View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Scholarship@Western

Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

2-7-2013 12:00 AM

Persistence and Anti-persistence: Theory and Software

Justin Quinn Veenstra
The University of Western Ontario

Supervisor
A. I. McLeod
The University of Western Ontario

Graduate Program in Statistics and Actuarial Sciences
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy

© Justin Quinn Veenstra 2013

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Longitudinal Data Analysis and Time Series Commons, and the Statistical Models Commons

Recommended Citation

Veenstra, Justin Quinn, "Persistence and Anti-persistence: Theory and Software" (2013). *Electronic Thesis and Dissertation Repository*. 1119.
https://ir.lib.uwo.ca/etd/1119

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

PERSISTENCE AND ANTI-PERSISTENCE: THEORY AND SOFTWARE

(Thesis format: Monograph)

by

Justin Veenstra

Graduate Program in Statistical and Actuarial Sciences

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

The School of Graduate and Postdoctoral Studies
Western University
London, Ontario, Canada

© Justin Quinn Veenstra 2013

WESTERN UNIVERSITY
School of Graduate and Postdoctoral Studies

CERTIFICATE OF EXAMINATION

Examiners:

. . . .
Dr. Hao Yu

. . . .
Dr. Serge Provost

. . . .
Dr. Pierre Duchesne

. . . .
Dr. John Koval

Supervisor:

. . . .
Dr. A. I. McLeod

The thesis by

Justin Quinn Veenstra

entitled:

Persistence and Anti-persistence: Theory and Software

is accepted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

. . . .
Date

. . . .
Chair of the Thesis Examination Board

Abstract

Persistent and anti-persistent time series processes show what is called hyperbolic decay. Such series play an important role in the study of many diverse areas such as geophysics and financial economics. They are also of theoretical interest. Fractional Gaussian noise (FGN) and fractionally-differenced white noise are two widely known examples of time series models with hyperbolic decay. New closed form expressions are obtained for the spectral density functions of these models. Two lesser known time series models exhibiting hyperbolic decay are introduced and their basic properties are derived. A new algorithm for approximate likelihood estimation of the models using frequency domain methods is derived and implemented in R. The issue of mean estimation and multimodality in time series, particularly in the simple case of one short memory component and one hyperbolic component is discussed. Methods for visualizing bimodal surfaces are discussed. The exact prediction variance is derived for any model that admits an autocovariance function and integrated (inverse-differenced) by integer d. A new software package in R, **arfima**, for exact simulation, estimation, and forecasting of mixed short-memory and hyperbolic decay time series. This package has a wider functionality and increased reliability over other software that is available in R and elsewhere.

Keywords: Time series analysis, long-memory, anti-persistence, ***R***, hyperbolic decay, multimodal log-likelihood in time series.

Contents

Certificate of Examination ii

Abstract iii

Contents vi

List of Figures vii

List of Tables ix

1 Introduction 1
- 1.1 Notation and Conventions in this Thesis . . . 1
  - 1.1.1 Naming Conventions . . . 2
- 1.2 Hyperbolic Decay in Time Series . . . 3
- 1.3 The Layout of the Dissertation . . . 4

2 Hyperbolic Decay Time Series Models 5
- 2.1 Introduction . . . 5
- 2.2 Four Different Types of Hyperbolic Decay Time Series . . . 5
  - 2.2.1 Fractionally Differenced White Noise (FD) . . . 5
  - 2.2.2 Fractional Gaussian Noise (FGN) . . . 6
  - 2.2.3 Power Law Spectrum (PLS) . . . 8
  - 2.2.4 Power Law Autocovariance (PLA) . . . 9
- 2.3 Model Estimation . . . 12
  - 2.3.1 Exact Likelihood . . . 12
  - 2.3.2 Whittle Likelihood . . . 13
  - 2.3.3 Statistical Inference . . . 14
  - 2.3.4 Illustrative Example . . . 17
  - 2.3.5 Extensions . . . 20
- 2.4 Conclusions . . . 20

3 On the Combination of ARMA and HD Processes 22
- 3.1 Introduction . . . 22
- 3.2 Computing the TACVFs . . . 22
  - 3.2.1 A Solution via Convolution . . . 22
    - 3.2.1.1 The Moving Average Case . . . 25
  - 3.2.2 On the Kullback-Liebler Divergence Between Distributions . . . 26

        - 3.2.2.1 The KL divergence between two normal distributions 27
        - 3.2.2.2 A Limit Theorem 28
- 3.3 On Properties of ARMA-HD Processes 32
    - 3.3.1 Laws of Large Numbers 32
    - 3.3.2 The Convergence of Means 36

4 Predictions and Their Error Variances **38**
- 4.1 Introduction 38
    - 4.1.1 The Models Considered in this Chapter 38
    - 4.1.2 Results Used in the Chapter 39
- 4.2 Derivations of Predictors and Their Error Variances 41
    - 4.2.1 The Case of Non-Integrated Series 42
        - 4.2.1.1 The Predictors 42
        - 4.2.1.2 The Prediction Error Variances 44
    - 4.2.2 The Case of Integrated Series 45
        - 4.2.2.1 The Predictors 45
        - 4.2.2.2 The Prediction Error Variances 47
        - 4.2.2.3 On the Value of d^* 50
- 4.3 Proof of Equivalence in AR Case 50
    - 4.3.1 Three Useful Lemmas 50
    - 4.3.2 Proof of Equivalence in the AR(p) Case 54
    - 4.3.3 Proof of Equivalence in the ARIMA($p, d, 0$) Case 55
- 4.4 Comparison of the Forms with Increasing n 56
    - 4.4.1 On the Predictions 57
        - 4.4.1.1 On Stationary Models 57
        - 4.4.1.2 On Non-stationary Models 58
    - 4.4.2 On the Prediction Error Variances 58
    - 4.4.3 The Incorrect Application of the Exact Form 59
    - 4.4.4 On Running Time 59
- 4.5 Examples 60
    - 4.5.1 Simple Symbolic Examples 60
    - 4.5.2 Some Numerical Examples 61
- 4.6 Conclusions 62

5 The arfima Package **63**
- 5.1 Introduction 63
- 5.2 The Need for Exact Maximum Likelihood 63
    - 5.2.1 Near Singular Matrices 63
- 5.3 On Other R Packages that Deal with ARFIMA Models 65
    - 5.3.1 The Haslett-Raftery Method and the **fracdiff** ***R*** Package 65
    - 5.3.2 The Approximations Used 65
        - 5.3.2.1 The Approximations Listed in the Paper 66
        - 5.3.2.2 The Code's Heuristics 67
- 5.4 What the **arfima** Package Can Do 67
    - 5.4.1 Calculating the Log-Likelihood, Simulating, and Forecasting 67

5.4.2 More on the **arfima** Package 69
5.5 Package Details 69
5.5.1 Dealing with the Estimation of μ_w 69
5.5.2 The Partial Autocorrelation Space for AR and MA Coefficients 70
5.5.3 Functions in the Package 70
5.5.3.1 The Choice of Optimizer 72
5.5.4 Considering the Critique of a Mode 72
5.6 Numerical Results 72
5.6.1 The `fracdiffMM` Script 73
5.6.2 River Flow Data 73
5.6.3 Temperature Data 74
5.6.4 Simulation Studies 77
5.7 Other Examples of Using **arfima** 84
5.7.1 Looking at Plots of the TACVF 84
5.7.2 Series J 86
5.7.3 A Prediction Example 90
5.8 On Multimodality in ARFIMA Models 92
5.9 Conclusions and Future Work 94

6 Visualizations and Multimodality **95**
6.1 Introduction 95
6.1.1 A Discussion of Mean Estimation and Visualizations 95
6.2 Visualizing a Log-Likelihood Surface 96
6.2.1 Technical Considerations 96
6.2.2 Extracting the Fitted Model from ***R*** 97
6.2.3 On the **simpleVis** Package 97
6.3 Suppositions on Multimodal Behaviour 98
6.3.1 On Apparent Persistence or Anti-persistence in Simple Models 100
6.3.1.1 Apparent Anti-persistence and the MA-Based Models 101
6.3.1.2 Apparent Persistence and the AR-Based Models 102
6.3.2 On More Complex Models 102
6.3.3 Dynamic Mean Estimation and Modes on the Boundaries 103
6.3.3.1 The Push To Persistence 104
6.3.3.2 On the Effect of Larger n and Mean Estimation 107
6.3.4 The Effect of Adding Noise to a Series 108
6.4 Conclusions and Future Work 109

7 Conclusion **112**
7.1 Future Work 113

Bibliography **114**

A Appendix to Chapter 2 **120**

Curriculum Vitae **130**

List of Figures

2.1 Comparison of the spectral density functions of the FGN (solid curve) and FD (dashed curve) models. In the left panel corresponds to strong long-memory with decay parameter $\alpha = 0.4$ and the right panel shows the anti-persistent case with $\alpha = 1.6$. Note that when $\alpha = 0.4$, $H = 0.8$ for FGN and $d = 0.3$ for FD while $\alpha = 1.6$ corresponds to $H = 0.2$ and $d = -0.3$ respectively. 7

2.2 The term c_a. 9

2.3 Comparing the expected Fisher information $I(\alpha)$ in the FD, PLS, and FGN models. The horizontal line at about 1.64 corresponds to the FD case, the curve shows $I(\alpha)$ for the PLS model and the plotted points show the estimated information for α based on 10^5 simulations of the FGN model. 15

2.4 Comparison of the relative likelihood functions for the different models estimated with exact MLE and using the Whittle approximate MLE. 19

3.1 Boxplots of the differences in absolute mean from zero on the $\log_{10}$ scale for 500 series generated with the same seeds, but with different n and d_f. The facet label is the value of n, the horizontal axis is the value of d_f, and absolute difference from zero is the vertical axis. 37

5.1 The differences in log-likelihoods with respect to `arfima` $\bar{w}$; this figure shows that for the most part **arfima** with sample mean subtracted does better in terms of exact likelihood than **fracdiff** and sometime itself with dynamic mean estimation. Note that either one or the other does as well or better than **fracdiff**. . . 83

5.2 The TACVF plot of the toy data set M, fit as ARFI, where one mode (mode 1) is anti-persistent with $\phi \simeq 0.93$ and $d_f \simeq -0.64$, and the other (mode 2) is persistent with $\phi \simeq 0.25$ and $d_f \simeq 0.09$ 85

5.3 The TACVF plot of the toy data set N, fit as ARFI with two very persistent modes: the first having $\phi \simeq 0.98$ and $d_f \simeq 0.45$, and the second having $\phi \simeq 0.92$ and $d_f \simeq 0.5$. It is obvious that mode 2 is spurious, as it hardly decays; we note that this "mode" has the optimizer trapped on the upper boundary for d_f. 87

5.4 The plots of the predictions associated with `fit`, a bimodal log-likelihood; this figure shows the modes for this particular fit are similar enough to give similar predictions 93

6.1 The TACVF plot of `fit1`, a FIMA model fitted to simulated data, with two modes, one of which has persistent parameters and one of which has anti-persistent parameters, although the TACVFs look very similar. Mode 1 is the persistent mode, with $\theta \simeq 0.96$ and $d_f \simeq 0.44$ while mode 2 has $\theta \simeq 0.26$ and $d_f \simeq -0.25$. 99
6.2 White noise modelled as FD with different means; $e_t \sim \text{NID}(0, 1)$ for $t = 1, \ldots, 1000$ was generated as e, and had $\boldsymbol{a} = \boldsymbol{e} - \bar{e}$ to have a mean of exactly zero up to machine epsilon. For $m_{i=1}^{301} = (-15, -14.9, \ldots, 14.9, 15)$, $\boldsymbol{e} - m[i]$ was fit as FD with the "true" mean set to zero, and $d_f[i]$ was recorded. The plot is m on the horizontal axis and d_f on the vertical axis. 105
6.3 A **simpleVis** representation of an ARFI process with $\text{NID}(0, \sigma_y^2)$ noise added to the series, with the top plot having $\sigma_y^2 = 0$, the middle having $\sigma_y^2 = 2$, and the bottom having $\sigma_y^2 = 4$. The fitted values were found using the **arfima** package. The bottom plot has 3 points of the optimization from **arfima** pushed to the boundaries hidden behind the peak at the back. We note that this occurs due to a push to persistence. 110
6.4 A **simpleVis** representation of a FIMA process with $\text{NID}(0, \sigma_y^2)$ noise added to the series, with the top plot having $\sigma_y^2 = 0$, the middle having $\sigma_y^2 = 0.25$, and the bottom having $\sigma_y^2 = 0.5$. The fitted values were found using the **arfima** package. The log-likelihood surface turns into one that describes zero mean white noise. 111

List of Tables

2.1 Exact and Whittle MLE estimates $\hat{\alpha}$, relative likelihood, R, and computer time required. 18
2.2 95% confidence intervals for α based on the likelihood-ratio test. 18

4.1 The MA(1) prediction error variance differences with symbolic θ and d, $n = 15$. 60
4.2 The ARFIMA(1, 0.45, 1) prediction error variances 62
4.3 The ARFIMA(1, 1 + 0.45, 1) prediction error variances 62

5.1 The AICs and order of the ARMA parameters chosen by the `arfima` function, the `fracdiffMM` script as chosen by exact **arfima** AIC, and the `fracdiffMM` script as chosen by the **fracdiff** AIC for seven riverflow data sets found in Hipel and McLeod [1994] 74
5.2 Model Specifications for the Simulation Studies 77
5.3 Model 1 RMSEs: it would seem that **fracdiff** is only finding one mode of this often bimodal surface, while the **arfima** fits are finding both. While this shows that **arfima** does much better, recall that we only have one parameter that **fracdiff** has multiple starts in. 78
5.4 Model 2 RMSEs: another case where **fracdiff** has trouble finding multiple modes when they exist; the **arfima** starred modes do very well 79
5.5 Model 3 RMSEs: we were sure that this surface was unimodal, which it seems to be from this table. **arfima** has a slight advantage, but the results are comparable. 79
5.6 Model 4 RMSEs: **fracdiff** finally seems to find multple modes: however, as we see from Figure 5.1 that the high modes found by **arfima** are superior; we also see superiority in the starred fits 79
5.7 Model 5 RMSEs: we see the same pattern as Table 5.6; **arfima** does better on the whole, although not by as much. We had thought this set of parameters to be unimodal; either we were wrong, or the overfitting with d_f induced modes: see the text. 80
5.8 Model 6 RMSEs: we were sure that this surface was unimodal, which it seems to be from this table. The results are comparable. 80
5.9 Model 7 RMSEs: this set of parameters was known to us to generally give a bimodal surface. We note that dynamic mean estimation did very poorly here, which we will discuss in Chapter 6. The mean subtracted version did much better. **fracdiff** seemingly only found one mode. 80

5.10 Model 8 RMSEs: this set of parameters, thought to possibly lead to bimodal surfaces, seems to lead to unimodal surfaces. We checked each fit for this particular case. The changes in the **arfima** estimates come from spurious modes on boundaries, which is sometimes a problem, especially when a task is automated. Surprisingly, the modes were only induced by subtracting the sample mean rather than dynamically estimating it. The latter did a poor job, while the former did a comparable job to **fracdiff**. See Chapter 6 for more on mode induction. 81
5.11 Model 9 RMSEs: a set of parameters leading to a unimodal surface we thought was possibly bimodal; once again, **arfima** has a small amount of mode induction, although this time it does much better than **fracdiff** 81
5.12 A Timings Table for the Different Fitting Methods by Model. Note that **fracdiff** does dominate 84

Chapter 1

Introduction

Persistent and anti-persistent time series are the two types of processes that exhibit what is called hyperbolic decay (HD). This is in terms of autocovariance structure: the autocovariances decay hyperbolically. Persistent processes, also called strongly-persistent or long-memory, show a positive long-range dependence between the observations, while anti-persistent ones reverse direction very often: they have strong negative autocorrelations. While it is possible to simply integrate (i.e. inverse difference) some anti-persistent time series to give them long memory, we believe most anti-persistent time series come about from differencing a process that is integrated not quite to unity. These processes need to be differenced for the sake of stationarity. Anti-persistent processes have some place in physics and economics, as well as in the study of random walks.

In this thesis, theory on persistent and anti-persistent processes is presented, as well as two ***R*** packages. The first, called **FGN** as in McLeod and Veenstra [2012], is updated to use all types of pure hyperbolic decay processes, which will be discussed in Chapter 2. A new package called **arfima** is presented, in which ARIMA models with various types of HD noise are used. This package uses exact maximum likelihood and we demonstrate that it improves on the existing **fracdiff** package which only provides approximate maximum likelihood when there is also an autoregressive-moving average (ARMA) component present. Chapter 5 discusses this package.

1.1 Notation and Conventions in this Thesis

Let the mean of the covariance stationary process $\boldsymbol{w}$ be μ_w. Then the k^{th} lag theoretical autocovariance of $\boldsymbol{w}$ is equal to

$$\begin{aligned} \gamma_w(k) &= \mathrm{E}[w_t w_{t-k}] - \mathrm{E}[w_t]\mathrm{E}[w_{t-k}] && (1.1) \\ &= \mathrm{E}[w_t w_{t-k}] - \mu_w^2 && (1.2) \\ &= \mathrm{E}[w_t w_{t+k}] - \mu_w^2 && (1.3) \\ &= \gamma_w(-k) && (1.4) \end{aligned}$$

Most often the convention used is that $\mu_w = 0$. The theoretical autocovariance function will be called the TACVF. The theoretical autocorrelation function, given as

$$\rho_w(k) = \gamma_w(k)/\gamma_w(0), \quad \forall k \in \mathbb{Z} \tag{1.5}$$

will be known as the TACF. When there is no possibility of confusion, we will let $\gamma(\cdot) = \gamma_w(\cdot)$ and similarly with ρ.

The TACVF and TACF can be used for simulating, fitting, and forecasting of time series data through the Durbin-Levinson and Trench algorithms as in McLeod et al. [2007b]. This will be outlined in §5.4.1.

If n is the length of the process, then γ_k will be defined as

$$\gamma_k' = (\gamma_w(k), \ldots, \gamma_w(n+k-1)). \tag{1.6}$$

We note here that the dependence on n is implicit, since n is fixed: therefore we index via k. The covariance matrix of n successive observations, Γ_n, is

$$\Gamma_n = [\gamma_w(i-j)]_{i,j=1,\ldots,n}. \tag{1.7}$$

This is the symmetric Toeplitz matrix of the lag 0 through lag $n-1$ autocovariances.

Two types of expectation operators are spoken of in this dissertation: the unconditional expectation, and the conditional expectation up to time t. The former will be denoted as E and the latter as E_t. For example $\mathrm{E}_t[w_{t+k}]$ means $\mathrm{E}[w_{t+k}|w_t, w_{t-1}, \ldots]$.

1.1.1 Naming Conventions

The naming conventions of this thesis should be mentioned. While they are introduced in the chapters in which they are described and some are well known, they are briefly gone over here.

The autoregressive operator of order p is denoted AR(p), and the moving average operator of order q is denoted MA(q). Similarly, the autoregressive (integrated) moving average of orders p, q and d, the last parameter being for integration, are denoted by ARMA(p, q) and ARIMA(p, d, q). The class of hyperbolic decay processes that are discussed in this thesis are denoted HD: the actual models are fractionally differenced white noise (FD), fractional Gaussian noise (FGN), power law spectrum (PLS), and the newly derived power law autocovariance (PLA). For the most part, the mixture of ARMA and HD processes are called ARMA-HD, with the process symbol instead of HD when a specific process is desired. The exception is FD: processes are called autoregressive fractionally integrated moving average processes, ARFIMA(p, d^*, q), where $d^* = d + d_f$. Most commonly, $d_f \in (-1, 0.5)$ is the fractional part, and $d \in \mathbb{Z}_{\geq 0}$ always is the integer part. Also to be introduced are the ARFI and FIMA models: these are the ARFIMA($1, d_f, 0$) and ARFIMA($0, d_f, 1$) models, respectively.

1.2 Hyperbolic Decay in Time Series

Let $\boldsymbol{w}$ be a covariance stationary time series with autocovariance function $\gamma_w(k) = \text{Cov}\,(w_t, w_{t-k})$. Then $\boldsymbol{w}$ has hyperbolic or power-law decay if

$$\lim_{k\to\infty} \gamma_w(k)/k^{-\alpha} = c_\alpha, \tag{1.8}$$

where $\alpha > 0$, and $c_\alpha > 0$ for $\alpha < 1$ and negative otherwise. With suitable parameters, the stationary ARFIMA time series model (Beran [1994], Palma [2007]) is a notable example of this class of models. In general, as required for any covariance stationary time series, the autocovariance function is assumed to be symmetric and to satisfy the non-negative-definite condition [Brockwell and Davis, 1991, Theorem 1.5.1] and hence the spectral density function exists and may be written,

$$\begin{aligned} f_\alpha(\lambda) &= \frac{1}{2\pi}\sum_{k=-\infty}^{\infty} \gamma_w(k) e^{ik\lambda} && (1.9)\\ &= \frac{1}{2\pi}\left(\gamma_w(0) + 2\sum_{k=1}^{\infty} \gamma_w(k)\cos(\lambda k)\right). && (1.10)\end{aligned}$$

When $\gamma_w(k)$'s in (1.9) and (1.10) are replaced by the autocorrelations, $\rho(k) = \gamma(k)/\gamma(0)$, $f_\alpha(\lambda)$ is referred to as the normalized spectral density function [Priestley, 1981, Equation 4.8.15].

If the spectral density function is specified, another sufficient condition for the existence of the time series may be given. Assuming that $f_\alpha(\lambda)$ is defined by (1.9), Wold's Theorem [Priestley, 1981, §4.8.3] implies that if the area under the normalized spectral density function over $(-\pi, \pi)$ is one and $f_\alpha(\lambda) \geq 0$ for $\lambda \in (-\pi, \pi)$, the time series exists and the autocovariances determined by $\gamma(k) = \int_{-\pi}^{\pi} e^{-i\lambda k} f_\alpha(\lambda) d\lambda$.

A time series is said to have (strong) persistence or long memory [Hipel and McLeod, 1994, §2.5.3] if

$$\sum_{k=-\infty}^{\infty} \gamma_w(k) = \infty. \tag{1.11}$$

Hence $\alpha \in (0, 1)$ corresponds to long memory or persistence. When the sum in (1.11) is finite, the process is said to be weakly or short-range dependent. A special type of short-range dependence occurs when $\alpha > 1$ and the time series is said to be anti-persistent. In the anti-persistent case,

$$\sum_{k=-\infty}^{\infty} \gamma_w(k) = 0. \tag{1.12}$$

When $\alpha = 1$, we set $c_\alpha = 0$. Short-range dependent models such as ARMA are included the $\alpha = 1$ case.

Many time series show observed spectra that appear governed by a power law, $f_\alpha(\lambda) \propto |\lambda|^{\alpha-1}$ where $0 < \alpha < 1$ and λ is typically in the low frequency range (Granger [1966], Wolfram [2002, p.969]). This provides another characterization of persistence and anti-persistence. More generally, for all $\alpha > 0$, let

$$\lim_{\lambda\to 0} f_\alpha(\lambda)/\lambda^{\alpha-1} = C_\alpha, \tag{1.13}$$

where $C_\alpha > 0$. This implies that as $\lambda \to 0$, $f_\alpha \to \infty$ or $f_\alpha \to 0$ according as the process is persistent or anti-persistent respectively. The conditions specified in (1.8) and (1.13) are equivalent [McLeod, 1998, Theorem 2] but it should be noted this equivalence of the power-law decay for the autocovariance function and spectral density functions as specified in (1.8) and (1.13) does not hold under more general assumptions [Yong, 1971, 1972].

As shown in [McLeod, 1998, Theorem 1], the time series may be written as a linear time series,

$$w_t = \psi(B)a_t, \tag{1.14}$$

where a_t is white noise with variance σ_a^2, $\psi(B) = 1 + \psi_1 B + \psi_2 B^2 + \ldots$, B is the backshift operator on t, and $\psi_k = O(k^{-(1+\alpha)/2})$. This linear time series is sometimes called a generalized linear process [Hannan, 1970, p. 210]. Then as in Box et al. [2008b, A3.1.14], the spectral density function may be written,

$$f_\alpha(\lambda) = \frac{\sigma_a^2}{2\pi}|\psi(e^{-i\lambda})|^2. \tag{1.15}$$

Theorem 1.1. *A hyperbolic Gaussian time series process with mean zero is ergodic, that is, the sample autocovariance at lag k, c_k, converges almost surely to $\gamma(k)$ as the series length n increases, where*

$$c_k = \frac{1}{n}\sum_{t=k+1}^{n} w_t w_{t-k}. \tag{1.16}$$

The proof follows directly from the generalized linear process representation and Hannan [1970, §IV, Theorem 6].

1.3 The Layout of the Dissertation

Chapter 2 discusses properties of HD models and explores their use through the **FGN** package, while Chapter 3, is about extension of ARMA processes driven by HD noise. Chapter 4 speaks of minimum mean square error prediction for any time series model that can be written in operator notation, which is extended to any model that admits an autocovariance function. Said chapter also introduces a new exact method for calculation of the prediction error variances of an integrated series. Chapter 5 is devoted to the **arfima** package and our comparisons with **fracdiff**. Chapter 6 lays out reasons for mulitmodality on a log-likelihood surface of time series data, as well as addressing technical issues with visualizations of such surfaces. It also presents a simple ***Mathematica*** package for viewing such surfaces with two parameter models called **simpleVis**. Finally Chapter 7 summarizes the thesis.

Chapter 2

Hyperbolic Decay Time Series Models

2.1 Introduction

The four types of hyperbolic decay time series are discussed in this chapter. The processes are introduced and properties are derived. Note that the appendix to this chapter has derivations in ***Mathematica*** along with other information that will be referred to in this chapter. It is Appendix A.

2.2 Four Different Types of Hyperbolic Decay Time Series

2.2.1 Fractionally Differenced White Noise (FD)

The fractionally differenced white noise model and its ARFIMA extension is currently one of the most widely used hyperbolic decay time series models [Box et al., 2008b, §10.3]. The FD model [Granger and Joyeux, 1980, Hosking, 1981] is derived from the model equation,

$$\nabla^{d_f} w_t = a_t \tag{2.1}$$

where $\nabla = (1-B)$, a_t is a white noise sequence with variance σ_a^2, and $d_f \in (-\infty, 0.5)$ (Dębowski [2011]). Usually the range $d_f \in (-1, 0.5)$ is used and that the process does not exist with d_f on the negative integers. The autocovariance function is given by (Hosking [1981], Dębowski [2011]), with $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$ being the Gamma function,

$$\gamma(k) = \sigma_w^2 \frac{\Gamma(k + d_f)\Gamma(1 - d_f)}{\Gamma(k - d_f + 1)\Gamma(d_f)} \tag{2.2}$$

$$= \sigma_w^2 \prod_{0<h\leq k} \frac{h - 1 + d_f}{h - d_f} \tag{2.3}$$

where $\sigma_w^2 = \gamma(0) = \sigma_a^2 \Gamma(1-2d_f)/\Gamma(1-d_f)^2$. The spectral density function may be written,

$$f_{d_f}(\lambda) = \frac{\sigma_a^2}{2\pi}\left(\sin\left(\frac{\lambda}{2}\right)\right)^{-2d_f} \tag{2.4}$$

and the FD model is hyperbolic with $\alpha = 1 - 2d_f$ and $c_\alpha = (-d_f)!/(d_f - 1)!$ [Hosking, 1981, Palma, 2007]. This model is useful in modelling financial/econometric series [Baillie, 1996, Baillie et al., 1996, Bhardwaj and Swanson, 2006, Tsay, 2010] and other differenced time series such as annual temperature changes Kärner [2001, 2002]. As a further illustration, Li and Li [2008], obtained $\hat{d}^* = d + \hat{d}_f = 0.71$ for a nonstationary ARFIMA model with $d = 1$ for absolute returns of the Dow Jones Industrial Average Index. This corresponds to a value of $\hat{d}_f = -0.29$ fit to the first differences of the series.

2.2.2 Fractional Gaussian Noise (FGN)

The first widely used hyperbolic time series model was FGN. It is defined by the discrete-time increments, $w_t = B_H(t) - B_H(t-1)$, in fractional Brownian motion, $B_H(t)$. This process was originally suggested by Kolmogoroff for modeling turbulence [Molchan, 2003] and subsequently for time series modeling by Mandelbrot and Van Ness [1968]. FGN has been used extensively in hydrology for modeling persistence and the Hurst effect [McLeod and Hipel, 1978, Hipel and McLeod, 1994] as well as many other areas Doukhan et al. [2003]. Fractional Brownian motion [Beran, 1994] may be defined

$$B_H(t) = s\int w_H(t,u)dB(u), \tag{2.5}$$

where $s > 0$ is a positive scaling factor, $B(u)$ Brownian motion and

$$w_H(t,u) = \begin{cases} 0, & \text{if } t < u, \\ (t-u)^{H-1/2}, & 0 \le u < t, \\ (t-u)^{H-1/2} - (-u)^{H-1/2}, & u < 0. \end{cases} \tag{2.6}$$

The definition and properties of $B_H(t)$ are discussed in more detail by Beran [1994], Taqqu [2003]. For FGN with parameter H,

$$\gamma_w(k) = \sigma_w^2\left((k+1)^{2H} - 2k^{2H} + (k-1)^{2H}\right)/2 \tag{2.7}$$

for $k > 0$ and $0 < H < 1$ [Beran, 1994, Taqqu, 2003]. When $H = 0.5$, FGN reduces to Gaussian white noise. For large k, [Beran, 1994, p.52], FGN is hyperbolic with $\rho(k) \approx H(2H-1)k^{2H-2}$ so $\alpha = 2(1-H)$ and we see that FGN is persistence or anti-persistent according as $H \in (0.5, 1)$ or $H \in (0, 0.5)$. Unlike FD, FGN is only defined for $\alpha \in (0, 2)$. Figure 2.1 below compares the spectral density functions for FGN and FD for typical persistent and anti-persistent cases. In general, the spectral density function for FGN with parameter H is similar to the corresponding spectral density of FD with parameter $d_f = H - 0.5$. But as discussed Cleveland [1994], it is difficult to judge accurately the difference between two steep curves as shown in the top panels in Figure 2.1. The bottom panels reveal that while the difference is not large, it is rapidly increasing in the strong long-memory case when λ is near zero. In the anti-persistent case, the difference between the curves is larger at the high frequencies. The spectral density function for FGN is computed using a new closed form expression given in Theorem 2.1.

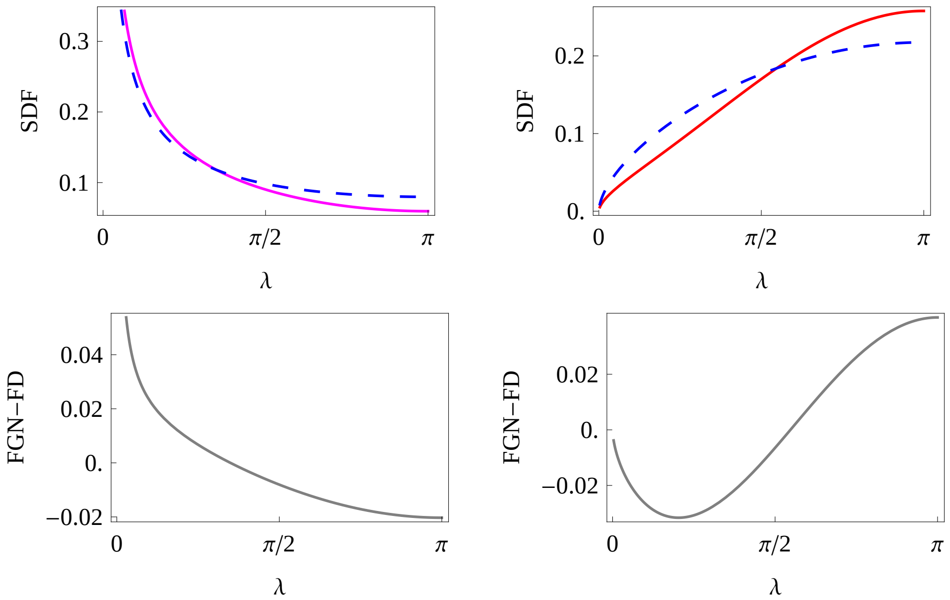


Figure 2.1: Comparison of the spectral density functions of the FGN (solid curve) and FD (dashed curve) models. In the left panel corresponds to strong long-memory with decay parameter $\alpha = 0.4$ and the right panel shows the anti-persistent case with $\alpha = 1.6$. Note that when $\alpha = 0.4$, $H = 0.8$ for FGN and $d = 0.3$ for FD while $\alpha = 1.6$ corresponds to $H = 0.2$ and $d = -0.3$ respectively.

Theorem 2.1. *For $\lambda \in (0, \pi)$,*

$$f_H(\lambda) = \frac{1}{4\pi}(A_1 + A_2 - 2A_3) \tag{2.8}$$

where $A_1 = e^{-i\lambda}\left(\Phi\left(e^{-i\lambda}, -2H, 0\right) + \Phi\left(e^{-i\lambda}, -2H, 2\right)\right)$, $A_2 = e^{i\lambda}\left(\Phi\left(e^{i\lambda}, -2H, 0\right) + \Phi\left(e^{i\lambda}, -2H, 2\right)\right)$, $A_3 = 2\left(Li_{-2H}\left(e^{-i\lambda}\right) + Li_{-2H}\left(e^{i\lambda}\right) - 1\right)$, $\Phi(z, s, a)$ is the Lerch zeta function,

$$\Phi(z, s, a) = \sum_{k=0}^{\infty} \frac{z^k}{(a+k)^s} \tag{2.9}$$

and Li_α denotes the polylogarithm,

$$Li_\alpha(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^\alpha}. \tag{2.10}$$

The derivation of this result using ***Mathematica*** is discussed in Appendix A. The spectral density may also be computed using a formula given in Beran [1994, Equation (2.17)],

$$f_H(\lambda) = \frac{\sigma_w^2}{\pi} \sin(\pi H)\Gamma(2H+1) \sum_{j=-\infty}^{\infty} |2\pi j + \lambda|^{-2H-1}, \tag{2.11}$$

where $\lambda \in (-\pi, \pi)$. It is verified in Appendix A, that (2.8) and (2.11) produce equivalent results. For small λ, Beran [1994, p. 52] derived the result,

$$f_H(\lambda) \approx \sigma_w^2/(2\pi)\sin(\pi H)\Gamma(2H+1)|\lambda|^{1-2H}. \tag{2.12}$$

The Lerch zeta and polylogarithm are special functions associated with the Riemann zeta function. An introduction to the extensive literature on these functions is available online [Wikipedia, 2012d,b] in more detail in the book by Srivastava and Junesang [2001]. Algorithms are available for computing the special functions in (2.9) and (2.10) in ***Mathematica*** and other widely used computing environments as well as in the GNU Scientific Library.[1] Our ***Mathematica*** demonstration on power-law decay time series computes $f_H(\lambda)$ [Veenstra and McLeod, 2012b]. We also provide an R library for the efficient computation of $f_H(\lambda)$ [McLeod and Veenstra, 2012]. The closed form expression in Theorem 2.1 is also useful for symbolic algebraic computation.

2.2.3 Power Law Spectrum (PLS)

Percival and Walden [2000, §7.6] suggested a time series model for which the spectral density function is proportional to $|\lambda|^{p-1}$ for $\lambda \in (-\pi, \pi)$. In this model, the model parameter p equals the decay, α, in (1.13).

Theorem 2.2. *The* PLS *model has spectral density function*

$$f_p(\lambda) = \frac{p\sigma_z^2}{2\pi^p}|\lambda|^{p-1}, \tag{2.13}$$

where $p > 0$ *and autocorrelation function,* $k > 0$,

$$\rho(k) = {}_1F_2\left(\frac{p}{2}; \frac{1}{2}, \frac{p}{2}+1; -\frac{1}{4}k^2\pi^2\right) \tag{2.14}$$

where ${}_{\mathrm{q}}F_{\mathrm{p}}$ *is the generalized hypergeometric function,*

$${}_{\mathrm{q}}F_{\mathrm{p}}(a_1, \ldots, a_{\mathrm{p}}; b_1, \ldots, b_{\mathrm{q}}, z) = \sum_{k=0}^{\infty} \frac{(a_1)_k \cdots (a_{\mathrm{p}})_k}{(b_1)_k \cdots (b_{\mathrm{q}})_k} \frac{z^k}{k!}. \tag{2.15}$$

The large-lag formula when $p \in (0, 1)$ *is*

$$\rho(k) = \frac{2^{p-1}\pi^{\frac{1}{2}-p}p\Gamma(p/2)}{\Gamma((1-p)/2))}k^{-p} + o(1) \tag{2.16}$$

The derivation of Theorem 2.2 is included in Appendix A.

---

[1] `http://www.gnu.org/software/gsl/`

2.2.4 Power Law Autocovariance (PLA)

The PLA model, first presented here, specifies that the autocovariance function at lag k is proportional to k^{-a}, where $0 < a < 3$ is the model parameter and $a = \alpha$. More precisely in terms of the autocorrelation function, the PLA model is defined by, for $|k| > 0$,

$$\rho(k) = c_a|k|^{-a}, \tag{2.17}$$

with

$$c_a = \begin{cases} -(2\zeta(a))^{-1}, & a \neq 1, \\ 0, & a = 1, \end{cases} \tag{2.18}$$

and $\zeta(a)$ is the Riemann zeta function [Titchmarsh and Heath-Brown, 1987],

$$\zeta(a) = \begin{cases} (1-2^{1-a})^{-1}\sum_{k=1}^{\infty}(-1)^{k-1}k^{-a}, & 0 < a < 1, \\ \sum_{k=1}^{\infty} k^{-a}, & a > 1. \end{cases} \tag{2.19}$$

where the two expressions in (2.19) are equivalent for $a > 1$. It may be shown algebraically or using ***Mathematica*** symbolics that c_a and $\rho(k)$ are continuous functions of a; see Appendix A This property is illustrated in Figure 2.2 below.

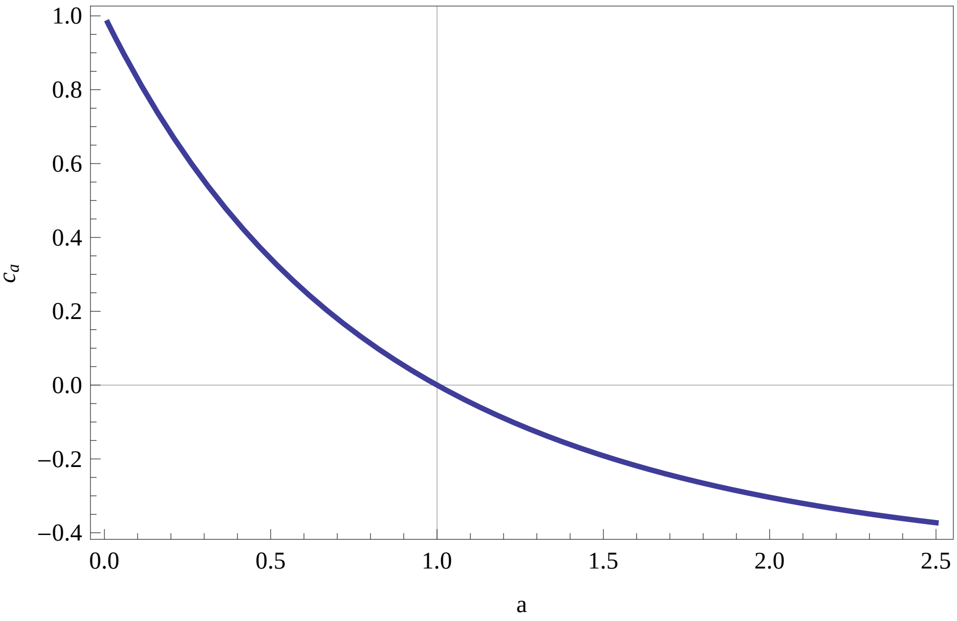


Figure 2.2: The term c_a.

Theorem 2.3. *The covariance stationary Gaussian time series defined by* (2.17) *exists.*

Proof. Let $\sigma_w^2 = 1$ and as such $\gamma_w = \rho_w$. We have that from Wold's theorem [Priestley, 1981] that γ_w as defined by (2.17) is an autocovariance function of a stationary process if and only if

$$\gamma_w(h) = \int_{-\pi}^{\pi} e^{i\nu h} dF_w(\nu) \tag{2.20}$$

for all $h \in \mathbb{Z}$ where F_w is a non-decreasing function on $[-\pi, \pi]$ with $F_w(-\pi) = 0$ and $F_w(\pi) = \sigma_w^2 = 1$.

We follow the regular definition of the spectral density function to find if there is some F_w that satisfies this. We have that

$$\begin{aligned} f_w(\lambda) &= \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \gamma_w(k) e^{-ik\lambda} && (2.21) \\ &= \frac{1}{2\pi} \left(1 - (2\zeta(a))^{-1} \sum_{k=-\infty, k\neq 0}^{\infty} k^{-a} e^{-ik\lambda} \right) && (2.22) \\ &= \frac{1}{2\pi} \left(1 - \zeta(a)^{-1} \sum_{k=1}^{\infty} k^{-a} \cos(k\lambda) \right) && (2.23) \\ &= \frac{1}{2\pi} \left(1 - \zeta(a)^{-1} (\mathrm{Li}_a(e^{i\lambda}) + \mathrm{Li}_a(e^{-i\lambda})) \right) && (2.24) \end{aligned}$$

as in Theorem 2.4.

Clearly if $F_w(\lambda)$ exists, we have

$$\gamma_w(h) = \int_{-\pi}^{\pi} e^{i\nu h} dF_w(\nu) \tag{2.25}$$

since, following (2.21) we have

$$\begin{aligned} (2\pi)^{-1} \int_{-\pi}^{\pi} e^{i\nu h} \sum_{k=-\infty}^{\infty} e^{-i\nu k} \gamma_w(k) d\nu &= (2\pi)^{-1}\, 2\pi \sum_{k=-\infty}^{\infty} I(k = h) \gamma_w(k) && (2.26) \\ &= \gamma_w(h) && (2.27) \end{aligned}$$

as required, with I being the indicator function and $h, k \in \mathbb{Z}$.

We can verify the requirements on F_w through the above. We have that $F_w(-\pi) = 0$ by definition. Since

$$\begin{aligned} \int_{-\pi}^{\pi} \cos(k\nu)\, d\nu &= 2 \sin(k\pi)\, k^{-1} && (2.28) \\ &= 0 \text{ for } k \in \mathbb{Z}_{>0} && (2.29) \\ \Rightarrow \int_{-\pi}^{\pi} f_w(\nu) d\nu &= (2\pi)^{-1} \left(\int_{-\pi}^{\pi} 1 - 0 \right) && (2.30) \\ &= 1 && (2.31) \end{aligned}$$

and as such $F_w(\pi) = 1$.

Now we must show that F_w is non-decreasing: that is, that $f_w(\lambda) \geq 0$ for $\lambda \in (-\pi, \pi)$. We note that this is equivalent, due to the nature of the trigonometric functions, to $f_w(\lambda) \geq 0$ for $\lambda \in (0, 2\pi)$.

