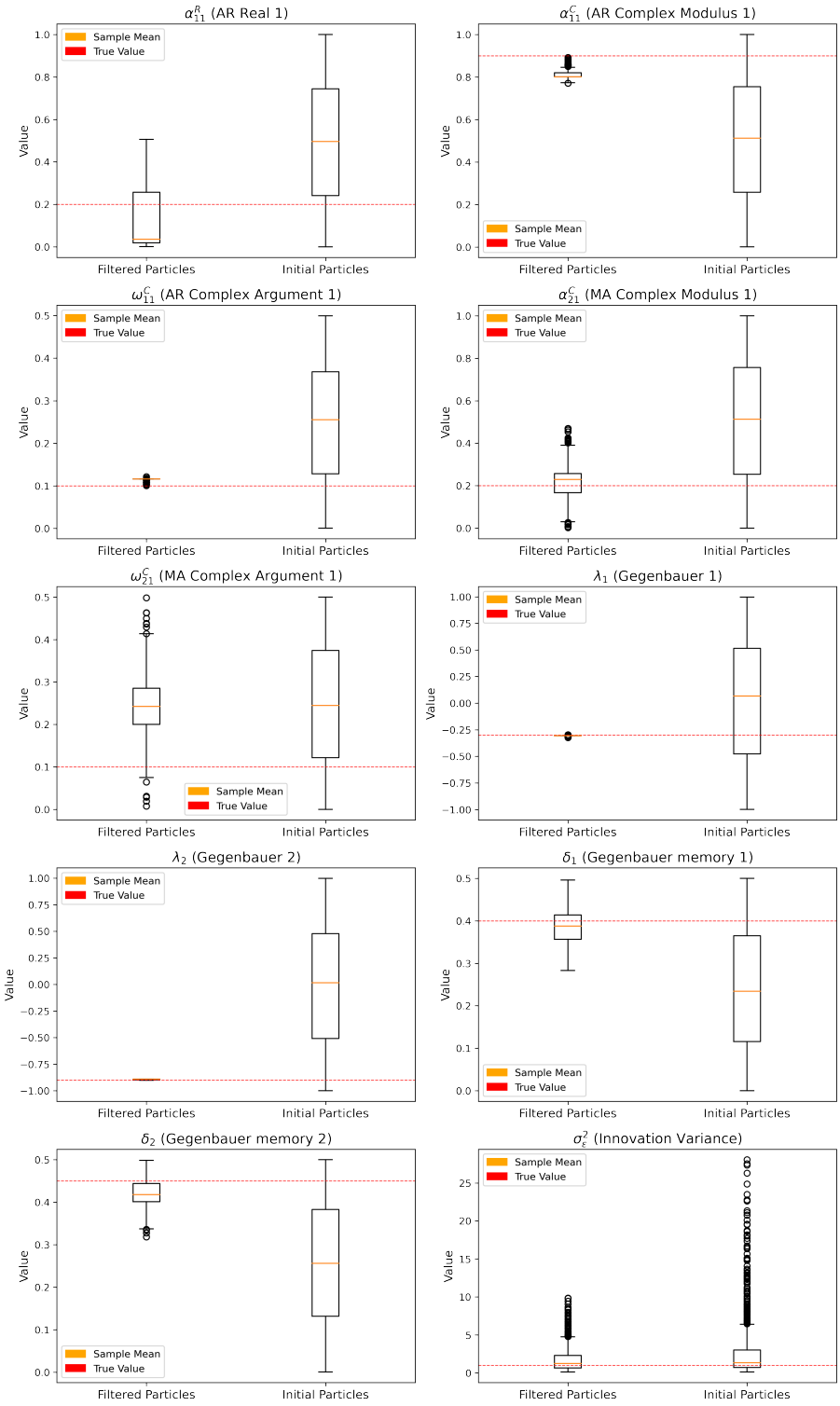


Figure 6.4: Box plots of the filtered (left) and initial (right) particles, with true parameter value (red).

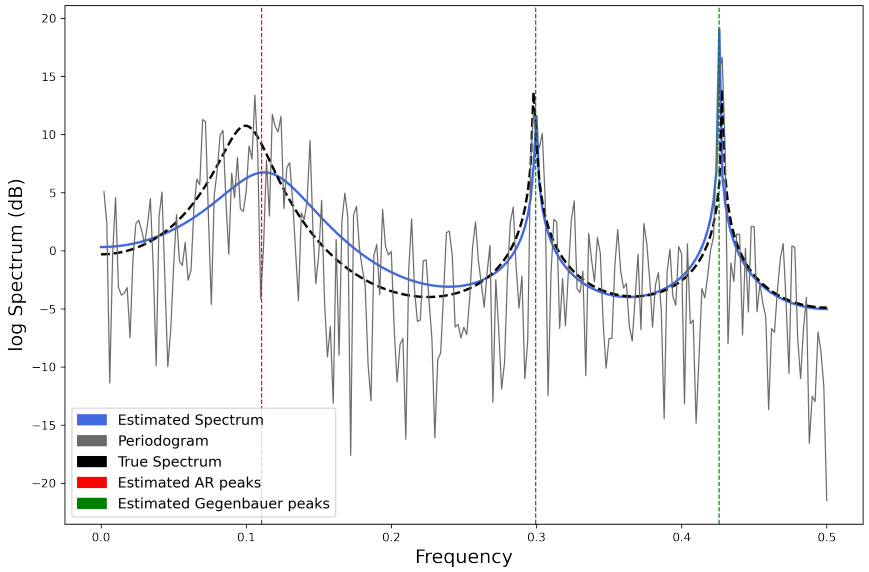


Figure 6.5: Logarithmic scale plot of the estimated spectrum (blue) with true spectrum (black dashed), sample spectral estimate (grey), and spectral peaks (red and green dashed).

6.2.2 U.S. Monthly CPI Inflation Data (1982-2022)

In this subsection, we demonstrate the steps in Subsection 6.1 on a real dataset. The data is the month-on-month United States Consumer Price Index (CPI) inflation data [4] (not seasonally adjusted) from May 1982 until April 2022. The data has length 480 and has been demeaned for the purpose of the analysis. Furthermore, the time series passed the ADF test for stationarity at the 1% level and was thus not differenced.

Since we are now working on real data, we do not assume optimal model specification and instead explore the fit of six different models, summarised in the table below. All models were run with 1000 particles for 100 linear schedule iterations and 5 MCMC updates per parameter, with the same proposal specifications as in Section 6.2.1. Excepting the first* model (3-factor Gegenbauer model using the periodogram), all models used the tapered periodogram with Hann taper as the sample spectral estimate. Note that this first* model is not directly comparable with the other models, since the Whittle likelihood is different. For brevity, we only output the estimated spectrum for each model; summary tables are provided in Appendix B.2.

