Multivariate Real-Time Signal Extraction

Marc Wildi and Tucker McElroy

May 12, 2019

Contents

Chapter 1

Introduction

1.1 Overview

1.1.1 Signals and Extraction

In the applications of time series analysis to macroeconomics, finance, and quality control it is essential to extract useful information about trends, turning points, and anomalies in real time. The practitioner does not have the luxury of sifting past data for structural breaks, indicators of regime change, or changes to volatility. Informative elections are contingent upon understanding the dynamics of various time series at time present. Because long-term movements, as well as aberrations, are defined in terms of the long-run behavior of a time series over past, present, and future, any analysis of the present state necessarily involves a degree of forecasting. This broad topic is referred to as real-time signal extraction.

A signal is any component of a time series that is deemed useful for a particular application. If long-term movements are of interest, the signal is a trend. If short-term fluctuations about a longer-term mean are of interest, the signal is a cycle. If shocks, due to rare terrorist events or natural disasters, are of interest, the signal consists of the extreme values. If regular patterns of an annual period, linked to cultural or meteorological patterns, are of interest, the signal is a seasonal component.

However, these signals are not directly observable at time present, because in each case their definition involves all the past and future values of a time series – but the future is unknown, and only part of the past is available to us. The statistical processes by which a signal is estimated from available data is referred to as extraction, and the residual from the signal extraction is referred to as the noise. Whereas signals can be estimated from historical, or past, sections of a time series, when effort is focused upon time present we refer to the analysis as real-time signal extraction.

Real-time signal extraction is considerably more challenging, and useful, than historical signal extraction. The difficulty lies in the uncertainty about the future, which is transmitted unto the signal extraction estimates themselves. One way to conceive of this difficulty is through the warring principles of timeliness and accuracy: should we procrastinate in providing our analysis of the present, we can increase the accuracy of signal extraction, but our answers become less

relevant, even as the present time rapidly drifts into the past. Conversely, extremely timely extractions suffer from greater future uncertainty, and are likely to exhibit inaccuracy.

There is a considerable body of literature addressing signal extraction, but this book focuses upon a particular methodology called Direct Filter Analysis (DFA). As the original development of DFA was univariate, the methodology's power was limited to the information content within a single time series. But because batches of time series can be closely linked, exhibiting correlated trends, common dynamics, or even predictive relationships, it is natural to expect that a multivariate extension of DFA to vector time series will more greatly facilitate informed decision making. The topic of this book is Multivariate Direct Filter Analysis (MDFA).

1.1.2 The Classic Model-Based Paradigm

Many signals can be formulated as weighted linear combinations of a time series, in which case the real-time signal extraction problem can be approached as a Linear Prediction Problem (LPP). In order to pose an LPP, a solution criterion is needed, and Mean Squared Error (MSE) is often used: one seeks a real-time signal extraction that has minimal MSE discrepancy with the actual target signal. Although an LPP can then be solved, the solution depends on knowing something about the dynamics in the time series process. The most venerable approach to understanding these dynamics is to posit a time series model, and fit this model to the observed data. This approach, which goes back to the work of Yule in the 1930s, is called the classic paradigm, being based upon a Model-Based Analysis (MBA).

An attractive feature of MBA is that analytical formulas for the LPP solutions can often be obtained, thereby facilitating computation. The philosophy underpinning the classic paradigm is that a Data Generation Process (DGP) exists – as a theoretical, or Platonic construct – to which the observed data closely adheres. Formally, the DGP is some stochastic process defined upon a probability space, and the observed data is a realization, or sample path, of the DGP. Statistical inference is involved with the science of identifying a model class for the DGP, narrowing down the class to a particular model (by eliminating contenders), and fitting that model via fixing values of the parameters. Successive applications of model diagnostics allow for refinements, and a process by which we can verify the validity of a postulated model. Of course, all of this is done on the basis of the single realization of the DGP.

While recognizing that any such model need not be correct, i.e., exactly match the DGP itself, such models can yet be useful to the extent to which they reflect important features in the data. Yet it is difficult to keep a model simple – which is necessay to its utility – and at the same time be sufficiently versatile to explain all the data's features. Moreover, the appellation of importance is subjective: a feature deemed important to one user may be irrelevant to another. This begs the question of customization: each user, with a distinct set of criteria and desired applications, could potentially stress the importance of a subset of features at the cost of de-emphasizing others. The classic paradigm ignores, or at least passes over, the issue of customization, and proposes a single all-purpose concept of utility: the minimization of one-step ahead forecast error MSE.

Another term for this classic conception of model utility is the Wold decomposition, which breaks a wide class of stochastic processes down in terms of a component that is completely

predictable from its own infinite past, and a second component fully describable in terms of one-step ahead forecast errors. Classical models can then be viewed as attempts to approximate the linear machinery in the Wold decomposition. However, were attention to focus upon an alternative utility, e.g., 10-step ahead forecasting, a different class of models would be suggested, with different apparatus for model selection, fitting, and evaluation.

However, customizing the modeling apparatus to allow for specific applications offers only a partial solution, because model mis-specification is the larger challenge. The full set of LPP solutions for a given time series is greatly constrained once a model is introduced, as only a particular subset of solutions can be obtained. If the model is badly mis-specified, the resulting LPP solution will be inadequate, even if the criteria for model selection are customized. This empirical disfunctionality motivated the genesis of DFA, which essentially provides access to a much wider pool of LPP solutions. Moreover, the basic DFA can be easily modified to allow for direct customization of real-time problems, according to whether users are concerned with timeliness, accuracy, or fidelity to the original signal (called smoothness).

1.1.3 The Scope of MDFA

Our critique of the classic paradigm has several facets. First, there is typically model mis-specification present. Second, the problem has typically not been structured properly, in the sense that the criteria used do not correspond to the relevant LPP, but rather to one-step ahead forecasting. Third, there is no specific customization of the model, in order to account for timeliness and accuracy. These weaknesses are actually linked together.

Model mis-specification is always present; the issue is whether it has a significant impact upon the objectives of analysis. For instance, a given model's mis-specification may have grave repercussions for certain problem structures, while being adequate for other LPPs. The given LPP of interest determines the gravity and impact of model mis-specification. Moreover, in the classic paradigm the one-step ahead forecasting LPP is solved, and it is merely hoped that timeliness and accuracy will be adequate for all users. Model parameters can be tweaked, or tuned, in order to indirectly modify timeliness and accuracy – but the relationships are indirect and often poorly understood. By building the timeliness-accuracy tradeoff directly into the DFA criterion, the optimality of an LPP solution for a customized application is assured.

These topics have been treated in Wildi and McElroy (2016, JTSE) in the case of univariate time series, which discusses at length the basic DFA (Sweave environment: replication). This book presents the generalized treatment of the multivariate LPP in Chapter 4. But before discussing customization in Chapter 7, we discuss the applications of forecasting and nowcasting, as well as the impact of data vintage, in Chapter 5). Then the basic LPP treatment is extended to nonstationary processes in Chapter 16, followed by a discussion of filter constraints (Chapter 6). This treatment is extended to the case of co-integration in Chapter 17. Applications to replicating and enhancing classical model-based approaches and HP/CF-filters are given in Chapter 9, while more sophisticated gain/loss structures are discussed in Chapter 11. Additional topics include inference (Chapter 12), regularization (Chapter 13), data revisions (Chapter 14), mixed-frequency data (Chapter 15), and adaptive filtering (Chapter 18).

1.2 The Style of the Book

This book was generated using Sweave, in accordance with the philosophy of scientific replicability. Throughout the text are portions of R code that can be pasted into an R script and directly run, given that the user has certain packages already installed. This installation is described below.

1.2.1 Setting the Paths

Begin by clearing the workspace:

```
> #rm(list=ls())
```

The R code in various chapters of this book requires installation of the following R packages:

```
> # Load packages: time series and xts
> #library(tseries)
> library(xts)
> # State-space models (will be replicated by MDFA)
> library(dlm)
> # Classic filter designs (be replicated by MDFA)
> library(mFilter)
> # Numerical package
> library(numDeriv)
> # Graphical package for recession-shading (empirical examples based on US-GDP)
> library(tis)
> # Library for tables
> library(Hmisc)
> require(xtable)
> #install.packages("devtools")
> library(devtools)
> # Load MDFA package from github
> devtools::install_github("wiaidp/MDFA")
> # MDFA package
> library(MDFA)
```

US-GDP data for the empirical examples can be retrieved either directly from Quandl (requiring a preliminary user registration) or from a local data folder, which is the default-setting:

```
> # Load fresh data from quandl: T/F
> #   Default-setting is False: the data will be loaded from local data folder
> load_from_quandl <- F
```

Paths to MDFA code, as well as to the US-GDP data, must be provided. It is assumed that the MDFA package is saved to a main folder containing subfolders labeled as DFA, MDFA, model-based, and data. The R code in the book generates pdf graphs that are saved in a separate folder, whose path is specified by *path.out*.

```
> # Set main path
> path.main <- paste(getwd(),"/Sweave/",sep="")
> #path.main <- "C:\\Users\\Tucker\\Documents\\MDFAbook\\"
> # Set paths to subfolders
>   # Path to Latex-folder: all pdfs generated by the R code are filed there
> path.out <- paste(path.main,"Latex/",sep="")
>   # Path to data (US-GDP)
> path.dat <- paste(path.main,"Data/",sep="")
>   # Path to code that is part of MDFA-Legacy project but not part of MDFA package
> path.pgm <- paste(path.main,"R/",sep="")
```

The univariate DFA code is the same as in DFA; all empirical examples are and will be fully compatible.