We must show that

$$1 - \zeta(a)^{-1} \sum_{k=1}^{\infty} k^{-a} \cos(k\lambda) \geq 0 \text{ for } a > 0, a \neq 1 \tag{2.32}$$

Recalling that $\zeta(a) = \sum_{k=1}^{\infty} k^{-a} > 0$ for $a \in (1, \infty)$, we have that we must show

$$\zeta(a) \geq \sum_{k=1}^{\infty} k^{-a} \cos(k\lambda) \tag{2.33}$$

which follows since $\cos(\nu) \leq 1 \ \ \forall \nu$.

For $a \in (0, 1)$ we have that $\zeta(a) < 0$ and we must show $\zeta(a) \leq \sum_{k=1}^{\infty} k^{-a} \cos(k\lambda) = g(a, \lambda)$. First, setting $\lambda = \pi$, through *Mathematica* we find

$$\sum_{k=1}^{\infty} k^{-a} \cos(k\pi) \quad = \quad 2^{1-a}\zeta(a) - \zeta(a) \tag{2.34}$$

and as $a \in (0, 1)$ we have $g(a, \pi) \geq \zeta(a)$. Now we must show $\lambda = \pi$ is a minimum for g.

We note that, if $a \in (0, 1)$ is fixed, as is logical, $g(a, \lambda) = g_a(\lambda)$

$$\frac{dg_a(\lambda)}{d\lambda} \quad = \quad \frac{1}{2}\left(i\mathrm{Li}_{a-1}\left(e^{i\lambda}\right) - i\mathrm{Li}_{a-1}\left(e^{-i\lambda}\right)\right) \tag{2.35}$$

which is equal zero if and only if

$$\mathrm{Li}_{a-1}\left(e^{i\lambda}\right) \quad = \quad \mathrm{Li}_{a-1}\left(e^{-i\lambda}\right) \tag{2.36}$$

Then

$$\mathrm{Li}_{a-1}\left(e^{i\lambda}\right) - \mathrm{Li}_{a-1}\left(e^{-i\lambda}\right) = 0 \tag{2.37}$$

$$\Leftrightarrow$$

$$\sum_{k=1}^{\infty} k^{1-a} \left(\cos(k\lambda) + i\sin(k\lambda) - (\cos(k\lambda) - i\sin(k\lambda))\right) = 0 \tag{2.38}$$

by a straightforward application of de Moivre's theorem, which means that, for all k,

$$\sin(k\lambda) = -\sin(k\lambda) \tag{2.39}$$

$$\Leftrightarrow$$

$$\lambda = h\pi, \quad \forall h \in \mathbb{Z} \tag{2.40}$$

On $\lambda \in (0, 2\pi)$ and recalling $a \in (0, 1)$ is fixed, we have that

$$\lim_{\lambda \to 0} g_a(\lambda) = \lim_{\lambda \to 2\pi} g_a(\lambda) \tag{2.41}$$

$$= \infty \tag{2.42}$$

since $\lim_{\lambda\to 0}\cos(k\lambda) = 1$ and similarly for $\lambda \to 2\pi$ as $k \in \mathbb{Z}$. As such these are global maxima at the boundaries of the support of the function. Then since $\lambda = \pi$ is the only other extrema on the support of g_a, it must be a global minima.

As such, the minimum of g_a is $\zeta(a)$, and so we have that $f_w(\lambda) \geq 0$ for $\lambda \in (-\pi, \pi)$. Indeed, due to the periodicity of trigonometric functions, we have $f_w(\lambda) \geq 0$ for $\lambda \in \mathbb{R}$.

□

Theorem 2.4. *The spectral density function for the* PLA *time series model may be written,*

$$f_a(\lambda) = \gamma(0)\left(1 - \frac{Li_a(e^{-i\lambda}) + Li_a(e^{i\lambda})}{2\zeta(a)}\right), \tag{2.43}$$

For λ small, $f_a(\lambda) \approx C_a\lambda^{a-1}$ where $C_a = 1$ for $a = 1$ and otherwise,

$$C_a = -\frac{\sin\left(\frac{\pi a}{2}\right)\Gamma(1-a)}{2\pi\zeta(a)} \tag{2.44}$$

Theorem 2.4 is derived in Appendix A.

2.3 Model Estimation

Statistical efficient model estimation is based on the method of maximum likelihood. For hyperbolic decay time series models and more general long-memory time series models, this method has been shown to be asymptotically efficient Fox and Taqqu [1986], Dahlhaus [1989]. In the case of ARMA models, maximum likelihood estimates (MLE) have been shown to be second-order efficient [Taniguchi, 1983] and it seems likely that this well-known advantage [Efron, 1975] of the maximum likelihood method is also shared with hyperbolic decay models although this has not been proved yet.

2.3.1 Exact Likelihood

Let $\boldsymbol{w}' = (w_1, \ldots, w_n)'$ denote a series of n successive observations from a hyperbolic decay model with mean μ, variance σ_w^2 and decay parameter α. We choose α to be the canonical parameter for the models in §2.2. The natural parameter in the respective models in §2.2 is given by $d = (1-\alpha)/2$, $H = 1 - \alpha/2$, $p = \alpha$ or $a = \alpha$ respectively. The sample mean $\bar{w}_n$ can be used in the place of μ_w under most circumstances, as in Chapter 6. Alternatively, an iterative algorithm as described in [McLeod and Zhang, 2008] can be used to obtain the exact MLE for μ_w.

Recall the exact Gaussian log-likelihood function may be written, after dropping constant terms,

$$\ell(\alpha, \sigma_w^2) = -\frac{1}{2}(\log\det(\Gamma_n) + \boldsymbol{w}'\Gamma_n^{-1}\boldsymbol{w}') \tag{2.45}$$

Let Ω_n denote the correlation matrix so that $\Gamma_n = \sigma_w^2 \Omega_n$ and let $g_n = \log\det(\Omega_n)$; then

$$\ell(\alpha, \sigma_w^2) = -\frac{1}{2}(n \log \sigma_w^2 + \boldsymbol{w}'\Omega_n^{-1}\boldsymbol{w}' + g_n). \tag{2.46}$$

Maximizing over σ_w^2 and dropping the additive constant, the concentrated log-likelihood can be written,

$$\ell_c(\alpha) = (-n/2)\log S/n - (1/2)g_n \tag{2.47}$$

where $S = \boldsymbol{w}'\Omega_n^{-1}\boldsymbol{w}'$ and $\sigma_w^2 = S/n$. A similar expression for the concentrated log-likelihood may be derived in the ARMA case using $\Gamma_n = \sigma_a^2 M_n$ and optimizing over σ_a^2 (McLeod [1977]). As pointed out by Li [1981], ℓ_c may be evaluated using the Durbin-Levinson algorithm. Although said algorithm has complexity $O(n^2)$, it is feasible provided n is not too large. Our ***R*** package [McLeod and Veenstra, 2012] uses this method to optimize $\ell(\alpha)$ and obtain the exact maximum likelihood estimates (MLE).

In the ARMA case, an approximate log-likelihood algorithm (McLeod and Zhang [2008]) based on using a high-order AR(P) approximation has complexity $O(P^2)$ in repeated likelihood evaluations after an initial setup. However an adequate approximation to long-memory hyperbolic models requires a large P and so this method is not very useful. As we will show, for large n the Whittle approximation is useful.

2.3.2 Whittle Likelihood

Whittle [1963] derived the likelihood approximation

$$\ell_w(\alpha, \sigma_a^2) = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\log 2\pi f(\lambda) + \frac{I(\lambda)}{f(\lambda)}\right)d\lambda, \tag{2.48}$$

where $f(\lambda)$ is the spectral density function and the periodogram,

$$I(\lambda) = \frac{1}{2\pi n}\left|\sum_{t=1}^{n} w_t e^{it\lambda}\right|^2. \tag{2.49}$$

He showed that with Gaussian short-range dependent time series maximizing ℓ_w produces asymptotically efficient estimates. Walker [1964] extended this asymptotic theory to the non-Gaussian case. Fox and Taqqu [1986], Dahlhaus [1989] further generalized and extended this asymptotic theory to long-memory time series.

Several algorithms for computing estimates based on the Whittle approximation have been discussed by [Priestley, 1981, §5.4.3] and [Hannan, 1970, §VI.5]. Beran [1994, Ch. 6] discusses Whittle approximate maximum likelihood method for long-memory time series and this method is implemented in R [Beran, 2011] for the FGN and FD models. In this section a new method is derived that we have implemented in our package [McLeod and Veenstra, 2012].

The spectral density function for the models in §2.2 may be expressed in the form,

$$f_\alpha(\lambda) = \begin{cases} \sigma_a^2(2\pi)^{-1}g_\alpha(\lambda), & \text{FD case,} \\ \sigma_w^2(2\pi)^{-1}g_\alpha(\lambda), & \text{FGN, PLS, PLA cases,} \end{cases} \tag{2.50}$$

In the FD case, the Whittle log-likelihood may be simplified using Kolmogoroff's formula [Brockwell and Davis, 1991]. After simplifying and defining the deviance, D, to be the negative of twice the log-likelihood,

$$D(\alpha, \sigma_a^2) = n \log \sigma_a^2 + \frac{1}{2\pi\sigma_a^2} \int_{-\pi}^{\pi} \frac{I(\lambda)}{g(\lambda)} d\lambda. \tag{2.51}$$

Approximating the integral using a Riemann sum at the Fourier frequencies,

$$D(\alpha, \sigma_a^2) = n \log \sigma_a^2 + \frac{2}{\sigma_a^2} \sum_{j=1}^{m} \frac{I_j}{g_j}, \tag{2.52}$$

where $I_j = I(\lambda_j)$, $g_j = g(\lambda_j)$, $\lambda_j = 2\pi j/n, j = 1, \ldots, m$, $m = [n/2]$. Maximizing over σ_a^2 and dropping the additive constant,

$$D(\alpha) = n \log \left(\frac{2}{n} \sum_{j=1}^{m} \frac{I_j}{g_j} \right), \tag{2.53}$$

and $\hat{\sigma}_a^2 = m^{-1} \sum_j I_j/g_j$. For the FGN and related cases, approximating (2.48) using a Riemann sum,

$$D(\alpha, \sigma_w^2) = 2 \sum_{j=1}^{m} \log(2\pi\sigma_w^2 g_j) + \frac{2}{\sigma_w^2} \sum_{j=1}^{m} \frac{I_j}{g_j}. \tag{2.54}$$

Setting $\partial D/\partial\sigma_w^2 = 0$ and solving, the MLE for σ_w^2 is obtained, $\hat{\sigma}_w^2 = m^{-1} \sum_j I_j/g_j$. Substituting for σ_w^2, simplifying and dropping the additive constant,

$$D(\alpha) = 2 \sum_{j=1}^{m} \log \left(\frac{2\pi}{m} g_j \sum_{i=1}^{m} \frac{I_i}{g_i} \right). \tag{2.55}$$

Approximate MLE are obtained by minimizing the appropriate deviance (2.53) or (2.55). Minimizing (2.55) produces a numerically different but asymptotically equivalent estimate as compared with the previous method [Beran, 1994, 2011].

2.3.3 Statistical Inference

Asymptotically we have $\sqrt{n}(\hat{\alpha} - \alpha)$ converges to a normal distribution with mean zero and variance σ_α^2, with $\sigma_\alpha^2 = I(\alpha)^{-1}$ and $I(\alpha)$ the Fisher large-sample information per observation. For the FD case, using a linear process approximation, Li [1981], Li and McLeod [1986] obtained, $I(\alpha) = \pi^2/6 \approx 1.64$. Using the Whittle approximation [Whittle, 1963],

$$I(\alpha) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left(\frac{\partial \log f_\alpha(\lambda)}{\partial \alpha} \right)^2 d\lambda. \tag{2.56}$$

For the PLS model, a closed form expression can be obtained as in Appendix A however due to the singularity it is difficult to evaluate (2.56) for the FGN and PLA models. Figure 2.3

compares $I(\alpha)$ in the FD and PLS models. For the FGN model, the value of $I(\alpha)$ was estimated by simulation and is also shown for selected values of α. It is interesting that $I(\alpha)$ differs so much between the different models.

Rather than using the expected information $I(\alpha)$, the observed information is often preferred [Cox, 2006, §6.6],

$$\hat{I}(\alpha) = \left(\frac{\partial \ell(\alpha)}{\partial \alpha}\right)\bigg|_{\alpha=\hat{\alpha}}. \tag{2.57}$$

Bootstrapping is another alternative both to estimate the standard error as well as to estimate more accurately the confidence interval [Efron and Tibshirani, 1993, Davison and Hinkley, 1997]. Although it is computationally intensive, it is quite feasible in many cases especially if a modern multi-core PC is used.

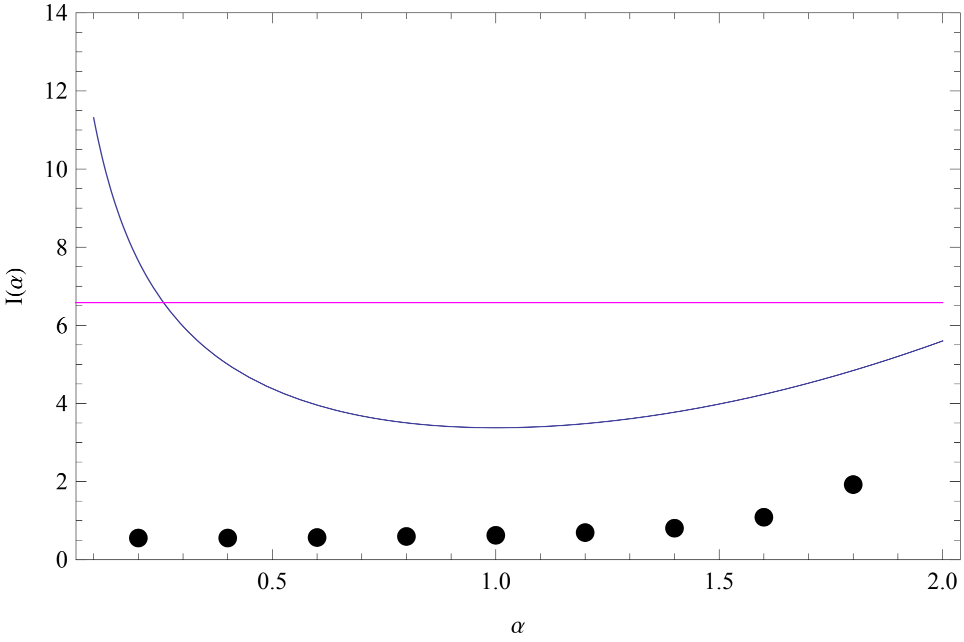


Figure 2.3: Comparing the expected Fisher information $I(\alpha)$ in the FD, PLS, and FGN models. The horizontal line at about 1.64 corresponds to the FD case, the curve shows $I(\alpha)$ for the PLS model and the plotted points show the estimated information for α based on 10^5 simulations of the FGN model.

The likelihood ratio test may also be used to obtain confidence intervals for the parameter and as a general rule, this method is often more accurate than methods based on estimating the standard error [Cox, 2006, §6.6]. The validity of this method for the models discussed in §2.2 follows from the established asymptotic theory [Fox and Taqqu, 1986, Dahlhaus, 1989].

An alternative to confidence intervals, likelihood inference provides an exact statistical inference method that was recommended by Barnard et al. [1962] for time series analysis. The general principles of this approach are described in the books by Royall [1997] and Sprott [2000]. This approach can also provide a graphical supplement to the likelihood-ratio approach to confidence intervals. The relative likelihood function, $R(\alpha)$, describes the plausibility of the parameter value α,

$$R(\alpha) = \frac{L(\alpha)}{L(\hat{\alpha})}, \tag{2.58}$$

or equivalently in terms of deviance,

$$R(\alpha) = \exp\left\{\left(D(\hat{\alpha}) - D(\alpha)\right)/2\right\}. \tag{2.59}$$

In the likelihood inference approach values of the parameter α for which $R(\alpha) < 0.05$ are relatively implausible and similarly $R(\alpha) < 0.01$ corresponds to even more implausible values. A plot of $R(\alpha)$ may used to access the range of plausible values of α. A $(1-\eta)\%$ confidence interval based on the likelihood-ratio test corresponds to the interval $R(\alpha) \geq \exp\chi_1^2(\eta)$, where $\chi_1^2(\eta)$ denotes the upper η quantile from a χ^2-distribution on 1 DF. Thus a 95% confidence interval for α corresponds to $R(\alpha) \geq 0.1465$. Our R package [McLeod and Veenstra, 2012] plots $R(\alpha)$ and obtains the 95% confidence interval using likelihood-ratio test method.

Box et al. [2008b], Li [2004] discuss the importance of model diagnostic checking and suggest the Ljung-Box portmanteau diagnostic check based on the residual autocorrelations,

$$\hat{r}(k) = \sum_{t=k+1}^{n} \hat{a}_t \hat{a}_{t-k} / \sum_{t=1}^{n} \hat{a}_t^2 \quad k = 1, 2, \cdots, \tag{2.60}$$

where $\hat{a}_t$ denotes residual or estimated innovation in (1.14).

For the FD model, Li and McLeod [1986] showed that the Ljung-Box statistic

$$Q_m = n(n+2)\sum_{k=1}^{m}(n-k)^{-1}\hat{r}(k)^2, \tag{2.61}$$

is approximately χ_{m-1}^2 distributed under the assumption of model adequacy. Another portmanteau diagnostic test [Peňa and Rodriguez, 2002, Mahdi and McLeod, 2012] may be written,

$$D_m = -n \log\det(\hat{R}_m), \tag{2.62}$$

where $\hat{R}_m$ is the $m \times m$ matrix $(\hat{r}_{i-j})$. A Monte-Carlo testing approach was used to show that D_m provides a more powerful diagnostic check for detecting model inadequacy due to long-memory with FD alternatives [Lin and McLeod, 2006]. We are unaware of portmanteau tests for other HD models.

For convenience we summarize the steps in the Monte-Carlo procedure making minor changes in the method given in Mahdi and McLeod [2012] for vector ARMA case. For the models discussed in this paper, we may use the standardized prediction residuals, $\hat{a}_t, t = 1, \ldots, n$. These residuals are discussed in [McLeod et al., 2007a, §2.8] and may be computed in R using the function `DLResiduals()` [McLeod et al., 2012].

1. Fit the model using exact or Whittle MLE. Compute the residuals, $\hat{a}_t, t = 1, \ldots, n$ and the residual autocorrelations (2.60) and the portmanteau test statistic, where $S_m = Q_m$ or D_m. Denote the observed value of this statistic by S_{obs}.

2. Select the number of Monte-Carlo simulations, N. Typically $100 \leq N \leq 1000$.

3. Simulate the model using the estimated parameters obtained in Step 1. Refit the simulated model using maximum likelihood to estimate the parameters, residuals and obtain the test statistic S_m.

4. Perform N replications of Step 3. Count the number of times, k, that the value of S_m is greater than or equal to S_{obs}.

5. The p-value for the test is $(k + 1)/(N + 1)$.

2.3.4 Illustrative Example

The Nile river flow minima, 660-1320, comprising $n = 663$ observations is a famous example of a time series that is well-fit by a long-memory time series model. This series was originally used by Hurst [1951] and some discussion of the data is given in Percival and Walden [2000, §5.9] and Beran [1994, §1.4]. The Hurst K statistic [Hipel and McLeod, 1994], $K = s^{-1}(\max_t R_t / \min_t R_t) = 0.825$, where $R_t, t = 1, \ldots, n$ is the cumulative range and s is the sample standard deviation, provides a simple, fast, and consistent estimate of H in the FGN model [Mandelbrot and Van Ness, 1968, Corollary 3.6].

In the Table 2.1, we compare the fits to this series using the four models in §2.2. Each model is fit using exact and Whittle MLE. For comparison purposes, the exact log-likelihood is computed for each of the fit model and the relative likelihood of the best fit vs each of the other fits is shown in the column with heading R. The computer time required for fitting is also shown. The best fit is given with FGN and then from better to worst are PLS, PLA, and FD. But even FD with the lowest value, $R = 61\%$, has high plausibility, so all the models fit about equally well in terms of likelihood. As expected the fit, in terms of exact likelihood, is only slightly less good when the Whittle MLE is used in each case. The timings indicate the feasibility of exact MLE for series of moderate length. The timings for the exact MLE were generally faster than for the approximate Whittle algorithm largely due to programming details. In the exact case, an interface to a C function was used. The ***R*** script for reproducing is provided in Appendix A. The ***R*** package **longmemo** [Beran, 2011] uses the Whittle method and produced estimates $\hat{H} = 0.837$ and $\hat{d} = 0.399$ for the FGN and FD models respectively. The estimates for α, 0.326 and 0.202 respectively, agree very closely. The ***R*** function `fracdiff()` [Fraley, 2012] uses exact MLE for the FD model and produces $\hat{d} = 0.393$.

Our R package also computes a 95% confidence interval and produces plots of the relative likelihood function for α. Table 2.2 compares the confidence intervals computed by solving the equation $R(\alpha) = 0.1465$. When the Whittle method is used, $R(\alpha)$ is computed using the deviance defined in (2.53) or (2.55). Using the expected information [Li and McLeod, 1986], $\sigma_{\hat{\alpha}} \approx 0.0605$ which implies the 95% confidence $(0.091, 0.329)$ for α in the FD model with exact MLE and this agrees precisely with the likelihood-ratio 95% confidence interval in Table 2.2.

Figure 2.4 compares the relative likelihood functions. We see that the likelihood functions are well behaved and approximately quadratic around the maximum and in such a case all the

	Exact MLE			Whittle MLE		
	$\hat{\alpha}$	R	time	$\hat{\alpha}$	R	time
FD	0.21	0.61	0.08	0.20	0.59	0.01
FGN	0.34	1.00	0.01	0.32	0.97	0.32
PLS	0.25	0.88	0.03	0.21	0.66	3.40
PLA	0.23	0.80	0.02	0.21	0.73	2.77

Table 2.1: Exact and Whittle MLE estimates $\hat{\alpha}$, relative likelihood, R, and computer time required.

	Exact	Whittle
FD	$(0.09, 0.33)$	(0.08, 0.32)
FGN	$(0.24, 0.43)$	(0.25, 0.39)
PLS	$(0.14, 0.36)$	(0.13, 0.29)
PLA	$(0.12, 0.34)$	(0.13, 0.29)

Table 2.2: 95% confidence intervals for α based on the likelihood-ratio test.

confidence interval methods discussed in §2.3.3 should agree.

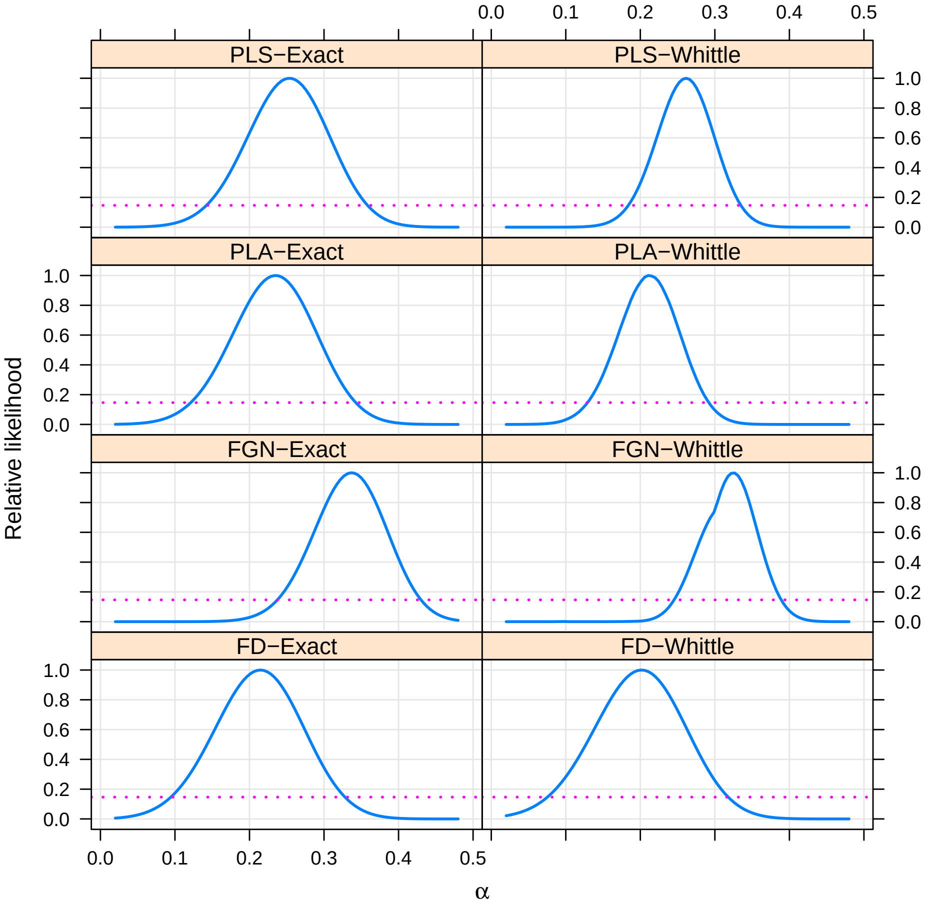


Figure 2.4: Comparison of the relative likelihood functions for the different models estimated with exact MLE and using the Whittle approximate MLE.

2.3.5 Extensions

A two-parameter version of each of the models in §2.2 can be obtained using the parameter τ defined in Theorem 2.5 below.

Theorem 2.5. *Let $\rho(k)$ denote the autocorrelation function at lag $k \geq 1$ for a stationary time series. Then the time series with variance $\gamma(0) = \tau\sigma_w^2 > 0$ and autocovariance function $\gamma(k) = \sigma_w^2\tau\rho(k), k \neq 0$ exists and is stationary.*

Proof. The result follows from the definition of the spectral density function in 1.10 since the resulting spectral density must be positive and integrate to σ_z^2 □

Using the parameter τ allows more flexibility in fitting and a simple method of including short-range autocorrelation. A more flexible family of models generated by convolving the autocovariance function for the models in §2.2 with a short-range dependent model such ARMA or the exponential spectrum model (Bloomfield [1973]). When the FD model is used, convolving with the ARMA, the ARFIMA model is obtained. Similarly convolving with the exponential spectrum model the FEXP model is obtained (Beran [1994, 1992], Craigmile and Guttorp [2011]). Convolution of the autocovariance function corresponds to multiplication of the spectral densities.

Time series models that exhibit strong seasonal or periodic persistence have been used in diverse applications [Gray et al., 1989, Porter-Hudak, 1990, Ray, 1993, Montanari et al., 2000]. A model with long-memory periodic autocorrelation with period s may be defined using any of the models in §2.2,

$$\gamma_s(k) = \begin{cases} 0, & \text{if} \quad \text{mod } (k, s) \neq 0, \\ \gamma(k/s), & \text{if} \quad \text{mod } (k, s) = 0. \end{cases} \tag{2.63}$$

In the case of FD models, the SARFIMA models are obtained [Porter-Hudak, 1990]. We will discuss this in more detail in Chapter 3.

2.4 Conclusions

In choosing which type of HD model to use to fit a series, we recommend using information criteria such as suggested by Akaike [1974] (AIC) and Schwarz [1978] (BIC) or variants thereof. We have noticed each model has advantages in certain situations.

A ***Mathematica*** demonstration Veenstra and McLeod [2012b] is provided that allows one to further explore properties of the models discussed in §2.2. This demonstration provides visualization of simulated time series as well as the sample and theoretical autocorrelations, partial autocorrelations, spectral density function and its estimates using the periodogram and the autoregressive spectral density estimator.

The Appendix [Veenstra and McLeod, 2012a], in non-interactive form in Appendix A provides detailed derivations, interactive displays comparing the autocorrelations and spectral density functions.

Li and Li [2008] and references therein discuss the estimation of long-memory models with heteroscedastic innovations.

Very long time series occur occasionally and for such time series wavelet analysis (Percival and Walden [2000], Moulines et al. [2006]) provides a useful approach. Also for very long time series, the Hurst K provides a simple and fast estimate.

The PLA may be useful in introducing students to the subject of long-memory time series in an introductory time series course along other simple one parameter models such as the first-order autoregression and exponentially smoothing. Similarly, when introducing frequency domain analysis, the PLS provides a simple illustration of a power-law spectrum.

Chapter 3

On the Combination of ARMA and HD Processes

3.1 Introduction

We now will look at the combination of ARMA structure with HD structure. By this we mean having an ARMA model being driven by noise that is HD. To do this, we must somehow meaningfully compute the TACVFs of these combined processes, which we discuss in §3.2.

3.2 Computing the TACVFs

We consider the various hyperbolic decay processes, and would like to have short term memory structure mixed with the long memory or anti-persistent processes for a richer class of models.

We could integrate these processes ARIMA-type processes as well, but we will not consider that for now. While the form of the ARFIMA process TACVF has a known structure (see, e.g. Sowell [1992]), it can be very complicated. Also, the combination of an ARMA structure with FGN or PLA noise has no known closed form, since those forms of noise cannot be written in operator notation.

3.2.1 A Solution via Convolution

We suppose that the general form of the ARMA-HD processes exist in the world and we wish to estimate them. This is not an unreasonable assumption, as we know, for example, the ARFIMA class of processes exists and are important. When we ask how the processes are to be estimated or simulated, it seems we can go no further if we prefer not to use Sowell's formulae or want to use the other types of HD processes. However, we have a proposition from Brockwell and Davis [1991] (Proposition 3.1.2) that comes to our aid which we will state without proof.

Proposition 3.1. *If $\boldsymbol{x}$ is a stationary process with theoretical autocovariance function γ_x and $\sum_{j=-\infty}^{\infty} |\psi_j| < \infty$, then the series*

$$w_t = \psi(B)x_t \tag{3.1}$$

is stationary and converges almost surely and in mean square to the same limit. Also, w_t has the TACVF

$$\gamma_w(h) = \sum_{j,k=-\infty}^{\infty} \psi_j\psi_k\gamma_x(h-j+k) \tag{3.2}$$

In particular, we can have the ψ_js defined by a stationary invertible ARMA process. In that case, we have

$$\psi(c) = \sum_{j=0}^{\infty} \psi_j c^j \tag{3.3}$$

$$= \frac{\theta(c)}{\phi(c)}, \tag{3.4}$$

which is convergent for $c \in \mathbb{C},\ |c| \leq 1$. We let y_t be this ARMA process, having without loss of generality zero mean. Then y_t can be written in random shock form $y_t = \sigma_a \sum_{j=0}^{\infty} \psi_j a_{t-j}$. Note that we have $\psi_{-j} = 0$ for $j > 0$. We let, once again without loss of generality, $\sigma_a = 1$. We note that $\gamma_y(k) = \sum_{j=0}^{\infty} \psi_j\psi_{j+k}$ for $k \geq 0$ and since we are dealing with weakly stationary processes we have $\gamma_y(-k) = \gamma_y(k)$. Then we note that the definition of the autocovariance function for w_t has

$$\gamma_w(h) = \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} \psi_j\psi_k\gamma_x(h-j+k) \tag{3.5}$$

$$= \sum_{k=-\infty}^{\infty}\sum_{j=0}^{\infty} \psi_j\psi_{j+k}\gamma_x(h-k) \tag{3.6}$$

$$= \sum_{k=-\infty}^{\infty} \gamma_y(k)\gamma_x(h-k) \tag{3.7}$$

where (3.7) is sometimes called the splitting method: see, eg. Palma [2007].

We now define convolution in the discrete case.

Definition 3.1 (Discrete Convolution)**.** *The convolution of two discrete sequences f and g, called h, is defined $\forall n = \mathbb{Z}$ as*

$$h(n) = \sum_{m=-\infty}^{\infty} f(m)g(n-m) \tag{3.8}$$

$$= \sum_{m=-\infty}^{\infty} f(n-m)g(m) \tag{3.9}$$

and as such we see that the autocovariance function of w_t is a discrete convolution of the autocovariance functions of y_t and x_t.

We introduce the following notation: call the symmetrized TACVF of up to lag m

$$\begin{aligned} \mathcal{S}(\gamma_w(0), \dots, \gamma_w(m)) &= (\gamma_w(m-1), \gamma_w(m-2), \dots, \gamma_w(0), \gamma_w(1), \dots, \gamma_w(m)) && (3.10) \\ &= (\gamma_w(-m+1), \gamma_w(-m+2), \dots, \gamma_w(0), \gamma_w(1), \dots, \gamma_w(m)) && (3.11) \end{aligned}$$

where (3.11) holds if the process is covariance stationary.

Then we have the following corollary to Proposition 3.1.

Corollary 3.1. *We can compute the first $m+1$ autocovariances of any* ARMA-HD *process arbitrarily exactly through the convolution of the respective symmetrized* ARMA *TACVF and symmetrized* HD *TACVF, both computed up to lag $L \geq 2m$.*

Proof. We recall that the autocovariances of an ARMA process decay geometrically: that is, $\gamma_y(k) = O(r^k)$ for $r \in (0,1)$. Also by definition the autocovariances of a HD process decay hyperbolically: $\gamma_x(k) = O(k^{-\alpha}) \subset O(1)$ for $\alpha \in (0,3)$.

We know the (3.7) holds. Thus we have that

$$\begin{aligned} \gamma_w(h) &= \sum_{k=-\infty}^{\infty} \gamma_y(k)\gamma_x(h-k) && (3.12) \\ &= \sum_{k=-L+1}^{L} \gamma_y(k)\gamma_x(h-k) + \sum_{k=-\infty}^{-L} \gamma_y(k)\gamma_x(h-k) + \sum_{k=L+1}^{\infty} \gamma_y(k)\gamma_x(h-k) && (3.13) \\ &= \sum_{k=-L+1}^{L} \gamma_y(k)\gamma_x(h-k) + O(r^L) && (3.14) \end{aligned}$$

for $h = 0, \dots, m$, where (3.14) is by the decay of the autocovariances.

Therefore, choosing L sufficiently large, we can have an arbitrarily close approximation to the TACVF of w_t (up to some function of machine epsilon) via symmetrizing and convolving the TACVFs of y_t and x_t. □

We note that we most often perform the convolution via the fast Fourier transform (FFT) and inverse FFT. Since said operations are most efficient on sizes of power of 2, we let the minimum value of L be such that $L = 2^c \geq 2m$, where c is the smallest integer for which the relation holds. We have $L+1$ terms in each TACVF: then the symmetrized TACVFs have length $2(L+1) - 2 = 2^{c+1}$.

We also have the following corollary about TACVFs and TACFs.

Corollary 3.2. *Any scalar multiplication by $a \in \mathbb{R}_{\neq 0}$ of the TACVFs in Corollary 3.1 before convolution can be undone via a scalar division by a. In particular, it does not matter whether we convolve TACVFs or TACFs or a combination thereof, as long as we multiply by the correct value to obtain the convolved TACF or TACVF.*

Proof. If we have $a \in \mathbb{R}$ and nonzero, it is easy to see that multiplying it to one of the TACVFs has all operations involving the multiplied values. Therefore we can just as easily divide by a at the end.

Secondly, suppose we have the TACVF of one process, and the TACF of the other. Then we can convolve the two and obtain something that seems nonsensical. However, if we remember that any TACF has 1 at lag 0, we must simply divide all lags of the convolved operator by the lag 0 term to obtain the TACF of the process defined by the convolution. Additionally, if we know the true value of $\gamma_w(0)$ of the convolved process, we can multiply it to obtain the TACVF. □

We note that we have only dealt with nonseasonal processes. We have the following corollary that lets us deal with seasonal processes.

Corollary 3.3. *Suppose we have the TACVFs of two processes, one of which should be seasonal. Then we have the following results:*

1. *We can "shift" the seasonal TACVF by its period, s, to obtain the seasonal TACVF with period s*
2. *We can then convolve the two TACVFs to make a multiplicative seasonal TACVF.*

Proof. By shifting we mean $\forall s \in \mathbb{Z}_{>1}$ letting the autocovariances of the seasonal process have $\gamma_s(ks) = \gamma_w(k)$ for $k \in \mathbb{Z}$, with γ_w being the TACVF of the associated nonseasonal process. We also have $\gamma_s(t) = 0$ for t not a multiple of s. Then the claim in 1 follows. Since the process defines the TACVF, with $B^k \mapsto B^{sk}$ in the process, we have that the γ_s is the theoretical autocovariance function of the seasonal process.

To prove 2, we note that a straightforward application of the methods applied in Corollary 3.1 gives convergence to the true TACVF with (as is done in our **arfima** package) $O\left(t^{L^\star} r^{L^*}\right)$ with $r, t \in (0, 1)$. Note that $L^\star$ and L^* in our package are both linear transformations of L that are greater than or equal L. □

We note that mixing seasonal and nonseasonal pure HD processes is not recommended. While it can be shown that the convolution of the TACVFs of said processes gives rise to the true TACVF, the size of L^* and $L^\star$ may become prohibitive. There are also identifiability issues. Naturally, the mixture of two or more HD processes with the same period (including the nonseasonal case) is not recommended for these reasons.

3.2.1.1 The Moving Average Case

A theorem of the calculation of the convolution of MA(q)-HD models is given.

Theorem 3.1. *We have that with* MA(q)-HD *models, as long as* $L > q + 1$, *the convolved TACVF is equal to the exact up to numerical errors.*

Proof. Since $L > q + 1$, we have that

$$\tilde{\gamma}_z(h) = \sum_{j=-L+1}^{L} \gamma_y(j)\gamma_w(h-j) \tag{3.15}$$

$$= \sum_{j=-q}^{q} \gamma_y(j)\gamma_w(h-j) \tag{3.16}$$

$$= \gamma_z(h) \tag{3.17}$$

where (3.16) holds since the MA(q) only has autocovariances up to lag q. □

3.2.2 On the Kullback-Liebler Divergence Between Distributions

The Kullback-Liebler (KL) divergence is an information-theoretic measure of discrepancy between two probability distributions. It measures the expected number of extra nats (or bits if we use the logarithm base 2) when trying to code samples from a probability distribution P rather using the distribution Q rather than P. That is, it is a measure of information lost when Q is used to approximate P. It is non-symmetric and does not satisfy the triangle inequality, and as such is not a metric. However, it is a way to measure the difference between the true distribution P and the approximate distribution Q.

Definition 3.2 (Kullback-Liebler Divergence)**.** *The KL divergence for P and Q two continuous distributions with p.d.f.s $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ respectively on $\mathbb{R}^n$ is given by*

$$D_{KL}(P\|Q) = \int_{\mathbb{R}^n} p(\boldsymbol{x}) \log\left(\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right) d\boldsymbol{x} \tag{3.18}$$

$$= \mathrm{E}_P\left[\log\left(\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right)\right] \tag{3.19}$$

We have the following lemmas.

Lemma 3.1. *For $x, y \in \mathbb{R}$ with $x > 0$ and $y \geq 0$ we have*

$$y - y\log y \leq x - y\log x \tag{3.20}$$

Proof. Using the convention that $0\log 0 \equiv 0$, this is clear when $y = 0$. Therefore choose $y > 0$. We have that Equation (3.20) is equivalent to

$$\log\left(\frac{x}{y}\right) \leq \frac{x}{y} - 1 \tag{3.21}$$

$$\Rightarrow \log t \leq t - 1, \quad t > 0 \tag{3.22}$$

with equality holding if and only if $t = 1$: that is, $x = y$. □

Lemma 3.2. *For any p.d.f.s $p(\boldsymbol{x}) \geq 0$ and $q(\boldsymbol{x}) > 0$ on $\mathbb{R}^n$, we have that*

$$-\int_{\mathbb{R}^n} p(\boldsymbol{x}) \log p(\boldsymbol{x}) d\boldsymbol{x} \leq -\int_{\mathbb{R}^n} p(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x} \tag{3.23}$$

with equality if and only if $p(\boldsymbol{x}) = q(\boldsymbol{x})$ almost everywhere.

Proof. By Lemma 3.1, for any $\boldsymbol{x} \in \mathbb{R}^n$, we have

$$p(\boldsymbol{x}) - p(\boldsymbol{x}) \log p(\boldsymbol{x}) \leq q(\boldsymbol{x}) - p(\boldsymbol{x}) \log q(\boldsymbol{x}) \tag{3.24}$$

$$\Rightarrow -\int_{\mathbb{R}^n} p(\boldsymbol{x}) \log p(\boldsymbol{x}) d\boldsymbol{x} \leq -\int_{\mathbb{R}^n} p(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x} \tag{3.25}$$

If there is equality, then

$$\int_{\mathbb{R}^n} \left(p(\boldsymbol{x}) - p(\boldsymbol{x}) \log p(\boldsymbol{x}) - q(\boldsymbol{x}) + p(\boldsymbol{x}) \log q(\boldsymbol{x})\right) d\boldsymbol{x} = 0 \tag{3.26}$$

$$\Rightarrow p(\boldsymbol{x}) - p(\boldsymbol{x}) \log p(\boldsymbol{x}) - q(\boldsymbol{x}) + p(\boldsymbol{x}) \log q(\boldsymbol{x}) = 0 \tag{3.27}$$

$$\Rightarrow p(\boldsymbol{x}) \overset{a.e.}{=} q(\boldsymbol{x}) \tag{3.28}$$

□

The following proposition holds. See, e.g. Cesa-Bianchi and Lugosi [2006].

Proposition 3.2. *$D_{KL}(P\|Q) \geq 0$ with equality if and only if $P = Q$ almost everywhere.*

Proof. First we note that

$$D_{KL}(P\|Q) = \int_{\mathbb{R}^n} p(\boldsymbol{x}) \log p(\boldsymbol{x}) d\boldsymbol{x} - \int_{\mathbb{R}^n} p(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x} \tag{3.29}$$

$$\geq 0 \tag{3.30}$$

where (3.30) holds by Lemma 3.2. Then we have that (3.28) completes the proof. □

3.2.2.1 The KL divergence between two normal distributions

We note that any Gaussian process can be thought of as a draw from a multivariate normal distribution with a given mean and covariance matrix equal to the Toeplitz matrix of its theoretical autocovariances.

We have the following proposition, which we derived for this thesis, although the result is not new (see, e.g. Wikipedia [2013].)