Table 6.1: Comparison of the six models

	Model Name	(R_p, C_p, R_q, C_q, k)	Parameters	Loglikelihood	BIC	Run Time
1*	Gegenbauer(3) (periodogram)	$(0, 0, 0, 0, 3)$	7	-327.0	702.3	117 min
2	Gegenbauer(3) (tapered)	$(0, 0, 0, 0, 3)$	7	-337.1	722.6	136 min
3	AR(4)	$(0, 2, 0, 0, 0)$	5	-365.5	765.7	122 min
4	ARMA(5,3)	$(1, 2, 1, 1, 0)$	9	-370.4	802.9	330 min
5	Gegenbauer(2)	$(0, 0, 0, 0, 2)$	5	-347.3	729.2	103 min
6	GARMA(1,2,3)	$(1, 0, 1, 1, 2)$	9	-343.6	749.5	344 min

[4] Data sourced from the Bureau of Labor Statistics:
https://beta.bls.gov/dataViewer/view/timeseries/CUUR0000SA0

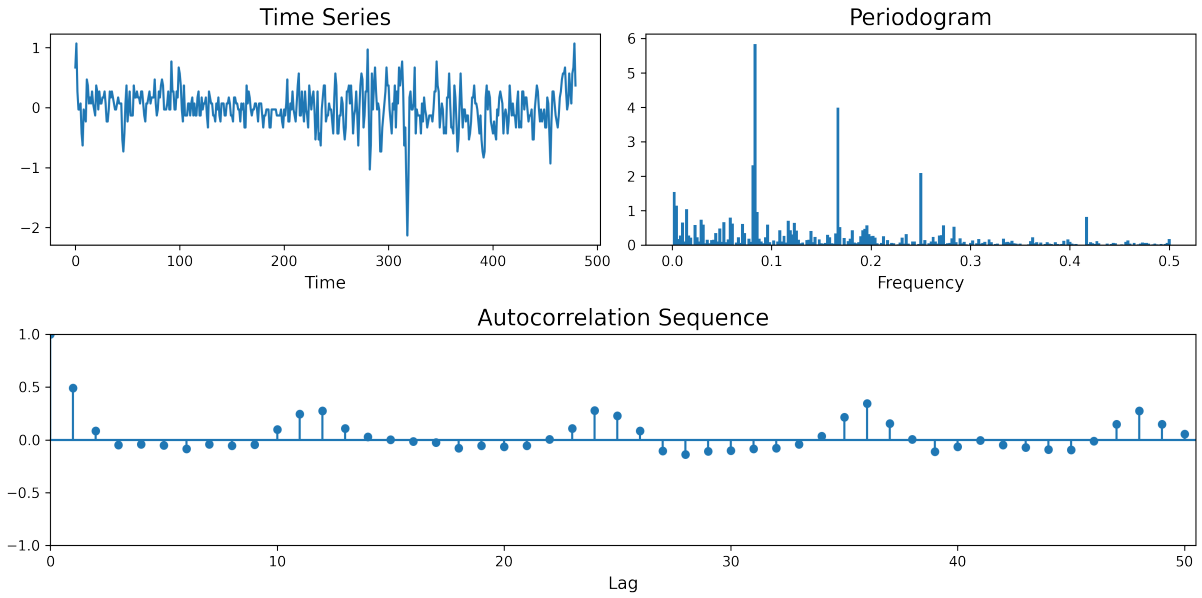


Figure 6.6: Plot of the demeaned CPI inflation time series and of its periodogram and autocorrelation sequence.

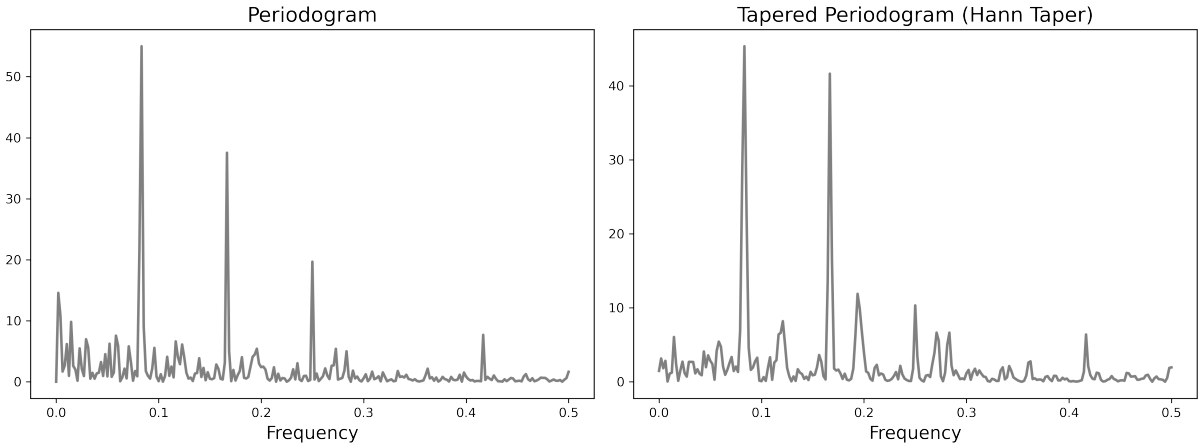
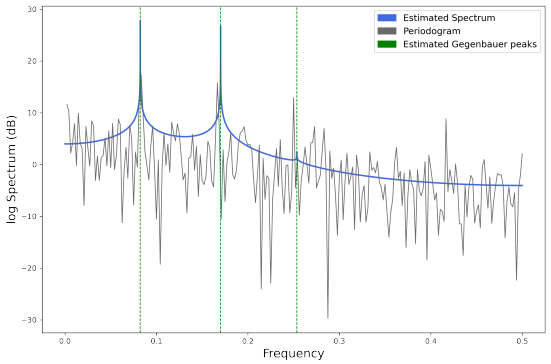
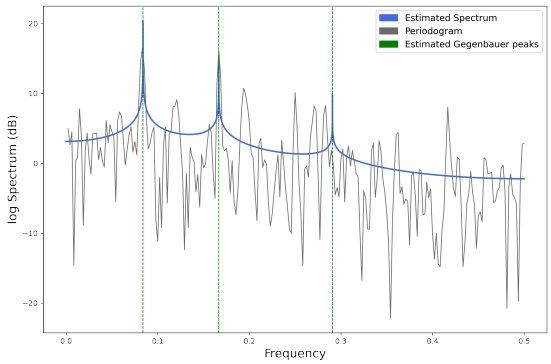


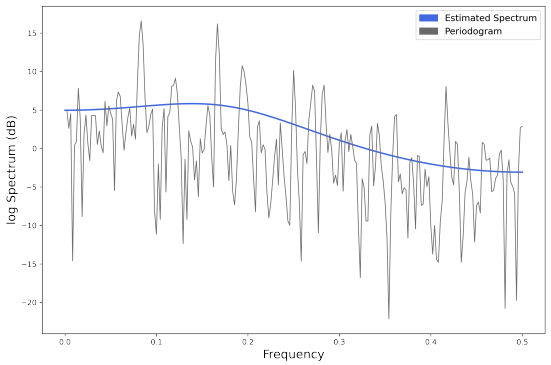
Figure 6.7: Comparison of the periodogram and tapered periodogram for the CPI inflation data.