1.2.2 DFA

We here briefly review the relevant facets of DFA, thereby providing an anchor for the MDFA discussion.

DFT and Periodogram

The Discrete Fourier Transform (DFT) and the periodogram are defined in Sections 2.2 and 2.3 of DFA. The following periodogram function – referred to as *per* below – in the MDFA package replicates these formulae. Note that frequency π is treated differently, depending on whether the sample size is odd or even; also, the value at frequency zero is scaled by $1/\sqrt{2}$, which is explained in later text.

```
> head(per,100)
```

```
1  function (x, plot_T)
2  {
3      len <- length(x)
4      per <- 0:(len/2)
5      DFT <- per
6      for (k in 0:(len/2)) {
7          cexp <- exp((0+1i) * (1:len) * 2 * pi * k/len)
8          DFT[k + 1] <- sum(cexp * x * sqrt(1/(2 * pi * len)))
9      }
10     if (abs(as.integer(len/2) - len/2) < 0.1)
11         DFT[k + 1] <- DFT[k + 1]/sqrt(2)
12     per <- abs(DFT)^2
13     if (plot_T) {
14         par(mfrow = c(2, 1))
15         plot(per, type = "l", axes = F, xlab = "Frequency", ylab = "Periodogram",
```

```
16             main = "Periodogram")
17         axis(1, at = 1 + 0:6 * len/12, labels = c("0", "pi/6",
18             "2pi/6", "3pi/6", "4pi/6", "5pi/6", "pi"))
19         axis(2)
20         box()
21         plot(log(per), type = "l", axes = F, xlab = "Frequency",
22             ylab = "Log-periodogram", main = "Log-periodogram")
23         axis(1, at = 1 + 0:6 * len/12, labels = c("0", "pi/6",
24             "2pi/6", "3pi/6", "4pi/6", "5pi/6", "pi"))
25         axis(2)
26         box()
27     }
28     return(list(DFT = DFT, per = per))
29 }
```

This function will be generalized in the new multivariate setting.

Basic DFA

A simple version of the DFA based on the MSE criterion alone – as proposed in Section 4.1 of DFA – is included in the MDFA package:

```
> # This function computes MSE DFA solutions
> # L is the length of the MA filter,
> # periodogram is the frequency weighting function in the DFA
> # Gamma is the transfer function of the symmetric filter (target) and
> # Lag is the lag-parameter: Lag=0 implies real-time filtering, Lag=L/2
> #     implies symmetric filter
> # The function returns optimal coefficients as well as the transfer
> #     function of the optimized real-time filter
> head(dfa_ms,100)
```

```
1  function (L, periodogram, Lag, Gamma)
2  {
3      periodogram[1] <- periodogram[1]/2
4      K <- length(periodogram) - 1
5      X <- exp(-(0+1i) * Lag * pi * (0:(K))/(K)) * rep(1, K + 1) *
6          sqrt(periodogram)
7      X_y <- exp(-(0+1i) * Lag * pi * (0:(K))/(K)) * rep(1, K +
8          1)
9      for (l in 2:L) {
10         X <- cbind(X, (cos((l - 1 - Lag) * pi * (0:(K))/(K)) +
11             (0+1i) * sin((l - 1 - Lag) * pi * (0:(K))/(K))) *
12             sqrt(periodogram))
```

```
13          X_y <- cbind(X_y, (cos((l - 1 - Lag) * pi * (0:(K))/(K)) +
14              (0+1i) * sin((l - 1 - Lag) * pi * (0:(K))/(K))))
15      }
16      xtx <- t(Re(X)) %*% Re(X) + t(Im(X)) %*% Im(X)
17      b <- as.vector(solve(xtx) %*% (t(Re(X_y)) %*% (Gamma * periodogram)))
18      trffkt <- 1:(K + 1)
19      trffkt[1] <- sum(b)
20      for (k in 1:(K)) {
21          trffkt[k + 1] <- (b %*% exp((0+1i) * k * (0:(length(b) -
22              1)) * pi/(K)))
23      }
24      return(list(b = b, trffkt = trffkt))
25 }
```

This function is nested in the multivariate MDFA, in the sense that the latter can replicate the former perfectly when suitably parametrized; see Section 4.6 below.

Customized DFA

A more general DFA function, called *dfa_analytic*, is proposed in Section 4.3.5 of DFA. Customization and the generic Accuracy-Timeliness-Smoothness (ATS) trilemma are presented in Sections 4.3 and 5 of DFA. This function is included in the MDFA package:

```
> head(dfa_analytic)
```

```
1 function (L, lambda, periodogram, Lag, Gamma, eta, cutoff, i1,
2     i2)
3 {
4     periodogram[1] <- periodogram[1]/2
5     lambda <- abs(lambda)
6     eta <- abs(eta)
```

The additional control parameters *lambda*, *eta* allow for customization of the filter, as discussed below in Chapter 7. The Boolean *i1* and *i2* can enforce useful filter constraints; see Chapter 6. This function is also encompassed by the MDFA.

1.2.3 MDFA

The R code for MDFA is more sophisticated than that of the DFA, and is correspondingly more complex and lengthy. As for the DFA package, the MDFA code can be sourced. We here briefly review the corresponding pieces.

Data Matrix

All time series are collected in a data-*matrix*, say X, which is organized as follows:

- the first column $X[,1]$ of X always corresponds to the target series: the target series $X[,1]$ is the time series to be forecasted, nowcasted or backcasted.

- Columns 2, 3, ... of X are allocated to the explanatory variables (more than one in a multivariate setting). If the target series is part of the set of explanatory variables (it does not have to be), then it must be assigned a specific column – by convention always the second one – in X, i.e., in this case the target series is entered twice, in the first column (target) and in the second column (explanatory data).

Example. Suppose we study a two-dimensional signal extraction problem, whereby the target series (first column) is part of the set of explanatory variables:

```
> set.seed(1)
> len <- 100
> target <- arima.sim(list(ar=0.9),n=len)
> explanatory_2 <- target+rnorm(len)
> explanatory <- cbind(target,explanatory_2)
> x <- cbind(target,explanatory)
> dimnames(x)[[2]] <- c("target","explanatory 1","explanatory 2")
> head(x)

       target explanatory 1 explanatory 2
[1,] 1.703613      1.703613     0.3191863
[2,] 1.398197      1.398197     3.2674879
[3,] 3.659995      3.659995     4.0850957
[4,] 3.254756      3.254756     3.0161086
[5,] 3.619020      3.619020     4.6775026
[6,] 3.285120      3.285120     4.1715424
```

For a one-step ahead forecast LPP, we might consider lagging both the explanatory variables:

```
> x<-cbind(x[,1],lag(x[,2:3],-1))
> dimnames(x)[[2]]<-c("target","lagged explanatory 1","lagged explanatory 2")
> head(x)

       target lagged explanatory 1 lagged explanatory 2
[1,] 1.703613                   NA                   NA
[2,] 1.398197             1.703613            0.3191863
[3,] 3.659995             1.398197            3.2674879
[4,] 3.254756             3.659995            4.0850957
[5,] 3.619020             3.254756            3.0161086
[6,] 3.285120             3.619020            4.6775026
```

By adopting the frequency-domain methods of this book, we can generalize this construction and avoid the introduction of missing values (denoted by NA in R). □

DFT

In contrast to the univariate DFA, where the LPP can be expressed in terms of the periodogram, the multivariate case requires the DFT of each time series in order to account for cross-sectional dependencies. These DFTs are complex-valued quantities, and the angular portion of the cross-spectrum provides information about the relative phase-shift of each explanatory time series. In the univariate case the relative phase-shift is irrelevant, because the target series and the explanatory series are identical. The scope of the method is extended in order to cover the mixed-frequency case, which is discussed in Chapter 15. Another facet, is that we allow for the possibility of integrated processes; see Chapter 16. In order to illustrate some of the new features we briefly look at the main DFT function called *spec_comp*:

```
> spec_comp

function (insamp, x, d)
{
    if (d == 1) {
        weight_func <- periodogram_bp(diff(x[1:insamp, 1]), 1,
            insamp - 1)$fourtrans
        if (length(weight_func) > 1) {
            for (j in 2:ncol(x)) {
                per <- periodogram_bp(diff(x[1:insamp, j]), 1,
                  insamp - 1)$fourtrans
                weight_func <- cbind(weight_func, per)
            }
        }
    }
    else {
        weight_func <- periodogram_bp(x[1:insamp, 1], 0, insamp)$fourtrans
        if (length(weight_func) > 1) {
            for (j in 2:ncol(x)) {
                per <- periodogram_bp(x[1:insamp, j], 0, insamp)$fourtrans
                weight_func <- cbind(weight_func, per)
            }
        }
    }
    colnames(weight_func) <- colnames(x)
    return(list(weight_func = weight_func))
}
<bytecode: 0x0000000023d07d68>
<environment: namespace:MDFA>
```

The inner loop tracks the columns of the data matrix X and the DFTs are stored in a matrix called *weight_func*, which is returned by the function. The matrix *weight_func* collects all DFTs;

the target series is always in the first column, whereas the DFTs of the explanatory series are in columns 2, 3, ... The function *periodogram_bp*, called in the above loop, is slightly more general than the DFA function *per* proposed in the previous section. In particular, it can handle various integration orders as well as seasonal peculiarities.