Proposition 3.3. *Let $P = \text{MVN}(\boldsymbol{\mu}_1, \Sigma_1)$ be a multivariate normal distribution of size n. Similarly, let $Q = \text{MVN}(\boldsymbol{\mu}_2, \Sigma_2)$ be another multivariate normal distribution of size n. Then the KL divergence between P and Q is*

$$D_{KL}(P\|Q) = \frac{1}{2}\left(\text{tr}\left(\Sigma_2^{-1}\Sigma_1\right) + (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)' \,\Sigma_2^{-1}\,(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) - \log\left(\det\left(\Sigma_2^{-1}\Sigma_1\right)\right) - n\right) \tag{3.31}$$

Proof. We have that

$$D_{KL}(P\|Q) = \mathrm{E}_P\left[\log p(\boldsymbol{x}) - \log q(\boldsymbol{x})\right] \tag{3.32}$$

$$= \frac{1}{2}\mathrm{E}_P\Big[-\log\left(\det\left(\Sigma_1\right)\right) - (\boldsymbol{x}-\boldsymbol{\mu}_1)'\Sigma_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + \log\left(\det\left(\Sigma_2\right)\right) + (\boldsymbol{x}-\boldsymbol{\mu}_2)'\Sigma_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)\Big] \tag{3.33}$$

$$= -\frac{1}{2}\log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - \frac{1}{2}\mathrm{E}_P\Big[(\boldsymbol{x}-\boldsymbol{\mu}_1)'\Sigma_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) - (\boldsymbol{x}-\boldsymbol{\mu}_2)'\Sigma_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)\Big] \tag{3.34}$$

$$= -\frac{1}{2}\log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - \frac{1}{2}\mathrm{E}_P\Big[\operatorname{tr}\Big(\Sigma_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1)(\boldsymbol{x}-\boldsymbol{\mu}_1)'\Big) - \operatorname{tr}\Big(\Sigma_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)(\boldsymbol{x}-\boldsymbol{\mu}_2)'\Big)\Big] \tag{3.35}$$

$$= -\frac{1}{2}\log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - \frac{1}{2}\operatorname{tr}\Big(\Sigma_1^{-1}\Sigma_1\Big) + \frac{1}{2}\mathrm{E}_P\Big[\operatorname{tr}\Big(\Sigma_2^{-1}\left(\boldsymbol{x}\boldsymbol{x}' - 2\boldsymbol{x}\boldsymbol{\mu}_2' + \boldsymbol{\mu}_2\boldsymbol{\mu}_2'\right)\Big)\Big] \tag{3.36}$$

$$= -\frac{1}{2}\log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - \frac{n}{2} + \frac{1}{2}\operatorname{tr}\Big(\Sigma_2^{-1}\left(\Sigma_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1' - 2\boldsymbol{\mu}_1\boldsymbol{\mu}_2' + \boldsymbol{\mu}_2\boldsymbol{\mu}_2'\right)\Big) \tag{3.37}$$

$$= -\frac{1}{2}\log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - \frac{n}{2} + \frac{1}{2}\operatorname{tr}\Big(\Sigma_2^{-1}\Sigma_1\Big) + \frac{1}{2}\operatorname{tr}\Big(\boldsymbol{\mu}_1'\Sigma_2^{-1}\boldsymbol{\mu}_1 - 2\boldsymbol{\mu}_1'\Sigma_2^{-1}\boldsymbol{\mu}_2 + \boldsymbol{\mu}_2'\Sigma_2^{-1}\boldsymbol{\mu}_2\Big) \tag{3.38}$$

$$= \frac{1}{2}\Big(\operatorname{tr}\Big(\Sigma_2^{-1}\Sigma_1\Big) + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)'\,\Sigma_2^{-1}\,(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) - \log\Big(\det\Big(\Sigma_2^{-1}\Sigma_1\Big)\Big) - n\Big) \tag{3.39}$$

□

3.2.2.2 A Limit Theorem

We note that the KL divergence is not a metric. Although it satisfies non-negativity and the equality constraint, it is not symmetric in its arguments nor does it satisfy the triangle inequality. Thus we introduce the total variation on the space $\mathfrak{F}$ of continuous distribution functions with support $D \subseteq \mathbb{R}^n$. Note that other definitions of distribution function spaces $\mathfrak{F}$ are relatively easy extensions.

Definition 3.3 (Total Variation)**.** *The total variation between any two distribution functions $P, Q \in \mathfrak{F}$ with densities $p(x), q(x) : D \mapsto \mathbb{R}$ respectively is defined as*

$$d_{TV}(P,Q) = \frac{1}{2}\int_D |p(\boldsymbol{x}) - q(\boldsymbol{x})| d\boldsymbol{x} \tag{3.40}$$

We have the following proposition, which is a relatively standard result (see, e.g. Cesa-Bianchi and Lugosi [2006]):

Proposition 3.4. *d_{TV} is a metric on $\mathfrak{F}$.*

Proof. We have that, more formally, $\mathfrak{F}$ is equivalent to $(\Omega, \mathcal{F}, M)$, where Ω is a set of events, $\mathcal{F}$ is a σ-algebra on Ω, and M is a probability measure on $\mathcal{F}$. The equivalence holds since while

$\mathfrak{F}$ is a space of distribution functions, they are defined by their densities. That is, any operation on this space is defined by the density $p(x)$ of $P \in \mathfrak{F}$. In particular we have, $\Omega = D$, $\mathcal{F}$ is Borel on D and M is the Lebesgue measure.

We note then that $L_1(\Omega, \mathcal{F}, M)$ is a normed vector space with norm

$$\|f\| = \int_D |f| dx \tag{3.41}$$

for all $f \in (\Omega, \mathcal{F}, M)$, and dx is with respect to M. Then the induced metric on this space is $d(f, g) = \|f - g\| = 2d_{TV}(F, G)$. As such we have that $(\Omega, \mathcal{F}, M)$ is a metric space under d_{TV}. Since $\mathfrak{F}$ is equivalent to $(\Omega, \mathcal{F}, M)$, we have that $\mathfrak{F}$ is a metric space under d_{TV}. □

We have the following theorem due to Pinsker (see, e.g. Cover and Thomas [1991]):

Theorem 3.2 (Pinsker's Inequality)**.** *We have that for all* $P, Q \in \mathfrak{F}$

$$d_{TV}^2(P, Q) \leq \frac{1}{2} D_{KL}(P \| Q) \tag{3.42}$$

Proof. We note that

$$u \log(u) - u + 1 \geq 0 \tag{3.43}$$

for all $u \in \mathbb{R}_{>0}$ is easy to show. Then define $g(u) = 3(u-1)^2 - (2u+4)(u \log(u) - u + 1)$. We have that $g(1) = g'(1) = 0$ and that by (3.43) that $g''(u) = -4u^{-1}(u \log(u) - u + 1) \leq 0$ for $u > 0$. As such $g(u) \leq 0$ for $u > 0$. Thus

$$3(u-1)^2 \leq (2u+4)(u \log(u) - u + 1) \tag{3.44}$$

Let $u(\boldsymbol{x}) = p(\boldsymbol{x})/q(\boldsymbol{x})$, where we must restrict q to be strictly positive on D. This also ensures that the KL divergence is defined. Then

$$\begin{aligned}
d_{TV}^2(P, Q) &= \frac{1}{4} \left[\int_D |p(\boldsymbol{x}) - q(\boldsymbol{x})| d\boldsymbol{x} \right]^2 && (3.45)\\
&= \frac{1}{4} \left[\int_D q(x) |u(\boldsymbol{x}) - 1| d\boldsymbol{x} \right]^2 && (3.46)\\
&\leq \frac{1}{12} \left[\int_D q(\boldsymbol{x}) \sqrt{2u(\boldsymbol{x}) + 4} \sqrt{u(\boldsymbol{x}) \log u(\boldsymbol{x}) - u(\boldsymbol{x}) + 1} d\boldsymbol{x} \right]^2 && (3.47)\\
&\leq \frac{1}{12} \left(\int_D q(\boldsymbol{x})(2u(\boldsymbol{x}) + 4) \right) \left(\int_D q(\boldsymbol{x})(u(\boldsymbol{x}) \log u(\boldsymbol{x}) - u(\boldsymbol{x}) + 1) d\boldsymbol{x} \right) && (3.48)\\
&= \frac{1}{12} \left(2 \int_D p(\boldsymbol{x}) d\boldsymbol{x} + 4 \right) \left(\int_D p(\boldsymbol{x}) \log u(\boldsymbol{x}) d\boldsymbol{x} - \int_D p(\boldsymbol{x}) d\boldsymbol{x} + 1 \right) && (3.49)\\
&= \frac{6}{12} \int_D p(\boldsymbol{x}) \log u(\boldsymbol{x}) d\boldsymbol{x} - \frac{6}{12} + \frac{6}{12} && (3.50)\\
&= \frac{1}{2} D_{KL}(P \| Q) && (3.51)
\end{aligned}$$

where we have (3.47) by (3.44) and (3.48) by the Cauchy-Schwartz Inequality. □

Then we have

Proposition 3.5. *If $D_{KL}(P\|Q_n) \to 0$ as $n \to \infty$ with $X \sim P$ and $X_n \sim Q_n$ for each n, then $X_n \overset{\mathcal{D}}{\to} X$. That is, X_n converges in distribution to X.*

Proof. We have that $D_{KL}(P\|Q_n) \to 0 \Rightarrow d_{TV}(P, Q_n) \to 0$ as $n \to \infty$. Since $\mathfrak{F}$ is a metric space under d_{TV}, the standard notions of convergence apply. That is, for any $\boldsymbol{x} \in D$ and

$$\begin{aligned}
D_{KL}(P\|Q_n) &\to 0 \text{ as } n \to \infty && (3.52)\\
&\Rightarrow \\
d_{TV}(P, Q_n) &= \frac{1}{2}\int_D |p(\boldsymbol{x}) - q_n(\boldsymbol{x})| d\boldsymbol{x} && (3.53)\\
&\to 0 \text{ as } n \to \infty && (3.54)\\
&\Leftrightarrow \\
q_n(\boldsymbol{x}) &\to p(\boldsymbol{x}) \text{ as } n \to \infty && (3.55)\\
&\Leftrightarrow \\
\Pr(X_n \leq \boldsymbol{x}) &\to \Pr(X \leq \boldsymbol{x}) \text{ as } n \to \infty && (3.56)\\
\text{i.e. } X_n &\overset{\mathcal{D}}{\to} X
\end{aligned}$$

□

Note that D_{KL} induces a topology on $\mathfrak{F}$ and as such we could have used a topological argument for convergence instead. However, this is less intuitive.

We will let the theoretical mean the exact and approximate distributions be the same. Then we have the following theorem:

Theorem 3.3. *Any stationary Gaussian series $\boldsymbol{w}$ generated by a given ARMA-HD process can have its covariance matrix estimated arbitrarily accurately (up to machine epsilon, the model, and the series itself) through Toeplitz matrix of the convolution of the correct an ARMA and HD processes. Thus we can approximate the distribution of w_t arbitrarily accurately, given that we know the parameters.*

Proof. We let $\Sigma_1 = \Gamma_n = [\gamma_w(i-j)]_{i,j=1}^n$ be the Toeplitz matrix for P, and $\Sigma_2 = \tilde{\Gamma}_{n,L} = [\tilde{\gamma}_{w,L}(i-j)]_{i,j=1}^n$ be the Toeplitz matrix for Q_L. Since $\gamma_w = \tilde{\gamma}_{w,L} + O(r^L)$, we have that $\Sigma_1 = \Sigma_2 + O(r^L)$, where $O(r^L)$ denotes a matrix of terms that are $O(r^L)$. We let the means of the exact and approximate distributions be equal. Then, with I_n being the $n \times n$ identity matrix,

$$D(P\|Q_L) = \frac{1}{2}\left(\operatorname{tr}\left(\Sigma_2^{-1}\Sigma_1\right) + (\mu_2 - \mu_1)' \, \Sigma_2^{-1} \, (\mu_2 - \mu_1) - \log\left(\det\left(\Sigma_2^{-1}\Sigma_1\right)\right) - n\right) \tag{3.57}$$

$$= \frac{1}{2}\left(\operatorname{tr}\left(\Sigma_2^{-1}\left(\Sigma_2 + O(r^L)\right)\right) - \log\left(\det\left(\Sigma_2^{-1}\left(\Sigma_2 + O(r^L)\right)\right)\right) - n\right) \tag{3.58}$$

$$= \frac{1}{2}\left(\operatorname{tr}\left(I_n + O(r^L)\right) - \log\left(\det\left(I_n + O(r^L)\right)\right) - n\right) \tag{3.59}$$

$$= \frac{1}{2}\left(n + O(r^L) - \log\left(1 + O(r^L)\right) - n\right) \tag{3.60}$$

$$= O(r^L) \tag{3.61}$$

Since $0 < r < 1$, we have that we can make the KL divergence between the approximate and the exact normal distributions arbitrarily small. Therefore we can say that the distributions are arbitrarily close: in fact, that as $L \to \infty$, the approximate converges to the exact (up to machine epsilon and the model). We note that rate of convergence is $O\left(r^{L/2}\right)$ under the L_1 metric. □

We now let P be a non-normal process with mean $\boldsymbol{\mu}$ and (auto-)covariance matrix Σ. We want to approximate it with the Q, a normal process. We have the following theorem.

Theorem 3.4. *The normal process that is "closest" to P in terms of KL divergence is $Q = \text{MVN}(\boldsymbol{\mu}, \Sigma)$.*

Proof. Let $\boldsymbol{\mu}_1$ be the mean of Q and Σ_1 be the covariance matrix of Q. Then we must show $\boldsymbol{\mu}_1 = \boldsymbol{\mu}$ and $\Sigma_1 = \Sigma$. Let n be the sample size.

$$D(P\|Q) = \mathrm{E}_P\left[\log p(\boldsymbol{x}) - \log q(\boldsymbol{x})\right] \tag{3.62}$$

$$= \mathrm{E}_P\left[\log p(\boldsymbol{x})\right] - \mathrm{E}_P\left[\log q(\boldsymbol{x})\right] \tag{3.63}$$

$$= \mathrm{E}_P\left[\log p(\boldsymbol{x})\right] - \frac{1}{2}\mathrm{E}_P\left[-(\boldsymbol{x} - \boldsymbol{\mu}_1)^T\Sigma_1^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_1) - \log|\Sigma_1| - n\log 2\pi\right] \tag{3.64}$$

$$= \mathrm{E}_P\left[\log p(\boldsymbol{x})\right] + \frac{1}{2}\left(\log|\Sigma_1| + n\log 2\pi + \mathrm{E}_P\left[(\boldsymbol{x} - \boldsymbol{\mu}_1)^T\Sigma_1^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_1)\right]\right) \tag{3.65}$$

Now

$$\mathrm{E}_P\left[(\boldsymbol{x} - \boldsymbol{\mu}_1)^T\Sigma_1^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_1)\right] = \mathrm{E}_P\left[(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])^T\Sigma_1^{-1}(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])\right] + (\mathrm{E}_P[\boldsymbol{x}] - \boldsymbol{\mu}_1)^T\Sigma_1^{-1}(\mathrm{E}_P[\boldsymbol{x}] - \boldsymbol{\mu}_1) \tag{3.66}$$

so the minimum for $\boldsymbol{\mu}_1$ is indeed reached when $\boldsymbol{\mu}_1 = \mathrm{E}_P[\boldsymbol{x}] = \boldsymbol{\mu}$.

Similarly, minimizing

$$\log|\Sigma_1| + \mathrm{E}_P\left[(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])^T\Sigma_1^{-1}(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])\right] = \log|\Sigma_1| + \mathrm{E}_P\left[\operatorname{tr}\left((\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])^T\Sigma_1^{-1}(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])\right)\right] \tag{3.67}$$

$$= \log|\Sigma_1| + \operatorname{tr}\left(\Sigma_1^{-1}\mathrm{E}_P\left[(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])(\boldsymbol{x} - \mathrm{E}_P[\boldsymbol{x}])^T\right]\right) \tag{3.68}$$

$$= \log|\Sigma_1| + \operatorname{tr}\left(\Sigma_1^{-1}\Sigma\right) \tag{3.69}$$

gives a minimum for Σ_1 as Σ. □

The above theorem allows us to use the same mean and covariance matrix when we approximate the true distribution, whatever it may be, with a Gaussian distribution.

3.3 On Properties of ARMA-HD Processes

We have Theorem 4.4.1 from Brockwell and Davis [1991], which we will state without proof.

Theorem 3.5. *If $\boldsymbol{y}$ is any zero mean stationary process with spectral distribution F_y, and $\boldsymbol{w}$ is the process*

$$w_t = \sum_{j=-\infty}^{\infty} \psi_j y_{t-j}, \text{ where } \sum_{j=-\infty}^{\infty} |\psi_j| < \infty \tag{3.70}$$

then $\boldsymbol{x}$ is stationary with spectral distribution function

$$F_w(\lambda) = \int_{(-\pi,\lambda]} \left| \sum_{j=-\infty}^{\infty} \psi_j e^{-ijv} \right|^2 dF_y(v), \quad -\pi \leq \lambda \leq \pi \tag{3.71}$$

In particular, the spectral density of any nonseasonal ARMA-HD process is of the form

$$f_w(\lambda) \sim \frac{\left|\theta\left(e^{-i\lambda}\right)\right|^2}{\left|\phi\left(e^{-i\lambda}\right)\right|^2} \lambda^{\alpha-1} \tag{3.72}$$

with λ near 0.

We then note it is an easy consequence of (3.72) that any ARMA-HD model is itself HD, and is persistent or anti-persistent dependent on the underlying HD process.

3.3.1 Laws of Large Numbers

We note that we will have a guarantee that the sample mean $\bar{w}_n$ converges in mean square and thus in probability to μ_w regardless of whether the process $\boldsymbol{w}$ is short memory, long memory, or anti-persistent. They will, of course, converge to μ_w at different rates. We need the results of this section for §5.5.1.

First note we have that, for any sequence of random variables $\boldsymbol{w}$,

$$\begin{aligned}\operatorname{Var}(\bar{w}_n) &= \frac{1}{n^2}\operatorname{Var}(w_1 + w_2 + \cdots + w_n) \\ &= \frac{1}{n^2}\sum_{s=1}^{n}\sum_{t=1}^{n}\operatorname{Cov}(w_s, w_t) \\ &= \frac{1}{n^2}\sum_{s-t=-n+1}^{n-1}(n-|s-t|)\gamma_w(s-t) \\ &= \frac{1}{n}\sum_{r=-n+1}^{n-1}\left(1-\frac{|r|}{n}\right)\gamma_w(r) \\ &= \frac{1}{n}\left(\gamma_w(0) + 2\sum_{r=1}^{n-1}\left(1-\frac{r}{n}\right)\gamma_w(r)\right)\end{aligned}$$

so that when the w_ts are uncorrelated, we have that $\operatorname{Var}(\bar{w}_n) = \sigma^2/n$.

Recall the following: if a process has short memory (ARMA), we have that $\gamma_w(k) \sim O(m^k)$ for $0 < m < 1$. If a process has hyperbolic decay with parameter $0 < \alpha < 3,\ \alpha \neq 1$, we have that $\gamma_w(k) \sim k^{-\alpha}$.

Then we have the following propositions.

Proposition 3.6. *If a process* $\boldsymbol{w} \sim$ ARMA, *we have that* $\operatorname{Var}(\bar{w}_n) = O(n^{-1})$.

Proof. We have that

$$\begin{aligned} n\operatorname{Var}(\bar{w}_n) &= \left(\gamma_w(0) + 2\sum_{r=1}^{n}\left(1-\frac{r}{n}\right)\gamma_w(r)\right) \\ &\sim \gamma_w(0) + \frac{2m\,(m^n + n - mn - 1)}{(m-1)^2 n} && (3.73) \\ &= \gamma_w(0) + \frac{2m\,(n(1-m))}{(1-m)^2 n} + \frac{2m\,(m^n - 1)}{(m-1)^2 n} && (3.74) \\ &\sim \gamma_w(0) + \frac{2m}{1-m} && (3.75)\end{aligned}$$

where (3.73) was found using ***Mathematica*** and (3.75) has n large. Then dividing by n, we obtain our result. □

Proposition 3.7. *If a process* $\boldsymbol{w}$ *is persistent, we have that* $\operatorname{Var}(\bar{w}_n) = O(n^{-\alpha})$, *recalling that for such a series we have* $0 < \alpha < 1$.

Proof. We have

$$\mathrm{Var}(\bar{w}_n) \sim \frac{\gamma_w(0)}{n} + \frac{c}{n}\sum_{r=1}^{n}\left(1-\frac{r}{n}\right)r^{-\alpha}\frac{n^{-\alpha}}{n^{-\alpha}} \tag{3.76}$$

$$= \frac{\gamma_w(0)}{n} + \frac{c}{n}n^{-\alpha}\sum_{r=1}^{n}\left(1-\frac{r}{n}\right)\left(\frac{r}{n}\right)^{-\alpha} \tag{3.77}$$

$$= \frac{\gamma_w(0)}{n} + cn^{-\alpha}\left(\frac{1}{n}\sum_{r=1}^{n}\left(1-\frac{r}{n}\right)\left(\frac{r}{n}\right)^{-\alpha}\right) \tag{3.78}$$

$$\sim \frac{\gamma_w(0)}{n} + cn^{-\alpha}\int_0^1 (1-x)x^{-\alpha}dx \tag{3.79}$$

$$= \frac{\gamma_w(0)}{n} + n^{-\alpha}\frac{c}{2-3\alpha+\alpha^2} \tag{3.80}$$

where in (3.76), we use Landau notation and $c > 0$ is a constant. The expression in (3.79) has the definition of the Riemann sum, and since $n^{-\alpha} > n^{-1}$, we have our result. Note that (3.79) only converges for $\alpha < 1$. □

Proposition 3.8. *If a process* $\boldsymbol{w}$ *is anti-persistent, we have that* $\mathrm{Var}(\bar{w}_n) = O(n^{-\alpha})$.

Proof. We first note that for a HD process, we have that

$$\lim_{\lambda\to 0} f_\alpha(\lambda)\lambda^{1-\alpha} \sim C_\alpha \tag{3.81}$$

$$\Rightarrow \lim_{n\to\infty} f_\alpha(1/n)n^{\alpha-1} \sim C_\alpha \tag{3.82}$$

Therefore in the following, we will let $n \to \infty$ as λ^{-1}.

We have from Brockwell and Davis [1991], Theorem 7.1.1 that if the process is stationary and $\sum_{h=-\infty}^{\infty}|\gamma_w(h)| < \infty$, we have that

$$n\mathrm{Var}(\bar{w}_n) \to \sum_{h=-\infty}^{\infty}\gamma_w(h) \tag{3.83}$$

Since we have $\sum_{h=-\infty}^{\infty} |\gamma_w(h)| = 2\gamma_w(0)$ for an anti-persistent process, we have

$$
\begin{aligned}
n\text{Var}(\bar{w}_n) &\rightarrow \sum_{h=-\infty}^{\infty} \gamma_w(h) && (3.84)\\
&= \sum_{h=-\infty}^{\infty} \int_{-\pi}^{\pi} e^{ih\lambda} f_\alpha(\lambda) d\lambda && (3.85)\\
\Rightarrow \text{Var}(\bar{w}_n) &\sim \sum_{h=-\infty}^{\infty} \int_{-\pi}^{\pi} e^{ih/n} f_\alpha(\lambda) d\lambda n^{-1} && (3.86)\\
&= \sum_{h=-\infty}^{\infty} \int_{-\pi}^{\pi} e^{ih/n} f_\alpha(\lambda) n^{\alpha-1} d\lambda n^{-\alpha} && (3.87)\\
&\sim n^{-\alpha} C_\alpha \int_{-\pi}^{\pi} e^{ih\lambda} d\lambda && (3.88)\\
&= n^{-\alpha} C_\alpha 2\pi && (3.89)\\
&\sim n^{-\alpha} && (3.90)
\end{aligned}
$$

where (3.85) is by the definition of an autocovariance at lag h and (3.88) comes from letting $n = O\left(\lambda^{-1}\right)$. □

Then, since we know that $\alpha = 1$ corresponds to short memory, we have $\text{Var}(\bar{w}_n) = O(n^{-\alpha})$ for $0 < \alpha < 3$.

Then the following theorem holds.

Theorem 3.6. *Suppose we have a stationary invertible series $\boldsymbol{w}$ that is either ARMA(p, q) or hyperbolic decay. Then the sample mean $\bar{w}_n$ converges in mean square and in probability to μ_x. Thus the Weak Law of Large Numbers holds for these type of processes.*

Proof. The definition of mean square convergence to μ_w has $\text{Var}(\bar{w}_n) \rightarrow 0$ as $n \rightarrow \infty$. Then by Propositions 3.6, 3.7 and 3.8, both hyperbolic decay and exponential decay processes are mean square convergent to μ_w. Then such processes also converge in probability to μ_w and the Weak Law holds. □

We will discuss the estimation of μ_w in Chapter 5 and examine the convergence of means in Chapter 6.

We will now show how subtracting off the mean of the series (or any value) changes the log-likelihood structure of the model associated with the series.

Lemma 3.3. *The log-likelihood structure of any stationary series changes when we subtract off any value from the series.*

Proof. We have that the log-likelihood for any stationary series $\boldsymbol{w}$ with $\mu_w = 0$ is

$$\ell(\Phi|\boldsymbol{w}) = -\frac{1}{2}\log(|\Gamma_n|) - \frac{1}{2}\boldsymbol{w}'\Gamma_n^{-1}\boldsymbol{w} \tag{3.91}$$

where Γ_n is the Toeplitz matrix of the (theoretical) autocovariances of the process as defined by the parameters Φ. Then if we subtract any value a from all points of our series, we have that

$$\ell(\Phi|\boldsymbol{w}) = -\frac{1}{2}\log\left(|\Gamma_n|\right) - \frac{1}{2}\left(\boldsymbol{w} - a\mathbf{1}_n\right)'\Gamma_n^{-1}\left(\boldsymbol{w} - a\mathbf{1}_n\right) \tag{3.92}$$

$$= -\frac{1}{2}\log\left(|\Gamma_n|\right) - \frac{1}{2}\left(\boldsymbol{w}'\Gamma_n^{-1}\boldsymbol{w}\right) - \frac{1}{2}\left(2a\boldsymbol{w}'\Gamma_n^{-1}\mathbf{1}_n - a^2\mathbf{1}_n\Gamma_n^{-1}\mathbf{1}_n\right) \tag{3.93}$$

So the log-likelihood changes in structure or “shape.”

□

Proposition 3.9. *Suppose we have a series* $\boldsymbol{w}$ *with theoretical but unknown mean* μ_w*. Then the difference of log-likelihoods between the true log-likelihood (where we subtract off* μ_w*) and the sample mean subtracted log-likelihood is*

$$\frac{1}{2}\left(2(\mu_w - \bar{w}_n)\boldsymbol{w}'\Gamma_n^{-1}\mathbf{1}_n - (\mu_w - \bar{w}_n)^2\mathbf{1}_n\Gamma_n^{-1}\mathbf{1}_n\right) \tag{3.94}$$

Proof. This is follows from Lemma 3.3. □

Corollary 3.4. *As* $n \to \infty$*, the difference between the concentrated log-likelihoods as in Proposition 3.9 tends to zero.*

Proof. This follows from the guarantee that the sample mean tends to the true mean in mean square. □

3.3.2 The Convergence of Means

As we noted in §3.3.1, as we alter d_f, the convergence of the means changes. We have simulated 500 series of an ARFIMA$(0.8, d_f, -0.4)$ process for each $d_f \in (-0.9, -0.8, \ldots, 0.4)$. We had the size of the series as 2000, and took n as 250, 500, 1000 and 2000. That is, if $n = 250$, we took the first 250 elements of the series. For each d_f, the seeds used were exactly the same.

Figure 3.1 shows that on average the absolute distance from zero with anti-persistent d_f decrease with increasing n. As n increases, the sample mean converges faster and so tends to be closer to its true value of zero. However, as the processes become more persistent, however, the effect of n on the differences diminishes. In other words, Figure 3.1 shows the effect of different d_f and n on the variability of the absolute mean of the series when the true mean is zero.

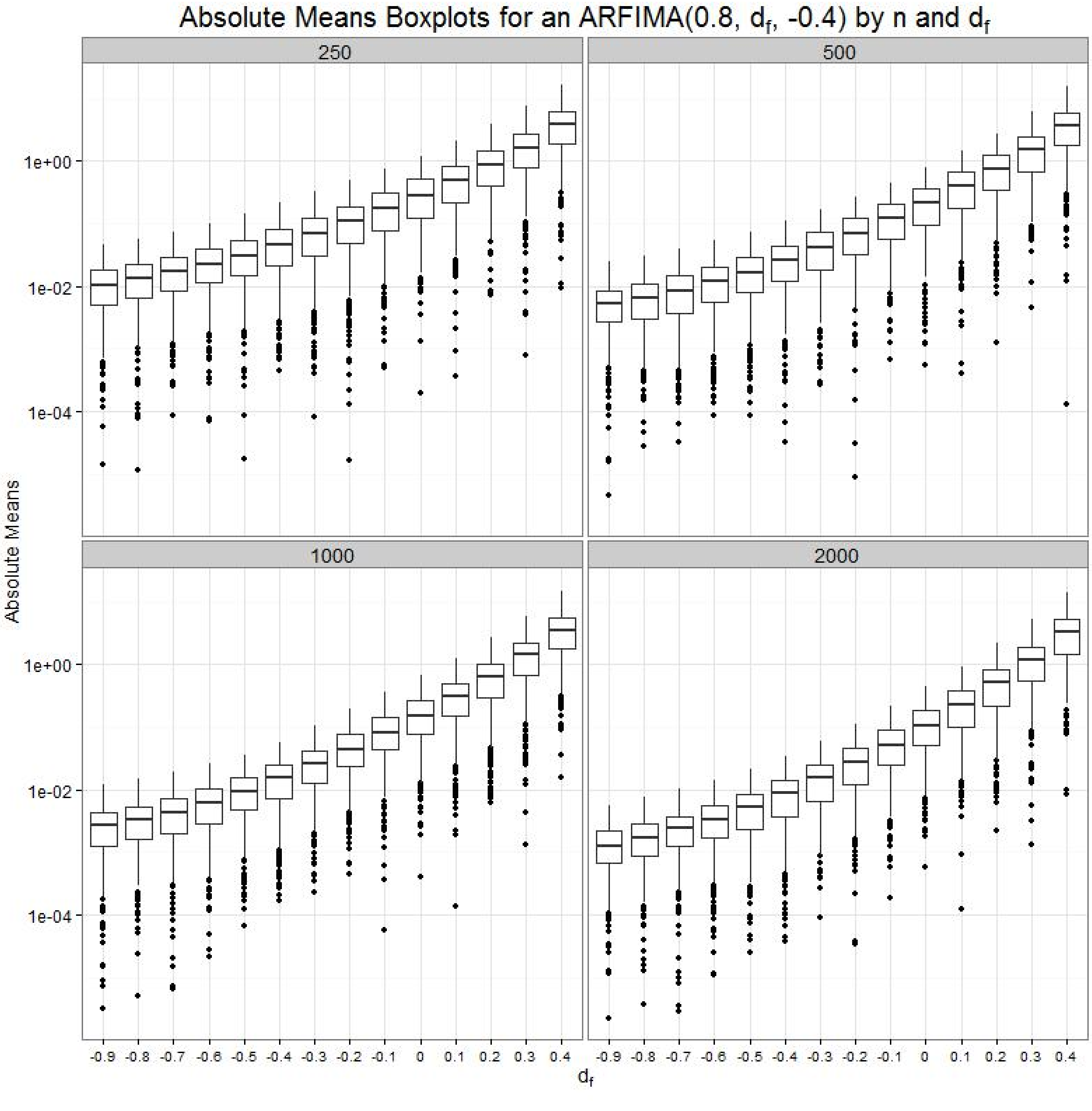


Figure 3.1: Boxplots of the differences in absolute mean from zero on the $\log_{10}$ scale for 500 series generated with the same seeds, but with different n and d_f. The facet label is the value of n, the horizontal axis is the value of d_f, and absolute difference from zero is the vertical axis.

Chapter 4

Predictions and Their Error Variances

4.1 Introduction

This chapter discusses the minimum mean square error (MMSE) prediction and the error variances as discussed in, for example, Box et al. [2008b] and McLeod et al. [2007a], for stationary series. We will call the former the limiting form and the latter the exact form of said predictions and error variances for reasons that will become clear. This chapter also will discuss the extension of the predictions and their error variances to non-stationary series, specifically those with a stochastic trend, deriving a new expression for the exact non-stationary error variances.

We will derive the limiting and exact form of the stationary and non-stationary predictions and prediction error variances in §4.2. We will show in §4.3.2 and §4.3.3 that the exact form of the variances are equivalent to the limiting form of the variances under some assumptions: specifically, that the processes are AR(p) processes integrated by $d^* \in (-1, \infty)$ with d^* in the exact formula, and the length of the series, n, has $n > p$.

We will discuss when it is inappropriate to use the exact form in §4.4.3 and the convergence of the exact form to the limiting form for all processes that can be written in operator notation in §4.4.

4.1.1 The Models Considered in this Chapter

For the limiting form of the prediction, and its error variances, we require the model to be able to be written in operator form: specifically as an MA(∞). Thus the class of ARFIMA(p, d^*, q) processes, with $d^* = d + d_f$, is the largest class of models available. These are of the form

$$\phi(B)\nabla^{d_f} w_t = \theta(B) a_t \tag{4.1}$$

with $w_t = \nabla^d z_t$. We have that $\boldsymbol{w}$ is the stationary series, $\boldsymbol{a}$ is white noise, and $\boldsymbol{z}$ is the possibly integrated series with $d \in \mathbb{Z}_{\geq 0}$. Then $\phi(B)$, $\theta(B)$ and d_f are the AR, MA, and fractional integration parameters respectively, constrained to be stationary and invertible. Specifically we have $d_f \in$

$(-1, 1/2)$. Then the MA(∞) parameters take the form of

$$\begin{aligned} w_t &= \psi(B)a_t && (4.2)\\ &= \left(\psi_0 + \psi_1 B + \psi_2 B^2 + \cdots\right) a_t && (4.3)\\ \Rightarrow \psi(B) &= \frac{\theta(B)}{\phi(B)\nabla^{d_f}} && (4.4) \end{aligned}$$

where $\psi_0 \equiv 1$. We note that, for example, Box et al. [2008b], have the ψ_js written in a recursive form for ARMA and ARIMA models: however, we believe that our form is more natural and easier to understand. Also, fractional differencing cannot be incorporated into the model in said recursive form.

We have that from, e.g., Cryer and Chan [2008], that since processes $z_t = \nabla^{d^*} w_t$ with $d^* > 1/2$ are not in statistical equilibrium, we cannot have an infinite past for this class of processes. This most often occurs, in the ARFIMA case, when $d_f \in (-1/2, 0)$ and $d = 1$ (that is, a fractionally integrated process needs to be differenced once to obtain an anti-persistent but stationary process), or in general, when we have any process differenced by $d \in \mathbb{Z}_{>0}$ to obtain stationarity. We note that with $d^* > 1/2$ we can still write the form of the process in terms of an MA(∞) by way of (4.4) even though we do not have an infinite past.

We note that the exact form of the prediction and its error variances are usable by any model that admits an autocovariance function. These include the HD class of processes we mentioned in Chapter 2 as well as the ARIMA-HD processes discussed in Chapter 3. The processes containing PLS, PLA and FGN terms cannot be written in operator form, and thus have no limiting form.

4.1.2 Results Used in the Chapter

In this section, we will introduce some of the results used in the chapter, with brief proofs. First we introduce the Yule-Walker equations, extended slightly to use Γ_n rather than Γ_p.

Proposition 4.1 (Yule-Walker Equations). *For an* AR(p) *process, we have, with* $\boldsymbol{\phi} = \left(\phi_1, \ldots, \phi_p, 0, \cdots, 0\right)'$, *a vector of length n,*

$$\begin{aligned} \Gamma_n \boldsymbol{\phi} &= \boldsymbol{\gamma}_1 && (4.5)\\ \sigma_a^2 &= \gamma_w(0) - \boldsymbol{\phi}' \boldsymbol{\gamma}_1 && (4.6) \end{aligned}$$

Proof. We have that for $n = p$

$$w_t = \sum_{i=1}^{p} \phi_i w_{t-i} + a_t \tag{4.7}$$

upon multiplying by w_{t+j} for $j = 0, \ldots, p$ and taking expectations that

$$\gamma_w(j) = \sum_{i=1}^{p} \phi_i \gamma_w(j-i) + I(j=0)\sigma_a^2 \tag{4.8}$$

where $I(\cdot)$ is the indicator function. This leads to the regular Yule-Walker equations.

For $n > p$, we note that as long as we remember that $\phi_k = 0$ for $k > p$, we have that the same set of operations give the result. □

Then we have the following corollary.

Corollary 4.1. *For an* AR(p) *process, we have that*

$$\psi_j = \sum_{i=1}^{p} \phi_i \psi_{|j-i|} \tag{4.9}$$

Proof. Since we know

$$\gamma_w(j) = \sum_{k=0}^{\infty} \psi_k \psi_{k+j} \tag{4.10}$$

$$\tag{4.11}$$

and

$$\sum_{i=1}^{p} \phi_i \gamma_w(j-i) = \sum_{i=1}^{p} \phi_i \gamma_w(|j-i|) \tag{4.12}$$

$$= \sum_{k=0}^{\infty} \psi_k \psi_{k+|j-i|} \tag{4.13}$$

this follows very easily from Proposition 4.1. □

We recall the following:

An inner product space V, is a vector space equipped with an inner product $< \cdot, \cdot >: V \to \mathbb{R}$, and a Hilbert space $\mathcal{H}$ is an inner product space that is complete (that is, every Cauchy sequence in $\mathcal{H}$ converges to an element of $\mathcal{H}$). A linear subspace of a Hilbert space is any $\mathcal{M} \subseteq \mathcal{H}$ such that for all elements in $\mathcal{M}$, every linear combination of these elements is also in $\mathcal{M}$. We say $\mathcal{M}$ is closed if all limit points of all sequences in $\mathcal{M}$ are in $\mathcal{M}$. We have $\mathcal{M}^{\perp}$ is called the orthogonal complement of $\mathcal{M}$ if $\forall x \in \mathcal{M}$ and $\forall y \in \mathcal{M}^{\perp}$, we have that $< x, y >= 0$. We note that the orthogonal complement of a subset of a Hilbert space $\mathcal{H}$ is a closed subspace of $\mathcal{H}$. Finally, we note that the norm of any $x \in \mathcal{H}$ is $\|x\| = \sqrt{< x, x >}$ and that a necessary and sufficient condition for $\|x - x_n\| \to 0$ is $\|x - x_n\|^2 \to 0$ for any sequence x_n and any point x in $\mathcal{H}$. In our particular Hilbert space, then, norm convergence is equivalent to convergence in mean square.

With these recollections in mind, we will now state the Projection Theorem (cf. Brockwell and Davis [1991], page 51) without proof.

Theorem 4.1 (The Projection Theorem)**.** *If $\mathcal{M}$ is a closed subspace of the Hilbert space $\mathcal{H}$ and $x \in \mathcal{H}$, then*

1. *there exists a unique element $\hat{x} \in \mathcal{M}$, the orthogonal projection of x onto $\mathcal{M}$, such that*

$$\|x - \hat{x}\| = \inf_{y \in \mathcal{M}} \|x - y\| \tag{4.14}$$

2. *$\hat{x} \in \mathcal{M}$ and $\|x - \hat{x}\| = \inf_{y \in \mathcal{M}} \|x - y\|$ if and only if $\hat{x} \in \mathcal{M}$ and $(x - \hat{x}) \in \mathcal{M}^{\perp}$*

The Hilbert space we will be discussing in this chapter is the space $L^2(\Omega, \mathcal{F}, P)$, where $(\Omega, \mathcal{F}, P)$ is a probability space with Ω a set of random variables, $\mathcal{F}$ a σ-algebra on Ω, and P the probability measure on $\mathcal{F}$. Under the inner product defined by $< X, Y > = E(XY)$, this is a set of equivalence classes with

$$C_\tau = \left\{ X : \mathrm{E}[X^2] = \int_\Omega X(\omega)^2 P(d\omega) < \infty \right\} \tag{4.15}$$

for $\tau \in \Theta$ and Θ an index set such that

$$P(X = Y) = 1, \quad \forall X, Y \in C_\tau \tag{4.16}$$

defines all random variables in C_τ as equivalent. That is, the equivalence classes define the space in that

$$L^2(\Omega, \mathcal{F}, P) \equiv \bigcup_{\tau \in \Theta} C_\tau \tag{4.17}$$

where the union is disjoint. We note that for time series random variables we will be using lower case letters instead of the usual upper case.

In $\mathcal{H} = L^2(\Omega, \mathcal{F}, P)$, we have, letting $\mathcal{M}_n$ be the closed linear subspace of $\mathcal{H}$ spanning $\boldsymbol{w} = \{w_n, \ldots, w_1\}$, $n \geq 1$, that Proj_n is the projection operator onto $\mathcal{M}_n$. We will see in §4.2 that $\mathrm{Proj}_n w_{n+k} = \mathrm{E}_n[w_{n+k}]$.

4.2 Derivations of Predictors and Their Error Variances

First we must prove the following proposition.

Proposition 4.2. *We have that with $\mathcal{H}$, $\boldsymbol{w}$, and $\mathcal{M}_n$ as in §4.1.2 that*

$$\begin{aligned} w_n(k) &= \mathrm{Proj}_n w_{n+k} &(4.18)\\ &= \mathrm{E}_n[w_{n+k}] &(4.19) \end{aligned}$$

Proof. We note that we must have $\mathrm{E}_n[w_n(k)] = w_n(k)$ by definition of the projection onto $\mathcal{M}_n$. As well, we have

$$\begin{aligned} \left\langle w_{n+k} - w_n(k), w_j \right\rangle &= \mathrm{E}\left[(w_{n+k} - w_n(k))w_j\right] &(4.20)\\ &= 0, \quad j = 1, \ldots, n &(4.21)\\ \Rightarrow \mathrm{E}\left[w_j\left(\mathrm{E}_n[w_{n+k} - w_n(k)]\right)\right] &= 0, \quad j = 1, \ldots, n &(4.22)\\ \Rightarrow \mathrm{E}_n[w_{n+k} - w_n(k)] &= 0 &(4.23)\\ \Rightarrow w_n(k) &= \mathrm{E}_n[w_{n+k}] &(4.24) \end{aligned}$$

as required. We have that (4.22) holds by the Law of Iterated Expectations and that (4.23) holds since the case where $P(w_j = 0) = 1, \ \forall j$ is excluded. □

4.2.1 The Case of Non-Integrated Series

4.2.1.1 The Predictors

We will now introduce exact k-step-ahead predictors, $k \geq 1$, for any process admitting an autocovariance function, with $d = 0$. That is, the process is stationary without differencing. This result is found in, e.g. Hipel and McLeod [1994].