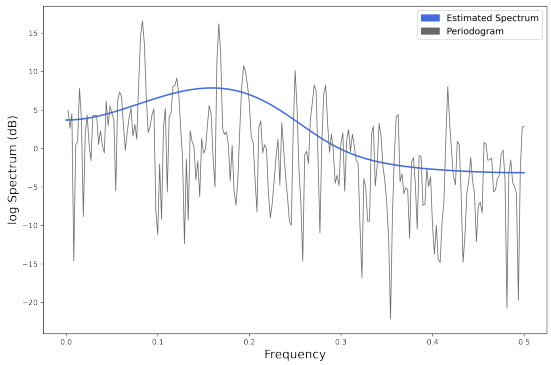
(a) 3-factor Gegenbauer model (periodogram)



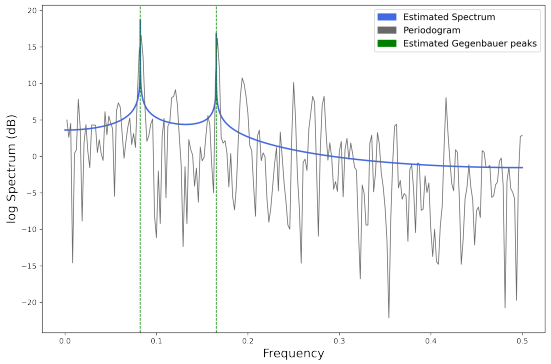
(b) 3-factor Gegenbauer model (tapered)



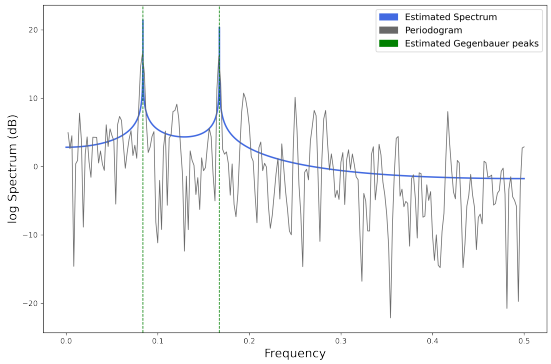
(c) AR(4) model with two pairs of complex roots



(d) ARMA(5,3) model



(e) 2-factor Gegenbauer model



(f) GARMA(1,2,3) model

Figure 6.8: Logarithmic scale plots of the estimated spectra for the six models.

Note from Figure 6.8f that the GARMA(1,2,3) model produces a spectral shape which is nearly indistinguishable from that of the 2-factor Gegenbauer process (Figure 6.8e) and has a slightly larger BIC than this much simpler model, despite taking more than three times as much to run. The marginally smaller BIC of the Gegenbauer(2) model is attributable to the lower number of parameters, since the loglikelihood is slightly higher for the GARMA(1,2,3) model. Thus, in practice we would choose the simpler model due to this positive efficiency-accuracy tradeoff. Moreover, note that the AR(4) and ARMA(5,3) models both fail to identify any peaks in the spectrum, and merely detect a large frequency range where the spectral power is higher. This illustrates the necessity of using Gegenbauer factors to accurately model this specific example and demonstrates the advantage of the GARMA approach over the simpler ARMA model.

7 Conclusions and Further Developments

7.1 Review and Conclusions

In this work we demonstrated the methodology for performing frequency-domain Bayesian inference for the GARMA model using Sequential Monte Carlo (SMC) Sampling. We first introduced the GARMA process, as a model that extends the short-range seasonal dependence modelled by the ARMA process to stationary time series which also exhibit long-memory. Next, we saw how analysing time series in the frequency domain can provide better insight into seasonal patterns than time-domain analysis. We then established a framework for Bayesian inference in the frequency domain using the Whittle likelihood. Next, we motivated SMC and showed how it can be applied to simulate from the posterior distributions of the GARMA model parameters. Finally, after briefly discussing the problem of simulating processes with given spectra, we proceeded to demonstrate the methodology on simulated, as well as real data.

Due to time and length constraints, we had to focus on establishing the core methodology, and as a result not enough attention could be devoted to more theoretical aspects, such as convergence, variance estimation, and model selection. Thus, there remain several important further extensions, of which we present several below. These extensions could form the basis of promising future research.

7.2 Hyperparameter Optimisation and Computational Improvements

Having devised an implementation of SMC in the frequency domain (see Sections 4 and 6), we now ask how we could improve the accuracy and efficiency of this implementation. One way is by tuning the hyperparameters of the SMC algorithm. We list these hyperparameters below and discuss how one may optimise them:

1. T, the number of SMC iterations or the number of bridging distributions. Ideally, this number would be chosen as large as possible to improve the tempering effect of likelihood annealing. However, too large a value could also delay the convergence of the sample, rendering the procedure inefficient. For a detailed discussion of this, see Neal [2001].

2. R_t, the number of MCMC iterations at the t-th bridging density. A possible way to tune this parameter would be to decide based on the change in entropy of the likelihood. This could be done by measuring the Kullback-Leibler divergence between the likelihood before and after each MCMC iteration and stopping once this decreases below a pre-defined threshold. Naturally, this creates an adaptive procedure, however the problem remains in choosing the optimal value for this threshold, which could potentially also be different for each bridging density.

3. The type of spacing in the sequence of exponents $\{\gamma_t\}$. Choosing these parameters is largely a modelling choice. Experimentation has shown that geometric spacing leads to faster convergence, with the resampling steps being more concentrated towards the end of the iterative cycle, when more of the likelihood is introduced at each iteration. On the other hand, linear spacing, as used in the two examples in Subsections 6.2.1 and 6.2.2, leads to a better exploration of the sample space, since the resampling and MCMC steps are performed at more regular intervals, including in the earlier iterations, when the particles have not yet converged to their final distribution.

4. The variance of the MCMC Truncated Normal proposals. This aspect was discussed earlier in Subsection 4.8 and remains an important open problem. We make the additional note that the procedure could be improved using adaptive proposals. In this case, we would reduce the proposal variance for the parameters which have been well identified, thus speeding the convergence of the algorithm. Deciding whether the distribution of a parameter has already converged to the corresponding marginal posterior may prove complicated, however. Moreover, if done too early in the iteration cycle, adaptive proposals increase the risk of convergence to local modes. Adaptive proposals are discussed in e.g. Jasra et al. [2007], who suggest dampening the proposal variance at each iteration t by a factor of $1/(1+\gamma_t) \in [1, 2]$, where γ_t is the temperature at iteration t, as in AIS.

5. The parameters of the prior distributions. In our case, we chose uniform priors on all parameters except the white noise variance, so there were no parameters to choose. For the Inverse Gamma prior on σ_ϵ^2, one could vary the shape and scale parameters α and β, while keeping in mind that a diffuse prior (lower value of α) is preferable if one wants this prior to be uninformative. Finally, one could choose different classes of priors. So far we have not found the issue of prior optimisation to be particularly emphasised in the literature in the context of MCMC, with some authors (see McCoy and Stephens [2004]) noting that posterior estimates are generally robust to the choice of prior.

A general way to tune these parameters would be through cross-validation. In the simplest approach, one would try to optimize the parameters independently, assessing performace based on a given metric (such as Monte Carlo variance or accuracy when training on simulated data) when varying a given hyperparameter over a specified grid, while holding the other parameters constant at some arbitrary values; these arbitrary values could later be replaced with their respective optima, as we more hyperparameters are optimised. There are, however, limitations to this approach, since the assumption of independence may not be justified, i.e. the joint optimal values for the hyperparameters may not coincide with the individual optimal values. Moreover, optimality cannot be characterised through a single metric, such as variance or running time, since in practice we would strive for a balance of these. Finally, this procedure would be computationally very expensive due to the need of running the SMC algorithm many times.

Additional computational gains can be obtained by parallelising the SIS procedure in the SMC algorithm and by making changes to the resampling procedure. Parallelisation has a direct impact on the computational time by distributing the computations to multiple CPU cores, which then execute in parallel. For details on this adaptation, see Chopin [2002] and Naesseth et al. [2019]. The resampling procedure also influences the efficiency of the algorithm, since it affects the variance of the weighs. In particular, Jasra et al. [2007] note that multinomial resampling (as used in our implementation) leads to high variability in the number of replicates of each particle, and consequently in the weights of the resampled particles. The two most common alternatives to multinomial resampling are residual resampling and stratified resampling. Jasra et al. [2007] list several additional methods, including stochastic remainder selection and systematic resampling. Due to space constraints, we do not detail these here and instead refer the interested reader to Petris et al. [2009], who discusses residual resampling, and to Cappé et al. [2007] and Naesseth et al. [2019], who discuss stratified resampling.