1.2.4 Using MDFA

A Versatile User Interface

MDFA is a generic forecast and signal extraction paradigm. Besides its capacity to replicate classical time series approaches, MDFA possesses unique features such as customization and regularization (Chapter 13); it can treat data revisions (Chapter 14), mixed-frequency problems (Chapter 15), and non-stationarity (Chapters 16 and 17. Accordingly, the user interface is more sophisticated than the precediing DFA package. Consider the head of the main estimation routine:

```
> head(mdfa_analytic)
```

```
1 function (L, lambda, weight_func, Lag, Gamma, eta, cutoff, i1,
2     i2, weight_constraint, lambda_cross, lambda_decay, lambda_smooth,
3     lin_eta, shift_constraint, grand_mean, b0_H0, c_eta, weight_structure,
4     white_noise, synchronicity, lag_mat, troikaner)
5 {
6     lambda <- abs(lambda)
```

Arguments such as *weight_func* (discussed above), the filter length (L), and the target specification *Gamma* are straightforward. But there are numerous additional control parameters: the relevance and the modus operandi of these will be discussed in this book.

Default Settings

For convenience, we store a so-called default setting of the parameters in a file called *control_default*. First we define the data (initialize the DFT matrix) and specify the filter length:

```
> weight_func <- matrix(rep(1:6,2),ncol=2)
> L <- 2
```

Given these two entries (DFT and filter length), the default-settings are as follows:

```
> d<-0
> lin_eta<-F
> lambda<-0
> Lag<-0
> eta<-0
> i1<-F
> i2<-F
> weight_constraint<-rep(1/(ncol(weight_func)-1),ncol(weight_func)-1)
```

```
> lambda_cross<-lambda_smooth<-0
> lambda_decay<-c(0,0)
> lin_expweight<-F
> shift_constraint<-rep(0,ncol(weight_func)-1)
> grand_mean<-F
> b0_H0<-NULL
> c_eta<-F
> weights_only<-F
> weight_structure<-c(0,0)
> white_noise<-F
> synchronicity<-F
> cutoff<-pi
> lag_mat<-matrix(rep(0:(L-1),ncol(weight_func)),nrow=L)
> troikaner<-F
```

This particular configuration will be used extensively in Chapter 4; it corresponds to the basic MSE criterion (i.e., no customization) without regularization, without design constraints, and without any *a priori* knowledge. Also, this configuration presumes a common identical sampling frequency (i.e., no mixed frequency data) and the absence of data revisions. The default settings can be obtained by sourcing the corresponding R file:

```
> source(file=paste(path.pgm,"control_default.r",sep=""))
```

For later use we source a convenient plotting function:

```
> source(file=paste(path.pgm,"mplot_func.r",sep=""))
```

Selected Calls: Classic MSE, Customization and Regularization

Selected calls of the classic MSE criterion – as well as calls utilizing the customization or regularization features – are available through dedicated functions in the MDFA package:

```
> head(MDFA_mse)
```

```
1 function (L, weight_func, Lag, Gamma)
2 {
3     cutoff <- pi
4     lin_eta <- F
5     lambda <- 0
6     eta <- 0
```

```
> head(MDFA_mse_constraint)
```

```
1 function (L, weight_func, Lag, Gamma, i1, i2, weight_constraint,
2     shift_constraint)
3 {
```

```
4     cutoff <- pi
5     lin_eta <- F
6     lambda <- 0
```

```
> head(MDFA_cust)
```

```
1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta)
2 {
3     lin_eta <- F
4     weight_constraint <- rep(1/(ncol(weight_func) - 1), ncol(weight_func) -
5         1)
6     lambda_cross <- lambda_smooth <- 0
```

```
> head(MDFA_cust_constraint)
```

```
1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta, i1,
2     i2, weight_constraint, shift_constraint)
3 {
4     lin_eta <- F
5     lambda_cross <- lambda_smooth <- 0
6     lambda_decay <- c(0, 0)
```

```
> head(MDFA_reg)
```

```
1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta, lambda_cross,
2     lambda_decay, lambda_smooth, troikaner = F)
3 {
4     lin_eta <- F
5     weight_constraint <- rep(1/(ncol(weight_func) - 1), ncol(weight_func) -
6         1)
```

```
> head(MDFA_reg_constraint)
```

```
1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta, lambda_cross,
2     lambda_decay, lambda_smooth, i1, i2, weight_constraint, shift_constraint,
3     troikaner = F)
4 {
5     lin_eta <- F
6     lin_expweight <- F
```

The heads of the corresponding functions differ in the number of additional arguments available when going from specific (MSE) to generic (reg). The following chapters of the book provide an understanding of the use of these functions.

Chapter 2

Linear Prediction Problems

2.1 Background on Stationary Vector Time Series

The reader should have a basic familiarity with multivariate time series analysis, such as that provided by Lütkepohl (2007). Our focus is on discrete-time stochastic processes taking values in $\mathbb{R}^n$, and such will be denoted $\{X_t\}$, i.e., a vector time series. Each X_t for a particular $t \in \mathbb{Z}$ is a random vector with n components, and the jth component will be denoted $X_{t,j}$ for $1 \leq j \leq n$. This can also be written as $X_{t,j} = e_j' X_t$, where e_j is the jth unit vector in $\mathbb{R}^n$. The union of these unit vectors is the $n \times n$ identity matrix, denoted 1_n.

In this book we are focused upon square integrable random variables, so that the classic Hilbert space projection theory (see, for example, Brockwell and Davis (1991)) can be applied. Occasionally, we consider vector time series $\{Y_t\}$ or $\{Z_t\}$, in which case the same conventions apply.When $\{X_t\}$ is weakly stationary, its autocovariance function (acf) is defined for $h \in \mathbb{Z}$ via

$$\Gamma(h) = \mathrm{Cov}[X_{t+h}, X_t],$$

which does not depend upon t by the stationarity assumption. Recall that $\Gamma(-h) = \Gamma(h)'$, and clearly

$$\Gamma_{jk}(h) = \mathrm{Cov}[X_{t+h,j}, X_{t,k}]$$

for $1 \leq j, k \leq n$. The spectral density is a complex matrix-valued function of $\omega \in [-\pi, \pi]$, defined as the Fourier Transform (FT) of the acf:

$$F(\omega) = \sum_{h \in \mathbb{Z}} \Gamma(h)\, z^h,$$

where we use the shorthand $z = e^{-i\omega}$. Clearly,

$$F(-\omega) = \sum_{h \in \mathbb{Z}} \Gamma(h)\, z^{-h} = \sum_{h \in \mathbb{Z}} \Gamma(-h)\, z^h = \sum_{h \in \mathbb{Z}} \Gamma(h)'\, z^h = F(\omega)',$$

which shows that the spectral density function (sdf) is Hermitian. In addition, its eigenvalues (for each ω) are real and non-negative. Given a bounded sdf (i.e., each F_{jk} has bounded modulus as

a function of ω), the acf can be recovered via inverse FT:

$$\Gamma(h) = \langle F \rangle_h = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega)\, e^{i\omega h}\, d\omega, \tag{2.1}$$

which uses the bracket notation to define the average integral of a function (of ω) multiplied by $e^{i\omega h} = z^{-h}$.