Proposition 4.3 (Best linear k-step ahead predictor for a finite past)**.** *With the definitions as in Proposition 4.2, and with $\mu_w = 0$ without loss of generality, we have that the best linear predictor of w_{n+k} based on $\boldsymbol{w}$ (in the MMSE sense) will have the form*

$$\begin{aligned} w_n(k) &= \mathrm{Proj}_n w_{n+k} && (4.25)\\ &= \boldsymbol{\phi}_n^{(k)\prime}\boldsymbol{w} && (4.26)\\ &= \sum_{i=1}^{n} \phi_{ni}^{(k)} w_{n-i+1} && (4.27) \end{aligned}$$

where

$$\boldsymbol{\phi}_n^{(k)} = \Gamma^{-1}\boldsymbol{\gamma}_k \tag{4.28}$$

Proof. We would like $\mathrm{E}\left[(w_{n+k} - \tilde{w}_n(k))^2\right]$ to be minimized with $\tilde{w}_n(k)$ being some linear combination of $\boldsymbol{w}$. In particular, we must have, by Theorem 4.1 and Proposition 4.2, that $\tilde{w}_n(k) = w_n(k)$. Thus we note again that

$$\mathrm{E}\left[(w_{n+k} - w_n(k))\, w_j\right] = 0, \quad j = 1, \ldots, n \tag{4.29}$$

which implies, by the linearity of the inner product,

$$\begin{aligned} \mathrm{E}\left[w_{n+k} w_j\right] &= \mathrm{E}\left[w_n(k) w_j\right] && (4.30)\\ &= \mathrm{E}\left[\sum_{i=0}^{n} \phi_{ni}^{(k)} w_{n+1-i} w_j\right] && (4.31) \end{aligned}$$

That is, since $j = 1, \ldots, n$, we have n equations resulting in

$$\boldsymbol{\gamma}_k' = \boldsymbol{\phi}_n^{(k)\prime}\Gamma_n \tag{4.32}$$

as required. □

Letting $\boldsymbol{\phi}_n^{(1)} = \boldsymbol{\phi}_n$, we note that in the AR(p) case with $p < n$ that we have $\boldsymbol{\phi}_n \equiv \boldsymbol{\phi}$.

Also, when $\mu_w \neq 0$, we have the exact form as

$$w_n(k) = \mu_w + \gamma_k' \Gamma_n^{-1}(\boldsymbol{w} - \mathbf{1}_n \mu_w) \tag{4.33}$$

To see this note that

$$\begin{aligned} \mathrm{E}_n[w_{n+k}] - \mu_w &= \mathrm{E}_n[w_{n+k} - \mu_w] && (4.34) \\ &= E_n[\tilde{w}_{n_k}] && (4.35) \\ &= \gamma_k' \Gamma_n^{-1} \tilde{\boldsymbol{w}} && (4.36) \\ &= \gamma_k' \Gamma_n^{-1}(\boldsymbol{w} - \mathbf{1}_n \mu_w) && (4.37) \end{aligned}$$

We note that this method of exact prediction is at its most efficient using the Durbin-Levinson Algorithm. Another way to write the prediction is by using the Innovations Algorithm. For derivations and the properties of these algorithms, one can see, for example, Brockwell and Davis [1991], Chapter 5.

We now will present the MMSE best linear k-step-ahead predictor for any process that can be written in MA(∞) form, from an infinite past, as given in, e.g. Box et al. [2008a].

Proposition 4.4 (Best linear k-step-ahead predictor from an infinite past)**.** *With $\boldsymbol{w} \sim$ ARFIMA being zero-mean, having an infinite past and the* MA(∞) *parameters as defined by* (4.4)*, we have that the MMSE best linear predictor is, for any t,*

$$w_t(k) = \sum_{j=0}^{\infty} \psi_{k+j} a_{t-j} \tag{4.38}$$

Proof. Suppose the best forecast given the infinite past is

$$w_t(k) = \sum_{j=0}^{\infty} \nu_{k+j} a_{t-j} \tag{4.39}$$

For this to be an MMSE forecast, we have to minimize

$$\begin{aligned} \mathrm{E}\left[(w_{t+k} - w_t(k))^2\right] &= \mathrm{E}\left[\left(\sum_{i=0}^{k-1} \psi_i a_{t-i}\right)^2\right] + \mathrm{E}\left[\left(\sum_{i=k}^{\infty} (\psi_i - \nu_i) a_{t-i}\right)^2\right] && (4.40) \\ &= \sigma_a^2 \sum_{i=0}^{k-1} \psi_i^2 + \sigma_a^2 \sum_{i=k}^{\infty} (\psi_i - \nu_i)^2 && (4.41) \end{aligned}$$

which is obviously minimized by taking $\nu_i = \psi_i$, $i \in \mathbb{Z}_{\geq k}$. We note that this definition also has $w_t(k) = \mathrm{E}_t[w_{t+k}]$, where E_t is with respect to the infinite past up to time t. This can be easily seen, since

$$\begin{aligned} w_{t+k} &= \sum_{i=0}^{k-1} \psi_i a_{t+k-i} + \sum_{i=k}^{\infty} \psi_i a_{t+k-i} && (4.42) \\ \Rightarrow \mathrm{E}_t[w_{t+k}] &= \sum_{i=k}^{\infty} \psi_i a_{t+k-i} && (4.43) \end{aligned}$$

since $E_t[a_{t+j}] = 0, \ \forall j > 0$ and $E_t[a_{t-j}] = a_{t-j}, \ \forall j \geq 0.$ □

It may seem obvious that the finite sample predictor is often more useful, and that there is another form of Proposition 4.4 that allows for a finite past: however, the infinite form expansion is quite often used to calculate prediction error variances, which we will talk about next.

4.2.1.2 The Prediction Error Variances

We will let $e_n(k) = w_{n+k} - w_n(k)$. Note that $e_n(k)$ is an unbiased estimate of 0 (whether the process has an finite or infinite past) and thus the variance is the mean squared error.

For the finite past prediction case, we have, as in McLeod et al. [2007a] and Brockwell and Davis [1991],

Proposition 4.5. *Under the conditions in Proposition 4.3, that*

$$\mathrm{Var}\left(e_n(k)\right) = \gamma_w(0) - \gamma_k' \Gamma_n^{-1} \gamma_k \tag{4.44}$$

Proof. The exact form of the variance of $e_n(k)$ is:

$$\begin{aligned}
\mathrm{Var}(e_n(k)) &= \mathrm{E}\left[(w_{n+k} - w_n(k))^2\right] && (4.45)\\
&= \mathrm{E}\left[w_{n+k}^2\right] - 2\mathrm{E}\left[w_{n+k} w_n(k)\right] + \mathrm{E}\left[w_n^2(k)\right] && (4.46)\\
&= \gamma_w(0) - 2\gamma_k' \Gamma_n^{-1} \mathrm{E}\begin{bmatrix} w_n w_{n+k} \\ \dots \\ w_1 w_{n+k} \end{bmatrix} + \gamma_k' \Gamma_n^{-1} \mathrm{E}\left[\boldsymbol{w}\boldsymbol{w}'\right] \Gamma_n^{-1} \gamma_k && (4.47)\\
&= \gamma_w(0) - 2\gamma_k' \Gamma_n^{-1} \gamma_k + \gamma_k' \Gamma_n^{-1} \gamma_k && (4.48)\\
&= \gamma_w(0) - \gamma_k' \Gamma_n^{-1} \gamma_k && (4.49)
\end{aligned}$$

□

For the infinite past, we have, once again from Box et al. [2008a]:

Proposition 4.6. *Under the conditions in Proposition 4.4, that*

$$\mathrm{Var}_\ell\left(e_n(k)\right) = \sigma_a^2 \sum_{i=0}^{k-1} \psi_i^2 \tag{4.50}$$

Proof. This is follows from the arguments presented in the proof of Proposition 4.4, specifically by (4.41). □

4.2.2 The Case of Integrated Series

4.2.2.1 The Predictors

Since we cannot have an infinite past for non-stationary integrated series, we will examine an algorithm to predict from this type of process. It is as follows.

Algorithm 4.1. *To predict from a non-stationary integrated series, $\boldsymbol{z}$, we perform the following steps:*

- *Difference and seasonally difference the series the appropriate number of times, say d and d_s. Call this series $\boldsymbol{w}$.*
- *Predict from $\boldsymbol{w}$, for k-step-ahead, from $w_n(1)$ to $w_n(k)$.*
- *Integrate the $w_n(h)$ for $h = 1, \ldots, k$, using the previous $w_n(j)$s and $\boldsymbol{z}$s.*

We note if, with $\nabla_s = (1 - B^s)$ and s the seasonality, $w_t = \nabla^d \nabla_s^{d_s} z_t$ is the form of $\boldsymbol{w}$, with $d, d_s \in \mathbb{Z}_{\geq 0}$ being the amount of nonseasonal and seasonal differencing, respectively, we have, for example,

$$\begin{aligned} z_{n+k} &= \sum_{j=1}^{d} \binom{d}{j} (-1)^{j+1} B^j z_{n+k} + \sum_{h=1}^{d_s} \binom{d_s}{h} (-1)^{h+1} B^{hs} z_{n+k} + \sum_{j=1}^{d} \sum_{h=1}^{d_s} \binom{d}{j} \binom{d_s}{h} (-1)^{j+h+1} B^{j+hs} z_{n+k} + w_{n+k} & (4.51) \\ &= f_k(\boldsymbol{z}, \widetilde{w_{n+k}}) & (4.52) \end{aligned}$$

is an expression in for z_{n+k} in terms of $\boldsymbol{z}$ and $\widetilde{w_{n+k}} = \{w_{n+k-1}, \ldots, w_{n+1}\}$. If we knew exactly $\widetilde{w_{n+k}}$ as well as $\boldsymbol{z}$, we could obtain z_{n+k} for $k = 1, 2, \ldots$. In the same way, we can have, with $z_n(-k) = z_{n-k}$ for $k \geq 0$ and $B^i z_n(k) = z_n(k - i)$

$$z_n(k) = f_k(\boldsymbol{z}, \widetilde{w_n(k)}) \tag{4.53}$$

and as such we can use f_k recursively to determine the values c_j and c_j^* such that

$$z_n(k) = \sum_{j=0}^{k-1} c_j w_n(k - j) + \sum_{j=0}^{k^*} c_j^* z_{n-j} \tag{4.54}$$

and note that there is a similar expression for z_{n+k}.

We have the following results about $z_n(k)$.

Proposition 4.7. *We have that $z_n(k) = \mathrm{E}_n [z_{n+k}]$. Moreover, $z_n(k)$ is an MMSE predictor.*

Proof. We note that, if $\epsilon_n(k) = z_{n+k} - z_n(k)$,

$$\begin{aligned}
\epsilon_n(k) &= z_{n+k} - z_n(k) && (4.55)\\
&= \sum_{j=0}^{k-1} c_j w_{n+k-j} + \sum_{j=0}^{k^*} c_j^* z_{n-j} - \left(\sum_{j=0}^{k-1} c_j w_n(k-j) + \sum_{j=0}^{k^*} c_j^* z_{n-j} \right) && (4.56)\\
&= \sum_{j=0}^{k-1} c_j \left(w_{n+k-j} - w_n(k-j) \right) && (4.57)\\
\Rightarrow \mathrm{E}_n[\epsilon_n(k)] &= \sum_{j=0}^{k-1} c_j \left(\mathrm{E}_n[w_{n+k-j}] - \mathrm{E}_n[w_n(k-j)] \right) && (4.58)\\
&= 0 && (4.59)
\end{aligned}$$

where (4.59) holds since the past of $\boldsymbol{z}$ necessarily includes the past of $\boldsymbol{w}$. Therefore we have that $\epsilon_n(k)$ is an unbiased estimate of zero and as such $z_n(k) = \mathrm{E}_n\left[z_{n+k}\right]$.

To show $z_n(k)$ is an MMSE predictor, we first have

$$\begin{aligned}
\mathrm{E}_n\left[z_j\left(z_{n+k} - z_n(k)\right)\right] &= E_n\left[z_j \epsilon_n(k)\right], \quad \forall j = 1, \ldots, n && (4.60)\\
&= \mathrm{E}_n\left[z_j \mathrm{E}_n\left[\epsilon_n(k)\right]\right] && (4.61)\\
&= 0 && (4.62)
\end{aligned}$$

where $\mathrm{E}_n\left[\epsilon_n(k)\right] = 0$ by (4.59) and (4.61) holds by the Law of Iterated Expectations.

Thus each z_j is orthogonal to $z_{n+k} - z_n(k)$. This means that $z_n(k) = \mathrm{Proj}_n z_{n+k}$ and as such the linear combination of z_js, $j = 1, \ldots n$, that make up $z_n(k)$ give a minimum mean square error.

□

We present an algorithm, called Z, in the programming language and environment ***R*** for computing the c_js, in Listing 4.1. It returns the c_js in reversed order in the vector val. We could adapt this relatively easily to produce the c_j^*s as well, but at the moment we will not do so.

```
Z <- function(k, d, ds, s) {

  if((d==0)&&(ds==0)) return(numeric(0))
  if(ds > 0 && s==0) stop("No period supplied")
  worker1 <- function(m, value, val) {
    if(m > 0) {
      val[m] <- val[m] + value
      if(d > 0) {
        for(i in 1:d) {
          val <- worker1(m-i, value*choose(d, i)*(-1)^(i+1), val)
        }
      }
      if(ds > 0) {
        for(j in 1:ds) {
          val <- worker1(m-j*s, value*choose(ds, j)*(-1)^(j+1), val)
        }
      }
      if(d > 0 && ds > 0) {
        for(i in 1:d) {
          for(j in 1:ds) {
            val <- worker1(m-i-j*s, value*choose(d, i)
              *choose(ds, j)*(-1)^(i+j+1), val)
          }
        }
      }
    }
    val
  }

  val <- numeric(k)
  val <- worker1(k, 1, val)

  val
}
```

Listing 4.1: Our Algorithm for Computing the c_js in ***R***

4.2.2.2 The Prediction Error Variances

We presented only the algorithm that produces the c_js in Listing 4.1 since they are all that is needed for the prediction error variances. We note that we can construct another, infinite, expansion of the c_js, even though we only have a finite past to work with. We see that since

$w_t = \nabla^d \nabla_s^{d_s} z_t$, we have

$$w_t = \nabla^{-d}\nabla_s^{-d_s} y_t \tag{4.63}$$
$$= \sum_{j=0}^{\infty} \binom{-d}{j} (-1)^j B^j \sum_{i=0}^{\infty} \binom{-d_s}{i} (-1)^i B^{si} z_t \tag{4.64}$$
$$= \sum_{h=0}^{\infty} c_h B^h z_t \tag{4.65}$$

which would quickly get us into trouble, since we have a finite past. However, we note that not only are $c_0, \ldots, c_{k-1}$ equal to the c_js given by the algorithm Z by construction, we have that since we use these values for the prediction errors, the z_t with $t \leq n$ subtract out. That is,

$$\epsilon_n(k) = \sum_{j=0}^{k-1} c_j(w_{n+k-j} - w_n(k-j)) + \sum_{j=k}^{n+k} c_j 0 + \sum_{j=n+k+1}^{\infty} B^j \emptyset \tag{4.66}$$
$$= \sum_{j=0}^{k-1} c_j(w_{n+k-j} - w_n(k-j)) \tag{4.67}$$

We note that this form is faster to compute, although there may be slight numerical errors. We also have that $c_{k+j} = c_j^*$ for $j = 0, \ldots, k^*$, and all we have to do is determine the correct k^* to integrate the series.

Then the values of $\mathrm{Var}(\epsilon_n(k))$ and $\mathrm{Var}_\ell(\epsilon_n(k))$ can be written as in the following propositions.

Proposition 4.8. *The exact form of the error variances is*

$$\mathrm{Var}(\epsilon_n(k)) = \sum_{j=0}^{k-1}\sum_{l=0}^{k-1} c_j c_l \left(\gamma_w(|j-l|) - \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l}\right) \tag{4.68}$$

Proof. It is easily seen that

$$\mathrm{Var}(\epsilon_n(k)) = \mathrm{E}\left[\left(\sum_{i=0}^{k-1} c_i \left(w_{n+k-i} - w_n(k-i)\right)\right)^2\right] \tag{4.69}$$
$$= \sum_{j=0}^{k-1}\sum_{l=0}^{k-1} c_j c_l \Big(\mathrm{E}[w_{n+k-j}w_{n+k-l}] - \mathrm{E}[w_n(k-j)w_{n+k-l}] - \mathrm{E}[w_n(k-l)w_{n+k-j}] + \mathrm{E}[w_n(k-j)w_n(k-l)]\Big) \tag{4.70}$$
$$= \sum_{j=0}^{k-1}\sum_{l=0}^{k-1} c_j c_l \left(\gamma_w(|j-l|) - \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l} - \gamma'_{k-l}\Gamma_n^{-1}\gamma_{k-j} + \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l}\right) \tag{4.71}$$
$$= \sum_{j=0}^{k-1}\sum_{l=0}^{k-1} c_j c_l \left(\gamma_w(|j-l|) - \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l}\right) \tag{4.72}$$

where (4.71) is a consequence of (4.33) through

$$\begin{aligned} \mathrm{E}[w_n(k-j)w_{n+k-l}] &= \gamma'_{k-j}\Gamma_n^{-1}\mathrm{E}\left[\boldsymbol{w}w_{n+k-l}\right] && (4.73)\\ &= \gamma'_{k-j}\Gamma_n^{-1}\mathrm{E}\begin{bmatrix} w_n w_{n+k-l} \\ \dots \\ w_1 w_{n+k-l} \end{bmatrix} && (4.74)\\ &= \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l} && (4.75) \end{aligned}$$

and

$$\begin{aligned} \mathrm{E}\left[w_n(k-j)w_n(k-l)\right] &= \gamma'_{k-j}\Gamma_n^{-1}\mathrm{E}[\boldsymbol{w}\boldsymbol{w}']\Gamma_n^{-1}\gamma_{k-l} && (4.76)\\ &= \gamma'_{k-j}\Gamma_n^{-1}\Gamma_n\Gamma_n^{-1}\gamma_{k-l} && (4.77)\\ &= \gamma'_{k-j}\Gamma_n^{-1}\gamma_{k-l} && (4.78) \end{aligned}$$

□

Proposition 4.9. *The limiting form of the integrated error variances is*

$$\mathrm{Var}_\ell(\epsilon_n(k)) = \sum_{j=0}^{k-1}\left(\sum_{i=0}^{j}\sum_{h=0}^{j} c_i c_h \psi_{j-i}\psi_{j-h}\right) \tag{4.79}$$

Proof. Although we do not have an infinite past, we can construct an infinite expansion

$$\begin{aligned} \psi^*(B) &= \frac{\psi(B)}{\nabla^d\nabla_s^{d_s}} && (4.80)\\ &= \sum_{i=0}^{\infty}\psi_i B^i \sum_{h=0}^{\infty}\begin{pmatrix} -d \\ h \end{pmatrix} B^h \sum_{g=0}^{\infty}\begin{pmatrix} -d_s \\ g \end{pmatrix} B^{gs} && (4.81)\\ &= \sum_{i=0}^{\infty}\sum_{l=0}^{\infty} B^{i+l}c_l\psi_i && (4.82)\\ &= \sum_{j=0}^{\infty} B^j \sum_{l=0}^{\infty} c_l\psi_{j-l} && (4.83)\\ &= \sum_{j=0}^{\infty} B^j \sum_{l=0}^{j} c_h\psi_{j-l} && (4.84) \end{aligned}$$

where Equation (4.84) holds since $\psi_i = 0$ for $i < 0$.

Also,

$$\begin{aligned} \mathrm{Var}_\ell(\epsilon_n(k)) &= \sum_{j=0}^{k-1}\left(\psi_j^*\right)^2 && (4.85)\\ &= \sum_{j=0}^{k-1}\left(\sum_{i=0}^{j} c_i\psi_{j-i}\right)^2 && (4.86)\\ &= \sum_{j=0}^{k-1}\left(\sum_{i=0}^{j}\sum_{h=0}^{j} c_i c_h \psi_{j-i}\psi_{j-h}\right) && (4.87) \end{aligned}$$

□

4.2.2.3 On the Value of d^*

We note that none of the above requires d to be a non-negative integer. The formulae above all apply if we have fractional d^* instead of d: that is, if $d^* = d + d_f$, where $d_f \in (-1, 1/2)$ and $d \in \mathbb{Z}_{\geq 0}$. Therefore the above derivation of the exact form and the equivalence proof for the integrated series both hold for $d^* \in (-1, \infty)$. Note that we require that $d^* > -1$ for the process to be invertible, so we can forecast from it. There is a caveat, however: while it may be tempting to forecast with, say, an AR(p) and use the "exact" formula with d_f, rather than forecast with an ARFI(p, d_f), this is in fact incorrect and not the exact forecast at all. We will discuss this in §4.4.3.

4.3 The Proof of Equivalence of the Prediction Error Variances for the AR Case

4.3.1 Three Useful Lemmas

These lemmas are useful in the equivalence proof.

Lemma 4.1. *We have that,* $\forall r, s \in \mathbb{Z}_{\geq 0}$,

$$\gamma_r \Gamma_n^{-1} \gamma_s = \gamma_s \Gamma_n^{-1} \gamma_r \tag{4.88}$$

Proof. We have that from Siddiqui [1958], the inverse of any covariance matrix is bisymmetric. Then (4.88) follows from symmetry. □

Lemma 4.2. *We have that for an* AR(p) *model that with* $n > p$

$$\gamma_w(m) - \sum_{h=1}^{p} \phi_h \gamma_w(m+h) = \psi_m \tag{4.89}$$

for all $m \geq 0$.

Proof. We proceed by complete induction. The base case is $k = 0$: we have it holds by Proposition 4.1.

The induction hypotheses are that (4.89) hold for all m, $0 \leq m \leq k$. The induction step is then to show that

$$\gamma_w(k+1) - \sum_{h=1}^{p} \phi_h \gamma_w(h+1+k) = \psi_{k+1} \tag{4.90}$$

We have

$$\gamma_w(k+1) - \sum_{h=1}^{p} \phi_h \gamma_w(h+1+k) = \sum_{j=0}^{\infty} \psi_j \left(\psi_{j+k+1} - \sum_{h=1}^{p} \phi_h \psi_{j+h+k+1} \right) \tag{4.91}$$

and thus an equivalent form of our induction hypothesis in (4.89) is

$$\sum_{j=0}^{\infty} \psi_j \left(\psi_{j+m} - \sum_{h=1}^{p} \phi_h \psi_{j+m+h} \right) = \psi_m \tag{4.92}$$

Let, for any integer $r \geq 0$,

$$\begin{aligned} a_j^{(r)} &= \psi_{j+r} - \sum_{h=1}^{p} \phi_k \psi_{j+r+h} && (4.93) \\ &= \sum_{i=1}^{p} \phi_i \left(\psi_{|j+r-i|} - \sum_{h=1}^{p} \phi_h \psi_{|j+r+h-i|} \right) && (4.94) \end{aligned}$$

Then (4.92) has

$$\sum_{j=0}^{\infty} \psi_j a_j^{(m)} = \psi_m \tag{4.95}$$

for $0 \leq m \leq k$. Thus

$$\begin{aligned} \sum_{j=0}^{\infty} \psi_j a_j^{(k+1)} &= \sum_{j=0}^{\infty} \psi_j \left(\sum_{i=1}^{p} \phi_i \left(\psi_{|j+k+1-i|} - \sum_{h=1}^{p} \phi_h \psi_{|j+h+1+k-i|} \right) \right) && (4.96) \\ &= \sum_{i=1}^{p} \phi_i \left(\sum_{j=0}^{\infty} \psi_j \left(\psi_{|j+k+1-i|} - \sum_{h=1}^{p} \phi_h \psi_{|j+h+1+k-i|} \right) \right) && (4.97) \\ &= \sum_{i=1}^{p} \phi_i \left(\sum_{j=0}^{\infty} \psi_j a_j^{(|k+1-i|)} \right) && (4.98) \\ &= \sum_{i=1}^{p} \phi_i \psi_{|k+1-i|} && (4.99) \\ &= \psi_{k+1} && (4.100) \end{aligned}$$

where (4.99) comes from (4.95), and (4.100) is from Corollary 4.1. □

Lemma 4.3. *We have that, for an* AR(p) *process,*

$$\gamma_r' \Gamma_n^{-1} \gamma_s - \gamma_{r+1}' \Gamma_n^{-1} \gamma_{s+1} = \psi_r \psi_s \tag{4.101}$$

Proof. We proceed by two-dimensional complete induction on r and s. Without loss of generality, we have $\sigma_a^2 = 1$ and we recall we have $\psi_0 = 1$ by definition.

Our base case is $r = s = 0$. We must show that

$$\gamma_0' \Gamma_n^{-1} \gamma_0 - \gamma_1' \Gamma_n^{-1} \gamma_1 = \psi_0 \psi_0 \tag{4.102}$$

This is, by Lemma 4.1,

$$\begin{aligned} \gamma_0 \Gamma_n^{-1} \gamma_0 - \gamma_1' \Gamma_n^{-1} \gamma_1 &= \gamma_0' \Gamma_n^{-1} \gamma_0 - \gamma_1' \Gamma_n^{-1} \gamma_1 \qquad (4.103) \\ &= \gamma_w(0) - \boldsymbol{\phi}' \gamma_1 \qquad (4.104) \\ &= 1 \qquad (4.105) \end{aligned}$$

where (4.105) is by Proposition 4.1 and so we have our base case.

We must show (4.101) holds for all $r, s \in Z_{\geq 0}$.

We begin by letting $s = 0$ and proceeding with the induction on r. Then the base case for r is done by the above.

We have that, for $0 \leq m \leq k$

$$\gamma_m' \Gamma_n^{-1} \gamma_0 - \gamma_{m+1}' \Gamma_n^{-1} \gamma_1 = \psi_m \psi_0 \tag{4.106}$$

and we must show the relation holds for $m = k + 1$.

Then we have that, since we are dealing with an autoregressive process, $\boldsymbol{\phi} = \boldsymbol{\phi}_n^{(0)} = \boldsymbol{\phi}_n^{(1)}$. As such

$$\begin{aligned} \gamma_{k+1}' \Gamma_n^{-1} \gamma_0 - \gamma_{k+2}' \Gamma_n^{-1} \gamma_1 &= \gamma_{k+1}' \boldsymbol{\phi} - \gamma_{k+2}' \boldsymbol{\phi} \qquad (4.107) \\ &= \sum_{i=1}^{p} \phi_i \left(\gamma_w(k+i) - \gamma_w(k+i+1) \right) \qquad (4.108) \\ &= \sum_{j=0}^{\infty} \psi_j \sum_{i=1}^{p} \phi_i \left(\psi_{j+k+i} - \psi_{j+k+i+1} \right) \qquad (4.109) \\ &= \sum_{j=0}^{\infty} \psi_j b_j^{(k+1)} \qquad (4.110) \end{aligned}$$

where

$$\begin{aligned} b_j^{(k+1)} &= \sum_{i=1}^{p} \phi_i \left(\psi_{j+k+i} - \psi_{j+k+i+1} \right) \qquad (4.111) \\ &= \sum_{l=1}^{p} \phi_l \sum_{i=1}^{p} \phi_i \left(\psi_{|j+k+i-l|} - \psi_{|j+k+i+1-l|} \right) \qquad (4.112) \end{aligned}$$

and we know that $\sum_{j=0}^{\infty} \psi_j b_j^{(m)} = \psi_m = \psi_m \psi_0$ for $0 \leq m \leq k$ from our induction hypotheses.

Then

$$\sum_{j=0}^{\infty}\psi_j b_j^{(k+1)} = \sum_{j=0}^{\infty}\psi_j \sum_{l=1}^{p}\phi_l \sum_{i=1}^{p}\phi_i \left(\psi_{|j+k+i-l|} - \psi_{|j+k+i+1-l|}\right) \tag{4.113}$$

$$= \sum_{l=1}^{p}\phi_l \left(\sum_{j=0}^{\infty}\psi_j \sum_{i=1}^{p}\phi_i \left(\psi_{j+k+i-l} - \psi_{j+k+i+1-l}\right)\right) \tag{4.114}$$

$$= \sum_{l=1}^{p}\phi_l \left(\sum_{j=0}^{\infty}\psi_j b_j^{(|k+1-l|)}\right) \tag{4.115}$$

$$= \sum_{l=1}^{p}\phi_l \psi_{|k+1-l|} \tag{4.116}$$

$$= \psi_{k+1} \tag{4.117}$$

$$= \psi_{k+1}\psi_0 \tag{4.118}$$

Now we let $s \in \mathbb{Z}_{\geq 0}$ be arbitrary and proceed with induction on r. The base case is complete by the above complete induction with $s = 0$, since r and s can be switched by Lemma 4.1.

Then we have as our induction hypotheses that for $0 \leq m \leq k$,

$$\gamma_m' \Gamma_n^{-1} \gamma_s - \gamma_{m+1}' \Gamma_n^{-1} \gamma_{s+1} = \psi_m \psi_s \tag{4.119}$$

and we must show the relation holds for $m = k + 1$.

Then,

$$\gamma_{k+1}' \Gamma_n^{-1} \gamma_s - \gamma_{k+2}' \Gamma_n^{-1} \gamma_{s+1} = \gamma_{k+1}' \boldsymbol{\phi}^{(s)} - \gamma_{k+2}' \boldsymbol{\phi}^{(s+1)} \tag{4.120}$$

$$= \sum_{i=1}^{p}\phi_{ni}^{(s)}\gamma_w(k+i) - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\gamma_w(k+i+1) \tag{4.121}$$

$$= \sum_{j=1}^{\infty}\psi_j \left(\sum_{i=1}^{p}\phi_{ni}^{(s)}\psi_{j+k+i} - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\psi_{j+k+i+1}\right) \tag{4.122}$$

$$= \sum_{j=1}^{\infty}\psi_j d_j^{(s,k+1)} \tag{4.123}$$

where

$$d_j^{(s,k+1)} = \left(\sum_{i=1}^{p}\phi_{ni}^{(s)}\psi_{j+k+i} - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\psi_{j+k+i+1}\right) \tag{4.124}$$

$$= \sum_{l=1}^{p}\phi_l \left(\sum_{i=1}^{p}\phi_{ni}^{(s)}\psi_{|j+k+i-l|} - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\psi_{|j+k+i+1-l|}\right) \tag{4.125}$$

and we know by our induction hypotheses that $\sum_{j=1}^{\infty}\psi_j d_j^{(s,m)} = \psi_m \psi_s$, with $s \in \mathbb{Z}_{\geq 0}$ arbitrary and $0 \leq m \leq k$.

Therefore,

$$\sum_{j=1}^{\infty}\psi_j d_j^{(s,k+1)} = \sum_{j=1}^{\infty}\psi_j\left(\sum_{l=1}^{p}\phi_l\left(\sum_{i=1}^{p}\phi_{ni}^{(s)}\psi_{|j+k+i-l|} - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\psi_{|j+k+i+1-l|}\right)\right) \tag{4.126}$$

$$= \sum_{l=1}^{p}\phi_l\left(\sum_{j=1}^{\infty}\psi_j\left(\sum_{i=1}^{p}\phi_{ni}^{(s)}\psi_{|j+k+i-l|} - \sum_{i=1}^{p}\phi_{ni}^{(s+1)}\psi_{|j+k+i+1-l|}\right)\right) \tag{4.127}$$

$$= \sum_{l=1}^{p}\phi_l\sum_{j=1}^{\infty}\psi_j d^{(s,|k+1-l|)} \tag{4.128}$$

$$= \sum_{l=1}^{p}\phi_l\psi_s\psi_{|k+1-l|} \tag{4.129}$$

$$= \psi_s\sum_{l=1}^{p}\phi_l\psi_{|k+1-l|} \tag{4.130}$$

$$= \psi_s\psi_{k+1} \tag{4.131}$$

and thus we have our result.

□

4.3.2 Proof of Equivalence in the AR(p) Case

We have the following theorem:

Theorem 4.2. *We have that* (4.50) *and* (4.49) *are equivalent in the* AR(p) *case. This includes the seasonal case when* $p^* = p + p_s s$.

Proof. We first note that $\gamma_w(0) = \sum_{j=0}^{\infty}\psi_j^2$. Then we must show that

$$\gamma_k'\Gamma_n^{-1}\gamma_k = \sum_{j=k}^{\infty}\psi_j^2 \tag{4.132}$$

which is true if and only if

$$\gamma_k'\Gamma_n^{-1}\gamma_k - \gamma_{k+1}'\Gamma_n^{-1}\gamma_{k+1} = \psi_k^2 \tag{4.133}$$

which follows from Lemma 4.3. □

For example, Brockwell and Dahlhaus [2004], Equation (54) has

$$v_n^{(h+1)} = v_n^{(h)} - \phi_{n+1,n+1}^{(h)}v_n^{(1)} \tag{4.134}$$

where $v_n^{(k)}$ is the k-step-ahead prediction error variance of the given series of length n. We must have $\phi_{n+1,n+1}^{(h)} = 0$ for any $h \in \mathbb{Z}_{\geq 0}$ with $n > p$. This is a consequence of Proposition 4.10.

4.3.3 Proof of Equivalence in the ARIMA$(p, d, 0)$ Case

We note that the nonintegrated case ($d = 0$) is a special case of this. Then we have the following theorem:

Theorem 4.3. *We have that the exact and limiting prediction error variances are the same for* ARIMA$(p, d, 0)$ *processes for all* $d \in \mathbb{Z}_{\geq 0}$.

Proof. Let Y_k be the k^{th} step ahead exact prediction error variance, and let X_k be the k^{th} step ahead limiting prediction error variance. That is,

$$Y_k = \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \left(\gamma_w(|i-l|) - \gamma'_{k-i} \Gamma_n^{-1} \gamma_{k-l} \right) \tag{4.135}$$

$$X_k = \sum_{j=0}^{k-1} \left(\sum_{i=0}^{j} \sum_{l=0}^{j} c_i c_l \psi_{j-i} \psi_{j-l} \right) \tag{4.136}$$

We also let

$$\begin{aligned} S_k &= Y_{k+1} - Y_k && (4.137) \\ &= \sum_{i=0}^{k} \sum_{l=0}^{k} c_i c_l \left(\gamma_w(|i-l|) - \gamma'_{k+1-i} \Gamma_n^{-1} \gamma_{k+1-l} \right) - \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \left(\gamma_w(|i-l|) - \gamma'_{k-i} \Gamma_n^{-1} \gamma_{k-l} \right) && (4.138) \\ &= \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \left(\gamma'_{k-i} \Gamma_n^{-1} \gamma_{k-l} - \gamma'_{k-i+1} \Gamma_n^{-1} \gamma_{k-l+1} \right) + 2c_k \sum_{i=0}^{k} c_i \left(\gamma_w(k-i) - \gamma'_1 \Gamma_n^{-1} \gamma_{k+1-i} \right) && (4.139) \\ &= M_k + 2c_k A_k && (4.140) \end{aligned}$$

through some careful algebra, and

$$\begin{aligned} T_k &= X_{k+1} - X_k && (4.141) \\ &= \sum_{j=0}^{k} \left(\sum_{i=0}^{j} \sum_{l=0}^{j} c_i c_l \psi_{j-i} \psi_{j-l} \right) - \sum_{j=0}^{k-1} \left(\sum_{i=0}^{j} \sum_{l=0}^{j} c_i c_l \psi_{j-i} \psi_{j-l} \right) && (4.142) \\ &= \sum_{i=0}^{k} \sum_{l=0}^{k} c_i c_l \psi_{k-i} \psi_{k-l} && (4.143) \\ &= \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \psi_{k-i} \psi_{k-l} + 2c_k \sum_{i=0}^{k} c_i \psi_{k-i} && (4.144) \\ &= N_k + 2c_k B_k && (4.145) \end{aligned}$$

Note that we have (by construction) that $Y_0 = X_0 = 0$ and that for any d, the ARIMA$(p, d, 0)$ has $Y_1 = X_1 = 1$. Therefore for any $k \in \mathbb{Z}_{>0}$ an equivalent statement to $Y_k = X_k$ is that $S_k = T_k$.

For arbitrary $k > 0$,

$$A_k = \sum_{i=0}^{k} c_i \left(\gamma_w(k-i) - \gamma_1' \Gamma_n^{-1} \gamma_{k+1-i}\right) \tag{4.146}$$

$$= \sum_{i=0}^{k} c_i \psi_{k-i} \tag{4.147}$$

$$= B_k \tag{4.148}$$

where (4.147) is from Lemma 4.2.

For arbitrary $k > 0$,

$$M_k = \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \left(\gamma_{k-i}' \Gamma_n^{-1} \gamma_{k-l} - \gamma_{k-i+1}' \Gamma_n^{-1} \gamma_{k-l+1}\right) \tag{4.149}$$

$$= \sum_{i=0}^{k-1} \sum_{l=0}^{k-1} c_i c_l \psi_{k-i} \psi_{k-l} \tag{4.150}$$

$$= N_k \tag{4.151}$$

where (4.150) is by Lemma 4.3. Then we have our result. □

We note that Equation (54) in Brockwell and Dahlhaus [2004] does not apply, since the results of said paper are restricted to stationary series. We will see that the application of the exact formula to fractional d^* as was mentioned in §4.2.2.3 is incorrect: we will see this in §4.4.3.

4.4 On the Comparison of the Exact Form and the Limiting Form as *n* Increases

We have the following proposition.

Proposition 4.10. *For an* AR(p) *process, we have that for all* $k \geq 1$ *and with* $n > p$ *that* $\phi_{ns}^{(k)} = 0$ *for* $s > p$.

Proof. We proceed by complete induction. We note that $k = 1$ is true. Then our induction hypotheses are that the above holds for $1 \leq m \leq k$, and we must show it holds for $m = k + 1$.

By properties of linear projection, we have that $\text{Proj}_n = \text{Proj}_n \text{Proj}_{n+g}$ for $g \geq 0$, since necessarily we have that $\mathcal{H}_n \subseteq \mathcal{H}_{n+g}$ with equality if and only if either $g = 0$ or $w_{n+1}, \ldots, w_{n+g} \in \mathcal{H}_n$.

Therefore,

$$
\begin{aligned}
w_n(k+1) &= \boldsymbol{\phi}^{(k+1)\prime}\boldsymbol{w} && (4.152)\\
&= \mathrm{Proj}_n w_{n+k+1} && (4.153)\\
&= \mathrm{Proj}_n \mathrm{Proj}_{n+k} w_{n+k+1} && (4.154)\\
&= \mathrm{Proj}_n w_{n+k}(1) && (4.155)\\
&= \mathrm{Proj}_n \sum_{i=1}^{p} \phi_i w_{n+k+1-i} && (4.156)\\
&= \sum_{i=1}^{p} \phi_i \mathrm{Proj}_n w_{n+k+1-i} && (4.157)\\
&= \sum_{i=1}^{p} \phi_i \sum_{j=1}^{p} \phi_{nj}^{(k+1-i)} w_{n+1-j} && (4.158)
\end{aligned}
$$

where (4.156) is by the base case, and (4.158) is by the induction hypotheses. Thus since only $w_{n-p+1}, \ldots, w_n$ are used by the predictor, we must have that $\phi_{ns}^{(k+1)} = 0$ for $s > p$. □

4.4.1 On the Predictions

4.4.1.1 On Stationary Models

We note we should have increasing agreement between the limiting form and the exact form as our finite past gets larger. Indeed we will show this is the case.

Theorem 4.4. *As $n \to \infty$, we have that the exact form of the prediction will give rise to the limiting form.*

Proof. Let us write w_{n+k} in terms of $\boldsymbol{\phi}_n^{(k)}$ and $\boldsymbol{w}$ as if it were an AR$(n+k-1)$ process. If we let the form of the expected value of w_{n+k}, which is $w_n(k)$, be our guide, we would write it as

$$w_{n+k} - \phi_{n1}^{(k)} w_n - \ldots \phi_{nn}^{(k)} w_1 = a_{n+k} \tag{4.159}$$

so we would get the same result as Proposition 4.3 if we were to take expected values with respect to $\boldsymbol{w}$.

We note that unless the process is AR(p) with $n > p$, as n increases, we get a better fit in our autoregressive approximation to whatever process we are considering. This is a standard result. It should also be clear that when we take expected values up to time n to obtain the k^{th}-step-ahead prediction, the prediction should become more accurate: not in the MMSE sense, since the predictor is already the MMSE predictor, but in that we have a longer series to predict from. Another way to see this is that $\mathcal{H}_n$ becomes a larger space. Note that this does not change anything for pure autoregressive processes of finite order, since as a consequence of Proposition 4.10 we only project onto the last p values of $\boldsymbol{w}$.

As $n \to \infty$, we have that the autoregressive approximation becomes exact: that is, for any t,

$$w_{t+k} - \sum_{j=k}^{\infty} \pi_j w_{t+k-j} = a_{t+k} \tag{4.160}$$

$$\Rightarrow \pi_k(B) w_{t+k} = a_{t+k} \tag{4.161}$$

for $\pi_k(B) = 1 - \sum_{j=k}^{\infty} \pi_j B^j$. We have $\psi_k(B) = 1 + \sum_{j=k}^{\infty} \psi_j B^j$ being defined by $\psi_k(B) = \pi_k^{-1}(B)$. Note that the usual definitions have $\pi(B) = \pi_1(B)$ and $\psi(B) = \psi_1(B)$.

Therefore we are left with

$$w_{t+k} = \psi_k(B) a_{t+k} \tag{4.162}$$

$$= a_{t+k} + \sum_{j=k}^{\infty} \psi_j a_{t+k-j} \tag{4.163}$$

and, upon taking expected values with respect up to time t, we obtain

$$w_t(k) = \sum_{j=k}^{\infty} \psi_j a_{t+k-j} \tag{4.164}$$

which is exactly the result of Proposition 4.4. □

Thus the limiting form and the exact form will become closer as $n \to \infty$: that is, their difference will tend to zero.

A consequence of this theorem and Proposition 4.10 is that only in the AR(p) case (and integer integrations thereof) do we need only a finite number past to predict the series "perfectly" by projection. For all other processes, as n gets larger, the prediction improves in the sense that the predictions will become more accurate due to a longer past. We note that only in the AR(p) case and its integer integrated forms does the prediction not change at all as n increases. For all other processes, the main idea to take hold of is that the predictions change as n increases.

The fact that the ARIMA($p, d, 0$) forecast for any integer $d \geq 0$ does not change as n increases as long as $n > p + d$ underlies the proof of Theorem 4.3.

4.4.1.2 On Non-stationary Models

Since we cannot have an infinite past for non-stationary models, we seem to be stuck. However, we note that as the past gets larger, even though it cannot tend to infinity, the autoregressive approximation gets better. Thus we see that the exact form of the predictions becomes more comparable to the limiting form: the difference between these forms will get closer to zero.

4.4.2 On the Prediction Error Variances

Since we note that the difference of the forms of the predictions tends to zero when n increases, we must have that the prediction error variances differences tend to zero in the same way.

We note that an AR process prediction error variances will be close (equal up to numerical errors) to its limiting form, and an MA process will be farther away. An ARMA process will have errors somewhere in between: that is, it will usually be smaller than the variance of an associated MA process even if the magnitude of $\phi = \phi_1$ is small, when comparing a MA(q) to an ARMA($1, q$) process. Note that fractional integration (correctly applied: that is, in the autocovariance function) will make the differences between the exact and the limiting variances larger.

4.4.3 The Incorrect Application of the Exact Form

We first note that in the limiting form the incorrect application is equivalent to the correct application, as is shown in (4.85).

We have shown that the exact form of the prediction variance is equal to the limiting form of the prediction variance in the ARFI(p, d^*) case, where $d^* \in (-1, \infty)$. However, having $d^* = d_f + d$, with d_f nonzero, the exact form is no longer exact if we put d^* in the exact formula (4.72). We may argue that since the ARFI(p, d_f) is stationary and invertible and as such we should use its autocovariance function. However, we also note that in the ARFI(p, d_f) case, the incorrect application of the exact form gives the limiting form. We note that from Proposition 4.10 and Theorem 4.4, especially from the consequences of the latter, that the exact prediction, and thus the error variances, will be different from the limiting one. The same arguments follow for other processes.