7.3 Demodulation

Having mentioned in Subsections 3.4 and 3.5 tapering and the convolution idea in (Sykulski et al. [2019]) as two methods for reducing the bias in the Whittle likelihood, we now turn to a complementary approach, known as demodulation. This approach, introduced in Olhede et al. [2004] and further expanded in McCoy et al. [2007] consists in shifting and reordering the Fourier grid, so that the Fourier frequencies used to calculate the discretised Whittle likelihood are better aligned with the peaks in the spectrum. This has the effect of reducing the bias near the spectral poles, which is particularly important for the Gegenbauer peaks, since the periodogram is no longer asymptotically unbiased at frequencies close to these poles (see Olhede et al. [2004]). Indeed, we saw in Figure 6.5 that unless the resolution of the frequency grid is high enough, the unbounded Gegenbauer peaks will appear bounded, since the values of the SDF at the G-frequencies will

be calculated as an interpolation of the values at the two nearest Fourier frequencies on either side. While we can plot the estimated spectrum on a finer grid after the parameters have been estimated, the level of detail in the periodogram or any other sample spectral estimate is limited by the length of the available data, and consequently so is the accuracy of the discretisation of the Whittle likelihood. Therefore, when obtaining more data is not possible, the natural solution is to rearrange the frequencies. Specifically, Olhede et al. [2004] suggest reordering and relabelling the Fourier frequencies based on their absolute distance to the nearest pole and then defining and computing at each frequency the relative bias of the periodogram, i.e. the expected value of the ratio of the periodogram and the theoretical spectrum of the chosen model. Olhede et al. [2004] establish the asymptotic properties of this relative bias, and McCoy et al. [2007] extend this result by devising a procedure for shifting the Fourier grid so as to minimise the relative bias. Finally, for a demonstration on data and for a comparative analysis of the effect of demodulation, we refer the reader to McCoy et al. [2007].

7.4 Model Selection and Trans-Dimensional SMC

We briefly discussed model selection using Bayes Factors and the BIC in the context of AIS in Subsection 4.6, and demonstrated this on the CPI data in Subsection 6.2.2. However, as we saw for this example, choosing a model using one of these metrics requires running and comparing several models, which is computationally expensive, especially if one has no prior reason to favour certain parametrisations, e.g. if there are no obvious peaks in the spectrum or if the peaks are located at close frequencies, causing them to merge. Thus, one may want to perform automatic model selection at some stage of the SMC algorithm, instead of during the post-processing of the resulting posterior sample.

The standard way for performing such trans-dimensional or multi-model inference is to replace the standard MCMC kernel with a Reversible-Jump MCMC kernel (RJMCMC). The core idea of this modification, due to Green [1995], is to incorporate a Jacobian term into the Metropolis-Hastings acceptance ratio. This term, which accounts for the change in dimensionality, relies on constructing bijections between each parameter space. The number of such bijections depends on the dimension of the largest parameter space. Specifically, if this space has dimension m, the original method in Green [1995] requires the specification of $\binom{m}{2}$ such bijections, which limits the scalability of this approach. Barker and Link [2013] show that the number of bijections can be reduced to m by defining a universal parameter ψ of dimension m (called the 'pallette'), such that all candidate models are augmented to the dimension of ψ (for an illustration of this, see Gelling et al. [2017]). Thus, models of lower dimensionality correspond to some entries of ψ being zero. Nonetheless, even with this modification, constructing these bijections and computing the Jacobian efficiently remains technically challenging in practice. For an application of trans-dimensional kernels to SMC sampling, we refer the reader to Del Moral et al. [2006]. For an application of RJMCMC to frequency-domain inference, see McCoy and Stephens [2004]. A chapter on RJMCMC can also be found in Liang et al. [2011].

7.5 Forecasting using the GARMA Model

We conclude the discussion on further developments with forecasting. Having presented in Section 5 two methods for simulating a GARMA process, it is natural to also ask how one could forecast from an estimated GARMA model. One could view the problem of forecasting as the inverse of simulation. When simulating from the functional form, we suggested in Subsection 5.3 filtering a white noise process through the impulse response filter given by the infinite MA representation of the GARMA process, obtained from Wold's theorem. In contrast, for forecasting, we are building on past values of the data. Thus, in general, we would need the infinite AR representation of the model in order to construct future values recursively. The existence of this representation is equivalent to the invertibility of the GARMA process. Since the MA component is assumed invertible, this in turn is equivalent to being able to write the Gegenbauer factors in autoregressive form. In order to do this, Woodward et al. [1998] extended the forecasting method in Box et al. [1976] for ARMA models to the k-factor Gegenbauer model. The core idea of this approach, which is known as the

Box-Jenkins method, is to equate:

$$\sum_{j=0}^{\infty} \mu_j B^j = \Theta^{-1}(B)\Phi(B) \sum_{i=1}^{k} (1 - 2\lambda_i B + B^2)^{\delta_i}$$
$$= \Theta^{-1}(B)\Phi(B) \sum_{i=1}^{k} \left(\sum_{l=0}^{\infty} C_l^{-\delta_i}(\lambda_i) B^l \right)$$

and then to identify the coefficients μ_j, $j = 0, 1, \ldots, n$ from a finite truncation up to a given length n. Once these coefficients have been calculated, one can forecast $h > 0$ steps ahead of a given time t_0 as follows:

$$\hat{Y}_{t_0+h} = - \sum_{j=1}^{t_0-1+h} \mu_j Y_{t_0+h-j}.$$

We note that this method is as general as the method of simulation from Section 5.3, in that it does not assume the Gaussianity of the process $\{Y_t\}$. However, while the Box-Jenkins method allows forecasting an arbitrary number of steps ahead, as in the case of simulation, it may prove inefficient due to the computational costs associated with inverting the MA polynomial and obtaining a sufficiently accurate truncation of the Gegenbauer polynomials. The computational cost can be reduced if one limits the scope to obtaining one-step-ahead forecasts, and then updating the forecasts when new data becomes available. Indeed, this is the approach in McCoy and Stephens [2004], who generate one-step-ahead forecasts within the sampling procedure, instead of as part of the post-processing of the sample. This has the additional advantage of not needing to obtain the time-domain representation of the estimated GARMA process, so that one can perform the entire procedure in the frequency domain. For a more in-depth theoretical discussion of forecasting with the k-factor Gegenbauer process, as well as an application to data, see Ferrara and Guégan [2001].

References

R. J. Barker and W. A. Link. Bayesian multimodel inference by RJMCMC: a Gibbs sampling approach. *The American Statistician*, 67:150 – 156, 2013.

G. E. P. Box, G. M. Jenkins, and H. Day. *Time Series Analysis: Forecasting and Control*. Holden-Day series in time series analysis and digital processing. Holden-Day, 1976. ISBN 9780816211043.

O. Cappé, S. J. Godsill, and É. Moulines. An overview of existing methods and recent advances in Sequential Monte Carlo. *Proceedings of the IEEE*, 95:899–924, 2007.

N. Chopin. A sequential particle filter method for static models. *Biometrika*, 89:539–552, 2002.

N. Chopin and O. Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer Series in Statistics. Springer International Publishing, 2020. ISBN 9783030478452.