The lag operator on a time series is denoted L, and is defined via the action

$$LX_t = X_{t-1}.$$

Powers of L are defined analogously, with $L^0 = 1$ (an operator identity) and negative powers yielding forward time shifts, i.e., leads. Matrix polynomials of L yield new operators that act upon a time series using the linearity principal. Thus, if $A(L) = \sum_{j=0}^{a} A_j\, L^j$ for $n \times n$ matrices A_j, then

$$A(L)\, X_t = \sum_{j=0}^{a} A_j\, X_{t-j}.$$

For many applications in this book, the polynomials are actually scalar, and can be interpreted as having coefficients A_j given by an identity matrix 1_n multiplied by a scalar coefficient a_j.

The spectral representation of a stationary square integrable vector time series is particularly useful. Assuming that $\mathbb{E}X_t = 0$ (the zero vector in $\mathbb{R}^n$) so that no fixed effects are present, we describe the stochastic process via

$$X_t = \int_{-\pi}^{\pi} e^{i\omega t}\, \mathcal{Z}(d\omega), \tag{2.2}$$

which is a stochastic integral computed with a Stieltjes measure $\mathcal{Z}(d\omega)$. This is an orthogonal increments process, which mean $\mathcal{Z}$ is a random measure defined on $[-\pi, \pi]^n$ that maps disjoint sets to independent random variables. The actual distribution of the random measure is not our concern, but the particular orthogonal increments process associated with $\{X_t\}$ has the property that

$$\mathrm{Cov}[\mathcal{Z}(d\omega), \mathcal{Z}(d\xi)] = (2\pi)^{-1}\, F(\omega)\, d\omega\, 1_{\{\omega=\xi\}}$$

where 1_A is the indicator for the set A. (Also recall that for complex variables, a covariance involves conjugation of the second argument.) This validates the expression

$$\mathrm{Cov}[X_{t+h}, X_t] = \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} e^{i\omega(t+h)}\, e^{-i\xi t}\, \mathrm{Cov}[\mathcal{Z}(d\omega), \mathcal{Z}(d\xi)] = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\omega)\, e^{i\omega h}\, d\omega = \Gamma(h).$$

As an example – that furnishes a basic building block for subsequent processes – we have *white noise*, which refers to a mean zero $\{X_t\}$ where F is constant, i.e., $F(\omega) = \Sigma$ for all ω, where Σ is real and symmetric and non-negative definite. Clearly, $\Gamma(h) = 0$ for $h \neq 0$ and $\Gamma(0) = \Sigma$. We denote this type of process with the notation $\mathrm{WN}(\Sigma)$.

The advantage of the spectral representation is that it quickly facillitates the understanding of linear filtering in the frequency domain. A multivariate filter maps one vector time series to another, and for now we suppose that both input and output belong to $\mathbb{R}^n$. Linear filters can be expressed as matrix Laurent series in L:

$$\Psi(L) = \sum_{\ell \in \mathbb{Z}} \psi(\ell)\, L^{\ell},$$

where each $\psi(\ell)$ is an $n \times n$ matrix. Individual entries of the matrix are denoted $\psi_{jk}(\ell)$, for $1 \le j,k \le n$. We also use the notation $[\Psi(L)]_r^s$ to denote $\sum_{\ell=r}^s \psi(\ell)\, L^\ell$. In the special case that $\Psi(L)$ is a power series in L, the only nonzero coefficients are for $\ell \ge 0$, so that no negative powers of L are featured, i.e., the filter only utilizes present and past data. Such a filter is called a *concurrent* filter.

The action of a linear filter on a weakly stationary time series, expressed in terms of the spectral representation, is

$$Y_t = \Psi(L)\, X_t = \sum_{\ell\in\mathbb{Z}} \psi(\ell)\, X_{t-\ell} = \sum_{\ell\in\mathbb{Z}} \psi(\ell) \int_{-\pi}^{\pi} e^{i\omega(t-\ell)}\, \mathcal{Z}(d\omega) = \int_{-\pi}^{\pi} e^{i\omega t}\, \Psi(e^{-i\omega})\, \mathcal{Z}(d\omega).$$

So the output time series $\{Y_t\}$ has orthogonal increments process $\Psi(z)\, \mathcal{Z}(d\omega)$, and in particular its sdf is

$$\Psi(z)\, F(\omega)\, \Psi(z)^*,$$

where $*$ denotes the conjugate transpose. Thus, it is very natural to analyze a filter in terms of the function $\Psi(e^{-i\omega})$, which is called the *frequency response function* (frf). In the scalar case, an frf can be further dissected via the polar decomposition of a complex number, yielding its *gain function* (the modulus) and the *phase function* (its angular portion). Note that the coefficients are recovered from the frf via the inverse FT:

$$\psi(\ell) = \langle \Psi(e^{-i\cdot}) \rangle_\ell.$$

It is well-known from Fourier theory that the degree of smoothness of a function at $\omega = 0$ corresponds to the degree of decay in the coefficients of its inverse FT. In particular, when the frf is smooth and flat in a neighborhood of the origin then the matrix norm of the coefficients $\psi(\ell)$ decays rapidly as $|\ell| \to \infty$. Conversely, discontinuities in the frf indicates slowly decaying coefficients.

Datasets are typically available as a finite set of contiguous regular measurements, denoted $\{x_1, x_2, \ldots, x_T\}$, where T is the length of sample. The data is viewed as a realization of the corresponding random vectors $\{X_1, X_2, \ldots, X_T\}$, or alternatively as a time window of the sample path $\{x_t\}$ corresponding to times $1, 2, \ldots, T$. Applying the *vec* operator to such a sample yields the full vector $\underline{X}$, which is given by

$$\underline{X} = \mathrm{vec}[X_1, X_2, \ldots, X_T].$$

The covariance matrix of this nT-dimensional random vector, in the stationary case, is block Toeplitz, Each block is $n \times n$, and the stth such block, for $1 \le s,t \le T$, is given by $\Gamma(s-t)$. Also, from the sample we can compute the *Discrete Fourier Transform* (DFT) via

$$\widetilde{X}(\omega) = T^{-1/2} \sum_{t=1}^{T} z^t\, X_t. \tag{2.3}$$

This can be computed for any $\omega \in [-\pi, \pi]$, though if we restrict to Fourier frequencies – of the form $2\pi j/T$ for integer j – then the various real and imaginary components of the DFT will be asymptotically uncorrelated, and also asymptotically normal. The multivariate *periodogram* is defined to be the rank one Hermitian matrix

$$\widehat{F}(\omega) = \widetilde{X}(\omega)\, \widetilde{X}(\omega)^*. \tag{2.4}$$

The periodogram furnishes a basic estimate of the spectral density F of the process. There is an empirical version of (2.1), where the periodogram is mapped to the sample autocovariance:

$$\widehat{\Gamma}(h) = \langle \widehat{F} \rangle_h = \frac{1}{2\pi} \int_{-\pi}^{\pi} \widehat{F}(\omega)\, e^{i\omega h}\, d\omega. \tag{2.5}$$

This is easily verified using the definition of sample autocovariance

$$\widehat{\Gamma}(h) = T^{-1} \sum_{t=1}^{T-h} X_{t+h}\, X_t'$$

for $h \geq 0$, and with $\widehat{\Gamma}(h) = \widehat{\Gamma}(-h)'$ for $h < 0$. Conversely, the periodogram is the FT of the sample autocovariances:

$$\widehat{F}(\omega) = \sum_{|h|<T} \widehat{\Gamma}(h)\, e^{-i\omega h}. \tag{2.6}$$

2.2 MSE Optimal Prediction Problems

2.2.1 The Linear Prediction Problem

We define the class of real-time estimation problems considered in this book. This chapter focuses upon the case of weakly stationary vector time series, but Chapter 16 makes extensions to difference stationary processes.

Definition 1 A **target** is defined to be the output of any known linear filter acting on the data process, i.e., $\{Y_t\}$ is a target time series corresponding to a given filter $\Psi(L)$ acting on a given observed time series $\{X_t\}$ if and only if we can write for all integers t

$$Y_t = \Psi(L) X_t.$$

We say that $\{Y_t\}$ is a **scalar target** if $\Psi(L)$ is a $1 \times n$-dimensional filter.

We are only interested in scalar targets. The reason is that if $\{Y_t\}$ is multivariate, we can treat each component series $\{Y_{t,j}\}$ for $1 \leq j \leq n$ in turn, so that without loss of generality we can just give the treatment for the scalar case.

Example 1: Multi-step Ahead Forecasting. Suppose that our goal is to forecast one of the component series h steps ahead, where $h \geq 1$ is the given *forecast lead*. Here, suppose that the series of interest is the first component, so that

$$Y_t = X_{t+h,1}$$

for all $t \in \mathbb{Z}$. This is indeed a scalar target, setting $\Psi(L) = L^{-h}\, e_1'$. That is, each $\psi(\ell)$ is a $1 \times n$ row vector, each of which are zero except $\psi(-h)$, which is given by e_1'.