4.4.4 On Running Time

For long series, the exact formulae take time. The Durbin-Levinson algorithm takes $O(n^2)$ floating point operations (flops) to invert the matrix Γ_n, while the fastest way to compute the prediction error variances for k steps ahead in the non-integrated case takes $O(kn^2)$, since it involves one matrix multiplication and no other computation contains said multiplication. For the integrated series, the fastest way to compute the prediction error variances involves two matrix multiplications. While this takes the same asymptotic number of flops, since the matrix computations can be separated, the increase in time spent is substantial.

The flop count for the limiting form is a polynomial in k, the form of which can be quite complicated. However, since most often $k << n$, the limiting form is much faster, while the limiting form as defined in (4.85) is faster still. For large n, using the limiting form (4.87) or (4.85) may be preferable.

4.5 Examples

4.5.1 Simple Symbolic Examples

Let us begin with a very simple example of an MA(1), with $n = 3$. Then $\gamma_0' = (1+\theta^2, -\theta, 0)$ and Γ_3 is the Toeplitz matrix defined by γ_0. It can be shown (easily in an algebraic programming language such as *Mathematica*) that

$$\Gamma_3^{-1} = \frac{1}{1+\theta^2+\theta^4+\theta^6}\begin{pmatrix} 1+\theta^2+\theta^4 & \theta+\theta^3 & \theta^2 \\ \theta+\theta^3 & 1+\theta^2+\theta^4 & \theta+\theta^3 \\ \theta^2 & \theta+\theta^3 & 1+\theta^2+\theta^4 \end{pmatrix} \tag{4.165}$$

Then for the one-step-ahead predictor, we have $\gamma_1' = (-\theta, 0, 0)$ and that

$$\begin{aligned} \mathrm{Var}(e_3(1)) &= \gamma_w(0) - \gamma_1'\Gamma_3^{-1}\gamma_1 && (4.166)\\ &= \frac{1+\theta^2+\theta^4+\theta^6+\theta^8}{1+\theta^2+\theta^4+\theta^6} && (4.167)\end{aligned}$$

again assuming (without loss of generality) that $\sigma_a^2 = 1$. Recalling that the ψ-weights expansion always has the one-step ahead predictor has a variance of σ_a^2, we note that Equation (4.167) is only equal to one when $\theta = 0$. Recall we will have as our h-step-ahead prediction error variance $1 + \theta^2$ for all $h > 1$.

However, we note that with $n = m \in \mathbb{Z}_{\geq 3}$ that we will have that the 1-step ahead prediction variance can be shown to be

$$\begin{aligned} \mathrm{Var}(e_m(1)) &= \gamma_w(0) - \gamma_1'\Gamma_m^{-1}\gamma_1 && (4.168)\\ &= \frac{1+\theta^2+\cdots+\theta^{2(m+1)}}{1+\theta^2+\cdots+\theta^{2m}} && (4.169)\end{aligned}$$

and as such when $m \to \infty$, we will have that $\mathrm{Var}(e_m(1)) \to 1$.

When we integrate the MA(1) process $d \in \mathbb{Z}_{>0}$ times, the difference between the exact and the approximate 1- to 4-step-ahead prediction error variances for an MA(1) with d symbolic are, for $n = 15$, and $a = \sum_{i=0}^{15} \theta^{2i}$

k	difference
1	θ^{32}/a
2	$d^2\theta^{32}/a$
3	$d^2(1+d)^2\theta^{32}/4a$
4	$d^2(1+d)^2(2+d)^2\theta^{32}/36a$

Table 4.1: The MA(1) prediction error variance differences with symbolic θ and d, $n = 15$

We note a similar pattern occurs with different k and n. The differences between the exact and approximate k^{th}-step-ahead ($k \geq 2$) MA(1) with size n then is postulated to be

$$\frac{\theta^{2n+2}\prod_{j=2}^{k}(j-2+d)^2}{\prod_{j=2}^{k}(k-1)^2\sum_{i=0}^{n}\theta^{2i}} \tag{4.170}$$

We note that when we write the MA(1) as an AR(∞), we have $\phi_j = -\theta^j$ for $j \geq 1$. Using *Mathematica* we have computed the AR(n) TACVFs for several n.

- $n = 3$

$$\left\{\frac{1+\theta^2}{1-\theta^6}, \frac{-\theta}{1-\theta^6}, \frac{-\theta^4}{1-\theta^6}\right\} \tag{4.171}$$

- $n = 5$

$$\left\{\frac{1+\theta^2}{1-\theta^{10}}, \frac{-\theta}{1-\theta^{10}}, 0, 0, \frac{-\theta^6}{1-\theta^{10}}\right\} \tag{4.172}$$

- $n = 21$

$$\left\{\frac{1+\theta^2}{1-\theta^{42}}, \frac{-\theta}{1-\theta^{42}}, 0, \ldots, 0, \frac{-\theta^{22}}{1-\theta^{42}}\right\} \tag{4.173}$$

where there are 15 zeroes in the ellipsis of (4.173). We note that the TACVF an AR(n) for even relatively small n becomes close to the true MA(1) TACVF.

4.5.2 Some Numerical Examples

We present the following tables, with prediction error variances of an ARFIMA($1, 0.45, 1$) and an ARFIMA($1, 1 + 0.45, 1$) process, with $\phi = 0.9$ and $\theta = -0.9$. We note that under certain circumstances in the real world, we may try to fit these as ARFIMA($1, 1 - 0.55, 1$) (that is, $d = 1$ and $d_f = -0.55$) and ARFIMA($1, 2 - 0.55, 1$) processes respectively. We note that d^* is the same regardless of how we formulate the problem. However, the prediction error variances are not. This leads to the question of which to fit, as it is possible we do not know the "true" value of d. In real world data analysis, it may make more sense to choose the simpler model, however.

The limiting columns in the tables are the values where (4.85) was used. When we have d equal to a value, we are applying the exact formula with that d. For example, when $d = 0.45$, we are applying the exact formula incorrectly. When we have d_f equal to a value, it is used in the autocovariance function.

We note that always the limiting form has the smallest error variances, as is expected. The incorrect application of the formula for fractional d is close to the limiting for $n = 10$ and the same when $n = 50$. We also note that the ARFIMA($1, d + 0.45, 1$) has a smaller variance than the ARFIMA($1, (d + 1) - 0.55, 1$) for $d = 0, 1$ for all lags. This is to be expected: even though the autocovariances worked with will be smaller for $d_f = -0.55$, with $d = 1$ or 2, the use of the exact integrated formula will tend to make the variances larger.

		$n = 10$			$n = 50$		
k	Limiting	$d = 0.45$	$d = 1, d_f = -0.55$	$d_f = 0.45$	$d = 0.45$	$d = 1, d_f = -0.55$	$d_f = 0.45$
1	1	1.1	1.097	1.123	1	1.004	1.003
2	6.063	6.245	6.335	6.346	6.063	6.108	6.093
3	13.66	13.9	14.25	14.18	13.66	13.81	13.76
4	22.91	23.18	24.01	23.77	22.91	23.25	23.14
5	33.19	33.48	35.02	34.52	33.19	33.82	33.62

Table 4.2: The ARFIMA(1, 0.45, 1) prediction error variances

		$n = 10$			$n = 50$		
k	Limiting	$d = 1.45$	$d = 2, d_f = -0.55$	$d = 1, d_f = 0.45$	$d = 1.45$	$d = 1, d_f = -0.55$	$d = 1, d_f = 0.45$
1	1	1.1	1.097	1.123	1	1.004	1.003
2	11.56	12.11	12.24	12.33	11.56	11.64	11.62
3	47.64	49.15	50.09	50.1	47.64	48.09	47.94
4	129.5	132.6	136.2	135.5	129.5	131.1	130.6
5	279.6	284.9	294.9	292.3	279.6	283.9	282.5

Table 4.3: The ARFIMA(1, 1 + 0.45, 1) prediction error variances

4.6 Conclusions

We have discussed predictions and prediction error variances for stationary and non-stationary processes with stochastic trend. We have given proofs of equivalence for the limiting form and the exact form of the variances under the assumption that the underlying process is strictly autoregressive.

Chapter 5

The arfima Package

5.1 Introduction

In this chapter the **arfima** ***R*** package is formally presented, which fits ARMA-HD models via exact maximum likelihood. This package also performs exact simulation and prediction. It seems to be the only ***R*** package that performs time series analysis with the Box-Jenkins transfer-function noise model and we generalize the noise to include ARFIMA. It is the first of its kind to look at multimodality in time series. The merits of the package will be discussed further in §5.4. While the **arfima** package can mix any of FD, FGN, and PLA noise with ARMA structure, note that all other ***R*** packages only use ARFIMA as their mixed models, and as such those models will be the focus of this chapter.

5.2 The Need for Exact Maximum Likelihood

In this section, the need for exact maximum likelihood in software is discussed. We have done many experiments with approximate likelihood methods, and found them to be very good in some ways and very bad in others. There are, of course, things to be desired in exact maximum likelihood, most notably speed of computations. There is also the problem of near-singular matrices at extreme points of a log-likelihood surface, which we will address next.

5.2.1 Near Singular Matrices

Beran [1994], page 108, notes that besides a large amount of computing time, exact maximum likelihood can be burdened by ill-conditioned matrices that are almost singular. The use of increasingly powerful (and precise) computers and the use of specialized algorithms mitigate these effects. As part of his example, Beran states that the correlation matrix for an FGN process with $H = 0.9$ for $n = 100$ (call this matrix D) has a determinant of approximately 5×10^{-39}; also, the largest eigenvalue divided by the smallest eigenvalue was approximately 222. We calculated the same results.

In Beran [1994], the ratio of eigenvalues was used to calculate the condition number of the matrix. Recall the condition number of a matrix is with respect to a norm: we use the ℓ_2-norm, as was done implicitly in Beran. Trefethen and Bau [1997] note that for this norm, the ratio of the largest and smallest singular values of a matrix give the condition number: in the case of Toeplitz matrices, these are the same ratios. It should also be noted that one generally looks at the condition number of the matrix relative to the size of the matrix: however, this point will be ignored. If the condition number of a matrix A is τ, one can expect to "lose" approximately $\log_{10}(\tau)$ significant digits by the inversion of the matrix - see, e.g. Cheney and Kincaid [2007]. For example, a perfectly well-conditioned matrix has $\tau = 1$ and loses no digits. For most machines today, the approximate machine precision is $\epsilon \approx 2.22 \times 10^{-16}$. For the example in Beran, $\tau = 222$ and $\log_{10}(\tau) = 2.35$. The matrix D was inverted with the **ltsa** function `TrenchInverse` to get $E \approx D^{-1}$. The maximum absolute difference of $DE - I_{200}$ was about 2.11×10^{-15}, and so it is noted that the rule of thumb is overestimating the number of digits lost, up to the machine precision in subtracting the elements of the matrices.

As a test, the correlation matrix of an FGN process with $H = 0.99$ and $n = 5000$, call this F, was computed. The determinant reported by ***R*** was 0, with $\tau = 245908.3 \Rightarrow \log_{10}(\tau) \simeq 5.39$. The inverse of said matrix was computed using the **ltsa** function `TrenchInverse`; when this was multiplied by the original matrix, the maximum absolute value of the product minus the identity matrix was within 2.58×10^{-13}. Notice that F is much larger than D and that it has a value of H much closer to the stationarity boundary, and yet loses only about three more digits in estimated precision and about two more in actual precision.

The covariance and correlation matrices for the FD case with $n = 5000$ and $d = 0.49$ were also calculated. The determinant of the covariance matrix was approximately 143. The determinant of the correlation matrix was presented as 0. Both matrices were inverted using the `TrenchInverse` algorithm of **ltsa**. With the correct scaling factor to account for the division of the theoretical autocovariance function by its first element, the two matrices had maximum difference reported as 0. However, this test is misleading, as the covariance and correlation matrices have the same condition number, with $\log_{10}(\tau) \simeq 5.13$. This again overestimates the loss in significant digits, since when both matrices are multiplied by their inverses and subtracted from the identity, the maximum absolute error is about 2.6×10^{-13} for the correlation matrix and 1.46×10^{-13} for the covariance matrix. The differences here are likely due to underflow.

The inverse and determinant for the log-likelihood are computed relatively efficiently using the Durbin-Levinson or Trench algorithm, withstanding for the most part poorly-conditioned matrices. However, not all likelihood values are necessarily computable: for example as above with and a FD series was generated with d_f set to 0.49, the log-likelihood was not computable for either of the **ltsa** functions `DLLoglikelihood` and `TrenchLoglikelihood` for one test series with the generating d_f. This particular series will be called M. However, the effect of this is sufficiently small. The exact algorithm had no problem finding the MLE for the data: we believe that for any given data set, the non-computable regions are very small. We have experimented with grids around this and a few other non-computable points, and found this to be true.

All of this addresses ill-conditioned matrices. However, while exact maximum likelihood may lose a few digits, it is as exact as it can be up to machine precision, while approximate maximum likelihood is by definition not exact. It is our belief that it is better to lose a few digits of precision than not be exact. Thus the only advantage that approximate ML has over exact is speed. While advantage can be considerable, we will show that approximate maximum likelihood may be seriously flawed.

5.3 On Other R Packages that Deal with ARFIMA Models

There are three ***R*** packages that deal with ARFIMA models we are currently aware of: **longmemo** and its extension, **afmtools**; and the **fracdiff** package. The former two maximize the Whittle log-likelihood, while the latter approximates exact maximum likelihood, except in the case of FD, where it can be exact. Since the **arfima** deals with exact maximum likelihood, **fracdiff** will be the focus of the comparisons in this chapter. Note also that **fracdiff** seems to be the most popular ***R*** package for ARFIMA models.

5.3.1 The Haslett-Raftery Method and the fracdiff *R* Package

The **fracdiff** package is widely used. It is very fast, one of the major things it has in its favour. However, its speed depends on two things: the first is that it is coded almost entirely in ***C*** generated from ***Fortran***. While it has been cleaned up by the maintainer of the package, this makes it hard to see what is truly being done, and thus hard to alter or extend. The second is that **fracdiff** uses several approximations and heuristics. While one major approximation is controllable through the `fracdiff` command, the others are not.

Note that the **fracdiff** approximations were first used by Haslett and Raftery [1989] for what was certainly a long memory time series with $n = 6574$ at each of 11 spatially correlated stations in a time of minimal computing power. These approximations usually serve well when considering long memory: using $M = 100$ in the below often yields a fairly good fit for strongly persistent processes, but not anti-persistent ones. Even when this particular approximation is removed, the **fracdiff** package often approximates the log-likelihood surface poorly.

5.3.2 The Approximations Used

The package **fracdiff** uses several approximations: as listed in the original Haslett and Raftery paper, and some further approximations and heuristics implemented in the C code that is interfaced to R. These are the approximations used for the fitting function, and do not include such issues as simulation. The simulation done by **fracdiff** can also be poor: this is not addressed in this thesis.

5.3.2.1 The Approximations Listed in the Paper

The procedure outlined in Haslett and Raftery [1989] is as follows, noting that said paper has been paraphrased: first, the approximate conditional mean and variance are calculated. In the paper, a weighted mean is used, since not only temporally but also spatially correlated sequences are discussed. The effect of this can be ignored, as both **arfima** and **fracdiff** only deal with temporal data. After some initial heuristics to identify p, q, and the estimated values of $\theta(B)$, $\phi(B)$, and d are found by methods that will not be detailed, the approximate conditional mean for each t is:

$$\hat{x}_t = \phi(B)\theta(B)^{-1} \sum_{j=1}^{t-1} \phi_{tj} x_{t-j} \tag{5.1}$$

where ϕ_{tj} are as in Chapter 4, from the FD process. The approximate conditional variance is

$$v_t = \sigma_x^2 \kappa \prod_{j=1}^{t-1} \left(1 - \phi_{jj}^2\right)$$

where κ is the ratio of the innovations variance to the variance of the associated ARMA(p, q) process. The approximate concentrated log-likelihood is then given by

$$\ell_c(\beta) = c - \frac{1}{2} n \log\left(\hat{\sigma}^2(\eta)\right) - \log\left(\sum_{t=1}^{n} v_t\right) \tag{5.2}$$

with c being a constant, η being all parameters of the model, and

$$\hat{\sigma}^2(\eta) = \frac{1}{n} \sum_{t=1}^{n} \frac{(x_t - \hat{x}_t)^2}{v_t}$$

Note the last term on the right of (5.2) is not mentioned in Haslett and Raftery [1989]: it was put here because it is in the code. This is once again an approximation, since the v_ts are defined by the FD process only.

Another approximation that Haslett and Raftery use is the restriction of the number of ϕ_{tj}s used. To avoid a large number of calculations of the ϕ_{tj}, a value M, usually set to 100, is given such that

$$\sum_{j=1}^{t-1} \phi_{tj} x_{t-j} \simeq \sum_{j=1}^{M} \phi_{tj} x_{t-j} - \sum_{j=M+1}^{t-1} \pi_j x_{t-j} \tag{5.3}$$

since $\phi_{tj} \sim -\pi_j$ for large j, and π_j is the j^{th} term in the infinite autoregressive representation of the FD process. Then another approximation is used: rather than calculate all π_js, the given algorithm uses

$$\sum_{j=M+1}^{t-1} \pi_j x_{t-j} \simeq M \pi_M \left(1 - \left(\frac{M}{t}\right)^d\right) \bar{x}_{M+1,t-1-M} \tag{5.4}$$

where $\bar{x}_{M+1,t-1-M} = \frac{1}{t-1-2M} \sum_{M+1}^{t-1-M} x_j$.

5.3.2.2 The Code's Heuristics

All of the approximations mentioned in the paper are implemented in the code. There are, however, some further heuristics. First, however, the approximation in (5.4) based on (5.3) is addressed. The only approximation that can be changed with a call to **fracdiff** is the value of M. In all the tests run in preparation for this thesis, the package throws an error when $n > 100$ when fitting anti-persistent models. However, since M is changeable, when M is set to the length of the series, this ceases to be an issue. When this is done, note that (5.1) has the exact likelihood for a FD process.

The code uses as an heuristic a multistep optimization that is very fast. It first estimates the ARMA parameters with $d = 0$ as well as filtering out the mean, and then estimates the optimal d using the output. All optimizations of d are done using the `fmin` algorithm of Brent [1973]. All long memory or anti-persistent effects from the given d are filtered out, and then estimates the ARMA parameters again. This is repeated until convergence.

While there can be a great deal more to be said about **fracdiff**, the discussion of the package's inner workings will be left as it stands. Most of **fracdiff**'s speed comes from approximations and heuristics, which gives rise to a problem that will be mentioned in §5.4.

5.4 What the arfima Package Can Do

In this section, the many uses of the ARFIMA ***R*** package will be outlined, including what it can do that other packages cannot.

As has been mentioned previously, **fracdiff**, as well as other comparable ***R*** packages, is approximate. One of the larger hindrances of this is that approximate methods do not reflect the loglikelihood surface precisely. This is a problem when there is multimodality, as well as normal parameter estimation. While **fracdiff** is certainly capable of estimating parameters well in certain conditions, there are other conditions under which **fracdiff** does very poorly.

In the **arfima** package, the data are passes to the `DLLoglikelihood` function from **ltsa** with the TACVF of the process generated by whatever point we are at: this is driven by the `optim` function in ***R***. This ensures exact maximum likelihood up to numerical errors.

5.4.1 Calculating the Log-Likelihood, Simulating, and Forecasting

Two algorithms that are used (via **ltsa**) in our package for calculating the likelihood, simulating, and forecasting will be mentioned. The first is the Durbin-Levinson algorithm, while the second is the Trench algorithm.

Recall from Chapter 4 that the best linear one-step ahead predictions are of the form $w_n(1) = \hat{w}_{n+1} = \phi_{n1}w_n + \ldots + \phi_{nn}w_1$, with mean squared errors of the prediction as $v_n = \mathrm{E}[(w_{n+1} - \hat{w}_{n+1})^2]$. The Durbin-Levinson algorithm is an efficient way of computing the $\boldsymbol{\phi}_n$ and v_n in $O(n^2)$ time.

See, e.g., Brockwell and Davis [1991] for a derivation.

The Trench algorithm is an efficient way to calculate the inverse and determinant of a Toeplitz matrix. It also requires $O(n^2)$ flops. The algorithm is in, e.g., Golub and Van Loan [1996].

The unrestricted likelihood for a series $\boldsymbol{w}$ can be written, up to constant terms,

$$L(\Phi^*|\boldsymbol{w}) \propto |\Gamma_n|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{w}-\mu_w\mathbf{1})' \,\Gamma_n^{-1}\,(\boldsymbol{w}-\mu_w\mathbf{1})\right) \tag{5.5}$$

Now suppose $\mu_w = 0$ without loss of generality. Then the unrestricted log-likelihood function is

$$\begin{aligned} \ell(\Phi^*|\boldsymbol{w}) &= c - \frac{1}{2}|\Gamma_n| - \frac{1}{2}\boldsymbol{w}'\Gamma_n^{-1}\boldsymbol{w} && (5.6) \\ &= c - \frac{1}{2}\log(\sigma_a^2) - \frac{1}{2}\log(g_n) - \frac{1}{2\sigma_a^2}\sum_{j=1}^{n}\frac{(w_j-\hat{w}_j)^2}{v_{j-1}} && (5.7) \\ &= c - \frac{1}{2}\log(\sigma_a^2) - \frac{1}{2}\log(g_n) - \frac{1}{2\sigma_a^2}S(\Phi) && (5.8) \end{aligned}$$

where $g_n = \prod_{t=0}^{n-1} v_t$ and (5.7) is by application of the ideas in Chapter 4.

Maximizing (5.8) with respect to σ_a^2 we obtain $\hat{\sigma}_a^2 = S(\Phi)/n$ as in Chapter 2. The concentrated likelihood function is then

$$\ell_c(\Phi|\boldsymbol{w}) = c - \frac{1}{2}\log(S(\Phi)/n) - \frac{1}{2}\log(g_n) \tag{5.9}$$

Note the similarity of (5.2) to (5.9). Recall, however, the $\hat{x}$'s in the Haslett-Raftery paper were computed using only the FD autocorrelations rather than the full model. The minimum mean squared error linear one-step-ahead predictors correspond to Haslett and Raftery's conditional means. Also recall that the likelihood of one set of parameter values Φ is always up to an additive constant.

In simulation, the Durbin-Levinson algorithm (for finding the best linear predictors) comes to our aid once more. Using the TACVF of the process and said algorithm, let

$$\begin{aligned} w_1 &\sim \mathrm{WN}(0, \gamma_w(0)) && (5.10) \\ w_t &= \phi_{t-1,1}w_{t-1} + \cdots + \phi_{t-1,t-1}w_1 + e_t && (5.11) \end{aligned}$$

with

$$e_t \sim \mathrm{WN}(0, v_{t-1}) \tag{5.12}$$

where the white noise is usually, although not always, Gaussian. If a non-zero mean for the series is desired, it is added to the series at the end.

For forecasting, recall (4.33)

$$w_n(k) = \mu_w + \gamma_k'\Gamma_n^{-1}(\boldsymbol{w} - \mathbf{1}\mu_w) \tag{5.13}$$

for which the Trench algorithm can be to obtain the inverse autocovariance matrix.

5.4.2 More on the arfima Package

As was mentioned earlier, the **arfima** package is capable of exact maximum likelihood estimation, simulation, and forecasting. As was noted in the introduction to this chapter, it is capable of transfer function modelling as in Box et al. [2008b]. Unlike other packages and due to its use of the TACVF, **arfima** can have mixed ARIMA-HD models for three of the four HD models. Since the PLS model autocovariance structure is difficult at best to evaluate exactly and efficiently, said model was left out.

The **arfima** package also deals with integer differencing, both seasonal and non-seasonal. It is the first package that deals with exact prediction when there is integer differencing required in the model - see Chapter 4.

Seasonal ARIMA-HD noise can be included in the models. Since the non-seasonal and seasonal can mix as in Chapter 3 we can have two different types of noise, from white noise to the HD noises available.

The **arfima** package is also the only package that we are aware of that performs multiple starts for an analysis of time series. Note that there can be more than one mode for certain time series models. We have also introduced a new visual diagnostic tool that allows us to check for spurious modes, which we will mention in §5.5.4.

5.5 Package Details

First we note that in the **arfima** package, the main fitting function, as well as all of the utility functions associated with it, are based on the assumption that there will be multiple modes. That is, unless the parameters are specified exactly to only have one starting point, the main fitting function of the package, `arfima`, will start the optimizations at multiple starting points. Every other function, aside from the simulation function, `arfima.sim`, is meant to deal with a possibly multimodal loglikelihood surface and thus multiple fits.

This was done as in certain circumstances, discussed in Chapter 6, multimodality of the loglikelihood surface tends to occur. We have several hypotheses about the nature of log-likelihood surfaces of data that are bimodal. While the inclusion of HD parameters in the model may make the surface multimodal, there are ARMA processes that are also multimodal. We stress that the appearance of multimodality on a loglikelihood surface is highly dependant on both the data and the model. This will be discussed in Chapter 6.

5.5.1 Dealing with the Estimation of μ_w

In some way the estimation of μ_w must be dealt with. This must be done since if the true mean of the series is non-zero, the estimates will often be far from the true parameters - of course, assuming an underlying structure. There are multiple ways to do this: the first is to simply use $\bar{w}$ (once $\boldsymbol{w}$ is stationary) as the estimate. This is certainly the simplest method, and in fact one option in **arfima**. There is a guarantee that $\bar{w} \to \mu_w$ in mean square and probability for all ARMA-HD models by Theorem 3.6. It should be noted that the **fracdiff** package filters out the mean of the series in estimation, and thus it doesn't matter whether the mean is subtracted out or not during the fit. The `arfima` function of the package dynamically estimates the mean as the default. There is also the option of iteratively estimating the mean, which is done with `itmean = TRUE` in the `arfima` function. This method is laid out in McLeod et al. [2007a]. The fourth was of fitting a mean is to set it to a specific value.

5.5.2 The Partial Autocorrelation Space for AR and MA Coefficients

When considering ARMA parameters, it is important to note that if p and/or q are greater than 1, the admissible space for the stationarity and/or invertibililty of the parameters becomes complex. Monahan [1984] introduced a formulation (and its inverse) for autoregressive coefficients such that all stationary transformed coefficients occur within the open p-cube $(-1,1)^p$. This is based on Barndorff-Nielsen and Schou [1973]. Due to duality (see, eg. McLeod [1984]), the MA parameters are invertible if under the parametrization if they are in the open q-cube $(-1,1)^q$.

We sometimes call this space the transformed space or "PACF" space. It is called this in reference to that it transforms the AR coefficients to their partial autocorrelation function coefficients via a transformation equivalent to the Yule-Walker equations. Note that it is in this space that is checked whether the AR or MA parameters are close to the stationarity and/or invertibility boundaries. This is also how the multistart procedures are performed: a grid of parameter values in the AR and MA transformed spaces are generated, as well as in the HD space if fitting an ARIMA-HD model. Then the AR and MA parameters are transformed back into their normal spaces to start the optimizations.

5.5.3 Functions in the Package

We will list the primary functions in the package, as well as what they are used for.

- `arfima.sim` - The simulation function. This function only simulates univariate time series, and regressions, transfer function data, and the like, have to be added manually. These capabilities may be added to a later version of the package.

- `arfima` - The fitting function. Can fit ARIMA-HD models to data, by default in multi-start. The type of HD process, as well as fitting only short-memory processes, are options. The number of starts is also an option. The default is to have FD as the HD process as well as 2 starting points for each variable, in a grid. This function can fit regression data and transfer functions (see, e.g. Box et al. [2008b]). Allows the mean to be fit dynamically, by the mean of the data or by a user provided theoretical mean. This function also allows a choice of optimizers (to use in `optim`), although generally the default, BFGS, is recommended, although this may cause some problems. A `numeach` option is available to allow the user how many starts the ARMA parameters get each, and how many starts the HD parameters get. The default is `numeach = c(2, 2)` for considerations of speed: the user may want to change this depending on the nature of the problem. Note also that multiple cores of the computer can be used if available, through the `cpus` command.

- `weed` - While the `arfima` function has (as default) automatic 'weeding' out of modes that are too close to each other (`autoweed = TRUE`), one may wish to call `weed` after the fit to get rid of modes that seem too close to each other. That is, if we set `autoweed = FALSE` since we want to see all modes, or if we believe there are too many modes, we can use the `weed` function to eliminate modes. The parameters for the weed function are listed in the documentation of the package: they can specify how far apart the modes have to be to be weeded, which space the distance is calculated in (either the untransformed space, the transformed PACF space, or both), and what p-norm to use (default is Euclidean with $p = 2$).

- `removeMode` - Allows the user to manually remove modes.

- `predict.arfima` - For each mode found by `arfima`, predicts from the data using MMSE forecasting and prediction standard errors (including with integer d and d_s), as well as (by default) a bootstrap forecast and prediction intervals based on the residuals of that particular mode's fit. Limiting standard deviations are also included if available: this occurs when the model is writable in operator format, in particular the ARFIMA class of models.

- `distance` - Calculates the distance between modes with respect to a $p-$norm (default has $p = 2$, Euclidean distance) in both the normal operator space and the transformed "PACF" space, as well as the HD parameter.

- `tacvf` - Extracts the TACVFs of the fitted object

- `tacfplot` - Plots the TACFs of different fits to the same data

- Utility functions such as `plot`, `print`, `fitted`, `residuals`, and `summary` are available for those objects that they make sense for

- Currently there are two data sets in the package: Series J from Box et al. [2008b] to illustrate the use of transfer functions, and tmpyr, Central England temperature data from 1659 to 1976, as given by Manley [1974] and Parker et al. [1992].

- `ARToPacf` and `PacfToAR` - The former transforms AR/MA parameters into the PACF space, while the latter transforms them back

Note that standard errors reported by the `arfima` function are calculated using the inverse of the Hessian matrix, as is often done. Also, when there is only ARFIMA or ARMA present, including the seasonal cases, we have derived and implemented the expected information matrix to allow for theoretical standard errors, seen in the `summary.arfima` function. The derivation of the information matrix takes the form of the one in Li and McLeod [1986].

The four ways of fitting the mean are as follows: `dmean = TRUE`, the default, dynamically fits the mean, where `dmean = FALSE` fits the mean with the sample mean. Having `dmean = b`, with b a number, fits the mean with that number, and having `itmean = TRUE` iteratively fits the mean. Note that having `dmean = TRUE` and `itmean = TRUE` will produce a warning, and the mean will be fit iteratively.

5.5.3.1 The Choice of Optimizer

The choice of optimizer can have a large impact on the modes that are found. We have as the default the BroydenŰFletcherŰGoldfarbŰShanno optimizer, otherwise known as BFGS. It is a quasi-Newton method - see, e.g. Wikipedia [2012a]. This method is generally recommended, as the modes found by said optimizer are generally more accurate according to visualizations we have performed. The one problem with BFGS is that the optimizer may become “trapped” on a boundary of the space that is optimized over (and corresponding to a boundary of the transformed space). The Nelder-Mead optimizer (see, e.g. Wikipedia [2012c]) does not usually have this problem, but as it is an approximate simplex method, the results of the optimization may not be exact. We have seen the Nelder-Mead optimizer find modes that are not apparent in visualization, as well as not finding true modes.

5.5.4 Considering the Critique of a Mode

Before the `tacfplot` function and visualization of surfaces was implemented, we suspected most, if not all, modes on boundaries of being spurious. One can now look at the now look at TACF plots to partially critique how well a certain mode fits the data. The usefulness of this will be seen especially in Chapter 6: similarities and differences in the modes can be seen by looking at the TACF plots. It can also be seen under certain conditions that some modes of a certain fit are spurious. There may also be confirmation that a mode is not spurious in a TACF plot, even though the mode is on a boundary.

Of course there are other things to consider when trying to classify a mode as spurious. Often one can try to visualize the surface in some manner as well, when this is possible: our routines for two-dimensional viewing of each mode by each variable and log-likelihood are not well developed and currently not part of the package.

5.6 Numerical Results

A number of numerical results, mostly comparing the **arfima** and **fracdiff** packages, are presented. We ran a function called `fracdiffMM`, based on the **arfima** package Note that **fracdiff** was not changed in any way: it was simply called from the script.

5.6.1 The `fracdiffMM` Script

The `fracdiffMM` script was written so the fitting function in **fracdiff** could be compared to the fitting function in **arfima**. It uses `fracdiff` function with multiple starting values. We cannot claim parity between the functions, since in the `fracdiff` function, only the ARMA parameters can be set. However, generally in our numerical studies we have set the number of starts for the ARMA parameters higher than we have for the **arfima** based fits.

We will now describe the script in more detail. It allows the user to set the values of p and q for the ARMA structure, and a value we will call m for the number of starts in each dimension. Note that in the code, m is called `numeach` as in **arfima**. If $m = 1$, `fracdiff` is called with no starting values: that is, a regular `fracdiff` fit. If $m > 1$, there were $m^{p+q} + 1$ starts for the `fracdiff` function. The m^{p+q} are the fits for the ARMA parameters: note that the starting points were a grid on the PACF space and transformed into the operator space, as in **arfima**. The additional 1 was a regular fracdiff fit, with no given starting parameters. This was done in case the regular `fracdiff` call gave a better fit. After some experimentation, we realized that sometimes this was the case. More often, however, at least one of the `fracdiff` fits with a given starting parameter would be the same, if not better, in terms of likelihood.

Recall that **fracdiff** does not report a mean in its fitting function, since the mean is filtered out. Since `fracdiff` fits always gave a lower log-likelihood when the sample mean was subtracted, the `fracdiffMM` script took the parameters from fracdiff and optimized the mean with respect to it.

The script was equipped with a weeding function, similar to the one in **arfima**. This was done so as to better see what the fits were doing.

5.6.2 River Flow Data

There are seven data sets from Hipel and McLeod [1994]. They are all river flow data. They were analyzed by `arfima` as well as the `fracdiffMM` script, for the highest AIC. The data sets are mentioned by the code from Hipel and McLeod [1994] only.

Note that many of the starting points in `fracdiffMM` with higher p and q led to the `fracdiff` optimizer failing. There were some cases where all starts failed for certain p and q combinations. These optimization errors are especially important to note, in that `fracdiff` gave some of these failed optimizations a higher log-likelihood than the ones that did not fail. Also, many of these gave non-stationary parameter estimates.

The arf as a subscript denotes values chosen by the **arfima** package. The fd is for the `fracdiffMM` fits, although the AICs are with respect to **arfima**'s log-likelihood calculated with optimal mean subtracted. Finally, the values in the $fd^{\star}$ columns are those chosen by the AIC built into package **fracdiff**; that is, with the log-likelihood as calculated by **fracdiff**. The exact AICs were then calculated for these chosen parameters.

Except for the MSTOUIS and NEUMENAS data sets, there is a difference in order selected between the three models. What is more important is that the GOTA, OGDEN, and RHINE

Data set	$(p,q)_{arf}$	AIC_{arf}	$(p,q)_{fd}$	AIC_{fd}	$(p,q)_{fd^\star}$	$AIC_{fd^\star}$
DANUBE	(1, 0)	1666.99	(0, 0)	1668.86	(0, 0)	1668.86
GOTA	(0, 2)	1334.16	(0, 2)	1334.28	(2, 0)	1335.00
MINIMUM	(3, 2)	-531.73	(0, 0)	-528.41	(0, 0)	-528.41
MSTOUIS	(1, 0)	1397.43	(1, 0)	1397.49	(1, 0)	1397.49
NEUMUNAS	(1, 0)	1207.22	(1, 0)	1207.23	(1, 0)	1207.23
OGDEN	(2, 3)	1177.74	(1, 0)	1178.71	(1, 2)	1180.33
RHINE	(2, 2)	1532.67	(0, 1)	1532.96	(0, 0)	1533.78

Table 5.1: The AICs and order of the ARMA parameters chosen by the `arfima` function, the `fracdiffMM` script as chosen by exact **arfima** AIC, and the `fracdiffMM` script as chosen by the **fracdiff** AIC for seven riverflow data sets found in Hipel and McLeod [1994]

data sets, where the AIC chosen by **fracdiff**'s log-likelihood is different than the exact AIC for **fracdiff**. This highlights our point that the **fracdiff** package does not follow the loglikelihood surface closely, and is not likely to find multiple modes well. As an aside we note that **arfima** performed at least slightly better than **fracdiff** in all cases.

It should be noted that when we subtract the sample mean from the series before evaluating the `fracdiff` log-likelihoods, usually the **arfima** fits with `dmean = FALSE` have lower AICs by a fair margin. The **arfima** AICs from the fits with no dynamic mean were fairly close to the **arfima** ones in Table 5.1.

5.6.3 Temperature Data

The temperature data are from Manley [1974] and Parker et al. [1992]. The data are from central England and best described by in `http://www.metoffice.gov.uk/hadobs/hadcet/`. We will examine the data from 1659 to 1976 ($n = 318$): these are the years analyzed by Hosking [1984] and Bhansali and Koboszka [2003].

Hosking [1984] suggested an ARFIMA$(1, d_f, 0)$ to fit these data, although the ARFIMA$(1, d_f, 1)$ gives a lower AIC. Hosking also notes that an ARMA$(1, 1)$ may be suitable. The data were fit to the ARFIMA$(1, d_f, 1)$ for the purposes of this chapter, using the **arfima** package and the `fracdifMM` script.

```
> library(arfima)
> library(fracdiff)
> source("MMfdandweed.R")
> data(tmpyr)
> fit.a <- arfima(tmpyr, order = c(1, 0, 1), numeach = c(3, 4), quiet = TRUE)
> fit.a
```

```
Number of modes: 3
```

```
Call:
arfima(z = tmpyr, order = c(1, 0, 1), numeach = c(3, 4), quiet = TRUE)

Coefficients for fits:
             Coef.1:    SE.1:      Coef.2:    SE.2:       Coef.3:
phi(1)       -0.748239   0.147232   0.987573   0.0186077   0.868822
theta(1)     -0.646645   0.172729   0.18729    0.12563     0.646297
d.f           0.277833   0.050068  -0.626011   0.126001   -0.0322342
Fitted mean   9.15695    0.158699   9.15326    0.070261    9.151
logl          174.879               173.842                173.011
sigma^2       0.335358              0.337585               0.3398
             SE.3:
phi(1)        0.0795705
theta(1)      0.194114
d.f           0.254969
Fitted mean   0.07298
logl
sigma^2
Starred fits are close to invertibility/stationarity boundaries
```

The `arfima` fit gives a trimodal log-likelihood surface. We note that the first (that is, the one with the highest log-likelihood), corresponds roughly to the result found by Hosking [1984] and Bhansali and Koboszka [2003]. The second mode is quite strongly anti-persistent in terms of the fractional differencing parameter, and the third corresponds closely to the ARMA$(1,1)$ found by Hosking.

Now we will look at a `fracdiffMM` fit.

```
> fit.fd.a <- fracdiffMM(tmpyr, p = 1, q = 1, numeach = 8)


Beginning fracdiff fits with 65 starting values.


> fit.fd.a <- weedfd(fit.fd.a)
> fit.fd.a


Number of modes: 1

Call:
fracdiffMM(z = tmpyr, p = 1, q = 1, numeach = 8)

Coefficients for fracdiff fits:
            Coef.1:    SE.1:
phi(1)       0.982476  NA
theta(1)     0.196839  NA
df          -0.614804  NA
muHat        9.15083    0.0575599
```

```
zbar         9.14346
logl.muHat   173.793
logl.zbar    173.785
sigma^2      0.334709
Starred fits are close to invertibility/stationarity boundaries
NAs occur when fracdiff cannot compute the correlation matrix
```

The `fracdiffMM` function on the full range of d_f finds only the second mode. It also had some sort of optimization difficulties, as the standard errors produced by `fracdiff` are all `NA`s. Note the number of starting points: 65.

Therefore we decide to use the `fracdiffMM` script with only the long-memory range for d_f.

```
> fit.fd.b <- fracdiffMM(tmpyr, p = 1, q = 1, numeach = 8, drange = c(0, 0.5))

Beginning fracdiff fits with 65 starting values.

> fit.fd.b <- weedfd(fit.fd.b)
> fit.fd.b

Number of modes: 2

Call:
fracdiffMM(z = tmpyr, p = 1, q = 1, numeach = 8, drange = c(0, 0.5))

Coefficients for fracdiff fits:
            Coef.1:    SE.1:       Coef.2:    SE.2:
phi(1)      -0.778109   0.149456    0.96781    0.0456132
theta(1)    -0.683728   0.125397    0.962403   0.0265489
df           0.27446    0.0480751   0.19541    0.0742388
muHat        9.15675    0.155862    9.1533     0.115241
zbar         9.14346                9.14346
logl.muHat   174.854                171.976
logl.zbar    174.85                 171.972
sigma^2      0.33394                0.337928
Starred fits are close to invertibility/stationarity boundaries
```

When the range is restricted to long memory only, `fracdiffMM` finds the highest mode, as well as a long memory mode that has almost redundant ϕ and θ. While it is possible that this is an actual mode, our tests so far have yet to confirm this; we have tried the `arfima` function with more starts, but no similar modes.

Once again note the importance of exact methods in computing the maximum likelihood estimator. The surface of the log-likelihood seems to have completely changed when we restricted the fractional differencing parameter to be long memory. One would hope that the full parameter range would be enough to find all modes on a surface. With approximate methods such as fracdiff, this seems not to be the case.

5.6.4 Simulation Studies

We ran nine different models with 25 simulations each and $n = 1000$. We then compared parameter estimates as well as a variant of the relative likelihood (2.58) between **fracdiff** and **arfima**. From the **arfima** package, we computed the fits with both dynamic mean estimation and sample mean to compare the results.

To compare the parameter estimates, we computed both the root mean squared error (RMSE) for each parameter. Recall the RMSE is defined as, for a given parameter η, with estimates $\hat{\eta}_s$, $s = 1, \ldots, S = 25$

$$RMSE_{\eta} = \sqrt{\left(\sum_{s=1}^{S} (\eta - \hat{\eta}_s)^2\right) \div S} \tag{5.14}$$

The variant of the relative likelihood that was used was the difference in log-likelihoods. This is simply $DLL(\alpha) = D(\alpha)/2$ in terms of the deviance, or the log relative likelihood. We used this as a measure for the reason that it is easier to visualize. We chose to do a slight alteration of the ideas in Chapter 2 in this chapter: rather than observe the log relative likelihood at some true or optimal value, we chose, for each data set, to subtract the the log-likelihood of the `arfima` fit where the sample mean was used from the `arfima` dynamic mean fit and the `fracdiffMM` fit.

That is, suppose $\ell_{\bar{w}}$ is the **arfima** highest log-likelihood for a particular fit with no dynamic mean. Also, ℓ is the highest log-likelihood from either the dynamic mean **arfima** fit or a `fracdiffMM` fit. Then we calculated $DLL = \ell - \ell_{\bar{w}}$ for each data set, and plotted the results in Figure 5.1.

Below is a table specifying the models.