D. Crisan and A. Doucet. Convergence of Sequential Monte Carlo Methods. Working Paper, 2007. Available at https://www.stats.ox.ac.uk/~doucet/crisain_doucet_convergenceofSMC2000.pdf.

P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

G. S. Dissanayake, M. S. Peiris, and T. Proietti. Fractionally differenced Gegenbauer processes with long memory: a review. *Statistical Science*, 2018.

A. Doucet, A. Smith, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Information Science and Statistics. Springer New York, 2001. ISBN 9780387951461.

L. Ferrara and D. Guégan. Forecasting with k-factor Gegenbauer processes: theory and applications. *Journal of Forecasting*, 20:581–601, 2001.

N. Gelling, M. R. Schofield, and R. J. Barker. R package rjmcmc: the calculation of posterior model probabilities from MCMC output. R Package Vignette, 2017. Available at https://mran.microsoft.com/snapshot/2017-09-30/web/packages/rjmcmc/vignettes/rjmcmcVig.pdf.

A. Gelman and X-L. Meng. Simulating normalizing constants: from Importance Sampling to Bridge Sampling to Path Sampling. *Statistical Science*, 13:163–185, 1998.

J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57: 1317–1339, 1989.

W. R. Gilks and C. Berzuini. Following a moving target—Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63, 2001.

W.R. Gilks, S. Richardson, and D. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC Interdisciplinary Statistics. Taylor & Francis, 1995. ISBN 9780412055515.

N. J. Gordon, D. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140:107–113(6), April 1993.

H. L. Gray, N. F. Zhang, and W. A. Woodward. On generalized fractional processes. *Journal of Time Series Analysis*, 10:233–257, 1989.

P. J. Green. Reversible Jump Markov Chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

J. R. M. Hosking. Fractional differencing. *Biometrika*, 68(1):165–176, 1981.

G. Huerta and M. West. Priors and component structures in autoregressive time series models. *Journal of the Royal Statistical Society*, 61:881–899, 1999a.

G. Huerta and M. West. Bayesian inference on periodicities and component spectral structure in time series. *Journal of Time Series Analysis*, 20:401–416, 1999b.

R. Hunt, S. Peiris, and N. C. Weber. Estimation methods for stationary Gegenbauer processes. *Statistical Papers*, 2022.

C. Hurvich. Whittle's approximation to the likelihood function. NYU Stern Handout, 2002. Available at https://pages.stern.nyu.edu/~churvich/TimeSeries/Handouts/Whittle.pdf.

A. Jasra, D. A. Stephens, and C. C. Holmes. On population-based simulation for static inference. *Statistics and Computing*, 17:263–279, 2007.

S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671 – 680, 1983.

A. Lee and N. Whiteley. Variance estimation in the particle filter. *Biometrika*, 2018.

D. Li, A. Clements, and C. Drovandi. Efficient Bayesian estimation for GARCH-type models via Sequential Monte Carlo. *Econometrics and Statistics*, 19:22–46, 2021.

F. Liang, C. Liu, and R. Carroll. *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples*. Wiley Series in Computational Statistics. Wiley, 2011. ISBN 9781119956808.

J.S. Liu, J.S. Liu, S. Liu, and S.L. Jun. *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. Springer New York, 2001. ISBN 9780387952307.

E. J. McCoy and D. A. Stephens. Bayesian time series analysis of periodic behaviour and spectral structure. *International Journal of Forecasting*, 20:713–730, 2004.

E. J. McCoy, S. C. Olhede, and D. A. Stephens. Non-regular likelihood inference for seasonally persistent processes. *arXiv: Methodology*, 2007.

C. A. Naesseth, F. Lindsten, and T Bo Schön. Elements of Sequential Monte Carlo. *Foundations and Trends in Machine Learning*, 12(3):307–392, 2019.

R. M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6:353–366, 1996.

R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.

S. C. Olhede, E. J. McCoy, and D. A. Stephens. Large-sample properties of the periodogram estimator of seasonally persistent processes. *Biometrika*, 91:613–628, 2004.

D. B. Percival. Simulating Gaussian random processes with specified spectra. *Computing Science and Statistics*, 24:534–538, 1992.

G. Petris, S. Petrone, and P. Campagnoli. *Dynamic Linear Models with R*. Use R! Springer New York, 2009. ISBN 9780387772387.

S. Rao and J. Yang. Reconciling the Gaussian and Whittle likelihood with an application to estimation in the frequency domain. *The Annals of Statistics*, 2021.

C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer New York, 2005. ISBN 9780387212395.

L. South, A. N. Pettitt, and C. C. Drovandi. Sequential Monte Carlo samplers with independent Markov Chain Monte Carlo proposals. *Bayesian Analysis*, 2019.

A. M. Sykulski, S. C. Olhede, A. Guillaumin, J. M. Lilly, and J. J. Early. The debiased Whittle likelihood. *Biometrika*, 2019.

P. Whittle. The analysis of multiple stationary time series. *Journal of the royal statistical society series b-methodological*, 15:125–139, 1953.

W. A. Woodward, Q. C., and H. L. Gray. A k-factor GARMA long-memory model. *Journal of Time Series Analysis*, 19, 1998.

H. Wu and Peiris S. An introduction to vector Gegenbauer processes with long memory. *Stat*, 7(1), 2018.

Appendices

A Mathematical Appendix

A.1 Notation

Below we give a list of the main notation used in this paper:

1. Time series notation:

   - t is the time, τ is the time lag, and f denotes the frequency. N is the length of the time series and $(f_j)_{j=1,\dots,N}$ are the Fourier frequencies. $\tilde{\omega}_{0j}$, $\tilde{\omega}_{1j}$, $\tilde{\omega}_{2j}$ denote the frequencies corresponding to the Gegenbauer peaks, the AR peaks, and the MA valleys, respectively.
   - $\{X_t\}$ and $\{Y_t\}$ denote discrete time stochastic processes or realisations thereof. The former is usually used in a theoretical context, and the second to denote a GARMA process or the time data assumed to be a realisation from this model (sometimes we also use y to denote this).
   - $\{\epsilon_t\}$ denotes a Gaussian white noise process with variance σ_ϵ^2.
   - p, q, k are the orders of the AR, MA, and Gegenbauer components. R_p, C_p denote the number of real and complex AR reciprocal roots, respectively, and similarly we use R_q, C_q for the MA roots.
   - $(\phi_j)_{j=1,\dots,p}$, $(\theta_j)_{j=1,\dots,q}$ are the coefficients of the AR and MA polynomials $\Phi(\cdot)$ and $\Theta(\cdot)$.
   - B is the backward shift operator.
   - $s_\tau = s(\tau)$ and $S_X(f)$ denote the autocovariance sequence (ACVS) and the spectral density (SDF) of the time series $\boldsymbol{X} = (X_1, \dots, X_N)$ with mean $\overline{X}$. $\rho(\tau)$ denotes the autocorrelation sequence. $\hat{s}_p(\tau)$ denotes the biased estimator of the ACVS, $\hat{S}_p(f)$ denotes the periodogram, and $\hat{S}_d(f)$ a tapered spectral estimator. $(Z_j = \hat{S}_p(f_j))_{j=1,\dots,N}$ is the sequence of periodogram values at the Fourier frequencies.
   - $\xi_1 = \left(\xi_1^R, \xi_1^C, \overline{\xi_1^C}\right)$ and $\xi_2 = \left(\xi_2^R, \xi_2^C, \overline{\xi_2^C}\right)$ denote the AR and MA real and complex reciprocal roots. $\alpha_1 = (\alpha_{11}^R, \dots, \alpha_{1R_p}^R, \alpha_{11}^C, \dots, \alpha_{1C_p}^C)$ and $\alpha_2 = (\alpha_{21}^R, \dots, \alpha_{2R_q}^R, \alpha_{21}^C, \dots, \alpha_{2C_q}^C)$ denote the moduli of these roots. $\omega_1 = (\omega_{11}^R = 0, \dots, \omega_{1R_p}^R = 0, \omega_{11}^C, \dots, \omega_{1C_p}^C)$ and $\omega_2 = (\omega_{21}^R = 0, \dots, \omega_{2R_q}^R = 0, \omega_{21}^C, \dots, \omega_{2C_q}^C)$ denote the corresponding arguments. $\lambda = (\lambda_1, \dots, \lambda_k)$ and $\delta = (\delta_1, \dots, \delta_k)$ denote the parameters of the k Gegenbauer components.