Example 2: Ideal Low-Pass. In order to estimate a trend from a given series, conceptually we wish to screen out all the higher frequency components in the data. With reference to the spectral representation, if $\Psi(z)$ is zero for all ω in a band of the higher frequencies, then $\{Y_t\}$ will only be composed of low frequency stochastic sinusoids. The simplest way to achieve such an output is to design the frf as an indicator function, involving a steep cutoff of noise frequencies; see Baxter and King (1999). This is viewed by some as the best possible definition of trend, and hence the filter is called the ideal low-pass. For scalar target, we have

$$\Psi(z) = 1_{[-\mu,\mu]}(\omega)\, e_1'$$

for some cutoff $\mu \in (0, \pi)$ that separates the pass-band from the stop-band. To understand this terminology of pass-band and stop-band, observe that the spectral representation of the scalar target is

$$Y_t = \int_{[-\mu,\mu]} e^{i\omega t}\, e_1' \mathcal{Z}(d\omega).$$

Here, the stochastic integration only includes frequencies in the pass-band $[-\mu, \mu]$, and all content belonging to the stop-band has been eliminated. The coefficients are given by

$$\psi(\ell) = \frac{\sin(\ell\mu)}{\pi\ell}\, e_1'$$

for $\ell \neq 0$ and $\psi(0) = \mu/\pi\, e_1'$.

Example 3: HP Low-pass. The Hodrick-Prescott (HP) filter (Hodrick and Prescott, 1997) is a low-pass filter appropriate for producing trends. A multivariate version of the HP low-pass (or just HP), associated with trend-irregular structural models, was proposed in McElroy and Trimbur (2015); the frf is given by

$$\Psi(z) = \Sigma_\mu \left(\Sigma_\mu + |1 - z|^4\, \Sigma_\iota \right)^{-1}$$

in the case that the matrices Σ_μ and Σ_ι have full rank. When Σ_μ has reduced rank, an alternative expression is available, but note that the frf is a continuous matrix-valued function of ω. The matrices Σ_μ and Σ_ι have an interpretation in terms of the econometric concept of trend co-integration, which is further explored in Chapter REF???. It is always assumed that Σ_ι has full rank, and hence we can rewrite as

$$\Psi(z) = Q \left(Q + |1 - z|^4\, 1_n \right)^{-1}$$

with $Q = \Sigma_\mu\, \Sigma_\iota^{-1}$ representing a matrix *signal-to-noise ratio*, or snr. This formula generalizes the univariate HP filter, which has frf

$$\Psi(z) = \frac{q}{q + |1 - z|^4}$$

for snr parameter $q > 0$. Small values of q correspond to trends that are buried in volatile white noise, and thus require much smoothing to recover. The filter perfectly reflects this need, because a small q indicates a steep drop in the frf (which takes value one at $\omega = 0$) as ω is increased from zero, and hence the filter coefficients decay slowly. Conversely, higher values of q – corresponding to highly salient trends – yield an frf that equals unity in a large neighborhood of the origin, with

coefficients that decay swiftly, indicating that little smoothing is needed to discover the trend. These observations carry over to the multivariate case, though we judge the size of the snr via a matrix norm (such as the maximum eigenvalue) of Q. Some of these eigenvalues can be zero, corresponding to the case that Σ_μ has reduced rank – this has the effect of generating trends that are collinear. In the case of a scalar target, where we seek a trend for the first input series $\{X_{t,1}\}$, we have

$$\Psi(z) = e_1' \, Q \left(Q + |1-z|^4 \, 1_n \right)^{-1}.$$

There are no known analytical formulas for the coefficients in the multivariate case, although in the univariate case they are available in McElroy (2008).

Example 4: HP High-pass. While the HP filter is used to extract trends, the residual is thought to measure the business cycle along with higher frequency oscillations in the data. Thus, taking the identity minus the HP low-pass yields the HP high-pass filter:

$$\Psi(z) = \left(Q + |1-z|^4 \, 1_n \right)^{-1} |1-z|^4.$$

The presence of the term $|1-z|^4$ indicates differencing by the $(1-L)^2$ and $(1-L^{-1})^2$; thus the HP high-pass will annihilate cubic polynomials, and generally reduces high order stochastic trends to stationarity.

As we see from these examples, the targets of real-time signal extraction are features of the stochastic process that are of interest to a particular user. Some scalar targets depend upon only a single component of the time series (Examples 1 and 2), whereas others may be defined in terms of all the components (Examples 3 and 4). However, these targets represent an ideal feature of the time series that typically we cannot compute in real-time.

Real-time refers to time present, wherein we have access to present and past information, but have great uncertainty about the future. This is an essential feature of human existence. Time series methodology provides tools to model and understand the flow of information from past to present to future, with the implicit viewpoint that whereas causality is to some degree present – past events have a causative impact on future events, but not vice versa – there are other facets governing present and future outcomes that are not traceable to a particular variable's past. In other words, knowing the past values of a component time series $\{X_{t,1}\}$ is not sufficient to flawlessly determine its future values. However, having other explanatory variables in play can reduce the uncertainty of the future; taking n higher, we may be able to reduce the errors in forecasts.

The concept of *Granger causality* can be used to parse these notions mathematically. We may consider other component series $\{X_{t,j}\}$ for $j \geq 2$ useful for determining the future of $\{X_{t,1}\}$ if the one-step ahead forecast mean square error (MSE) is reduced, in which case we say that Granger causality is present. In such a scenario it can be proved that the one-step ahead forecast MSE arising from utilizing $\{X_{t,1}\}$ alone is greater than that obtained using the additional series. Hence, there is benefit to increasing n with additional ancillary series so long as they are helpful for forecasting. For real-time estimation problems, we seek to determine the best possible estimates of a target given a relevant collection of ancillary series.

More formally, the real-time estimation problem is concerned with projecting the target Y_t onto the available data $X_{t:} = \{X_t, X_{t-1}, \ldots\}$, i.e., the semi-infinite past. This formulation presumes that we have access to relevant ancillary series, and that we have access to all present and past values. In practice, databases only extend back a few decades, and the infinitely remote past represents merely an idyll useful for mathematical simplicity. The linear estimation problem seeks a linear estimate of the form

$$\widehat{Y}_t = \sum_{\ell \geq 0} \widehat{\psi}(\ell)\, X_{t-\ell} = \widehat{\Psi}(L)\, X_t,$$

which shows that we seek a linear (time-invariant) concurrent filter $\widehat{\Psi}(L)$, applied to $\{X_t\}$. We desire that the error in approximating the target with the available data be small with respect to MSE. If $\{X_t\}$ were Gaussian, we could view our estimate as the conditional expectation $\widehat{Y}_t = \mathbb{E}[Y_t | X_{t:}]$, with the coefficients $\{\widehat{\psi}(\ell)\}$ selected to minimize the MSE of the approximation error $Y_t - \widehat{Y}_t$. However, in our treatment in this book we do not presume Gaussian structure, and are not concerned with conditional expectations *per se*; rather, we seek linear solutions with minimal MSE.

Definition 2 The **Linear Prediction Problem** (LPP) seeks the minimal MSE linear estimate that solves the real-time estimation problem arising from a scalar target. That is, the LPP involves determining causal $\widehat{\Psi}(L)$ such that the prediction error

$$Y_t - \widehat{Y}_t = \left[\Psi(L) - \widehat{\Psi}(L)\right] X_t$$

has mean zero and minimal MSE.

Example 1: Multi-step Ahead Forecasting. The LPP corresponds to optimal h-step forecasting, and the forecast error is $[L^{-h} e_1' - \widehat{\Psi}(L)]\, X_t$. Note that although $\Psi(L)$ only involves one component series $\{X_{t,1}\}$, the real-time concurrent filter $\widehat{\Psi}(L)$ can involve all n component series.

Example 3: HP Low-pass. The LPP attempts to determine an optimal real-time trend estimate, where the target trend – sometimes called the historical trend – is defined through the HP low-pass filter. Here, both the target filter $\Psi(L)$ and the real-time concurrent filter $\widehat{\Psi}(L)$ involve all n component series.

2.2.2 Solution to the Linear Prediction Problem

When the data process is itself causal and linear, it is possible to give an explicit solution to the LPP in terms of the Wold decomposition (Brockwell and Davis, 1991). All purely nondeterministic weakly stationary (mean zero) processes have a Wold decomposition $X_t = \Theta(L)\epsilon_t$, where $\{\epsilon_t\}$ is WN(Σ) and $\Theta(L) = \sum_{\ell \in \mathbb{Z}} \theta(\ell)\, L^\ell$. When $\theta(\ell) = 0$ for all $\ell < 0$, the process is called *causal*. First, the error in the LPP is denoted $E_t = Y_t - \widehat{Y}_t$, which is clearly mean zero and covariance stationary, in fact having spectral representation

$$E_t = \int_{-\pi}^{\pi} e^{i\omega t} \left[\Psi(z) - \widehat{\Psi}(z)\right] \mathcal{Z}(d\omega). \tag{2.7}$$

With these preliminaries, we can state the solution to the LPP.