Model	ϕ	θ	d_f
1	$\emptyset$	0.94	0.42
2	$(0.8, 0.19)$	0.94	0.42
3	$(0.8, -0.2)$	$\emptyset$	0.3
4	$(0.7, 0.29)$	$(0.9, 0.09)$	0
5	$(0.7, -0.3)$	$(0.4, -0.2)$	0
6	$(0.8, -0.2)$	$\emptyset$	-0.4
7	0.96	$\emptyset$	-0.6
8	0.96	0.4	-0.6
9	$(0.96, 0.03)$	$\emptyset$	-0.6

Table 5.2: Model Specifications for the Simulation Studies

The models were chosen for specific reasons. Models 1, 4 and 7 were chosen as series generated from them were likely to be multimodal from our previous experience. Models 2, 8, and 9 were chosen as they were similar to said models (2 is similar to 1, while 8 and 9 are similar to 7). It was expected that the addition of extra parameters would either create modes or make modes more difficult to find.

Models 3, 5, and 6 were chosen as they were very likely to have one mode; thus the fits from **arfima** and **fracdiff** could also compete on a unimodal surface.

The simulations were performed in the following way: the 25 seeds were chosen randomly (subject to an overriding seed, 4563, and sampled from the numbers 1 to 10000 by R), so that each model had the same seeds generating the data. Then the `arfima` function and `fracdiffMM` function from our script were run on the data.

The RMSEs from the mode with the highest log-likelihood were computed. Also, since we knew the models, we did a search of all modes from each fit to see which mode was closest to the true generating parameters. These are the rows that are starred.

The starred rows are the only way we can really tell if one of the modes found is close to the generating parameters. We note that in a multimodal surface it is possible that the mode corresponding to the generative parameters is not the highest mode. This also leads to a criterion for seeing if multiple modes are found by each package without looking at the individual fits: if the unstarred RMSE is higher than the starred RMSE, then there are modes found with a higher log-likelihood than the mode closest to the generative parameters. The only way for this criterion to fail is for all of the series generated, the log-likelihood of the mode closest to the generative parameters is always the higher one. This, while possible, is extremely unlikely.

The **arfima** fits had `numeach = c(3, 4)`, while the `fracdiffMM` fits had 6 starts for the ARMA parameters: we recall there is no way to select the number of starts for the fractional differencing parameter without changing the **fracdiff** package itself. Note that the mean had no difference on the `fracdiffMM` fits parameter estimation that we could control.

Method	$\theta = 0.94$	$d_f = 0.42$
`arfima`$_{(3,4)}$, $\hat{\mu}^\star$	0.02	0.036
`arfima`$_{(3,4)}$, $\hat{\mu}$	0.198	0.205
`arfima`$_{(3,4)}$, $\bar{w}^\star$	0.033	0.036
`arfima`$_{(3,4)}$, $\bar{w}$	0.135	0.138
`fracdiffMM`$^\star_{(6,1)}$	0.48	0.486
`fracdiffMM`$_{(6,1)}$	0.48	0.486

Table 5.3: Model 1 RMSEs: it would seem that **fracdiff** is only finding one mode of this often bimodal surface, while the **arfima** fits are finding both. While this shows that **arfima** does much better, recall that we only have one parameter that **fracdiff** has multiple starts in.

Method	$\phi_1 = 0.8$	$\phi_2 = 0.19$	$\theta = 0.94$	$d_f = 0.42$
`arfima`$_{(3,4)}$, $\hat{\mu}^{\star}$	0.053	0.047	0.027	0.064
`arfima`$_{(3,4)}$, $\hat{\mu}$	0.097	0.097	0.156	0.239
`arfima`$_{(3,4)}$, $\bar{\varpi}^{\star}$	0.078	0.047	0.093	0.056
`arfima`$_{(3,4)}$, $\bar{\varpi}$	0.096	0.096	0.155	0.237
`fracdiffMM`$^{\star}_{(6,1)}$	0.178	0.175	0.331	0.474
`fracdiffMM`$_{(6,1)}$	0.178	0.175	0.331	0.474

Table 5.4: Model 2 RMSEs: another case where **fracdiff** has trouble finding multiple modes when they exist; the **arfima** starred modes do very well

Method	$\phi_1 = 0.8$	$\phi_2 = -0.3$	$d_f = 0.3$
`arfima`$_{(3,4)}$, $\hat{\mu}^{\star}$	0.054	0.028	0.053
`arfima`$_{(3,4)}$, $\hat{\mu}$	0.054	0.028	0.053
`arfima`$_{(3,4)}$, $\bar{\varpi}^{\star}$	0.054	0.028	0.053
`arfima`$_{(3,4)}$, $\bar{\varpi}$	0.054	0.028	0.053
`fracdiffMM`$^{\star}_{(6,1)}$	0.057	0.029	0.054
`fracdiffMM`$_{(6,1)}$	0.057	0.029	0.054

Table 5.5: Model 3 RMSEs: we were sure that this surface was unimodal, which it seems to be from this table. **arfima** has a slight advantage, but the results are comparable.

Method	$\phi_1 = 0.7$	$\phi_2 = 0.29$	$\theta_1 = 0.9$	$\theta_2 = 0.09$	$d_f = 0$
`arfima`$_{(3,4)}$, $\hat{\mu}^{\star}$	0.134	0.119	0.136	0.108	0.075
`arfima`$_{(3,4)}$, $\hat{\mu}$	1.366	0.69	1.608	0.447	0.627
`arfima`$_{(3,4)}$, $\bar{\varpi}^{\star}$	0.197	0.158	0.199	0.176	0.16
`arfima`$_{(3,4)}$, $\bar{\varpi}$	1.377	0.732	1.55	0.489	0.585
`fracdiffMM`$^{\star}_{(6,1)}$	0.642	0.415	0.778	0.322	0.25
`fracdiffMM`$_{(6,1)}$	0.967	0.462	1.062	0.326	0.253

Table 5.6: Model 4 RMSEs: **fracdiff** finally seems to find multple modes: however, as we see from Figure 5.1 that the high modes found by **arfima** are superior; we also see superiority in the starred fits

Method	$\phi_1 = 0.7$	$\phi_2 = -0.3$	$\theta_1 = 0.4$	$\theta_2 = -0.2$	$d_f = 0$
$\texttt{arfima}_{(3,4)}, \hat{\mu}^\star$	0.245	0.115	0.269	0.134	0.213
$\texttt{arfima}_{(3,4)}, \hat{\mu}$	0.623	0.434	0.797	0.141	0.605
$\texttt{arfima}_{(3,4)}, \bar{\omega}^\star$	0.291	0.145	0.238	0.132	0.212
$\texttt{arfima}_{(3,4)}, \bar{\omega}$	0.637	0.438	0.793	0.142	0.545
$\texttt{fracdiffMM}^\star_{(6,1)}$	0.478	0.195	0.392	0.176	0.201
$\texttt{fracdiffMM}_{(6,1)}$	0.631	0.217	0.533	0.15	0.231

Table 5.7: Model 5 RMSEs: we see the same pattern as Table 5.6; **arfima** does better on the whole, although not by as much. We had thought this set of parameters to be unimodal; either we were wrong, or the overfitting with d_f induced modes: see the text.

Method	$\phi_1 = 0.8$	$\phi_2 = -0.2$	$d_f = -0.4$
$\texttt{arfima}_{(3,4)}, \hat{\mu}^\star$	0.053	0.028	0.052
$\texttt{arfima}_{(3,4)}, \hat{\mu}$	0.053	0.028	0.052
$\texttt{arfima}_{(3,4)}, \bar{\omega}^\star$	0.052	0.028	0.051
$\texttt{arfima}_{(3,4)}, \bar{\omega}$	0.052	0.028	0.051
$\texttt{fracdiffMM}^\star_{(6,1)}$	0.052	0.028	0.052
$\texttt{fracdiffMM}_{(6,1)}$	0.052	0.028	0.052

Table 5.8: Model 6 RMSEs: we were sure that this surface was unimodal, which it seems to be from this table. The results are comparable.

Method	$\phi = 0.96$	d_f = -0.6
$\texttt{arfima}_{(3,4)}, \hat{\mu}^\star$	0.685	0.698
$\texttt{arfima}_{(3,4)}, \hat{\mu}$	0.699	0.715
$\texttt{arfima}_{(3,4)}, \bar{\omega}^\star$	0.009	0.024
$\texttt{arfima}_{(3,4)}, \bar{\omega}$	0.14	0.155
$\texttt{fracdiffMM}^\star_{(6,1)}$	0.218	0.224
$\texttt{fracdiffMM}_{(6,1)}$	0.218	0.224

Table 5.9: Model 7 RMSEs: this set of parameters was known to us to generally give a bimodal surface. We note that dynamic mean estimation did very poorly here, which we will discuss in Chapter 6. The mean subtracted version did much better. **fracdiff** seemingly only found one mode.

Method	$\phi = 0.96$	$\theta = 0.4$	$d_f = -0.6$
`arfima`$_{(3,4)}$, $\hat{\mu}^\star$	0.153	0.095	0.191
`arfima`$_{(3,4)}$, $\hat{\mu}$	0.153	0.095	0.191
`arfima`$_{(3,4)}$, $\bar{w}^\star$	0.017	0.106	0.127
`arfima`$_{(3,4)}$, $\bar{w}$	0.026	0.108	0.141
`fracdiffMM`$^\star_{(6,1)}$	0.029	0.12	0.155
`fracdiffMM`$_{(6,1)}$	0.029	0.12	0.155

Table 5.10: Model 8 RMSEs: this set of parameters, thought to possibly lead to bimodal surfaces, seems to lead to unimodal surfaces. We checked each fit for this particular case. The changes in the **arfima** estimates come from spurious modes on boundaries, which is sometimes a problem, especially when a task is automated. Surprisingly, the modes were only induced by subtracting the sample mean rather than dynamically estimating it. The latter did a poor job, while the former did a comparable job to **fracdiff**. See Chapter 6 for more on mode induction.

Method	$\phi_1 = 0.96$	$\phi_2 = 0.03$	$d_f = -0.6$
`arfima`$_{(3,4)}$, $\hat{\mu}^\star$	0.052	0.049	0.046
`arfima`$_{(3,4)}$, $\hat{\mu}$	0.06	0.056	0.049
`arfima`$_{(3,4)}$, $\bar{w}^\star$	0.056	0.052	0.051
`arfima`$_{(3,4)}$, $\bar{w}$	0.06	0.056	0.049
`fracdiffMM`$^\star_{(6,1)}$	0.171	0.053	0.172
`fracdiffMM`$_{(6,1)}$	0.171	0.053	0.172

Table 5.11: Model 9 RMSEs: a set of parameters leading to a unimodal surface we thought was possibly bimodal; once again, **arfima** has a small amount of mode induction, although this time it does much better than **fracdiff**

When considering all models, we see that **arfima** most often does better than **fracdiff**, although there are a few models where they are comparable. These particular models are Models 3, 6, and 8 in Tables 5.5, 5.8, and 5.10 respectively. In Models 3 and 6, all fitting methods performed about the same, which was actually what we expected: we did not see much chance for multimodality in these models. In Model 8 we note that ARFIMA dynamic mean estimation fits were much worse than anything **fracdiff** produced. The sample mean fits were comparable, however.

There are multiple modes found by the fits based on package **fracdiff**, although only in Models 4 and 5, in Tables 5.6 and 5.7. However, the modes found are invariably inferior in that their starred RMSEs do not compare with the RMSEs found by those from **arfima**, as well as from the differences in log-likelihoods shown in Figure 5.1. In these two models, there can be quite a few optimization failures on **fracdiff**'s part. We note that since these models are overfit by ARFIMA$(2, d_f, 2)$ models rather than the ARMA fits that they are, we could have used the option `lmodel = "n"` in **arfima** to model these. The whole point of these two models was to see how the two packages compared when overfitting, however.

We were surprised by the apparent multimodality in Model 5, as it was unexpected. While there were a fair number of modes "trapped" on the boundaries, and some modes that would have been eliminated with a call to `weed` with a larger radius, as it turned out, there was some real multimodality there. We suspect that at least part of this was the inclusion of the fractional differencing parameter in the fit. Note that the highest mode, even if on a boundary, is unlikely to be spurious, while the mode closest to the true parameters will usually only be close to a boundary if the true values are.

It is also certain that **arfima** sometimes got caught on the boundary. However, this is not likely to give a higher likelihood. Note that this is a completely different problem than the optimization failures associated with **fracdiff**: those can sometimes have a higher reported log-likelihood as computed by the package than those optimizations that do not fail.

There are three take away messages from the RMSEs. One is that on most, but not all, fairly simple log-likelihood surfaces, **fracdiff** is fairly comparable to **arfima**. Recall how poorly **fracdiff** did on model 1 in Table 5.3. The second is that dynamic mean estimation in **arfima** should be used with caution: for example, in models 7 and 8, dynamic mean estimation caused the "hiding" of one mode from the `arfima` function. It is also possible we did not do these fits with enough starting points. We note again that in some cases, dynamic mean estimation induces multimodality where there should be none. The third is that approximate maximum likelihood, such as is done by **fracdiff**, can be very misleading. A multimodal surface may not be recognized, and the mode or modes found by such methods may be completely incorrect.

Looking at Table 5.12, we see that, especially for complicated models, the multistart fits can be very slow. This is true for `fracdiffMM` fits as well as `arfima` fits. It is a given that any fit based on **fracdiff** is going to be faster than a fit based on **arfima**. However, we note that the `cpus` option in the `arfima` command will mitigate this. We also note that unfortunately, due to the curse of dimensionality, that usually more complex model require higher `numeach` options to find all modes of a loglikelihood surface.

Looking at Figure 5.1, we see that while `arfima` optimized with $\bar{w}_n$ subtracted first may do

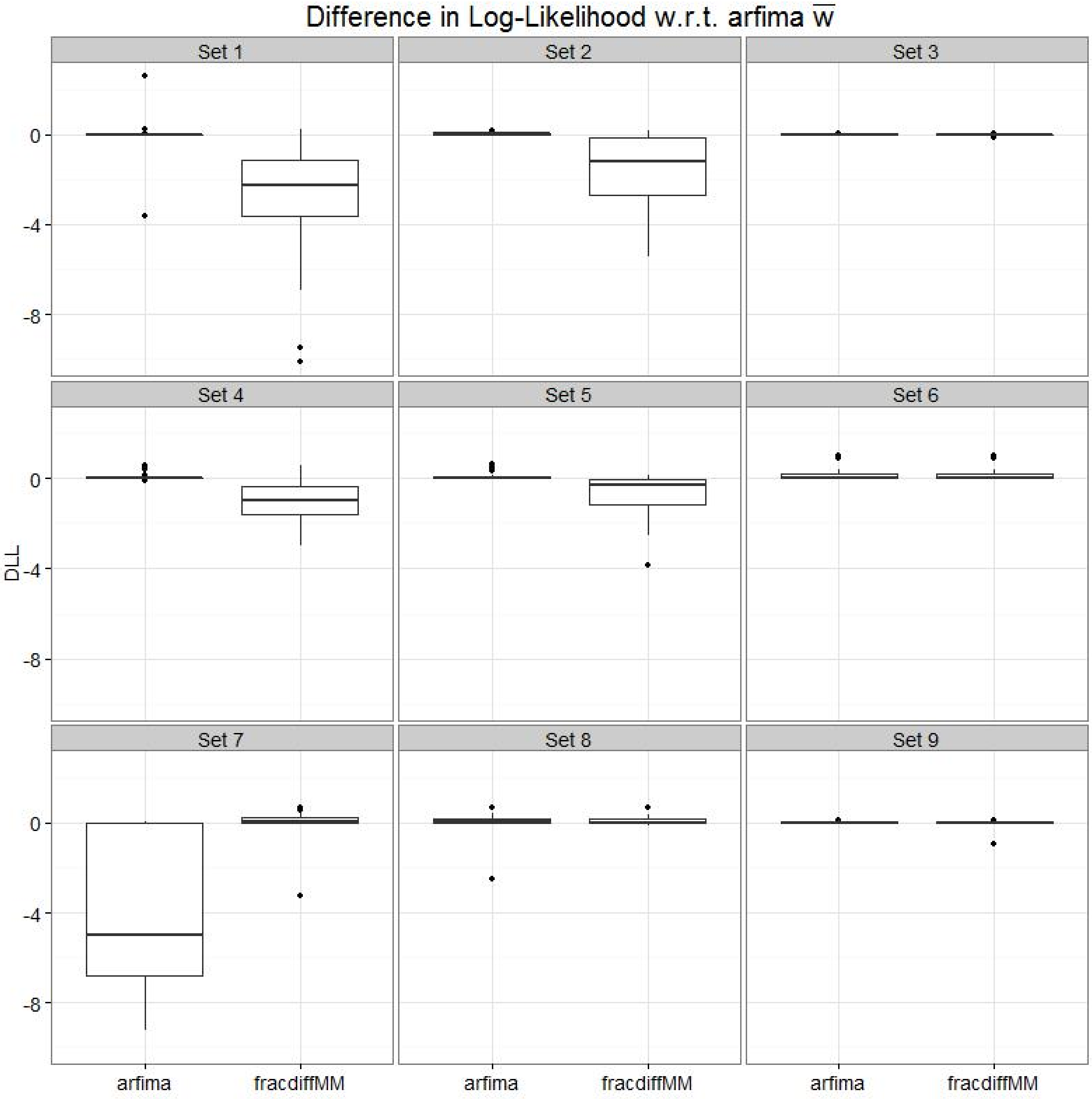


Figure 5.1: The differences in log-likelihoods with respect to `arfima` $\bar{w}$; this figure shows that for the most part **arfima** with sample mean subtracted does better in terms of exact likelihood than **fracdiff** and sometime itself with dynamic mean estimation. Note that either one or the other does as well or better than **fracdiff**.

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9
`arfima`$_{(3,4)}$, $\hat{\mu}$	64.31	1.1e+04	744.2	1.6e+05	1.4e+05	770.1	63.51	744.5	710.5
`arfima`$_{(3,4)}$, $\bar{w}$	46.43	7574	502.9	1e+05	9.5e+04	452.8	42.98	480.9	480.5
`arfima`$_{(3,4)}$, 0	52.07	7240	482.8	1e+05	9.6e+04	451.8	43.99	480.4	481.4
`fracdiffMM`$_{(6,1)}$	0.612	892.4	24.57	3.5e+04	3.5e+04	16.66	0.572	20.54	19.94

Table 5.12: A Timings Table for the Different Fitting Methods by Model. Note that **fracdiff** does dominate

much better than the mean being dynamically estimated (although once again, we may need more starting points), both of the `arfima` fits do better than the `fracdiffMM` fits on the whole. Note that we have also looked at regular `fracdiff` fits, which tend to do worse than the `fracdiffMM` fits in terms of log-likelihood.

5.7 Other Examples of Using arfima

We leave comparisons of **fracdiff** and **arfima** now, and look at other things that the **arfima** package can do. To give an example of everything would be superfluous, so we limit ourselves to three examples. We will look to the TACVF plot, Series J from Box et al. [2008b], and a prediction example. We note we could extract residuals and regression residuals from the Series J example; however, we will keep to simple examples.

5.7.1 Looking at Plots of the TACVF

As was mentioned in §5.5.4, a TACVF or TACF plot can show different things about a fit. Suppose, for example, we have a data set called M. We would like to fit it with an ARFI model.

```
> M <- as.ts(read.csv('M.csv', header = FALSE))
> fitM <- arfima(M, order = c(1, 0, 0), numeach = c(4, 3), dmean = FALSE, quiet = TRUE)
> fitM


Number of modes: 2

Call:
arfima(z = M, order = c(1, 0, 0), numeach = c(4, 3), dmean = FALSE, quiet = TRUE)

Coefficients for fits:
         Coef.1:     SE.1:       Coef.2:     SE.2:
phi(1)    0.932142    0.0656451   0.250765    0.21206
d.f      -0.639208    0.14273     0.0890651   0.173688
zbar      0.0045379               0.0045379
logl      5.42952                 5.20584
```

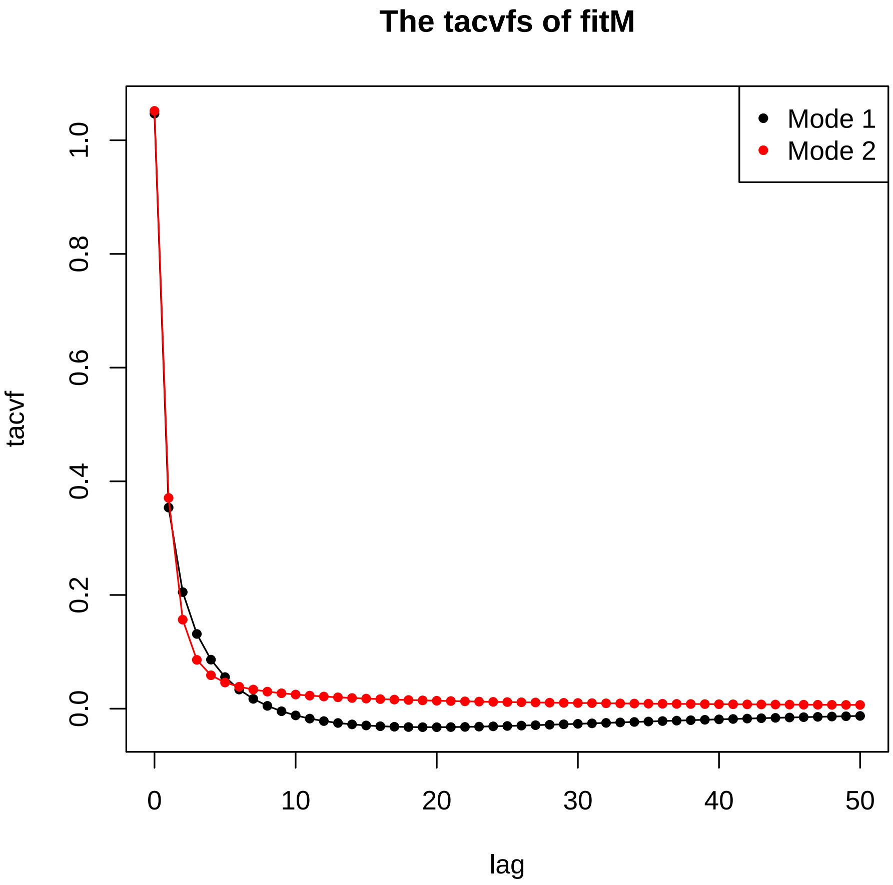


Figure 5.2: The TACVF plot of the toy data set M, fit as ARFI, where one mode (mode 1) is anti-persistent with $\phi \simeq 0.93$ and $d_f \simeq -0.64$, and the other (mode 2) is persistent with $\phi \simeq 0.25$ and $d_f \simeq 0.09$

```
sigma^2   0.908822                0.917966
Starred fits are close to invertibility/stationarity boundaries
```

The mode with the higher log-likelihood is anti-persistent, while the mode with the higher log-likelihood is persistent. This will be explained in Chapter 6: however, lacking that knowledge, we may want to see what is going on.

```
> plot(tacf(fitM), maxlag = 50)
```

Since the TACVFs in Figure 5.2 do not look very dissimilar, it does not look like the modes are incorrect. This will be discussed in more detail in Chapter 6.

Suppose also that there were data called N that we thought might be bimodal under ARFI. We check with

```
> N <- as.ts(read.csv('N.csv', header = FALSE))
> fitN <- arfima(N, order = c(1, 0, 0), dmean = FALSE, quiet = TRUE)
> fitN

Number of modes: 2

Call:
arfima(z = N, order = c(1, 0, 0), dmean = FALSE, quiet = TRUE)

Coefficients for fits:
         Coef.1:    SE.1:        Coef.2*:   SE.2*:
phi(1)    0.982523   0.00704204   0.924918   0.00782083
d.f       0.44569    0.0286124    0.499981   6.32458e-07
zbar     -185.181                -185.181
logl      4.49635                -22.028
sigma^2   0.983474                1.0311
Starred fits are close to invertibility/stationarity boundaries
```

The second mode is very close to the boundary for d_f. Therefore, we investigate the TACVF plot.

```
> plot(tacvf(fitN), maxlag = 50)
```

In Figure 5.3 we can see for certain that mode 2 is spurious. Once again we will address this issue in Chapter 6.

We simulated both M and N as ARFI. M had $n = 100$, $\phi = 0.98$ and $d_f = -0.69$. N had $n = 1000$, $\phi = 0.98$, and $d_f = 0.45$.

5.7.2 Series J

We will look at Series J, one of the data sets in the package, taken from Box et al. [2008b]. It is analysed using transfer functions (also known as dynamic regression), and as such no function in **fracdiff** can fit the two series. We will compare our results to those found in Box et al. [2008b]. We note that the **arfima** package does allow differencing in transfer function modelling, as well as in ordinary regression; however, we will not pursue such notions here.

First, however, we must present the model. A transfer function is a model of the form

$$Y_t = \sum_{i=1}^{k} \delta_i^{-1}(B)\omega_i(B)B^{b_i}X_{i,t} \tag{5.15}$$

where B is the backshift operator, $\delta_i(z) = 1 - \delta_{i,1}z - \ldots - \delta_{i,r_i}z^{r_i}$, $\omega_i(z) = \omega_{i,0} - \omega_{i,1}z - \ldots - \omega_{i,s_i}z^{s_i}$, and $b_i \in \mathbb{Z}_{\geq 0}$. We note that this is somewhat similar to regular regression. There is no error term, but if $\delta_i(z) = 1$, $\omega_i(z) = \beta_i$, and $b_i = 0$ for all i, we would have something akin to

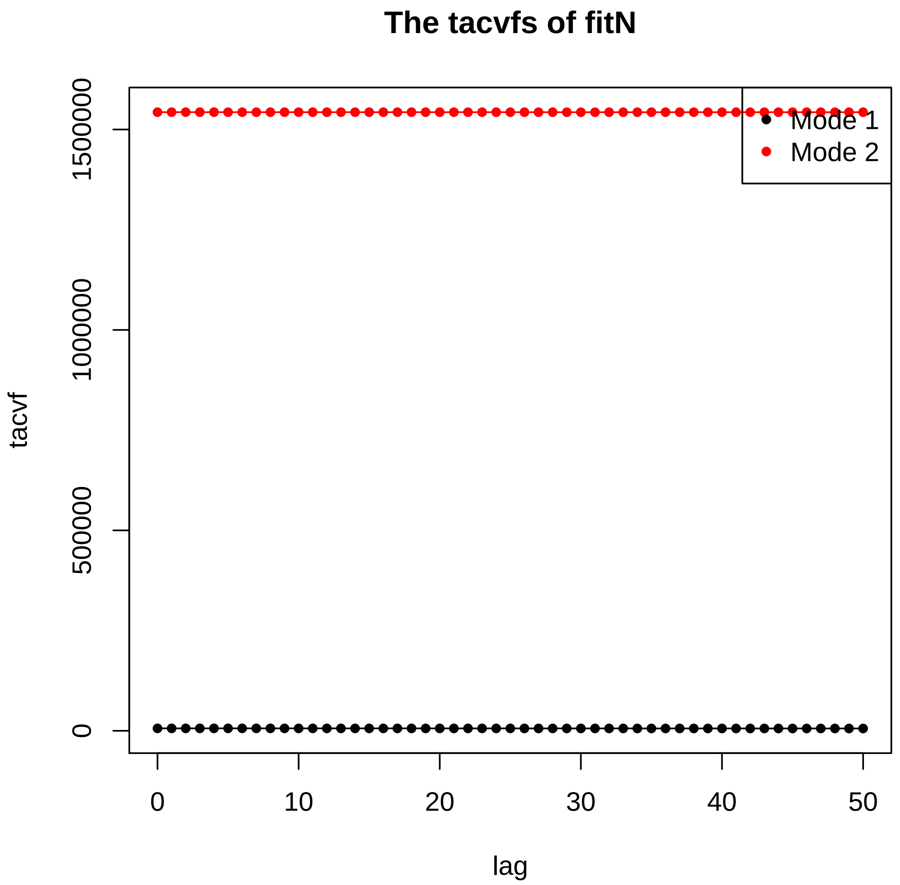


Figure 5.3: The TACVF plot of the toy data set N, fit as ARFI with two very persistent modes: the first having $\phi \simeq 0.98$ and $d_f \simeq 0.45$, and the second having $\phi \simeq 0.92$ and $d_f \simeq 0.5$. It is obvious that mode 2 is spurious, as it hardly decays; we note that this “mode” has the optimizer trapped on the upper boundary for d_f.

a regression problem. Note that the X_is have to be zero-mean, and so we must subtract the sample mean before the fit.

If we let $k = 1$ for ease of notation, we have

$$Y_t = \delta^{-1}(B)\,(\mu + \omega(B)X_t) \tag{5.16}$$

$$\Rightarrow \delta(B)Y_t = \mu + \omega(B)X_t \tag{5.17}$$

looks somewhat like an ARMA equation. We note that this is deterministic, which does not often reflect reality. We allow ARMA-HD noise to be present also, so that

$$Y_t = \delta^{-1}(B)\omega(B)X_t + N_t \tag{5.18}$$

with usually $N_t \sim$ ARMA, although in our package it is possible to have N_t be any type of process it can estimate. If w_t is white noise or hyperbolic decay, we have that

$$\phi(B)N_t = \theta(B)w_t \tag{5.19}$$

as usual.

Therefore, to evaluate the log-likelihood of a transfer function at a given set of parameters, we have a general procedure to follow: for each t, calculate $\mathcal{Y}_t$:

$$\delta(B)\mathcal{Y}_t = \omega(B)X_t \tag{5.20}$$

and set $N_t = Y_t - \mathcal{Y}_t$. Then we use, for example, the Durbin-Levinson recursion to evaluate the log-likelihood of (5.19). To fit a transfer function, assuming we know the order of the parameters, we pass this process to the optimizer. Note that the mean of the X_ts must be zero, and thus the mean of the N_t is zero by definition.

The $k > 1$ case is a somewhat more complicated, but does follow relatively easily.

Our example follows. Note that the mean is estimated dynamically, and fit an AR(2) model with $r = s = 2$ and $b = 3$.

```
> data(SeriesJ)
> attach(SeriesJ)
> fitTF.a <- arfima(YJ, order= c(2, 0, 0), xreg = XJ, reglist = list(regpar
+  = c(2, 2, 3)), lmodel = "n", quiet = TRUE)

note: transfer functions do not work with dynamic mean:  setting dmean to FALSE

Please note for transfer functions the means of each X variable

must be 0: subtracting mean from each X

> fitTF.a
```

```
Number of modes: 1

Call:
arfima(z = YJ, order = c(2, 0, 0), lmodel = "n", xreg = XJ,

 reglist = list(regpar = c(2, 2, 3)), quiet = TRUE)

Coefficients for fits:
             Coef.1:     SE.1:
phi(1)        1.52835     0.0463078
phi(2)       -0.630086    0.0489888
omega(0).X1  -0.532506    0.199447
omega(1).X1   0.370941    0.140933
omega(2).X1   0.509586    0.0736911
delta(1).X1   0.564777    0.145028
delta(2).X1  -0.0110946   0.148634
zbar          53.5091
logl          424.311
sigma^2       0.056657
phi_p(1)      0.93759
phi_p(2)     -0.630086
Starred fits are close to invertibility/stationarity boundaries
```

We note that the result is fairly comparable to Box et al. [2008b]. Dynamic mean estimation has not been implemented for transfer functions, as it does not make much sense, nor does the iteratively fitted mean for any type of regression for the same reason.

Making reference to Box et al. [2008b], it is suggested to set $r = 1$ since δ_2 = `delta(2).X1` is much smaller than its standard error. This fit is below. Note that the AR parameters in `fitTF.a` and `fitTF.b` are very close.

```
> fitTF.b <- arfima(YJ, order= c(2, 0, 0), xreg = XJ, reglist = list(regpar
+  = c(1, 2, 3)), lmodel = "n", dmean = FALSE, quiet = TRUE)

Please note for transfer functions the means of each X variable must be 0: subtracting mean fr

> fitTF.b

Number of modes: 1

Call:
arfima(z = YJ, order = c(2, 0, 0), dmean = FALSE, lmodel = "n", xreg = XJ, reglist = list(regpa
    2, 3)), quiet = TRUE)

Coefficients for fits:
             Coef.1:     SE.1:
```

```
phi(1)        1.52827     0.0462889
phi(2)       -0.630126    0.0489617
omega(0).X1  -0.531872    0.0388622
omega(1).X1   0.379109    0.0732022
omega(2).X1   0.517518    0.100915
delta(1).X1   0.549398    0.107704
zbar          53.5091
logl          424.308
sigma^2       0.0566584
phi_p(1)      0.937517
phi_p(2)     -0.630126
Starred fits are close to invertibility/stationarity boundaries
```

```
> detach(SeriesJ)
```

When we fit Series J with FD instead of white noise, we find that a multimodal surface is induced. In particular, modes are only found on the boundaries. This points to the need for more experimentation with transfer function data and hyperbolic decay noise.

5.7.3 A Prediction Example

There are some functions in our package only touched on in this chapter. However, we thought it would not be complete without a prediction example. We will illustrate the use of the new algorithm for prediction variances.

We will have a multimodal integrated series as our example, and show the differences between the limiting (standard) error variances and the exact error variances. We will see that the last mode is spurious since it is nearly non- identifiable and close to boundaries, and thus we remove it. The below is the fit.

```
> set.seed(34564)
> sim <- arfima.sim(1000, model = list(phi = 0.95, dfrac = -0.8, theta = 0.4, dint = 1))
> fit <- arfima(sim, order = c(1, 1, 1), numeach = c(3, 3), dmean = FALSE, quiet = TRUE)
> fit
```

```
Number of modes: 3

Call:
arfima(z = sim, order = c(1, 1, 1), numeach = c(3, 3), dmean = FALSE, quiet = TRUE)

Coefficients for fits:
          Coef.1:     SE.1:       Coef.2:     SE.2:       Coef.3*:
phi(1)     0.914579    0.0321209   0.0539853   0.158067   -0.999049
theta(1)   0.60054     0.135189    0.241601    0.191687   -0.998438
d.f       -0.550523    0.173471   -0.0284763   0.0643638  -0.0729256
```

```
zbar      -0.0103921             -0.0103921             -0.0103921
logl      -25.2126               -29.4844               -40.0282
sigma^2    1.05367                1.06394                1.08659
          SE.3*:
phi(1)     6.32457e-07
theta(1)   6.32457e-07
d.f        0.0275752
zbar
logl
sigma^2
Starred fits are close to invertibility/stationarity boundaries
```

```
> fit <- removeMode(fit, 3)
> pred <- predict(fit, n.ahead = 10, seed = 3456)
> pred
```

```
$`Mode 1`
$`Mode 1`$`Forecasts and SDs`
                  1         2         3         4         5
Forecasts   -11.09469 -11.11273 -11.13722 -11.15676 -11.17034
Exact SD      1.02663   1.29185   1.50636   1.69681   1.87007
Limiting SD   1.02648   1.29147   1.50570   1.69582   1.86870
                  6         7         8         9        10
Forecasts   -11.17861 -11.18253 -11.18296 -11.18068 -11.17632
Exact SD      2.02910   2.17565   2.31102   2.43628   2.55234
Limiting SD   2.02731   2.17341   2.30831   2.43306   2.54859

$`Mode 1`$`Bootstrap Replicates`
[1] 1000

$`Mode 1`$`Bootstrap Predictions and Intervals`
                        1         2         3         4         5
Upper 95%          -9.36462  -8.60604  -8.06558  -7.86305  -7.44342
Prediction (Mean) -11.17990 -11.24376 -11.20989 -11.22769 -11.25877
Lower 95%         -13.42202 -13.87088 -14.26490 -14.65213 -15.01275
                        6         7         8        9        10
Upper 95%          -7.06662  -6.65929  -6.40752  -6.3401  -6.20256
Prediction (Mean) -11.23925 -11.23293 -11.23498 -11.2627 -11.20637
Lower 95%         -15.33750 -15.68663 -15.97977 -16.1908 -16.38041


$`Mode 2`
$`Mode 2`$`Forecasts and SDs`
                  1         2         3         4         5
Forecasts   -11.07621 -11.06844 -11.07082 -11.07510 -11.08043
Exact SD      1.03148   1.31063   1.52999   1.71844   1.88608
Limiting SD   1.03147   1.31063   1.52999   1.71844   1.88607
```

```
                     6          7          8          9         10
Forecasts   -11.08651 -11.09316 -11.10025 -11.10769 -11.11542
Exact SD      2.03842   2.17891   2.30988   2.43299   2.54947
Limiting SD   2.03841   2.17890   2.30988   2.43298   2.54946

$`Mode 2`$`Bootstrap Replicates`
[1] 1000

$`Mode 2`$`Bootstrap Predictions and Intervals`
                          1         2        3         4         5
Upper 95%          -9.35358  -8.61712  -8.0893  -7.93631  -7.45889
Prediction (Mean) -11.18926 -11.26570 -11.2422 -11.26738 -11.31004
Lower 95%         -13.34294 -13.95533 -14.3785 -14.64938 -15.06371
                          6         7         8         9        10
Upper 95%          -6.96989  -6.70237  -6.64643  -6.67434  -6.51856
Prediction (Mean) -11.29711 -11.29943 -11.30841 -11.34946 -11.29920
Lower 95%         -15.26453 -15.64839 -15.91643 -16.03681 -16.31518
```

We see that in this case there is very little difference between exact and approximate standard deviations in this case. We look at the plots of the different predictions below in Figure 5.4.

```
> plot(pred)
```

5.8 On Multimodality in ARFIMA Models

We have noted that in ARFIMA models, there appears to be multimodality. We are certain of this in the ARFI and FIMA cases: we have visualized these cases in ***Mathematica***, one of the topics covered in Chapter 6. We are also quite certain of this in the case where the ARMA structure is more complex: since we are working with exact maximum likelihood, and the multimodal structure of the log-likelihood surface is apparent in the output of our package, we can be sure that multiple modes exist. We have seen this with white noise driving the processes as well: that is, multimodality seems to occur fairly naturally in complex enough ARMA models.

For the purposes of this section, we have that we subtract the sample mean off of the series before we observe the nature of the multimodal surface. As we have displayed in this chapter and will discuss more in Chapter 6, the dynamic estimation of the mean can induce or mask modes. In said chapter, we will also note that not subtracting the mean can mask modes as well, and induce modes on the boundaries.

When we delve deeper into the multimodal structure of an ARFIMA or ARMA model's log-likelihood, we have noticed that the MA(∞) coefficients for each mode are usually similar to each other. We believe that this is the most apparent cause for multimodality: since the parameters are alike at each mode, necessarily we have that the log-likelihood will act in a similar

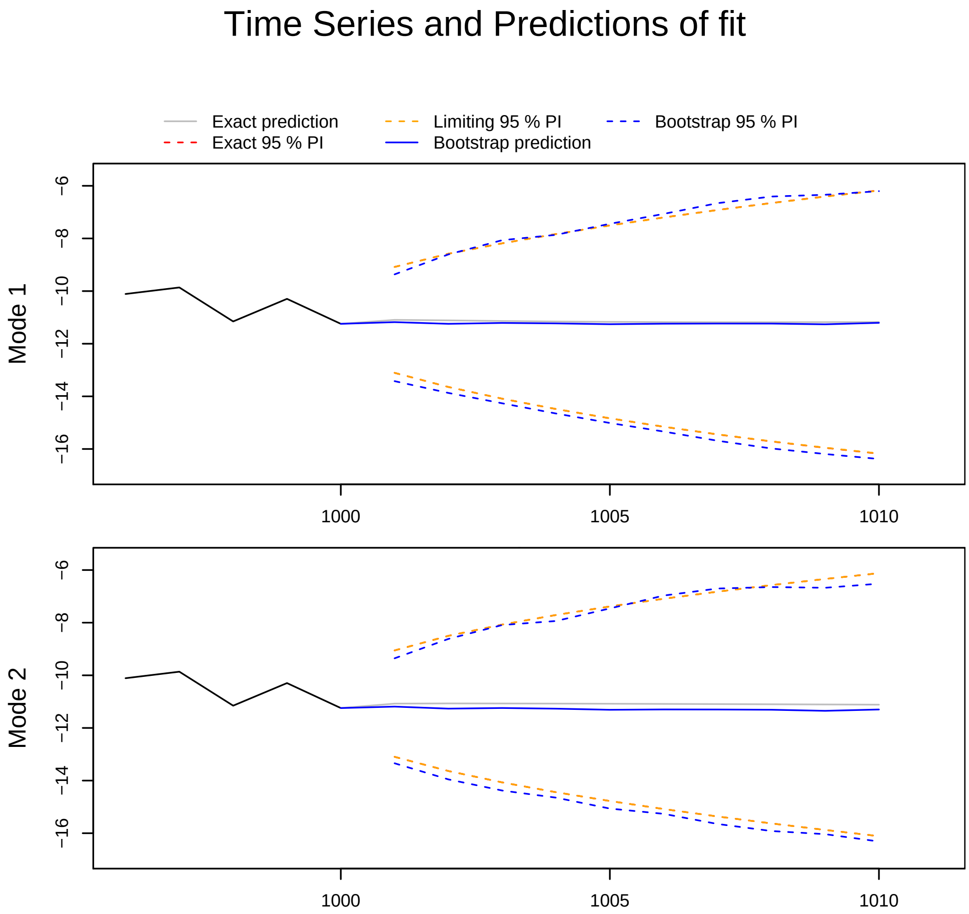


Figure 5.4: The plots of the predictions associated with `fit`, a bimodal log-likelihood; this figure shows the modes for this particular fit are similar enough to give similar predictions

way and rise to a maximum at these points. Of course, since the nature of the neighbourhood at each mode can be quite different, there will be differences.

However, we have also noticed multimodal likelihood surfaces for ARMA models driven by the other types of HD noise. Since these models cannot be written in operator notation and thus cannot have MA(∞) expansions, we note that there is an another underlying cause for multimodal log-likelihood surfaces. We hypothesize in Chapter 6 that this is similarity between TACVFs of modes on the surface.

5.9 Conclusions and Future Work

We have demonstrated the efficacy of our **arfima** package for ***R***. In particular, we have demonstrated that exact maximum likelihood clearly outperforms approximate maximum likelihood as is done by **fracdiff**. Not only are our exact methods easier to extend into seasonal data, transfer functions, and time series regression to name but a few, we have that the likelihood surface is very badly approximated in some ways by **fracdiff**.

For future work, we intend on extending our package to taking into account missing data via the EM algorithm, as well as looking more deeply into transfer functions and long memory, likely through simulation. We would possibly like to implement basic visualization in ***R***. We have a ***Mathematica*** package that does do visualizations, which will be presented along with our suppositions on multimodality, in Chapter 6.

Chapter 6

Visualizations and Multimodality

6.1 Introduction

In Chapter 5, we took the existence of multiple modes on various log-likelihood surfaces as a matter of course. In this chapter, we discuss and present the visualization of log-likelihood surfaces of simple models, to show true multimodal surfaces, as well as discussing our suppositions on the causes of multimodality. There is a companion ***Mathematica*** package, called **simpleVis**, to this chapter for visualizations.