2. Bayesian Inference notation:

   - $\boldsymbol{\eta}$ is the vector of $m = R_p + 2C_p + R_q + 2C_q + 2k + 1$ model parameters, as given in (3.2.1).
   - $p(\boldsymbol{\eta})$ is the prior on $\boldsymbol{\eta}$, $L(\boldsymbol{\eta}|y)$ is the likelihood given data y with $l(\boldsymbol{\eta}|y)$ the corresponding loglikelihood and $p(\boldsymbol{\eta}|y)$ is the posterior. $l_W(\eta)$ is the Whittle likelihood and $l_{DW}(\eta)$ its discretisation.

3. SMC notation:

   - T is the number of SMC iterations or bridging densities and $R_t = R$ is the number of MCMC iterations at the t-th SMC iteration, if resampling happens at this iteration. N is the number of particles or samples of the estimated parameter $\boldsymbol{\eta}$.
   - $(\pi_t)_{t=1,\dots,T}$ is a sequence of T bridging or interpolating densities $\pi_t(\boldsymbol{\eta}|y) = \pi(\boldsymbol{\eta})L(\boldsymbol{\eta}|y)^{\gamma_t}$, with $\pi_0 = \pi(\boldsymbol{\eta})$ being the prior and $\pi_T = \pi(\boldsymbol{\eta}|y)$ the target (the posterior). We refer to the sequence $0 = \gamma_0 < \gamma_1 < \cdots < \gamma_T = 1$ as the temperatures, schedule, or spacing.
   - $\boldsymbol{\eta}_t = (\boldsymbol{\eta}_t^i)_{i=1,\dots,N}$ are the particles at the t-th SMC iteration, with $w_t = (w_t^i)_{i=1,\dots,N}$ and $\tilde{w}_t = (\tilde{w}_t^i)_{i=1,\dots,N}$ the corresponding normalised and unnormalised importance weights, respectively.

A.2 Distributions Used

In Subsection 3.3 we discussed the prior specification on each parameter and in Subsection 4.8 we proposed the use of a Truncated Normal proposal for evolving the particles within the MCMC kernel. Firstly, with the exception of σ_ϵ^2, all parameters are limited to a finite range. As such, both the prior and the proposal must have the same bounded support. For the prior, the Uniform distribution is a simple and uninformative choice, as it assigns equal probability to every value in its range and has a constant probability density function. For the proposal, the Truncated Normal distribution, due to its similar parametrisation to the Gaussian distribution, makes it easy and intuitive to sample new candidates in a range centred at the current estimate with a probability directly controlled by the variance of this proposal. In the case of σ_ϵ^2, since the range is only bounded from below by 0, but lower values are favoured, it is natural to employ a diffuse, heavy-tailed Inverse Gamma distribution and a Truncated Normal proposal with zero lower bound and infinite upper bound centred at the current estimate. Below, we give the probability density functions of these distributions:

- Uniform Distribution:

$$X \sim Uniform(a, b) \quad \text{if} \quad f_X(x) = \frac{1}{b-a} \quad \text{for} \quad a \leq x \leq b$$

  and 0 otherwise, where $-\infty < a < b < +\infty$.

- Truncated Normal Distribution:

$$X \sim TN(\mu, \sigma^2, a, b) \quad \text{if} \quad f_X(x) = \frac{1}{\sigma} \frac{\phi\left(\frac{x-\mu}{\sigma}\right)}{\Phi\left(\frac{b-\mu}{\sigma}\right) - \Phi\left(\frac{a-\mu}{\sigma}\right)} \quad \text{for} \quad a \leq x \leq b$$

  and 0 otherwise, where $-\infty \leq a < b \leq +\infty$ and $\mu \in \mathbb{R}$ and $\sigma^2 \geq 0$ denote the location and scale parameters, respectively. Here, $\phi(\cdot)$ and $\Phi(\cdot)$ denote the probability density and the cumulative density functions of a standard Gaussian ($N(0, 1)$) random variable:

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \quad \text{for} \quad x \in \mathbb{R} \quad \text{and} \quad \Phi(x) = \int_{-\infty}^{x} \phi(y) dy.$$

  The proposal considered for the innovation variance σ_ϵ^2 simply takes $a = 0$ and $b = +\infty$. Note that this is different from the Half Normal distribution, which is a particular case of the Folded Normal distribution, folded at the origin. Also important to note is that when $a \leq \mu \leq b$, which is always the case in the context of our problem, the mode of the Truncated Normal distribution is μ. This property makes the Truncated Normal a suitable way to generate proposals centred around a given value.

- Inverse Gamma Distribution:

$$X \sim IG(\alpha, \beta) \quad \text{if} \quad f_X(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{1}{x}\right)^{\alpha+1} \exp\left(-\frac{\beta}{x}\right) \quad \text{for} \quad x > 0$$

  and 0 otherwise, where $\alpha > 0$, $\beta > 0$ denote the shape and scale parameters, respectively. The Inverse Gamma Distribution is the distribution of the reciprocal of a Gamma distributed random variable, i.e. if $Y \sim Gamma(\alpha, \beta)$, then $X = 1/Y \sim IG(\alpha, \beta)$.

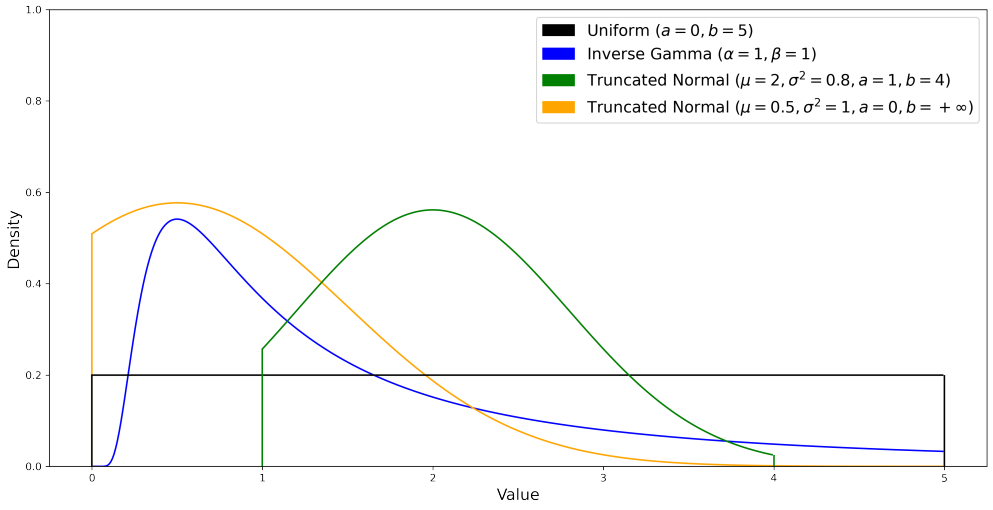


Figure A.1: Examples of the used probability densities.