Proposition 1 *Suppose that $\{X_t\}$ is mean zero and weakly stationary with spectral representation (2.2), and moreover is causal, expressed as $X_t = \Theta(L)\,\epsilon_t$. Then the solution to the LPP posed by a scalar target $Y_t = \Psi(L)\,X_t$ is given by*

$$\widehat{\Psi}(L) = \sum_{\ell \geq 0} \psi(\ell)\, L^{\ell} + \sum_{\ell < 0} \psi(\ell)\, [\Theta(L)]_{-\ell}^{\infty}\, L^{\ell}\, \Theta(L)^{-1}. \tag{2.8}$$

Moreover, the minimal MSE is given by

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \sum_{\ell > 0} \psi(-\ell)\, z^{-\ell} [\Theta(z)]_0^{\ell-1}\, \Sigma\, [\Theta(z)]_0^{\ell-1^*} \sum_{\ell > 0} \psi(-\ell)\, z^{\ell}\, d\omega. \tag{2.9}$$

Proof of Proposition 1. In order for a linear solution to be MSE optimal, it is sufficient that the resulting error process be uncorrelated with the data $X_{t:}$. If we can show that the real-time signal extraction error process $\{E_t\}$ depends only on future innovations, then by the causality of $\{X_t\}$ the error process must be uncorrelated with $X_{t:}$, establishing optimality. The filter error of the putative solution is given by

$$\begin{aligned} \Psi(L) - \widehat{\Psi}(L) &= \sum_{\ell < 0} \psi(\ell)\, L^{\ell} \left(1 - [\Theta(L)]_{-\ell}^{\infty}\, \Theta(L)^{-1}\right) \\ &= \sum_{\ell < 0} \psi(\ell)\, L^{\ell}\, [\Theta(L)]_0^{-(\ell+1)}\, \Theta(L)^{-1}. \end{aligned}$$

Applying this to $\{X_t\}$ yields

$$E_t = \sum_{\ell=1}^{\infty} \psi(-\ell)\, [\Theta(L)]_0^{\ell-1}\, \epsilon_{t+\ell}.$$

Noting that $[\Theta(L)]_0^{\ell-1}$ is an order $\ell - 1$ polynomial in L, and is applied to $\epsilon_{t+\ell}$, it is apparent that E_t is a linear function of future innovations $\{\epsilon_{t+1}, \epsilon_{t+2}, \ldots\}$. Computing the variance of E_t yields the expression for the minimal MSE. □

Remark 1 The formula (2.9) gives us a lower bound on the MSE when we use sub-optimal proxies for $\widehat{\Psi}(L)$.

As indicated by Remark 1, the result of Proposition 1 is chiefly useful when we know $\Theta(L)$. However, this is rarely the case in practice: a classical parametric approach involves formulating a time series model, fitted using the Gaussian likelihood, and finally computing the LPP solution in terms of the fitted model. Alternatively, one might consider fitting a specified model such that the LPP MSE is minimized. A more broad nonparametric approach involves considering classes of concurrent filters and directly minimizing the LPP MSE over this class – this is the methodology of Direct Filter Analysis (DFA).

Illustration 1: VAR(1). Consider an LPP where the true process $\{X_t\}$ is a Vector Autoregression (VAR) of order 1. This process can be described via

$$X_t = \Phi\, X_{t-1} + \epsilon_t$$

for a matrix Φ with all eigenvalues bounded by one in modulus (Lütkepohl, 2007). It is known that the VAR(1) has the causal representation $\Theta(L) = (1 - \Phi\, L)^{-1}$. Because for $\ell < 0$

$$[\Theta(L)]_{-\ell}^{\infty} = \sum_{j=-\ell}^{\infty} \Phi^j\, L^j = \Phi^{-\ell}\, L^{-\ell}\, (1 - \Phi\, L)^{-1},$$

we find that (2.8) reduces to

$$\widehat{\Psi}(L) = \sum_{\ell \geq 0} \psi(\ell)\, L^{\ell} + \sum_{\ell < 0} \psi(\ell)\, \Phi^{-\ell}.$$

The second term in this expression we denote by $A_{\Psi}(\Phi)$. Hence, the optimal concurrent filter is determined by applying the filter to past data and modifying the present weight $\psi(0)$ by adding the quantity $A_{\Psi}(\Phi)$. In the case of h-step ahead forecasting of the first time series (Example 1), $\widehat{\Psi}(L) = A_{\Psi}(\Phi) = e_1'\, \Phi^h$.

2.3 Model Fitting via LPP MSE Minimization

Here we study the mechanics of fitting a parametric model such that the LPP MSE is minimized. In the case of the one-step ahead forecasting MSE, this is related to Whittle estimation of vector time series models (c.f., Taniguchi and Kakizawa (2000)). We will focus on the class of separable causal linear models, wherein the innovation variance Σ is governed by a separate set of parameters from those describing the power series $\Theta(L)$. The model is essentially described through a particular class of power series $\Theta_{\vartheta}(L)$, parameterized by a vector ϑ belonging to some model parameter manifold. Hence the model sdf is

$$F_{\vartheta}(\omega) = \Theta_{\vartheta}(z)\, \Sigma\, \Theta_{\vartheta}(z)^*.$$

However, the model may be misspecified: the process' sdf is denoted $\widetilde{F}$, and may not belong to the model class. The goal of model fitting is to determine ϑ such that F_{ϑ} is a good approximation to $\widetilde{F}$. Clearly, knowing ϑ does not fully determine F_{ϑ} because Σ remains unknown; however, the methods described below provide for estimates of Σ in terms of ϑ and the process. From the proof of Proposition 1 we know that the filter error satisfies

$$E_t = \sum_{\ell=1}^{\infty} \psi(-\ell)\, L^{-\ell}\, [\Theta(L)]_0^{\ell-1}\, \Theta_{\vartheta}(L)^{-1}\, X_t.$$

In other words, we have an error filter $\Xi_{\vartheta}(L)\, \Theta_{\vartheta}(L)^{-1}$, where

$$\Xi_{\vartheta}(L) = \sum_{\ell=1}^{\infty} \psi(-\ell)\, L^{-\ell}\, [\Theta(L)]_0^{\ell-1},$$

such that for any choice of ϑ we can compute filter errors $\{E_t\}$. Note that these are not in general to be interpreted as residuals, and they need not be white noise. But we can seek to minimize their variance. In practice, the calculation of such filter errors may require a truncation of the error filter, because the finite sample $X_1, X_2, \ldots, X_T$ is available, not the entire infinite past. The

error filter is $1 \times n$, and for any ϑ and any Hermitian function G we can compute

$$J_\Psi(\vartheta, G) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \Xi(z)\, \Theta_\vartheta(z)^{-1}\, G(\omega)\, \Theta_\vartheta(z)^{-1^*}\, \Xi(z)^*\, d\omega = \mathrm{tr}\{\langle G\, K_\vartheta \rangle_0\}$$
$$K_\vartheta(z) = \Theta_\vartheta(z)^{-1^*}\, \Xi(z)^*\, \Xi(z)\, \Theta_\vartheta(z)^{-1}.$$

Then $\mathrm{Var}[E_t] = J_\Psi(\vartheta, \widetilde{F})$. As we seek to minimize the variability in the filter errors, we can take $J_\Psi(\vartheta, \widetilde{F})$ as our criterion function. However, this will only determine the proximity of the model to the true sdf, which is unavailable to us – in order to compute actual parameter estimates, we must utilize the data to approximate the true sdf. There are basic results giving asymptotic normality for simple functionals of the periodogram, and therefore this crude estimator of the true sdf is sufficient for our purposes. We propose $J_\Psi(\vartheta, \widehat{F})$ as an estimator of $J_\Psi(\vartheta, \widetilde{F})$, and intend that the respective minimizers have the same relationship. Namely, if $\vartheta(\widetilde{F})$ is the unique minimizer of $J_\Psi(\vartheta, \widetilde{F})$ and $\vartheta(\widehat{F})$ is the unique minimizer of $J_\Psi(\vartheta, \widehat{F})$, then $\vartheta(\widehat{F})$ is an estimator of $\vartheta(\widetilde{F})$, which is called the *pseudo-true value* (PTV). From the PTV and estimator, we can also compute the innovation covariance matrix by the formulas

$$\Sigma(\widetilde{F}) = \langle \Theta_{\vartheta(\widetilde{F})}(z)^{-1}\, \widetilde{F}\, \Theta_{\vartheta(\widetilde{F})}(z)^{-1^*} \rangle_0$$
$$\Sigma(\widehat{F}) = \langle \Theta_{\vartheta(\widehat{F})}(z)^{-1}\, \widehat{F}\, \Theta_{\vartheta(\widehat{F})}(z)^{-1^*} \rangle_0.$$

In the special case that the model is correctly specified, there exists some $\widetilde{\vartheta}$ and $\widetilde{\Sigma}$ such that $\widetilde{F}(\omega) = \Theta_{\widetilde{\vartheta}}(z)\, \widetilde{\Sigma}\, \Theta_{\widetilde{\vartheta}}(z)^*$; it is shown below that the PTV matches the truth.