The models we will consider for visualizations have either an AR or MA (i.e. short memory) component, as well as a HD component. When we talk about multimodality, we will discuss these, as well as a little on more general cases.

It must be made clear that when we talk about mode induction on boundaries later in this chapter, this is partly a problem with the BFGS optimizer. The Nelder-Mead optimizer in ***R*** does not get caught on the boundaries as often, since it is a simplex-based method. However, we were unsatisfied with the Nelder-Mead optimizer in most cases, since not only did it miss modes verified by visualizations, it induced modes in the middle of the surface that were not there.

6.1.1 A Discussion of Mean Estimation and Visualizations

We restrict ourselves to either letting the surface be defined by the true mean, which we can talk about if we simulate, or by the mean of the series. We note that in some cases, the surfaces with the true mean and the mean of the series can be quite different. For visualizations, we restrict ourselves to no dynamic mean estimation.

We restrict ourselves in this way for multiple reasons. The first is that, as mentioned in §5.5.1, the likelihood structure of the series changes when we subtract a mean. This is gives rise to two problems. The first is technical: we would have to visualize two surfaces, for example, if we had two modes. The second is more serious: since we dynamically estimate the mean, we have that technically we should be subtracting a different mean for every single point on the surface. We can see this by noting that for any given set of parameter values that generate the likelihood, we can optimize to find the best mean for that set of parameter values. This is, in fact what we do in the `fracdiffMM` script since `fracdiff` does not report a mean. We discuss this in Section 5.6.1. However, the change in mean seems to have no difference on the parameters of a fit involving the **fracdiff** package, since it is filtered out. It remains a problem for the **arfima** package. We discuss this in more detail in Chapter 5. As we discuss in said chapter and later in this one, the dynamic estimation of the mean can induce or mask multimodal structure on a likelihood surface. We have not investigated the iteratively fitted mean, although we suspect that such estimation has similar problems to dynamic mean estimation.

6.2 Visualizing a Log-Likelihood Surface

There are several reasons we keep the visualizations to one short memory component and one HD component. The first is that the log-likelihood surfaces with this type of model, combined with the correct (generating) parameters or data structure, can give rise to a multimodal surface, usually bimodal. We have seen this in simulations and with real data. While more complex, or at least less parsimonious, model surfaces also seem to exhibit multimodalilty, they are harder to visualize. We believe that the only two parameter (excluding the mean) model that is capable of a multimodal likelihood are models of this sort. This will be discussed in §6.3.

6.2.1 Technical Considerations

There are several technical considerations to viewing the log-likelihood surface of an ARMA-HD model fit to data. The first we will discuss is computation time. As the parameter space becomes larger and invariably more complex, the computation of any grid or surface will take exponentially more time. This can be slightly mitigated by computing maximum and minimum value in each dimension and only plotting in the hyperrectangle containing all of the mode's points for a given surface. If there were enough modes, we could compute the convex hull containing all mode points: this space would be smaller still. Another option would be to only compute the surface local to the modes. In cases of higher dimensionality, this is likely the best approach.

In a larger space, if we chose the last option, we would also have to compute a lattice of points around each mode in all dimensions. For viewing purposes, we would have to vary one or two parameters (for a 2- or 3D plot respectively) and hold all of the others fixed for each mode. This would be relatively cumbersome. Also, due to the nature of the PACF space as described in §5.5.2, the surface with untransformed coefficients would be relatively messy to view. It is true we could view the modes in the PACF space: however, we would have to keep the transformation in mind.

6.2.2 Extracting the Fitted Model from *R*

We have written the ***R*** script `extractFits` to extract the fitted values and the series for relevant **arfima** fits. This script checks that the series is the same, as well as extracting information about which mean was used and what HD-type parameter was estimated.

6.2.3 On the simpleVis Package

We introduce the **simpleVis** package. It takes output from the **arfima** package with one common short memory parameter and any or all of the HD parameters. We do this so we can change the model type in mid-view to view where the fitted model would be in the other parameter types with the asymptotic relationships.

Note that the **simpleVis** package still needs some work. While it is fully functional, the interface is still in its early stages.

We will now briefly outline the functions of the package.

- `Importer` - Imports information from the file created by the `extractFits` script. Also outputs information on the imported fits, such as the means, the number of modes associated with each fit, the range of the parameters (ϕ or θ and the HD parameters in terms of α), and the index of each fit.

- `SelectFit` - Allows the user to select the fit used as the underlying model. That is, which type of HD process is used, which mean is subtracted, and thus which surface will be plotted.

- `AddFit` - Allows the user to add fit information from the other fits to see where the modes of these additional fits would be on the currently selected surface. Will output the colour used to plot the points of this fit.

- `ShowLL` - Shows the log-likelihood surface with the fits selected.

- `ChangeOffsets` - Allows the user to change the offsets, that is the value subtracted from the lowest mode's log-likelihood value and the one added to the highest mode's value to ease the viewing of the plot

- `Replot` - Completely replots the log-likelihood with all the fits added and the current offsets. Warning: this operation takes more time than the `ShowLL` function and thus it is recommended that the user make sure they are satisfied with the information currently in the fit.

6.3 Suppositions on Multimodal Behaviour

We must stress that the ideas in this section are hypotheses only: they are suppositions based on tests we have run and empirical evidence we have accumulated. While there is some theoretical basis for our beliefs, said basis is not complete. More research needs to be done to understand what is going on.

We will restrict ourselves to the single short memory parameter driven by HD noise for most of this section. As we have mentioned, these seem to be the most parsimonious models wherein a multimodal log-likelihood surface occurs. With large enough n these surfaces tend to be bimodal only; also, with large enough n, we have noticed that one mode has anti-persistent parameters, while the other has persistent parameters. This is not always the case: with small enough n, we can have two anti-persistent modes. Since we are simulating, however, we can keep the same seed and model, but increase n: as we do this, invariably the less anti-persistent mode (that is, with the smaller α) becomes persistent. Therefore, for the rest of this section, we will have one mode as persistent and the other as anti-persistent.

However, we have made mention of the fact previously that the TACVFs of the two modes are often quite similar. This is most apparent in the MA-HD case: we will look at a FIMA model below. The TACVFs up to lag 50 are plotted in Figure 6.1.

```
> library(arfima)
> set.seed(45345)
> seed <- sample(1:3468356, 1)
> n <- 1000
> set.seed(seed)
> sim1 <- arfima.sim(n, model = list(theta = 0.95, dfrac = 0.42))
> fit1 <- arfima(sim1, order= c(0, 0, 1), dmean = FALSE, quiet = TRUE)
> fit1

Number of modes: 2

Call:
arfima(z = sim1, order = c(0, 0, 1), dmean = FALSE, quiet = TRUE)

Coefficients for fits:
          Coef.1:     SE.1:       Coef.2:     SE.2:
theta(1)   0.958185    0.0119221   0.256312      0.0620499
d.f        0.444102    0.032259   -0.254657    0.0445759
zbar       0.0170442               0.0170442
logl      -5.92563                -12.9146
sigma^2    1.0116                  1.02747
Starred fits are close to invertibility/stationarity boundaries

> plot(tacvf(fit1), maxlag = 50)
```

We see that both of the TACVFs in Figure 6.1 look anti-persistent. Meanwhile, if we look at

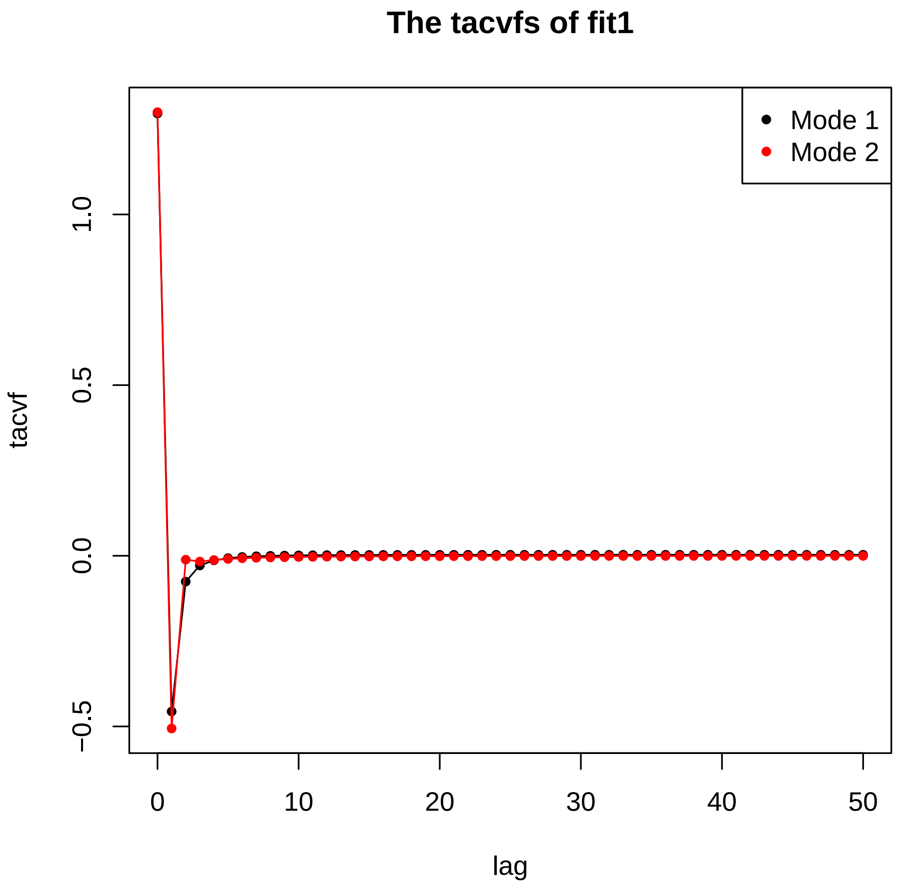


Figure 6.1: The TACVF plot of `fit1`, a FIMA model fitted to simulated data, with two modes, one of which has persistent parameters and one of which has anti-persistent parameters, although the TACVFs look very similar. Mode 1 is the persistent mode, with $\theta \simeq 0.96$ and $d_f \simeq 0.44$ while mode 2 has $\theta \simeq 0.26$ and $d_f \simeq -0.25$.

Chapter 5, in particular at the Figure 5.2, we notice that the two TACVFs look similar also. This brings us to our supposition on the existence of bimodal surfaces for this type of models: that of apparent persistence or anti-persistence.

6.3.1 On Apparent Persistence or Anti-persistence in Simple Models

We clarify what we mean by apparent persistence and anti-persistence. We know that for any truly persistent process, which includes ones induced by the fitted values of a persistent mode, the sum of the autocovariances will be infinite. Likewise, for any anti-persistent process, we will have said sum equalling zero: the sum of the lag one to infinity autocovariances will be equal to negative one half of the lag zero autocovariance. For any finite series and thus for any corresponding TACVF of the correct length, any persistent process will never have its autocovariance sum go to infinity as long as the process is stationary. However, there are more than one way for said sum to become large. In a similar manner, an anti-persistent mode's parameters will likely not give rise to an exact zero a for TACVF of the correct length. In this case as well, there are more than one way for a TACVF to be small.

Since we restrict ourselves to a single short memory parameter for now, we note that we can write a closed form expression for each of the models driven by any HD noise. The MA case is simple. With $\boldsymbol{y} \sim \textsc{ma}(1)$ and $\boldsymbol{x} \sim \textsc{hd}$, $\boldsymbol{w} \sim \textsc{ma-hd}$ has, $\forall k \in \mathbb{Z}$

$$\begin{aligned}\gamma_w(k) &= \sum_{i=-1}^{1} \gamma_y(i)\gamma_x(k-i) && (6.1)\\ &= (1+\theta^2)\gamma_x(k) - \theta\left(\gamma_x(k-1) + \gamma_x(k+1)\right) && (6.2)\end{aligned}$$

We note that the autocovariance function of the AR-HD process is much more difficult to derive. With $\boldsymbol{y} \sim \textsc{ar}(1)$ and $\boldsymbol{x} \sim \textsc{hd}$, $\boldsymbol{w} \sim \textsc{ar-hd}$ has, $\forall k \in \mathbb{Z}$

$$\begin{aligned}\gamma_w(k) &= \sum_{i=-\infty}^{\infty} \gamma_y(i)\gamma_x(k-i) && (6.3)\\ &= \sum_{i=-\infty}^{\infty} \frac{\phi^{|i|}}{1-\phi^2}\gamma_x(k-i) && (6.4)\\ &= \frac{1}{(1-\phi)(1+\phi)} \sum_{i=-\infty}^{\infty} \phi^{|i|}\gamma_x(k-i) && (6.5)\end{aligned}$$

which seems intractable. However, we examine Box et al. [2008b], pages 430-431, to derive the expression of the autocorrelation function of an ARFI model:

$$\rho_w(k) = \frac{\rho_x(k)}{1-\phi} \frac{{}_2F_1\left(1, d_f - k; 1 - k - d_f; \phi\right) + {}_2F_1\left(1, k + d_f; 1 + k - d_f; \phi\right) - 1}{{}_2F_1\left(1, d_f + 1; 1 - d_f; \phi\right)} \qquad (6.6)$$

where now $x \sim \textsc{fd}$ only. The ${}_2F_1$ denotes the hypergeometric function, which is defined in

Chapter 2. We have the lag 0 autocovariance being

$$\gamma_w(0) = \frac{\sigma_a^2 \gamma_w (1-2d_f)}{\gamma_w (1-d_f)^2} \frac{{}_2F_1\left(1+d_f, 1; 1-d_f; \phi\right)}{1+\phi} \tag{6.7}$$

We note that we can get other expressions using ***Mathematica***: specifically, for each lag for a PLA or FGN process. However, there does not seem to be an explicit closed form for any given lag for either process type.

We note that while for each n there are two sets of ranges for the parameters for a bimodal surface to occur: one will always be with the short memory parameter close to 1, while the HD parameter can be in quite a large area, depending on which type of noise drives the process. We know the autocovariance structures of the HD parameters can be quite different, although the fitted value autocovariance structures (when mixed with short memory) often appear similar. The other mode's parameters, however, can fluctuate wildly.

6.3.1.1 Apparent Anti-persistence and the MA-Based Models

As we have stated, given sufficient n, there is always an anti-persistent mode and a persistent mode. In the MA-based models with a bimodal log-likelihood surface, the persistent mode looks anti-persistent, as we saw in Figure 6.1. There is a very specific reason for this: for the n given, θ is close enough to 1 for the sum of the TACVFs to be small. Recalling (6.2), we have that

$$\begin{aligned}
\lim_{\theta\to 1} \sum_{k=1}^{\infty} \gamma_w(k) &= \sum_{k=1}^{\infty} \left(2\gamma_x(k) - \gamma_x(k+1) - \gamma_x(k-1)\right) && (6.8)\\
&= 2\gamma_x(1) + 2\sum_{k=1}^{\infty} \gamma_w(k+1) - \gamma_x(0) - \gamma_x(1) - \sum_{k=1}^{\infty} \gamma_w(k+1) - \sum_{k=1}^{\infty} \gamma_w(k+1) && (6.9)\\
&= \gamma_x(1) - \gamma_x(0) && (6.10)\\
&= -\frac{1}{2}\left(2\gamma_x(0) - 2\gamma_x(1)\right) && (6.11)\\
&= -\frac{1}{2}\left(\lim_{\theta\to 1}\left((1+\theta^2)\gamma_x(0) - 2\theta\gamma_x(1)\right)\right) && (6.12)\\
&= -\frac{1}{2}\left(\lim_{\theta\to 1}\gamma_w(0)\right) && (6.13)\\
& && (6.14)
\end{aligned}$$

and as such, when $\theta = 1$ and regardless of α (as long as $0 < \alpha < 3$), we have that the process is anti-persistent. The model is on the invertibility boundary, however. Still, if θ were close to 1 and we were to limit the sum to lags 0 to $n-1$, we would have an autocovariance function that sums to something quite small if n is not too large. As n increases, the sum would get larger. However, this can be mitigated by having θ closer to 1. As $n \to \infty$, we would require $\theta \to 1$ for the log-likelihood surface to remain bimodal. If $n \to \infty$ and θ and the HD parameter were to remain fixed, we would have that the anti-persistent mode would become less and prominent, until it dropped off of the surface. This will occur since the autocovariance structures become less similar, and the anti-persistent mode is usually our non-generative mode for MA-HD models. However, due to the fragility of the anti-persistent mode, which we will see in Section 6.3.4, we have known it to be the one to disappear in the few cases we have seen where said mode was the generative one.

6.3.1.2 Apparent Persistence and the AR-Based Models

Looking at (6.5) and keeping §6.3.1.1 in mind, we have that we should want to have apparent persistence by having an anti-persistent α in an AR-HD model and sending ϕ to ± 1. As it turns out we would in fact only succeed were we to send ϕ to 1. We note this is likely due to the complex nature of the structure of any AR-HD process autocovariance. We know that sending ϕ to -1 will not create a persistent mode due to experimentation and the spectral density of the process.

The spectral density of an AR-HD model model has the form, with λ close to 0, as

$$f_w(\lambda) \sim \left|1 - \phi e^{-i\lambda}\right|^{-2} \lambda^{\alpha-1} \tag{6.15}$$

where we note that if we take the limit as ϕ approaches 1 before taking $\lambda \to 0$, for $0 < \alpha < 3$, we have that this will tend to infinity. As we have already noted in Chapter 3, in particular (3.88), we have that there is an implicit inverse relationship between λ and n. We do note that it can be different than the one implied in (3.88): however, it is enough for us to say that $n_\lambda \to \infty$ as $\lambda \to 0$ for $n_\lambda = 1/h(\lambda)$, where $h(0) = 0$, $h(t) > 0$ for $t > 0$ and $h(t)$ is monotone increasing for $t \in [0, \pi)$. Thus the spectral density at some λ of some fitted value will be close to the sum of the TACVFs at the same fitted value and some n. As n increases, we have that the fitted value of ϕ will have to be closer to 1 for the anti-persistent mode to appear persistent. If $n \to \infty$ and ϕ were to remain unchanged, we will have that the persistent mode becomes less prominent, until it drops off the surface. Once again, the autocovariance structure become dissimilar, and since the persistent mode is the non-generative mode for most AR-HD models we have tested, we know that it will disappear as the fit becomes more accurate. Oddly enough, in the relatively few cases where the generative mode is the persistent one in our experiments, some of the time it is still the persistent mode that drops off the surface. Loss of modes in this type of model will usually happen much more slowly than the disappearance of the anti-persistent mode in the above.

6.3.2 On More Complex Models

We note that the more complex a model is, the more modes it is likely to have. Looking at the spectral density of an ARMA-HD model for λ close to 0 (6.16) (which we recall from Chapter 3),

$$f_w(\lambda) \sim \frac{\left|\theta\left(e^{-i\lambda}\right)\right|^2}{\left|\phi\left(e^{-i\lambda}\right)\right|^2} \lambda^{\alpha-1} \tag{6.16}$$

we have that there is likely to be apparent anti-persistence when $\theta(1) \to 0$ and apparent persistence when $\phi(1) \to 0$. Indeed, there are likely to be more modes, depending on the complexity of the model.

For ARMA models, multimodal log-likelihood surfaces also occur. However, since there is no longer anti-persistence or persistence, the number of modes likely changes less drastically with n, although we have not explored this.

6.3.3 Dynamic Mean Estimation and Modes on the Boundaries

We note that even a small change in mean can cause modes to be hidden, especially an anti-persisent mode in an MA-HD model. For example, suppose we subtracted the negative of the mean whose TACVFs are shown in Figure 6.1. Then the fitted surface would only have one mode, as below:

```
> fit1a <- arfima(sim1, order= c(0, 0, 1), dmean = -0.170442, numeach = c(4, 3), quiet = TRUE)
> fit1a


Number of modes: 1

Call:
arfima(z = sim1, order = c(0, 0, 1), numeach = c(4, 3), dmean = -0.170442, quiet = TRUE)

Coefficients for fits:
          Coef.1*:   SE.1*:
theta(1)   0.96615    0.00887356
d.f        0.492493   0.0108477
Set mean  -0.170442
logl      -8.27005
sigma^2    1.01441
Starred fits are close to invertibility/stationarity boundaries
```

Just as subtracting a constant mean can hide a mode, dynamically estimating one can hide a mode. This is logical, in that if the optimizer is started in the wrong place, we can have the same effect. Since the optimizer will change the mean, it is possible that as far as the other parameters are concerned, the wrong direction is attempted: changing the value that is subtracted will usually make the other parameters fluctuate, especially making the HD parameter more persistent, as we will see below in Figure 6.2. As the mean eventually changes, we could find that we have missed where a mode should be. We have seen this only occasionally, however. Since there are multiple starts in the **arfima** package, unless there are very few starting points, most often any mode found by a set mean of the mean of the series will be found by the dynamic mean.

On the other end of the spectrum to losing a mode is the possibility that one or modes may be induced. As the optimizer goes close to the boundary, the mean moves around and can trap the optimizer at the boundary. This problem can occur when means are fixed as well, but not nearly as much as when the mean is estimated dynamically. We believe this occurs only on boundaries in which the TACVF is persistent, and all of our tests confirm this, as well as the theory we will present in §6.3.4.

6.3.3.1 The Push To Persistence

We note that in all of our experiments, every time a mean has been taken to be different than the sample mean, the farther the mean goes from the sample mean, the more persistent or apparently persistent a mode will become. In the MA-based models whose generative parameters give two modes will almost invariably have their parameters pushed towards the middle of the square defined by the MA parameter and the HD parameter, and modes will be lost. The exact opposite happens with AR-based models: every starting point is pushed towards a boundary. Modes may be lost in this case, but more often, modes are induced. We will give what seems to be the reason for this in §6.3.4.

This push to persistence is seen clearly in Figure 6.2.

For Figure 6.2, $\boldsymbol{e} \sim \text{NID}(0, I_{1000})$ was generated, after which we set $a_t = e_t - \bar{e}$ for all t. Then values of from -15 to 15 with increments of 0.1 were subtracted off of $\boldsymbol{e}$, an FD model was fit to the data, and the value of d_f recorded. The plot is the mean subtracted off by the value of d_f obtained. This illustrates nicely how the fitted values become more persistent with an increasing absolute value of the mean.

What we believe happens is, especially when the mean changes dynamically, that there will be no place for the optimizer to go but more and more persistent values. The fit gets more persistent and the log-likelihood can get larger when the series is not fixed. This can also often lead to vastly different estimates of the mean.

Even when the series is centered, the likelihood may go up if the fitted values are more persistent. The reason this occurs in our package is that when the optimizer gets to the stationarity and/or invertibility boundaries, the likelihood drops off sharply as there is a large negative penalty for meeting those boundaries. Then the optimizer literally gets stuck, as in the above. This rarely happens with no dynamic mean estimation, although if there are a lot of starting points, it may occur as the starting point for one or more optimizations may have nowhere else to go. We have found this type of optimization penalty is usually quite good: much better than, say, constrained optimization in the PACF space of §5.5.2. As is mentioned in §5.5.4, we can usually identify a spurious mode by looking at the TACF plot, although this does not always seem to be the case.

Estimation of anti-persistent modes may change with a small change in mean, as below.

```
> set.seed(234534)
> seed <- sample(6:22452, 1)
> set.seed(seed)
> sim <- arfima.sim(1000, model = list(theta = 0.98, dfrac = 0.1))
> arfima(sim, order = c(0, 0, 1), numeach = c(4, 3), quiet = TRUE)
```

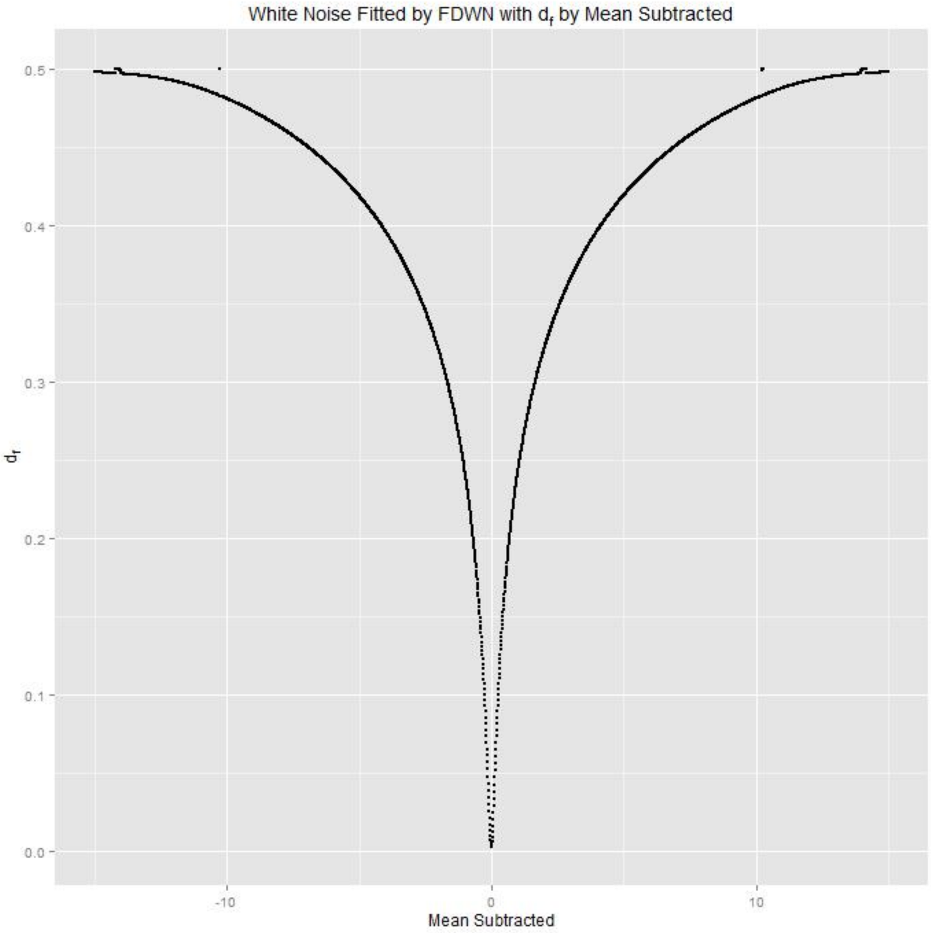


Figure 6.2: White noise modelled as FD with different means; $e_t \sim \text{NID}(0,1)$ for $t = 1,\ldots,1000$ was generated as e, and had $\boldsymbol{a} = \boldsymbol{e} - \bar{e}$ to have a mean of exactly zero up to machine epsilon. For $m_{i=1}^{301} = (-15, -14.9, \ldots, 14.9, 15)$, $\boldsymbol{e} - m[i]$ was fit as FD with the "true" mean set to zero, and $d_f[i]$ was recorded. The plot is m on the horizontal axis and d_f on the vertical axis.

```
Number of modes: 2

Call:
arfima(z = sim, order = c(0, 0, 1), numeach = c(4, 3), quiet = TRUE)

Coefficients for fits:
             Coef.1:       SE.1:        Coef.2:       SE.2:
theta(1)      0.976266      0.00814961   0.158567      0.0614347
d.f           0.0944361     0.0308876   -0.733791      0.0444421
Fitted mean  -0.000103622   0.00141905  -0.000559015   0.000368124
logl          26.8342                    23.0737
sigma^2       0.94734                    0.952867
Starred fits are close to invertibility/stationarity boundaries


> arfima(sim, order = c(0, 0, 1), dmean = FALSE, numeach = c(4, 3), quiet=TRUE)

Number of modes: 2

Call:
arfima(z = sim, order = c(0, 0, 1), numeach = c(4, 3), dmean = FALSE, quiet = TRUE)

Coefficients for fits:
          Coef.1:      SE.1:        Coef.2:      SE.2:
theta(1)   0.973299     0.00848568   0.328282     0.070616
d.f        0.0948949    0.0312415   -0.570908     0.0499456
zbar       0.00176024                0.00176024
logl       26.0864                   13.991
sigma^2    0.948849                  0.971471
Starred fits are close to invertibility/stationarity boundaries
```

The above output from the **arfima** shows how the anti-persistent mode may change with different means as compared to a persistent one. The persistent mode had a change in mean comparable to the anti-persistent mean between the dynamic mean and the sample mean, while its fitted values for θ and d_f barely changed. The anti-persistent mode had a large change in its fitted values. Note that we have seen even more extreme examples of this, such as the loss of the anti-persistent mode. The persistent mode always remains much closer to where it was with the change in mean.

For an anti-persitent mode or process, the dynamically fitted mean may be closer to the true mean than the sample mean. We have noted that it is the anti-persistent modes that are sensitive to these changes. See the below.

```
> set.seed(4567)
> seed <- sample(346:365845, 1)
> set.seed(seed)
> sim <- arfima.sim(1000, model = list(dfrac = -0.9))
> arfima(sim, dmean = FALSE, quiet = TRUE)
```

```
Number of modes: 1

Call:
arfima(z = sim, dmean = FALSE, quiet = TRUE)

Coefficients for fits:
         Coef.1:      SE.1:
d.f      -0.763203     0.0198202
zbar     -0.00206433
logl     -1.59719
sigma^2   1.00004
Starred fits are close to invertibility/stationarity boundaries


> arfima(sim, quiet = TRUE)


Number of modes: 1

Call:
arfima(z = sim, quiet = TRUE)

Coefficients for fits:
             Coef.1:       SE.1:
d.f          -0.907423      0.0279488
Fitted mean  -4.52612e-05   0.000178066
logl          23.739
sigma^2       0.949108
Starred fits are close to invertibility/stationarity boundaries
```

From the above, there are three things to notice about the fit with the dynamically estimated mean as compared to the one where the mean of the series was used. The log-likelihood is higher, as should be expected; the fitted mean is smaller; and the fitted parameter is much closer to the true generating parameter.

6.3.3.2 On the Effect of Larger n and Mean Estimation

We have seen cases wherein modes are either hidden or induced when the sample mean is subtracted, and when the mean is dynamically estimated, the modes were about where we expected them to be from past observations. This usually happens as n increases. This seems counterintuitive at first. As we have seen in Figure 3.1, we have that anti-persistent modes generally have means closer to 0, and that this becomes more pronounced as n increases. We know that persistent modes can deal much more easily with a change in mean. When the sample mean is subtracted, since the variance of the sample mean in all HD modes is $O(n^{-\alpha})$, difference of the mean from 0 will greatly reduce the log-likelihood of an anti-persistent mode. While this is true of all modes, the divergence between the variances of two modes where $\alpha < 1$

and $\alpha > 1$ quickly becomes apparent. It should be clear that this will affect MA-based models much more than AR-based models.

We have observed that mean induction can also happen more in the sample mean subtracted case than the dynamic mean case as n gets larger. This most often occurs when in an AR-based model's apparent persistent, that is, anti-persistent, mode gets pushed towards persistence. This usually does not happen in terms of the HD parameter, but in terms of ϕ. As was noted, the value of ϕ tending to 1 gives a persistent effect, which tends to send all starts with $\alpha > 1$ to become more apparently persistent in terms of ϕ and less persistent in terms of α.

6.3.4 The Effect of Adding Noise to a Series

We know for a persistent mode on the surface generated by a series $\boldsymbol{w}$, we have that, if its TACVF is $\gamma_w(\cdot)$, we have

$$\lim_{n\to\infty} \sum_{i=-n}^{n} \gamma_w(i) = \infty \tag{6.17}$$

while for an anti-persistent mode, we have

$$\lim_{n\to\infty} \sum_{i=-n}^{n} \gamma_w(i) = 0 \tag{6.18}$$

Recall also that

$$\mathrm{Var}(\bar{w}_n) = O(n^{-\alpha}) \tag{6.19}$$

for any HD process.

However, suppose we contaminate the true process. Suppose we add independent white noise, $\boldsymbol{y} \sim \mathrm{WN}(\mathbf{1}, I_n\sigma_y^2)$ to $\boldsymbol{w}$, and as such end up with a contaminated process $\boldsymbol{x}$, with $\boldsymbol{x} = \boldsymbol{w} + \boldsymbol{y}$. Then, if the process $\boldsymbol{w}$ is persistent (ignoring the possibility of a multimodal surface for now), we have

$$\begin{aligned} \lim_{n\to\infty} \sum_{i=-n}^{n} \gamma_x(i) &= \lim_{n\to\infty} \sum_{i=-n}^{n} \gamma_w(i) + \sigma_y^2 && (6.20) \\ &= \infty && (6.21) \end{aligned}$$

and

$$\begin{aligned} \mathrm{Var}(\bar{x}_n) &= \mathrm{Var}(\bar{w}_n) + \mathrm{Var}(\bar{y}_n) && (6.22) \\ &= O(n^{-\alpha}) + O(n^{-1}) && (6.23) \\ &= O(n^{-\alpha}) && (6.24) \end{aligned}$$

If the process $\boldsymbol{w}$ is anti-persistent, however,

$$\lim_{n\to\infty}\sum_{i=-n}^{n}\gamma_x(i) = \lim_{n\to\infty}\sum_{i=-n}^{n}\gamma_w(i) + \sigma_y^2 \tag{6.25}$$

$$= \sigma_y^2 \tag{6.26}$$

and

$$\mathrm{Var}(\bar{x}_n) = \mathrm{Var}(\bar{w}_n) + \mathrm{Var}(\bar{y}_n) \tag{6.27}$$

$$= O(n^{-\alpha}) + O(n^{-1}) \tag{6.28}$$

$$= O(n^{-1}) \tag{6.29}$$

and as such we lose the anti-persistence of the process when we add noise. The effect of this depends upon the value of σ_y^2. If we consider a pure HD process, we note that from (5.11) as σ_y^2 increases, the anti-persistent process will be masked by the white noise much more quickly than a persistent one. Also, an anti-persistent process will be masked more quickly with increasing n while a persistent process tends to do the reverse.

Since we have finite n, we note that any added white noise with mean 0 will tend to make the series more like white noise. The effect of this increases with σ_y^2. If n does not increase, it is logical that both modes will become like white noise: however, the effect on the number of modes seems to differ between the MA- and AR-based models. In the former, the modes will move towards the middle of the parameter space and a mode will be lost, while the latter will have modes pushed to persistence and modes may be induced. We will see this in Figures 6.3 and 6.4.

The figures below are fairly typical of what we have seen when a fixed series has added noise. The first set is a simulated ARFI model being fit, with Gaussian mean zero noise and variances of 0 (no noise), 2, and 4. The second is a simulated FIMA model being fit, with Gaussian mean and variances of 0 (no noise), 0.25, and 0.5. Note the large difference in the noise variances: this shows how much more fragile an anti-persistent mode can be, whether real or apparent.

6.4 Conclusions and Future Work

The basis for the **simpleVis** package was actually the start of a ***Mathematica*** package we called **hdVis**. However, we quickly ran into problems that we discussed in §6.2.1. We would still like to finish the package, although each of the technical considerations we mentioned in said section have to be addressed. We are also considering adding simple plotting capabilities to our ***R*** package **arfima** that we mentioned in Chapter 5.

We would like to more closely look at the effect of multimodality on prediction, as well as the placement of the mean likelihood estimator (MeLE) as in McLeod and Quenneville [2001]. We would also like to look more closely at the asymptotics of the modes for more complex models, such as the ARMA and the ARMA-HD.

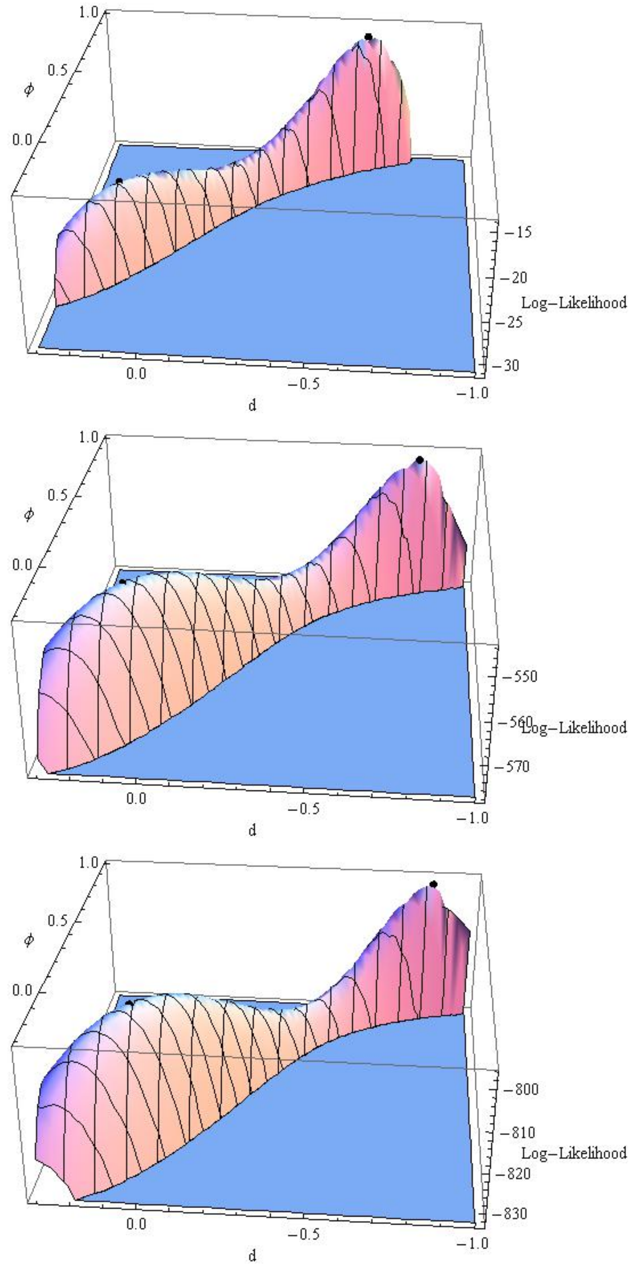


Figure 6.3: A **simpleVis** representation of an ARFI process with NID$(0, \sigma_y^2)$ noise added to the series, with the top plot having $\sigma_y^2 = 0$, the middle having $\sigma_y^2 = 2$, and the bottom having $\sigma_y^2 = 4$. The fitted values were found using the **arfima** package. The bottom plot has 3 points of the optimization from **arfima** pushed to the boundaries hidden behind the peak at the back. We note that this occurs due to a push to persistence.

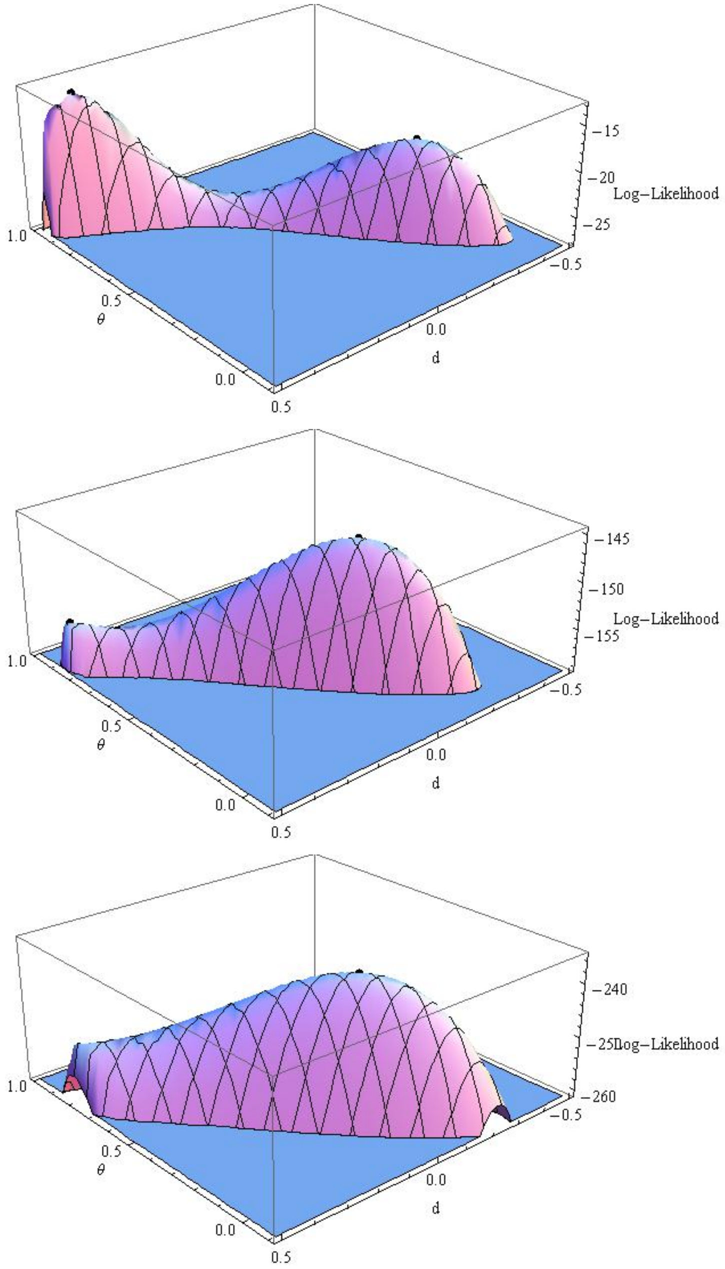


Figure 6.4: A **simpleVis** representation of a FIMA process with NID$(0, \sigma_y^2)$ noise added to the series, with the top plot having $\sigma_y^2 = 0$, the middle having $\sigma_y^2 = 0.25$, and the bottom having $\sigma_y^2 = 0.5$. The fitted values were found using the **arfima** package. The log-likelihood surface turns into one that describes zero mean white noise.

Chapter 7

Conclusion

In this thesis, we have discussed theoretical and numerical properties of hyperbolic decay (HD) time series, both persistent and anti-persistent. We derived the exact for for the spectral density function (SDF) of fractional Gaussian noise (FGN), the theoretical autocovariance function of power-law spectrum (PLS), and introduced a new HD model we called power-law autocovariance (PLA). We proved the existence of PLA, as well as deriving its SDF. We discussed inference on pure HD processes, with an example.

Furthermore, we delved into the mixture of ARMA structure with HD noise. We proved that a convolution of ARMA and HD autocovariance functions gives rise to an ARMA-HD autocovariance function within $O(r^L)$ and machine epsilon to the true autocovariance function with L being the length of the autocovariance function. Then we used Kullback-Liebler divergence to show that if the series was Gaussian, we can approximate the distribution of the series arbitrarily exactly in the same way.

We looked at minimum-mean-square-error (MMSE) predictors in the case of stationary sequences and their integration by arbitrary $d \in \mathbb{Z}_{>0}$, as well deriving a new, exact formula for prediction error variances of the integrated series. We also proved that the exact formula and the often-used limiting formula are equivalent for the ARIMA$(p, d^*, 0)$ case with $d^* \in (-1, \infty)$.

Our ***R*** **arfima** package was introduced. Said package is likely one of the more versatile time series packages for said environment, having many capabilities. These capabilities include simulation, fitting, and forecasting via exact methods and have multiple starts as the default. The package can also perform regression with autocorrelated errors, including transfer functions. We compared it to the popular **fracdiff** package with **arfima** showing superiority in all but speed.