A.3 Tapering

In Subsection 3.4 we introduced tapering or windowing as a way to reduce the bias in the periodogram. We then proceeded to use tapering with a Hann taper in the two examples in Section 6. Here, we briefly explain the effect of tapering and illustrate this effect for the Hann taper against the rectangular taper used to compute the periodogram.

A data taper defined over an interval $[a, b]$ is a function $h_t = h(t)$ which usually has the following properties:

1. h_t is symmetric about $(a + b)/2$, where it also achieves its maximum value.
2. $h_t = 0$, $\forall t \notin [a, b]$.
3. h_t is often given in square-normalised form, i.e. such that $\int_a^b h^2(t)dt = 1$. For sampled processes (discrete time series) X_t, we usually consider a partition of length N of the interval $[a, b]$, so that $\sum_{t=1}^{N} h_t^2 = 1$.

The effect of tapering is to narrow the frequency response of the data, when one is interested in the spectrum only within a certain frequency interval. Thus, when the original time series $\{X_t\}$ is multiplied componentwise through the data taper $\{h_t\}$, the spectral power is reduced outside the range of the taper and dampened in the regions where the taper has low values, thus reducing the spectral leakage from the frequencies of interest. Recall from Subsections 3.4 and 3.5 that the direct spectral estimator with taper h_t of the true spectrum $S_X(f)$ and its expectation are given by:

$$\hat{S}_d(f) = \left| \sum_{t=1}^{N} h_t X_t e^{-2\pi i f t} \right|^2, \ f \in [-1/2, 1/2] \quad \text{and} \quad \mathbb{E}\{\hat{S}_d(f)\} = \int_{-1/2}^{1/2} \mathcal{H}(f - \nu) S_X(\nu) d\nu$$

where $\mathcal{H}(f) = \left| \sum_{t=1}^{N} h_t e^{-2\pi i f t} \right|^2$, $f \in [-1/2, 1/2]$ is the DFT of the taper h_t.

The default taper is the rectangular (also known as boxcar or Dirichlet) taper:

$$h_t^0 = \begin{cases} \frac{1}{\sqrt{N}} & 1 \leq t \leq N \\ 0 & \text{otherwise} \end{cases}$$

An alternative taper is the Hann taper, whose discretised form with amplitude 1 is given by:

$$h_t^H = \begin{cases} \frac{1}{2}\left[1 - \cos\left(\frac{2\pi t}{N}\right)\right] = \sin^2\left(\frac{\pi t}{N}\right) & 0 \leq t \leq N \\ 0 & \text{otherwise} \end{cases}$$

The comparative effect of these two windows is illustrated in Figure A.2 below.

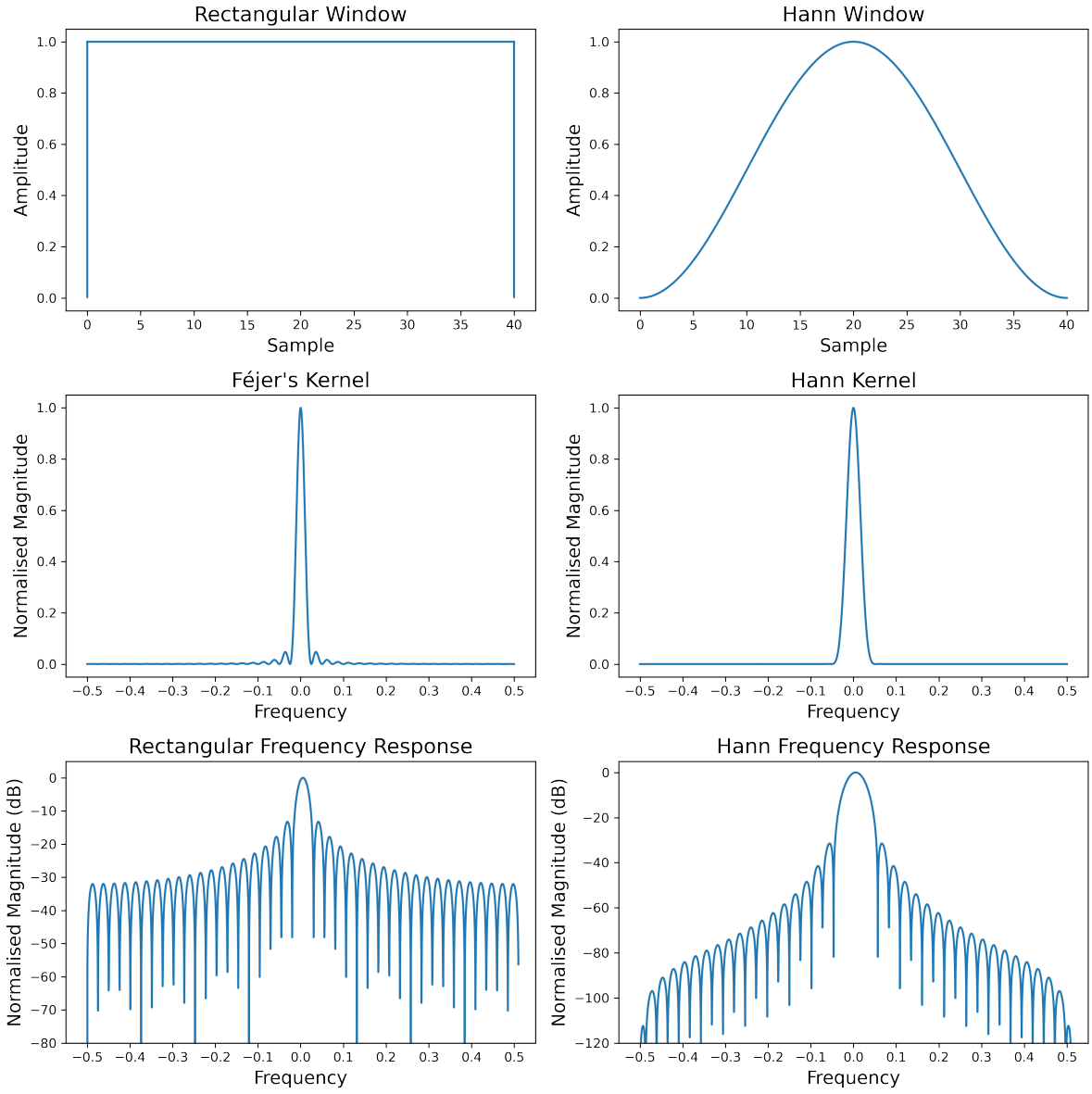


Figure A.2: Effect of windowing using the rectangular and the Hann taper for a sample of size $N = 40$.

B Tables

Below we provide tables with the main summary statistics of the estimated parameter distributions for the examples in Section 6. For the simulated GARMA(3,2,2) process we also provide the estimated AR and MA polynomials calculated using the method described in Subsection 5.1. It is straightforward to reconstruct the functional form of the process from these polynomials and the estimated Gegenbauer parameters using equation (2.5.1).

B.1 Simulated GARMA(3,2,2) Process

Table B.1: Summary statistics of the estimated parameter distributions for the GARMA(3,2,2) example.