Proposition 2 *Given an LPP Ψ and the criterion function $J_\Psi(\vartheta, \widetilde{F})$, if the model is correctly specified and the minimizer $\vartheta(\widetilde{F})$ is unique then it equals the true parameter $\widetilde{\vartheta}$, and $\Sigma(\widetilde{F}) = \widetilde{\Sigma}$.*

Proof of Proposition 2. Because the model is correct, the criterion function becomes

$$J_\Psi(\vartheta, \widetilde{F}) = \langle \Xi(z)\, \Theta_\vartheta(z)^{-1}\, \Theta_{\widetilde{\vartheta}}(z)\, \widetilde{\Sigma}\, \Theta_{\widetilde{\vartheta}}(z)^*\, \Theta_\vartheta(z)^{-1^*}\, \Xi(z)^* \rangle_0,$$

which for $\vartheta = \widetilde{\vartheta}$ achieves the minimal value:

$$J_\Psi(\widetilde{\vartheta}, \widetilde{F}) = \langle \Xi(z)\, \widetilde{\Sigma}\, \Xi(z)^* \rangle_0.$$

Because the minimizer is unique by assumption, $\vartheta(\widetilde{F}) = \widetilde{\vartheta}$. Plugging this into the formula for $\Sigma(\widetilde{F})$, we see that it equals $\widetilde{\Sigma}$. □

Example 1: Multi-step Ahead Forecasting. For h-step ahead forecasting, only $\psi(-h)$ is nonzero, so that $\Xi(L) = L^{-h}\, e_1'\, [\Theta(L)]_0^{h-1}$. Hence, the criterion function J_Ψ fits models so as to minimize (in the frequency domain) h-step ahead forecast error of the first series. In the special case that $h = 1$, the criterion function is

$$J_\Psi(\vartheta, G) = e_1'\, \langle \Theta_\vartheta(z)^{-1}\, G\, \Theta_\vartheta(z)^{-1^*} \rangle_0\, e_1.$$

If we were to compute such a measure for all n series, and sum over the n criteria, we would obtain the concentrated Whittle likelihood, namely

$$\mathrm{tr}\, \{\langle \Theta_\vartheta(z)^{-1}\, G\, \Theta_\vartheta(z)^{-1^*} \rangle_0\}.$$

See the discussion in McElroy and Findley (2015). This connection justifies viewing J_Ψ as a generalization of the Whittle likelihood from one-step ahead forecasting to more general real-time LPPs.

It is possible to conduct inference from the PTVs on the basis of the estimates $\vartheta(\widehat{F})$, and thereby assess model fit. In order to formulate our result, we assume that the PTVs are not on the boundary of the parameter set (otherwise the limit theory is non-standard; c.f., Self and Liang (1987)), and that they are unique. We also assume that the Hessian $H(\vartheta) = \nabla\nabla' J_\Psi(\vartheta, \widetilde{F})$ of J_Ψ is positive definite at the PTV. The so-called Hosoya-Taniguchi (HT) conditions of Hosoya and Taniguchi (1982) impose sufficient regularity on the process $\{X_t\}$ for our purposes; these conditions require that $\{X_t\}$ is a causal filter of a higher-order martingale difference. A simpler limiting variance expression is available if the fourth order cumulant function of $\{X_t\}$ is zero.

Theorem 1 *Suppose that $\vartheta(\widetilde{F})$ exists uniquely in the interior of the model parameter space, and that $H(\vartheta(\widetilde{F}))$ is positive definite. Suppose that $\{X_t\}$ has finite fourth moments, conditions (HT1)-(HT6) of Taniguchi and Kakizawa (2000, pp.55-56) hold, and that the fourth order cumulant function of $\{X_t\}$ is zero. Then the estimator is consistent for the PTV, and*

$$\sqrt{T}\,\left(\vartheta(\widehat{F}) - \vartheta(\widetilde{F})\right) \stackrel{\mathcal{L}}{\Longrightarrow} \mathcal{N}\left(0, H(\vartheta(\widetilde{F}))^{-1}\, V(\vartheta(\widetilde{F}))\, H(\vartheta(\widetilde{F}))^{-1}\right)$$

as $T \to \infty$, where

$$V_{jk}(\vartheta) = tr\{\langle \partial_j K_\vartheta(z)\, \widetilde{F}\, \partial_k K_\vartheta(z)\, \widetilde{F}\rangle_0\}.$$

Proof of Theorem 1. A Taylor series exapansion of the gradient of $J_\Psi(\vartheta, \widehat{F})$ and $J_\Psi(\vartheta, \widetilde{F})$ yields the asymptotic expression

$$\sqrt{T}\,\left(\vartheta(\widehat{F}) - \vartheta(\widetilde{F})\right) = o_P(1) - H(\vartheta(\widetilde{F}))^{-1}\, \mathrm{tr}\,\{\langle (\widehat{F} - \widetilde{F})\, \nabla K_\vartheta\rangle_0\},$$

where the trace operator acts upon the spectral matrices, for each component of the gradient operator. Our assumptions allow us to apply Lemma 3.1.1 of Taniguchi and Kakizawa (2000) to the right hand expression, yielding the stated central limit theorem. □

Illustration 1: VAR(1). When the model is a VAR(1), the parameter vector ϑ describes the entries of Φ in such a way that the matrix is stable (i.e., all eigenvalues have magnitude less than one). The error filter can be expressed

$$\begin{aligned}
\Xi(L) &= \sum_{\ell=1}^{\infty} \psi(-\ell)\, L^{-\ell} \sum_{k=0}^{\ell-1} \Phi^k\, L^k \\
&= \sum_{\ell=1}^{\infty} \psi(-\ell)\, L^{-\ell}\, \left(1 - \Phi^\ell\, L^\ell\right)\, (1 - \Phi\, L)^{-1} \\
&= \left(\sum_{\ell=1}^{\infty} \psi(-\ell)\, L^{-\ell} - A_\Psi(\Phi)\right)\, (1 - \Phi\, L)^{-1}.
\end{aligned}$$

It follows that

$$J_\Psi(\vartheta, G) = \sum_{\ell,k>0} \psi(-\ell)\,\langle G\rangle_{\ell-k}\,\psi(-k)' - A_\Psi(\Phi)\sum_{k>0}\langle G\rangle_{-k}\,\psi(-k)' - \sum_{\ell>0}\psi(-\ell)\,\langle G\rangle_{\ell}\,A_\Psi(\Phi)' + A_\Psi(\Phi)\,\langle G\rangle_0\,A_\Psi(\Phi)',$$

which is easily computed. Optimization with respect to ϑ is straightforward. In the special case of h-step ahead forecasting, the criterion further simplifies to

$$J_{L^{-h}}(\vartheta, G) = e_1'\,\langle G\rangle_0\,e_1 - e_1'\,\Phi^h\,\langle G\rangle_{-h}\,e_1 - e_1'\,\langle G\rangle_h\,\Phi^{h\prime}\,e_1 + e_1'\,\Phi^h\,\langle G\rangle_0\,\Phi^{h\prime}\,e_1,$$

and any ϑ such that

$$e_1'\,\Phi^h = e_1'\,\langle G\rangle_h\,\langle G\rangle_0^{-1}$$

is a critical point. If the model is correctly specified, then setting G equal to the spectral density of the VAR(1) yields a minimal possible MSE of

$$e_1'\,\left(\Gamma(0) - \Phi^h\,\Gamma(0)\,\Phi^{h\prime}\right)\,e_1.$$

It is known that Whittle estimation corresponds to the $h = 1$ case, with a criterion function given by the determinant of the forecast MSE matrix (McElroy and Findley, 2015), and in essence incorporates forecast error from all n series. In the above, the criterion only depends on the performance of the first series, which allows a practitioner to focus on parameter values that sacrifice performance on the other $n - 1$ series in order to achieve superior results for the first series.

Exercise 1: Correct VAR(1) LPP. Simulate a sample of size $T = 100$ from a bivariate VAR(1) process with

$$\Phi = \begin{bmatrix} 1 & .5 \\ -.2 & .3 \end{bmatrix}$$

and Σ equal to the identity. The eigenvalues are .8 and .5. Then utilize the LPP criterion $J_\Psi(\vartheta, G)$ to fit a VAR(1) model (let ϑ correspond to the four entries of Φ) with both the 2-step ahead forecasting LPP (Example 1) and the ideal low-pass LPP (Example 2) with $\mu = \pi/24$, where G is given by the periodogram of the sample. Do the parameter estimates appear to be consistent? Repeat for $T = 200$ and $T = 500$.