Finally, we talked about technical aspects of visualizing a log-likelihood surface. We visualized simple surfaces to aid in understading of bimodal surfaces that occur with one short memory parameter and one HD parameter. We hypothesized on the cause of multimodality, which came to the effect of finite sample sizes and the apparent persistence or anti-persistence of modes on the log-likelihood surface. We also looked at the effect of mean estimation and added noise to a log-likelihood surface.

7.1 Future Work

As future work, we would like to speed up the **arfima** package and add more capabilities to it, such as inference capabilities like the **FGN** package. We also would like to add the PLS model to **arfima**. We would like to investigate further the causes of multimodality, and finish the **hdVis** package mentioned in Chapter 6 to do so.

Bibliography

H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

R. R. Baillie. Long memory processes and fractional integration in econometrics. *Journal of Econometrics*, 73:5–59, 1996.

R. T. Baillie, C.-F. Chung, and M. A. Tieslau. Analysing inflation by the fractionally integrated arfima-garch model. *Journal of Applied Econometrics*, 11(1):23–40, 1996.

G. A. Barnard, G. M. Jenkins, and C. B. Winsten. Likelihood inference and time series. *Journal of the Royal Statistical Society. Series A (General)*, 125(3):pp. 321–372, 1962.

O. E. Barndorff-Nielsen and G. Schou. On the parametrization of autoregressive models by partial autocorrelations. *Journal of Multivariate Analysis*, 3:408–419, 1973.

J. Beran. A goodness-of-fit test for time series with long range dependence. *Journal of the Royal Statistical Society B*, 54(3):749–760, 1992.

J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, 1994.

J. Beran. *longmemo: Statistics for Long-Memory Processes*, 2011. URL `http://CRAN.R-project.org/package=longmemo`. R package version 1.0-0. Access date: 2012-09-15.

R. J. Bhansali and P. S. Koboszka. *Theory and Applications of Long-Range Dependence*, chapter Prediction of Long-Memory Time Series, pages 355–368. Birkhäuser Boston Inc., 2003.

G. Bhardwaj and N. R. Swanson. An empirical investigation of the usefulness of arfima models for predicting macroeconomic and financial time series. *Journal of Econometrics*, 131:539 – 578, 2006.

P. Bloomfield. An exponential model for the spectrum of a scalar time series. *Biometrika*, 60 (2):217–226, 1973. ISSN 00063444.

G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. John Wiley and Sons, 2008a.

G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Wiley, New York, 4th edition, 2008b.

R. P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, 1973.

P. J. Brockwell and R. Dahlhaus. Generalized Levinson-Durbin and Burg algorithms. *Journal of Econometrics*, 118:129–149, 2004.

P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*. Springer-Verlag, 1991.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006. ISBN 0521841089.

E. W. Cheney and D. Kincaid. *Numerical Mathematics and Computing*. International student edition. Brooks/Cole, 2007.

W. P. Cleveland. *The Elements of Graphing Data*. Hobart Press, 2nd edition, 1994.

Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991. ISBN 0-471-06259-6.

D.R. Cox. *Principles of Statistical Inference*. Cambridge University Press, 2006.

P. F. Craigmile and P. Guttorp. Space-time modelling of trends in temperature series. *Journal of Time Series Analysis*, 32:378–395, 2011.

J. D. Cryer and K. S. Chan. *Time Series Analysis: With Applications in R*. Springer Texts in Statistics. Springer, 2008.

R. Dahlhaus. Efficient parameter estimation for self-similar processes. *The Annals of Statistics*, 17(4):1749–1766, 1989.

A.C. Davison and D.V. Hinkley. *Bootstrap Methods and their Application*. Cambridge University Press, 1997.

L. Dębowski. On processes with hyperbolically decaying autocorrelations. *Journal of Time Series Analysis*, 32(5), 2011.

P. Doukhan, G. Oppenheim, and M. S. Taqqu, editors. *Theory and Applications of Long-Range Dependence*. Birkhäuser Boston Inc., 2003.

B. Efron. Defining the curvature of a statistical problem (with applications to second order efficiency). *The Annals of Statistics*, 3(6):1189–1242, 1975.

B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

R. Fox and M. S. Taqqu. Large-sample properties of parameter estimates for strongly dependent stationary gaussian time series. *The Annals of Statistics*, 14(2):517–532, 1986.

C. Fraley. *fracdiff: Fractionally differenced ARIMA*, 2012. URL `http://CRAN.R-project.org/package=fracdiff`. R package version 1.4-2. Access date: 2012-09-15.

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

C. W. J. Granger. The typical spectral shape of an economic variable. *Econometrica*, 34(1): 150–161, 1966.

C. W. J. Granger and R. Joyeux. An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis*, 1(1):15–30, 1980.

H. L. Gray, N. Zhang, and W. A. Woodward. On generalized fractional processes. *Journal of Time Series Analysis*, 10(3):233–257, 1989. ISSN 1467-9892. Correction: Volume 15, Number 5 (1994).

E. J. Hannan. *Multiple Time Series*. Wiley, 1970.

J. Haslett and A. E. Raftery. Space-time modelling with long-memory dependence: Assessing ireland's wind power resource. *Journal of the Royal Statistical Society, Series C*, 38(1):1–50, 1989.

K. W. Hipel and A. I. McLeod. *Time Series Modelling of Water Resources and Environmental Systems*. Elsevier, 1994.

K. W. Hipel and A. I. McLeod. *Time Series Modelling of Water Resources and Environmental Systems*. Elsevier, Amsterdam, 1994. URL `http://www.stats.uwo.ca/faculty/aim/1994Book/default.htm`.

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

J. R. M. Hosking. Modeling persistence in hydrological time series using fractional differencing. *Water Resources Research*, 20(12):1898–ï£¡1908, 1984.

H. E. Hurst. Long-term storage capacity of reservoirs. *Transactions of the American Society of Civil Engineers*, 116:770–808, 1951.

O. Kärner. Comment on Hurst exponent. *Geophysical Research Letters*, 28(19):3825–3826, 2001.

O. Kärner. On nonstationarity and antipersistency in global temperature series. *Journal of Geophysical Research*, 107(D20-4415), 2002.

G. Li and W. K. Li. Least absolute deviation estimation for fractionally integrated autoregressive moving average time series models with conditional heteroscedasticity. *Biometrika*, 95 (2):399–414, 2008.

W. K. Li. *Topics in Time Series Analysis*. PhD thesis, Western University, 1981.

W. K. Li. *Diagnostic Checks in Time Series*. Chapman and Hall/CRC, New York, 2004.

W. K. Li and A. I. McLeod. Fractional time series modelling. *Biometrika*, 73(1):217–221, 1986.

J.-W. Lin and A. I. McLeod. Improved Pena-Rodriguez portmanteau test. *Computational Statistics and Data Analysis*, 51:1731–1738, 2006.

E. Mahdi and A. I. McLeod. Improved multivariate portmanteau test. *Journal of Time Series Analysis*, 33(2):211–222, 2012.

B. B. Mandelbrot and J. W. Van Ness. Fractional Brownian motions, fractional noises and applications. *SIAM Review*, 10(4):422–437, 1968.

G. Manley. Central England temperatures: monthly means 1659 to 1973. *Quarterly Journal of the Royal Meteorological Society*, 100:389–405, 1974.

A. I. McLeod. Improved Box-Jenkins estimators. *Biometrika*, 64(3):531–534, 1977.

A. I. McLeod. Duality and other properties of multiplicative autoregressive-moving average models. *Biometrika*, 71:207–211, 1984.

A. I. McLeod. Hyperbolic decay time series. *Journal of Time Series Analysis*, 19(4):473–483, 1998.

A. I. McLeod and K. W. Hipel. Preservation of the rescaled adjusted range, Part 1, A reassessment of the Hurst phenomenon. *Water Resources Research*, 14:491–508, 1978.

A. I. McLeod and B. Quenneville. Mean likelihood estimators. *Statistics and Computing*, 11 (1):57–65, January 2001. ISSN 0960-3174. doi: 10.1023/A:1026509916251. URL `http://dx.doi.org/10.1023/A:1026509916251`.

A. I. McLeod and J. Q. Veenstra. *FGN: Fractional Gaussian Noise and simple models for hyperbolic decay time series*, 2012. URL `http://CRAN.R-project.org/package=FGN`. R package version 2.0. Access date: 2012-09-15.

A. I. McLeod and Y. Zhang. Faster ARMA maximum likelihood estimation. *Computational Statistics and Data Analysis*, 52:2166–2176, 2008.

A. I. McLeod, H. Yu, and Z. L. Krougly. Algorithms for linear time series analysis: with R package. *Journal of Statistical Software*, 23(5):1–26, 2007a.

A. I. McLeod, H. Yu, and Z. L. Krougly. Algorithms for linear time series analysis: with R package. *Journal of Statistical Software*, 23(5):1–26, 2007b.

A. I. McLeod, H. Yu, and Z. Krougly. *ltsa: Linear time series analysis*, 2012. URL `http://CRAN.R-project.org/package=ltsa`. R package version 2.0. Access date: 2012-09-15.

G. M. Molchan. *Theory and Applications of Long-Range Dependence*, chapter Historical Comments Relatted to Fractional Brownian Motion, pages 5–38. Birkhäuser Boston Inc., 2003.

J. F. Monahan. A note on enforcing stationarity in autoregressive-moving average models. *Biometrika*, 71(2):403–404, 1984.

A. Montanari, R. Rosso, and M. S. Taqqu. A seasonal fractional ARIMA model applied to the Nile River monthly flows at aswan. *Water Resources Research*, 36:1249–1259, 2000.

E. Moulines, F. Roueff, and M. S. Taqqu. A wavelet whittle estimator of the memory parameter of a non-stationary gaussian time series. *The Annals of Statistics*, 11(4):1925–1956, 2006.

W. Palma. *Long-Memory Time Series: Theory and Methods*. John Wiley and Sons, 2007.

D. E. Parker, T. P. Legg, and C. K. Folland. A new daily central england temperature series, 1772-1991. *International Journal of Climatology*, 12:317–342, 1992.

D. B. Percival and A. T. Walden. *Wavelet Methods for Time Series Analysis*. Cambridge University Press, 2000.

D. Peňa and J. Rodriguez. A powerful portmanteau test of lack of fit for time series. *Journal of American Statistical Association*, 97:601–610, 2002.

S. Porter-Hudak. An application of the seasonal fractionally differenced model to the monetary aggregates. *Journal of the American Statistical Association*, 85(410):338–344, 1990.

M. B. Priestley. *Spectral Analysis and Time Series: Univariate Series*. Academic Press, 1981.

B. K. Ray. Long-range forecasting of ibm product revenues using a seasonal fractionally differenced arma model. *International Journal of Forecasting*, 9(2):255 – 269, 1993.

R. Royall. *Statistical Evidence: A Likelihood Paradigm*. Chapman and Hall, New York, 1997.

Gideon Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461–464, 1978.

M. M. Siddiqui. On the inversion of the sample covariance matrix in a stationary autoregressive process. *The Annals of Mathematical Statistics*, 22(2):585–588, 1958.

F. Sowell. Maximum likelihood estimation of stationary univariate fractionally integrated time series models. *Journal of Econometrics*, 53:165–188, 1992.

D. A. Sprott. *Statistical Inference in Science*. Springer, New York, 2000.

H.M. Srivastava and C. Junesang. *Series Associated With the Zeta and Related Functions*. Kluwer, 2001.

M. Taniguchi. On the second order asymptotic efficiency of estimators of gaussian arma processes. *The Annals of Statistics*, 36:157–169, 1983.

M. S. Taqqu. *Theory and Applications of Long-Range Dependence*, chapter Fractional Brownian Motion and Long-Range Dependence, pages 39–42. Birkhäuser Boston Inc., 2003.

E.C. Titchmarsh and D.R. Heath-Brown. *The Theory of the Riemann Zeta-Function*. Oxford University Press, 1987.

L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. Number 50. Society for Industrial Mathematics, 1997.

R. S. Tsay. *Analysis of Financial Time Series*. Wiley, New York, 3rd edition, 2010.

J. Veenstra and A. I. McLeod. *Appendix: Hyperbolic Decay Time Series Models*, 2012a. URL `http://onlinelibrary.wiley.com/journal/10.1111/%28ISSN%291467-9892`. Journal of Time Series Analysis Online Website.

J. Veenstra and A. I. McLeod. *Time Series for Power-Law Decay*, 2012b. URL `http://demonstrations.wolfram.com/TimeSeriesForPowerLawDecay/`. Wolfram Demonstrations Project.

A. M. Walker. Asymptotic properties of least squares estimates of the parameters of the spectrum of a stationary non-deterministic time series. *Journal of the Australian Mathematical Society*, 4:363–384, 1964.

P. Whittle. Estimation and information in stationary time series. *Arkiv för Matematik*, 23(2): 423–434, 1963.

Wikipedia. Bfgs method, 2012a. URL `http://en.wikipedia.org/wiki/BFGS`. [Online; accessed 9-February-2013].

Wikipedia. Lerch zeta function, 2012b. URL `http://en.wikipedia.org/wiki/Lerch_zeta_function`. [Online; accessed 4-December-2012].

Wikipedia. Nelder-mead method, 2012c. URL `http://en.wikipedia.org/wiki/Nelder%E2%80%93Mead_method`. [Online; accessed 9-February-2013].

Wikipedia. Polylogarithm, 2012d. URL `http://en.wikipedia.org/wiki/Polylogarithm`. [Online; accessed 4-December-2012].

Wikipedia. Multivariate normal distribution, 2013. URL `http://en.wikipedia.org/wiki/Multivariate_normal_distribution`. [Online; accessed 9-February-2013].

S. Wolfram. *A New Kind of Science*. Wolfram Media, 2002.

C.H. Yong. On the asymptotic behavior of trigonometric series i. *Journal of Mathematical Analysis and Applications 381,- 14 (1972)*, 33:23–34, 1971.

C.H. Yong. On the asymptotic behavior of trigonometric series ii. *Journal of Mathematical Analysis and Applications 381,- 14 (1972)*, 38:1–14, 1972.

Appendix A: Chapter 2

This Appendix is also available as an online document that can be used with *Mathematica* or the freely available *Mathematica* Reader. It is intended to provide this appendix as a supplement on the journal website when the paper from this chapter is published.

Contents

- Table of Asymptotically Equivalent Parameters
- Derivation of SDF for FGN
- Numerical Comparisons with the Previous Method (FGN)
- Autocovariance Function of PLS
- The PLA Process
- Comparing SDF of Four Types Hyperbolic Decay Time Series Models
- Comparing the TACF of Four Types Hyperbolic Decay Time Series Models
    - Interactive graphical comparison
    - Interactive tabular comparison
- Fisher Information
- Comparing the snr for HD Models

Table of Asymptotically Equivalent Parameters

Given	Asymptotic Equivalent
a	$\{H \to 1 - \frac{a}{2}, d \to \frac{1-a}{2}\}$
H	$\{d \to -\frac{1}{2} + H, a \to 2 - 2H\}$
d	$\{H \to \frac{1}{2} + d, a \to 1 - 2d\}$

Note that a is the parameter for PLA, H is for FGN, and d is for FD; for PLS we have p = a.

Derivation of SDF for FGN

We use radial frequency definition for frequency so the spectral density function (SDF) is defined by the Fourier transformation,

$$f(\lambda) = \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \gamma_k e^{-i k \lambda}$$

$$= \frac{1}{2\pi}\left(\gamma_0 + 2\sum_{k=1}^{\infty}\gamma_k \cos(\lambda k)\right)$$

and the inverse transformation gives the ACVF

$$\gamma_k = \int_{-\pi}^{\pi} f(\lambda)\, e^{i k \lambda}\, d\lambda$$

- **FGN**

  May be defined by its autocovariance function

  $$\gamma_k = \tfrac{1}{2}\gamma_0\left((k+1)^{2H} - 2k^{2H} + (k-1)^{2H}\right),\ k > 0$$

  $$f(\lambda) = \frac{1}{2\pi}\left(\gamma_0 + 2\sum_{k=1}^{\infty}\gamma_k \cos(\lambda k)\right) \tag{1}$$

- **Theorem**

  The spectral density function for FGN may be written,

  $$\tfrac{1}{4\pi}\left(e^{-i\lambda}\left(\Phi\left(e^{-i\lambda}, -2H, 0\right) + \Phi\left(e^{-i\lambda}, -2H, 2\right)\right) + \right.$$
  $$\left. e^{i\lambda}\left(\Phi\left(e^{i\lambda}, -2H, 0\right) + \Phi\left(e^{i\lambda}, -2H, 2\right)\right) - 2\left(\mathrm{Li}_{-2H}\left(e^{-i\lambda}\right) + \mathrm{Li}_{-2H}\left(e^{i\lambda}\right) - 1\right)\right)$$

- **Proof**

```
A1 = Sum[(1 + Abs[k])^(2 H) Cos[k λ], {k, 1, ∞}];

A2 = Sum[k^(2 H) Cos[k λ], {k, 1, ∞}];

A3 = Sum[(Abs[k] - 1)^(2 H) Cos[k λ], {k, 1, ∞}];

f = 1 / (2 π) FullSimplify[1 + (A1 - 2 A2 + A3), Assumptions → H > 0 && H < 1 && λ > 0 && λ < Pi];

sdfFGN[λ_, H_] := Evaluate[f // Re];

TraditionalForm[sdfFGN[λ, H]]
```

$$\frac{1}{4\pi}\mathrm{Re}\Big($$
$$e^{-i\lambda}\left(\Phi\left(e^{-i\lambda}, -2H, 0\right) + \Phi\left(e^{-i\lambda}, -2H, 2\right)\right) + e^{i\lambda}\left(\Phi\left(e^{i\lambda}, -2H, 0\right) + \Phi\left(e^{i\lambda}, -2H, 2\right)\right) - 2\left(\mathrm{Li}_{-2H}\left(e^{-i\lambda}\right) + \mathrm{Li}_{-2H}\left(e^{i\lambda}\right) - 1\right)\Big)$$

Numerical Comparisons with the Previous Method (FGN)

Beran (eqn. 2.17, p.53) the spectral density function is given by,

$$f(\lambda) = (\gamma_0/\pi)\sin(\pi H)\ \Gamma(2H+1)(1-\cos\lambda)\sum_{k=-\infty}^{\infty}|2\pi k + \lambda|^{-2H-1} \tag{2}$$

- **Computing SDF of FGN with Beran's R package longmemo**

  The R package (Beran, 2011-06-15) function specFGN() implements eqn. (2) and returns the standardized version of $f(\lambda)$, that is, $h(\lambda) = f(\lambda)/\mathbb{C}(H)$, where $\mathbb{C}(H) = \sigma_a^2/\sigma_z^2$. The following R script uses the function specFGN() to compute $f(\lambda)$ in eqn. (2) with $\sigma_z^2 = 1$.

```
sdfFGNB <- function(H, n){
 ans <- specFGN(H, n)
 (ans$spec)*(ans$theta1)
```

```
  }
> HS <- c(0.05, 0.2,  0.7, 0.9, 0.98)
> m <- 200
> #compute at these test frequencies
> LD <- c(1/m, 33/m, 66/m, 99/m)*2*pi
> tb <- matrix(numeric(length(LD)*length(HS)), ncol=length(HS))
> dimnames(tb)<-list(paste("ld =",round(LD,4)), paste("H =", HS))
>   for (j in 1:length(HS))
+     for (i in 1:length(LD))
+         tb[, j] <- sdfFGNB(HS[j], n=m)[c(1,33,66,99)]
> round(tb,5)
```

```
            H = 0.05 H = 0.2 H = 0.7 H = 0.9 H = 0.98
ld = 0.0314  0.00108 0.01044 0.63836 1.31356  0.53401
ld = 1.0367  0.05153 0.11305 0.14977 0.07451  0.01723
ld = 2.0735  0.12192 0.20285 0.10504 0.03750  0.00763
ld = 3.1102  0.15601 0.24266 0.09196 0.02857  0.00549
```

---

longmemo: Statistics for Long-Memory Processes (Jan Beran) – Data and Functions. 2011-06-15, Version 1.0-0. http://cran.r-project.org/web/packages/longmemo/index.html

Computing SDF of FGN with our *Mathematica* Function

	0.05	**0.2**	**0.7**	**0.9**	**0.98**
0.0314159	0.00112	0.01045	0.63836	1.31356	0.53401
1.03673	0.08779	0.11678	0.14978	0.07451	0.01723
2.07345	0.23134	0.21411	0.10504	0.03750	0.00763
3.11018	0.30368	0.25786	0.09197	0.02857	0.00549

Conclusion

For the persistent case $H \in (0.5,\ 1)$, Beran's method works well. It is less accurate in the more extreme anti-persistent cases where $H \leq 0.2$. Presumably this accuracy could be improved by increasing the number of terms used in the summation in eqn. (2).

Autocovariance Function of PLS

The autocorrelation function at lag k is given by

```
Integrate[ⅇ^(-ⅈ λ k) p / (2 π^p) Abs[λ]^(p-1), {λ, -π, π}, Assumptions → p > 0 && k ∈ Integers]
```

$$\text{HypergeometricPFQ}\left[\left\{\frac{p}{2}\right\}, \left\{\frac{1}{2}, 1+\frac{p}{2}\right\}, -\frac{1}{4} k^2 \pi^2\right]$$

A note of the derivation of the sdf of PLS

```
Integrate[λ^(p-1), {λ, 0, π}, Assumptions → p > 0]
```

$$\frac{\pi^p}{p}$$

Hence,

```
sdfPLS[λ_, p_] := p / (2 π^p) λ^(p-1)
TraditionalForm[sdfPLS[λ, p]]
```

$$\frac{1}{2} p\, \pi^{-p} \lambda^{p-1}$$

Large-lag formula

```
AcfPLS[p_, k_] := HypergeometricPFQ[{p / 2}, {1 / 2, 1 + p / 2}, - (k^2 * Pi^2) / 4];
```

For the persistent case, $0 < p < 1$, we obtain

```
Limit[κ^p AcfPLS[p, κ], κ → ∞, Assumptions → {p > 0 && p < 1}]
```

$$\frac{2^{-1+p}\, p\, \pi^{\frac{1}{2}-p}\, \mathrm{Gamma}\left[\frac{p}{2}\right]}{\mathrm{Gamma}\left[\frac{1}{2}-\frac{p}{2}\right]}$$

For the anti-persistent case, *Mathematica* obtains:

```
Limit[κ^p AcfPLS[p, κ], κ → ∞, Assumptions → {p > 1 && κ ∈ Integers}]
```

$$p\, \pi^{-p}\, \mathrm{Cos}\left[\frac{p\,\pi}{2}\right]\, \mathrm{Gamma}[p]$$

The PLA Process

Let z_t, $t = 1, 2, \ldots$ be a covariance stationary Gaussian time series with variance σ_z^2 and autocovariance function $\gamma_k = \gamma_0\, \rho_k$,

where for $k > 0$, $\rho_k = c_a\, k^{-a}$, where

$$c_a = \begin{cases} (-2\,\zeta(\alpha))^{-1} & a \neq 1, \\ 0 & a = 1, \end{cases} \tag{3}$$

where $a \in (0, \infty)$ is the model parameter, and $\rho_{-k} = \rho_k$.

Recall $\zeta(\alpha)$ denotes the Reimann zeta function that is defined by

$$\zeta(a) = \begin{cases} \sum_{k=1}^{\infty} k^{-a} & a \in (1, \infty) \\ \left(1 - 2^{1-a}\right)^{-1} \sum_{k=1}^{\infty} k^{-a} (-1)^{k-1} & a \in (0, 1) \end{cases} \tag{4}$$

The autocorrelation of the HD Model with parameter a is defined for $k > 0$ by $\rho_k = c_a\, k^{-a}$, where

Lemma: Continuity of ρ_k

We have as a lemma that $\rho_k = c_a\, k^{-a}$ is a continuous function for $a \in (0, \infty)$. For proof we verify that lim $c_a \to 0$ as $a \to 1$.

```
Limit[(-2 Zeta[a])^-1, a → 1]
```

```
0
```

```
Limit[(-2 Zeta[a])^-1, a → 1, Direction → 1]
```

```
0
```

```
Limit[(-2 Zeta[a])^-1, a → 1, Direction → -1]
```

```
0
```

Derivation of the SDF of the PLA Process

To establish the existence of the PLA process, we must show that the function

$$f(\lambda) = \frac{\sigma_w^2}{2\pi}\left(1 + 2\sum_{k=1}^{\infty}\gamma_k \cos(\lambda\, k)\right) \tag{5}$$

has $f(\lambda) \geq 0$ for $\lambda \in (-\pi, \pi)$ and $\int_{-\pi}^{\pi} f(\lambda)\, d\lambda = \sigma_w^2$. That is, we must show it is a spectral density function, which we do in Theorem 2.3 of the main text, based on the following. First we must derive $f(\lambda)$, letting $\sigma_w^2 = 1$ without loss of generality:

```
f = (1 + 2 Sum[- (1 / (2 Zeta[a])) k^-a Cos[k λ], {k, 1, ∞}]) / 2 π
```

$$\frac{1}{2}\pi\left(1 + \frac{-\text{PolyLog}[a, e^{-i\lambda}] - \text{PolyLog}[a, e^{i\lambda}]}{2\,\text{Zeta}[a]}\right)$$

```
TraditionalForm[f]
```

$$\frac{1}{2}\pi\left(1 + \frac{-\text{Li}_a\left(e^{-i\lambda}\right) - \text{Li}_a\left(e^{i\lambda}\right)}{2\,\zeta(a)}\right)$$

```
G[a_, ν_] := Integrate[(1 + (-PolyLog[a, ⅇ^(-ⅈ λ)] - PolyLog[a, ⅇ^(ⅈ λ)]) / (2 Zeta[a])) / (2 π), {λ, -π, ν}]
```

Then we must show that G[a, -π] = 0 and G[a, π] = 1.

```
G[a, -π]
```

```
0
```

```
G[a, π]
```

```
1
```

With the arguments in Theorem 2.3, the PLA process exists. Then we have that

$$f(\lambda) = \frac{1}{2}\pi\,\sigma_w^2\left(1 - \frac{\text{Li}_a\left(e^{-i\lambda}\right) + \text{Li}_a\left(e^{i\lambda}\right)}{2\,\zeta(a)}\right)$$

which is Theorem 2.4.

■ Asymptotic formula for small λ

```
f0 = Series[f, {λ, 0, 1}, Assumptions → α > 0 && λ ∈ {-π, π}] // Normal;
```

```
c = FullSimplify[f0 / λ^(α-1), Assumptions → {λ > 0, α > 0}]
```

$$-\frac{\left(8 + (-2+\alpha)\,(-1+\alpha)\,\lambda^2\right)\text{Gamma}[1-\alpha]\,\text{Sin}\left[\frac{\pi\alpha}{2}\right]}{8\,\text{Zeta}[\alpha]}$$

```
Limit[c, λ → 0]
```

$$-\frac{\text{Gamma}[1-\alpha]\,\text{Sin}\left[\frac{\pi\alpha}{2}\right]}{\text{Zeta}[\alpha]}$$

```
% // TraditionalForm
```

$$-\frac{\sin\left(\frac{\pi\alpha}{2}\right)\Gamma(1-\alpha)}{\zeta(\alpha)}$$

```
TraditionalForm[sdfPLASmall[λ, α]]
```

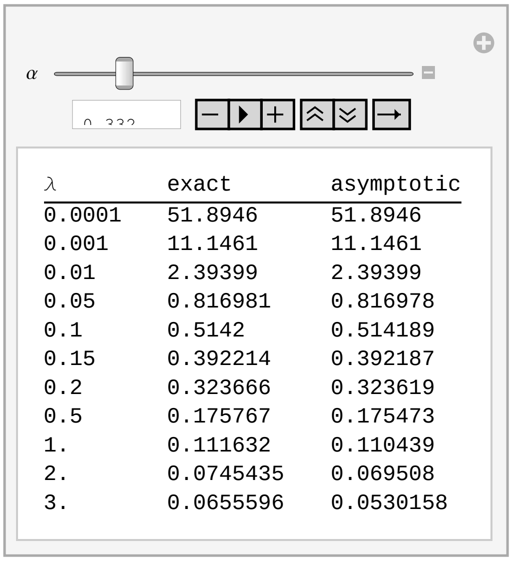
$$-\frac{\lambda^{\alpha-1}\sin\left(\frac{\pi\alpha}{2}\right)\Gamma(1-\alpha)}{2\pi\,\zeta(\alpha)}$$

$$f(\lambda) \approx -\frac{\lambda^{\alpha-1}\sin\left(\frac{\pi\alpha}{2}\right)\Gamma(1-\alpha)}{2\pi\zeta(\alpha)}$$

```
sdfPLA[λ_, α_] :=
  Re[(1 + (-PolyLog[α, E^((-I) * λ)] - PolyLog[α, E^(I * λ)]) / (2 * Zeta[α])) / (2 * Pi)];
```

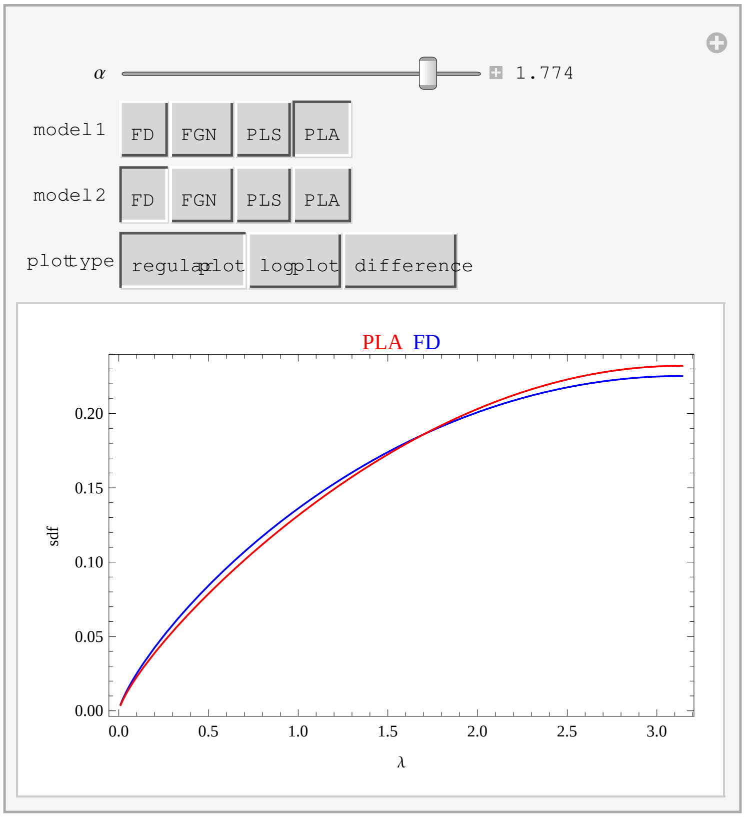
```
sdfPLASmall[λ_, α_] := -((Gamma[1 - α] * Sin[(Pi * α) / 2]) / Zeta[α]) λ^(α-1) / (2 π);
```

- **Dynamic Table Comparing Exact and Asymptotic SDF**



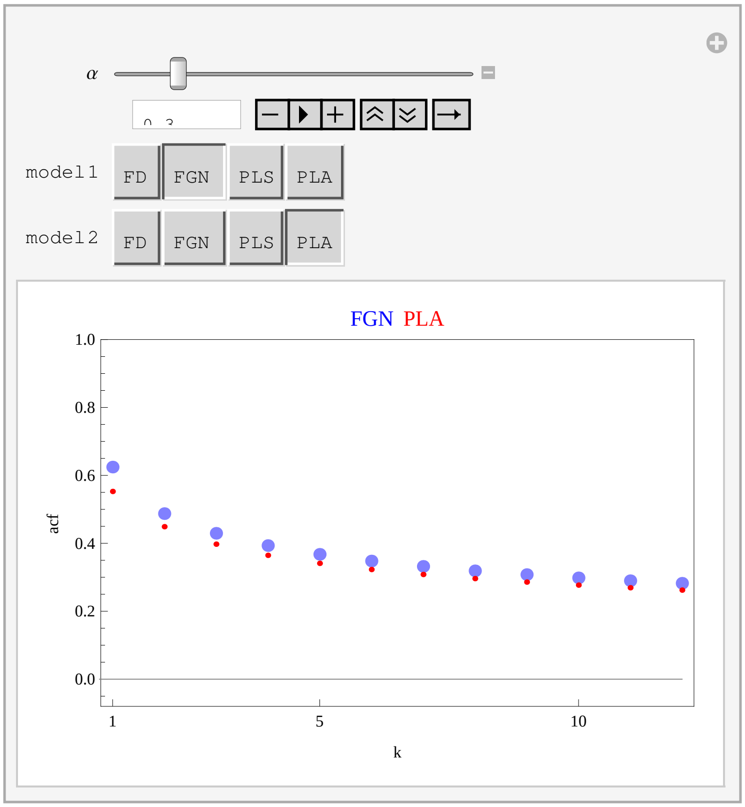
λ	exact	asymptotic
0.0001	51.8946	51.8946
0.001	11.1461	11.1461
0.01	2.39399	2.39399
0.05	0.816981	0.816978
0.1	0.5142	0.514189
0.15	0.392214	0.392187
0.2	0.323666	0.323619
0.5	0.175767	0.175473
1.	0.111632	0.110439
2.	0.0745435	0.069508
3.	0.0655596	0.0530158

Comparing SDF of Four Types Hyerbolic Decay Time Series Models

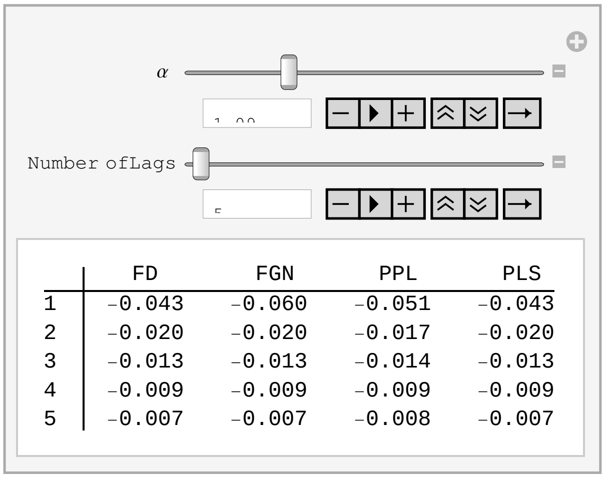


Comparing the TACF of Four Types Hyperbolic Decay Time Series Models

- **Interactive Graphical Comparison**



- **Interactive Tabular Comparison**



	FD	FGN	PPL	PLS
1	−0.043	−0.060	−0.051	−0.043
2	−0.020	−0.020	−0.017	−0.020
3	−0.013	−0.013	−0.014	−0.013
4	−0.009	−0.009	−0.009	−0.009
5	−0.007	−0.007	−0.008	−0.007

Fisher Information

The Fisher information for the PLS model can be written,

$$I(p) = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\frac{d}{d p}\log f(p,\ \lambda)\right)^2 d\lambda \tag{6}$$

```
df = D[ Log[ p / (2 π^p) λ^(p-1)]] /. p → α
```

$$\mathrm{Log}\left[\frac{1}{2}\pi^{-\alpha}\,\alpha\,\lambda^{-1+\alpha}\right]$$

```
FI = FullSimplify[(1 / π) Integrate[df^2, {λ, 0, π}, Assumptions → p > 0]]
```

$$2 + \mathrm{Log}[2]^2 + \alpha\,(-4 + 2\,\alpha + \mathrm{Log}[4]) + \mathrm{Log}\left[\frac{1}{4\,\pi^2}\right] + (\mathrm{Log}[4] - \alpha\,(-4 + 2\,\alpha + \mathrm{Log}[4]))\,\mathrm{Log}[\pi] +$$
$$(-1 + \alpha)^2\,\mathrm{Log}[\pi]^2 + \left(2\,(-1 + \alpha)\,(-1 + \mathrm{Log}[\pi]) + \mathrm{Log}\left[\frac{\pi^{-\alpha}\,\alpha}{4}\right]\right)\mathrm{Log}[\pi^{-\alpha}\,\alpha]$$

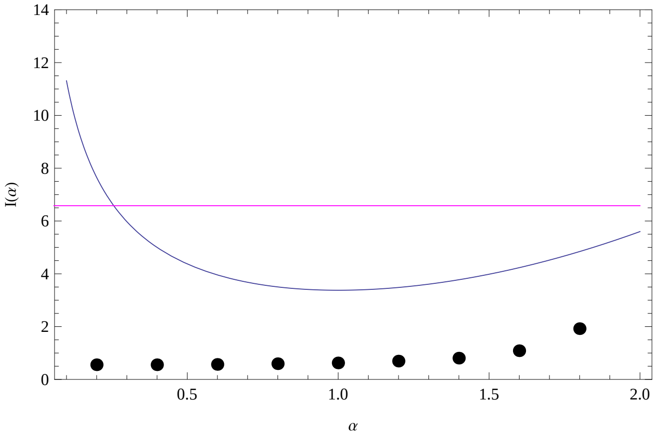
In the FD case, $I(d) = \pi^2/6$ and since $d = 1/2 - \alpha/2$, $I(\alpha) = 2\,\pi^2/3 = 6.58$, and we note that *Mathematica* cannot compute the Fisher information for the PLA or FGN models.

By computer simulation (McLeod, Yu & Krougly 2007, Table 14) we found that with $n = 2000$ and using 10^5 simulations for the FGN model,

H	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
n Var(H)	0.13	0.23	0.31	0.36	0.4	0.42	0.44	0.45	0.45

Since $\alpha = 2 - 2\,H$, $\mathrm{Var}(\hat{\alpha}) = 4\,\mathrm{Var}(\hat{H})$, hence $I(\alpha) = \left(4\,n\,\mathrm{Var}(\hat{H})\right)^{-1}$.

We have the following plot of the information of the FGN (black), FD (magneta) and PLS (blue) models.



Comparing the snr for HD Models

The snr defined by, $\mathrm{snr} = \sigma_z^2/\sigma_a^2$, is ratio total variance divided by the variance of the one-step ahead forecast and so it indicates the predictability of the time series, $\mathrm{snr} \geq 1$ and the larger snr is the better the prediction. Sometimes the coefficient of determination, R^2, has been used for this purpose as well (Nelson, 1976) and $R^2 = 1 - 1/\mathrm{snr}$. For the linear time series in , $\mathrm{snr} = 1\big/\sum_k \psi_k^2$ or equivalently in in terms of the spectral density function,

$$\mathrm{snr} = \frac{\int_{-\pi}^{\pi} f(\lambda)\, d\lambda}{\frac{1}{2\pi}\int_{-\pi}^{\pi} \log 2\pi\ f(\lambda)\, d\lambda} \tag{7}$$

For the FGN, PLS and PLA models, the snr may be obtained using eqn. (7) and for the FD model the snr may be computed directly from the definition.

The table below compares the snr for the four HD models with $\alpha = 0.1,\ 0.25,\ 0.5,\ 0.75,\ 0.9,\ 1,\ 1.1,\ 1.25,\ 1.5,\ 1.75,\ 2,\ 2.25,\ 2.5,\ 2.75,\ 2.9$. The FGN has the largest snr while FD and PLA have the lowest (depending on whether the process is persistent or anti-persistent) but the differences become very small provided that α is not too extreme. When $\alpha = 0.1$, there is about a 25% relative difference, that is, $(\mathrm{snr(FGN)} - \mathrm{snr(FD)})/\mathrm{snr(FD)} \approx 0.25$. Note that the PLA model has a larger snr than the FD model when the process is persistent and a smaller snr when the process is anti-persistent. The ∞ symbol in the table is to show the FGN model is not defined for $\alpha \geq 2$. The snr of all models is trivially equal 1 when $\alpha = 1$.

α	FD	FGN	PLS	PLA
0.10	3.64	4.55	4.07	3.81
0.25	1.76	2.04	1.89	1.80
0.50	1.18	1.26	1.21	1.19
0.75	1.03	1.05	1.04	1.03
0.90	1.00	1.01	1.01	1.00
1.00	1.00	1.00	1.00	1.00
1.10	1.00	1.01	1.00	1.00
1.25	1.02	1.04	1.03	1.02
1.50	1.08	1.15	1.10	1.07
1.75	1.16	1.37	1.21	1.15
2.00	1.27	∞	1.36	1.23
2.25	1.41	∞	1.55	1.32
2.50	1.57	∞	1.79	1.42
2.75	1.77	∞	2.09	1.51
2.90	1.90	∞	2.31	1.56

Curriculum Vitae

Name: Justin Veenstra

Education:

Ph.D. in Statistics, 2008 - 2013
Western University
London, Ontario, Canada
Supervisor: A. I. McLeod

M.Math in Statistics, 2006 - 2007
University of Waterloo
Waterloo, Ontario, Canada
Supervisor: S. Chenouri

Hon. B.Sc. in Statistics and Mathematics, 1999 - 2005
University of Toronto
Toronto, Ontario, Canada

Honours and Awards:

Western Graduate Research Scholarship: 2008 - 2012

President's Scholarship, University of Waterloo: 2006 - 2007

NSERC PGS M: 2006 - 2007

Trinity College Entrance Scholarship: 1999 - 2000

Software, Publications, and Presentations:

J. Veenstra and A. I. McLeod (2012). arfima: A package for exact simulation, estimation, and forecasting of long-memory time series. `http://CRAN.R-project.org/package=arfima`. R package version 1.2-4.

A. I. McLeod and J. Veenstra (2012). FGN: Fractional Gaussian Noise, estimation and simulaton. `http://CRAN.R-project.org/package=FGN`. R package version 2.0.

J. Veenstra and A. I. McLeod. Time Series for Power-Law Decay, `http://demonstrations.wolfram.com/TimeSeriesForPowerLawDecay/`. Wolfram Demonstrations Project.

C. M. T. Greenwood, S. Sun, J. Veenstra, N. Hamel, B. Niell, S. Gruber, W. D. Foulkes (2010). How old is this mutation? A study of three Ashkenazi Jewish founder mutations. BMC Genetics 11:39.

W. Xu, C. Taylor, J. Veenstra, S. B. Bull, M. Corey, C. M. T. Greenwood (2005). Recursive Partitioning Models for Linkage in COGA Data. BMC Genetics 6(Suppl 1): S38.

W. Xu, C. Taylor, J. Veenstra, S. B. Bull, M. Corey, C. M. T. Greenwood (2004). Recursive partitioning models for linkage to the COGA data. Poster presented at International Genetic Analysis Workshop 14, Noordwijkerhout, Netherlands. (Prepared most of the poster, did not present.)

Relevant Work Experience:

Data Scientist, Environics Analytics, 2012 - Present

Teaching Assistant, Western University, 2008 - 2012

Statistical Consultant for the Social Science Network & Data Services, Western University (UWO), 2008 - 2009

Statistical Consultant, Private Practice, 2006 - 2012

Teaching Assistant, University of Waterloo, 2006

Research Student, Hospital for Sick Children, 2004 - 2005

Research Student, Department of Mathematics, University of Toronto, 2002 - 2004

Research Student, Department of Astronomy, University of Toronto, 2000

Mathematics and Statistics Tutor, Private Practice, 1996 - 2012