	Parameter	Minimum	Maximum	Mean	Median	Mode	True Value
1	α_{11}^R (AR Real 1)	0.001	0.506	0.109	0.036	0.019	0.2
2	α_{11}^C (AR Complex Modulus 1)	0.771	0.892	0.812	0.801	0.801	0.9
3	ω_{11}^C (AR Complex Argument 1)	0.100	0.122	0.115	0.116	0.116	0.1
4	α_{21}^C (MA Complex Modulus 1)	0.002	0.470	0.212	0.228	0.257	0.2
5	ω_{21}^C (MA Complex Argument 1)	0.008	0.498	0.244	0.242	0.200	0.1
6	λ_1 (Gegenbauer 1)	-0.323	-0.297	-0.306	-0.306	-0.306	-0.3
7	λ_2 (Gegenbauer 2)	-0.902	-0.891	-0.894	-0.891	-0.891	-0.9
8	δ_1 (Gegenbauer memory 1)	0.283	0.496	0.385	0.388	0.408	0.4
9	δ_2 (Gegenbauer memory 2)	0.319	0.499	0.420	0.418	0.418	0.45
10	σ_ϵ^2 (Innovation Variance)	0.152	9.845	1.765	1.258	2.789	1

Table B.2: Estimated AR polynomial for the GARMA(3,2,2) example.

	AR Coefficient	Value
1	ϕ_0	1.000
2	ϕ_1	-1.329
3	ϕ_2	0.792
4	ϕ_3	-0.072

	AR Polynomial
Expression	$\Phi(z) = 1 - \phi_1 z^1 - \phi_2 z^2 - \phi_3 z^3$
Value	$\Phi(z) = 1 - 1.329 z^1 + 0.792 z^2 - 0.072 z^3$

Table B.3: Estimated MA polynomial for the GARMA(3,2,2) example.

	MA Coefficient	Value
1	θ_0	1.000
2	θ_1	-0.015
3	θ_2	0.045

	MA Polynomial
Expression	$\Theta(z) = 1 - \theta_1 z^1 - \theta_2 z^2$
Value	$\Theta(z) = 1 - 0.015 z^1 + 0.045 z^2$

B.2 U.S. Monthly CPI Inflation Data

B.2.1 3-Factor Gegenbauer Model (Periodogram)

Table B.4: Summary statistics of the estimated parameter distributions for Model 1.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	λ_1 (Gegenbauer 1)	-0.973	0.897	0.458	0.481	0.481
2	λ_2 (Gegenbauer 2)	-0.409	0.876	0.727	0.869	0.869
3	λ_3 (Gegenbauer 3)	-0.949	0.993	0.131	-0.011	-0.023
4	δ_1 (Gegenbauer memory 1)	0.002	0.329	0.167	0.154	0.254
5	δ_2 (Gegenbauer memory 2)	0.036	0.324	0.201	0.218	0.248
6	δ_3 (Gegenbauer memory 3)	0.001	0.311	0.122	0.119	0.025
7	σ_ϵ^2 (Innovation Variance)	1.113	1.699	1.399	1.396	1.361

B.2.2 3-Factor Gegenbauer Model (Tapered)

Table B.5: Summary statistics of the estimated parameter distributions for Model 2.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	λ_1 (Gegenbauer 1)	-0.766	0.875	0.471	0.500	0.500
2	λ_2 (Gegenbauer 2)	-0.562	0.915	-0.050	-0.136	-0.253
3	λ_3 (Gegenbauer 3)	-0.236	0.882	0.795	0.864	0.864
4	δ_1 (Gegenbauer memory 1)	0.008	0.298	0.169	0.169	0.142
5	δ_2 (Gegenbauer memory 2)	0.012	0.283	0.141	0.141	0.098
6	δ_3 (Gegenbauer memory 3)	0.058	0.276	0.198	0.201	0.195
7	σ_ϵ^2 (Innovation Variance)	1.204	2.011	1.501	1.495	1.487

B.2.3 AR(4) Model

Table B.6: Summary statistics of the estimated parameter distributions for Model 3.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	α_{11}^C (AR Complex Modulus 1)	0.002	0.584	0.240	0.211	0.082
2	α_{12}^C (AR Complex Modulus 2)	0.000	0.570	0.280	0.330	0.470
3	ω_{11}^C (AR Complex Argument 1)	0.002	0.266	0.135	0.153	0.174
4	ω_{12}^C (AR Complex Argument 2)	0.000	0.274	0.139	0.153	0.153
5	σ_ϵ^2 (Innovation Variance)	1.375	2.099	1.691	1.684	1.629

B.2.4 ARMA(5,3) Model

Table B.7: Summary statistics of the estimated parameter distributions for Model 4.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	α_{11}^{R} (AR Real 1)	0.001	0.841	0.532	0.530	0.804
2	α_{11}^{C} (AR Complex Modulus 1)	0.003	0.888	0.471	0.496	0.496
3	α_{12}^{C} (AR Complex Modulus 2)	0.005	0.759	0.270	0.283	0.289
4	ω_{11}^{C} (AR Complex Argument 1)	0.001	0.495	0.230	0.226	0.348
5	ω_{12}^{C} (AR Complex Argument 2)	0.003	0.499	0.165	0.121	0.096
6	α_{21}^{R} (MA Real 1)	0.005	0.890	0.574	0.596	0.814
7	α_{21}^{C} (MA Complex Modulus 1)	0.014	0.869	0.534	0.562	0.824
8	ω_{21}^{C} (MA Complex Argument 1)	0.002	0.497	0.286	0.347	0.348
9	σ_{ϵ}^{2} (Innovation Variance)	1.378	2.068	1.670	1.664	1.626

B.2.5 2-Factor Gegenbauer Model

Table B.8: Summary statistics of the estimated parameter distributions for Model 5.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	λ_1 (Gegenbauer 1)	0.335	0.880	0.681	0.515	0.507
2	λ_2 (Gegenbauer 2)	0.327	0.906	0.683	0.856	0.869
3	δ_1 (Gegenbauer memory 1)	0.100	0.309	0.180	0.177	0.142
4	δ_2 (Gegenbauer memory 2)	0.103	0.312	0.178	0.175	0.163
5	σ_{ϵ}^{2} (Innovation Variance)	1.242	1.949	1.556	1.552	1.476

B.2.6 GARMA(1,2,3) Model

Table B.9: Summary statistics of the estimated parameter distributions for Model 6.

	Parameter	Minimum	Maximum	Mean	Median	Mode
1	α_{11}^{R} (AR Real 1)	0.002	0.640	0.238	0.245	0.245
2	α_{21}^{R} (MA Real 1)	0.011	0.447	0.171	0.159	0.159
3	α_{21}^{C} (MA Complex Modulus 1)	0.001	0.354	0.098	0.085	0.053
4	ω_{21}^{C} (MA Complex Argument 1)	0.000	0.386	0.065	0.057	0.021
5	λ_1 (Gegenbauer 1)	0.851	0.874	0.865	0.864	0.864
6	λ_2 (Gegenbauer 2)	0.337	0.531	0.495	0.497	0.497
7	δ_1 (Gegenbauer memory 1)	0.049	0.364	0.216	0.214	0.207
8	δ_2 (Gegenbauer memory 2)	0.107	0.374	0.228	0.225	0.207
9	σ_{ϵ}^{2} (Innovation Variance)	1.297	1.927	1.544	1.546	1.413

C Code

The datasets and the Python code used to generate the figures and tables can be found at https://github.com/Matthieu-Papahagi/SMC-Frequency. The code is made available open-source and citations should be made in reference to this paper. For questions regarding the code and the data, e-mail matthieu-georges.papahagi18@imperial.ac.uk.