Exercise 2: Incorrect VAR(1) LPP. Simulate a sample of size $T = 100$ from a bivariate Vector Moving Avergage process of order one, or VMA(1), given by

$$X_t = \epsilon_t + \Theta\,\epsilon_{t-1},$$

where

$$\Theta = \begin{bmatrix} 1 & 0 \\ .4 & 2 \end{bmatrix}$$

and Σ equals the identity. Use the VAR(1) LPP criterion of Exercise 1 to fit the mis-specified VAR(1) to this process, for both the 2-step ahead forecasting LPP and the ideal low-pass LPP. Why do the parameter estimates not match those of Θ? Repeat for $T = 200$ and $T = 500$, and explain your results.

Chapter 3

Introduction to the Multivariate Direct Filter Analysis

Chapter 2 introduced the LPP and its ideal solution. This chapter extends the discussion, by determining optimal solutions without restricting to parametric models.

3.1 Background on Multivariate Filtering

Recall the spectral representation of $\{X_t\}$ via (2.2). Because there are n series in the orthogonal increments process $\mathcal{Z}$, we have

$$X_{t,j} = \int_{-\pi}^{\pi} e^{i\omega t}\, \mathcal{Z}_j(d\omega)$$

for each $1 \leq j \leq n$, and hence for a scalar target $\{Y_t\}$ we have

$$Y_t = \sum_{j=1}^{n} \int_{-\pi}^{\pi} e^{i\omega t}\, \Psi_{1j}(e^{-i\omega})\, \mathcal{Z}_j(d\omega). \tag{3.1}$$

Each of the functions $\Psi_{1j}(e^{-i\omega})$ is complex scalar-valued, and can be decomposed in terms of its gain and phase functions. We here provide some background on these functions, because they provide an interpretation of the action of the linear filter on the input time series.

Any complex number ζ is decomposed in terms of its real $\Re\zeta$ and imaginary $\Im\zeta$ parts:

$$\zeta = \Re\zeta + i\,\Im\zeta.$$

The *magnitude* of ζ is defined via

$$|\zeta| = \sqrt{\Re\zeta^2 + \Im\zeta^2}.$$

If this is positive, then $\zeta/|\zeta|$ is a complex number with unit modulus, and hence can be represented as $\exp\{-i\,\mathrm{Arg}\zeta\}$ for some angle in $[0, 2\pi]$ known as $\mathrm{Arg}\zeta$, or the *angular portion* or ζ. It follows that

$$\zeta = |\zeta|\, \exp\{-i\,\mathrm{Arg}\zeta\},$$

which is known as the Polar decomposition of ζ. Sometimes it is of interest to use a negative Polar decomposition based upon the negative magnitude, which can still be written in terms of $\text{Arg}\zeta$ via

$$\zeta = -|\zeta| \exp\{-i\,[\pi + \text{Arg}\zeta]\},$$

using $e^{-i\pi} = -1$. The angular portion of ζ can be directly computed from the real and imaginary parts of ζ via

$$\text{Arg}\zeta = \arctan\left(\frac{-\Im\zeta}{\Re\zeta}\right).$$

Now when ζ is a function of $\omega \in [-\pi, \pi]$, then the magnitude and angular portions also become functions of ω. In particular, a scalar frequency response function has a magnitude function (called the gain function) and angular function (called the phase function). In the case of some scalar filter $\Psi(B)$ (e.g., the component filter $\Psi_{1j}(B)$) we obtain

$$\Psi(e^{-i\omega}) = |\Psi(e^{-i\omega})| \exp\{-i\,\text{Arg}\Psi(e^{-i\omega})\}.$$

At $\omega = 0$, we know the frequency response function is $\Psi(1) = \sum_{\ell\in\mathbb{Z}} \psi(\ell)$, which is real; hence the phase function at $\omega = 0$ must be an integer multiple of π. It is advantageous to ensure the phase function takes the value zero, as this will facilitate the definition of the phase delay function discussed below. We can ensure this condition by allowing the gain function to be signed. Denoting these by $A(\omega)$ (for amplitude, or signed gain) and $\Phi(\omega)$ (for continuous phase), we have

$$\Psi(e^{-i\omega}) = A(\omega) \exp\{-i\,\Phi(\omega)\}. \tag{3.2}$$

There may be frequencies ω for which the frf equals zero; because the real and imaginary parts of zero are both zero, there is an indeterminancy to the angular portion. However, because the frf is continuous in ω (which follows from summability of the coefficients) we should define the phase function at the zeroes such that it is continuous. By adjusting the angular portion by an integer multiple of π, we can ensure that it will be a continuous function of ω; this adjustment can be compensated by inserting a sign change in the gain function. In this way, the signed gain and continuous phase functions can be computed: both A and Φ will be continuous functions of ω, and $\Phi(0) = 0$ as well. Substituting (3.2) into the spectral representation (3.1) of the target, where A_j and Φ_j are the gain and phase functions of $\Psi_{1j}(e^{-i\omega})$, yields

$$Y_t = \sum_{j=1}^{n} \int_{-\pi}^{\pi} e^{i\omega\,[t-\omega^{-1}\,\Phi_j(\omega)]}\, A_j(\omega)\, \mathcal{Z}_j(d\omega).$$

This representation is interpreted as follows: the target is the sum of n filtered series, where each orthogonal increments process $\mathcal{Z}_j$ has been dilated by the signed gain function A_j, and the timing of the sinusoidal component $e^{i\omega t}$ has been delayed by $\omega^{-1}\,\Phi_j(\omega)$. This quantity, called the *phase delay function*, is well-defined in a neighborhood of zero, as seen in the following result.

Proposition 3 *If the scalar filter $\Psi(B)$ satisfies $\sum_{\ell\in\mathbb{Z}} \ell\,\psi(\ell) < \infty$ and the phase function is continuously defined, then the phase delay function*

$$\phi(\omega) = \frac{\Phi(\omega)}{\omega}$$

is well-defined for $\omega \in [-\pi, \pi]$, and

$$\phi(0) = \dot{\Phi}(0) = \frac{\sum_{\ell\in\mathbb{Z}} \ell\,\psi(\ell)}{\sum_{\ell\in\mathbb{Z}} \psi(\ell)}.$$

Proof of Proposition 3. First note that the signed gain function is even, and hence $\dot{A}(0) = 0$. Differentiating (3.2) and evaluating at zero yields

$$-i \sum_{\ell \in \mathbb{Z}} \ell \, \psi(\ell) = \frac{\partial}{\partial \omega} \Psi(e^{-i\omega})|_{\omega=0} = \dot{A}(0) \, e^{-i \, \Phi(0)} + A(0) \, e^{-i \, \Phi(0)} \, (-i \, \dot{\Phi}(0)).$$

Using $\dot{A}(0) = 0$, $\Phi(0) = 0$, and $A(0) = \Psi(1)$ yields

$$\dot{\Phi}(0) = \frac{\sum_{\ell \in \mathbb{Z}} \ell \, \psi(\ell)}{\sum_{\ell \in \mathbb{Z}} \psi(\ell)}. \quad \square$$

The amplitude effects can be understood as dilations of the input spectral densities. If the spectral density matrix of the input process is denoted F, then F_{jj} is the spectral density of the jth component input series; its contribution to the target output involves the increment

$$A_j(\omega) \, \mathcal{Z}_j(d\omega),$$

and the associated spectral density is

$$|A_j(\omega)|^2 \, F_{jj}(\omega).$$

There are approximate empirical versions of these relations, which can be described in terms of the DFT. Applying the definition (2.3) to the scalar output $\{Y_t\}$, and utilizing (3.1), we obtain

$$\widetilde{Y}(\xi) = \int_{-\pi}^{\pi} T^{-1/2} \sum_{t=1}^{T} e^{i(\omega - \xi)t} \, \Psi(e^{-i\omega}) \, \mathcal{Z}(d\omega).$$

Note that the summation is bounded as $T \to \infty$ unless $\omega = \xi$; it can be shown that the variance of the difference between $\widetilde{Y}(\xi)$ and $\Psi(e^{-i\xi}) \, \widetilde{X}(\xi)$ tends to zero, so that we have the approximate result

$$\widetilde{Y}(\omega) \approx \Psi(e^{-i\omega}) \, \widetilde{X}(\omega). \tag{3.3}$$

Utilizing (2.4), we obtain an approximate relation of periodograms:

$$\widehat{F}_Y(\omega) \approx \Psi(e^{-i\omega}) \, \widehat{F}_X(\omega) \, \Psi(e^{i\omega})'. \tag{3.4}$$

3.2 Multivariate Direct Filter Analysis of the LPP

We can now discuss a more general solution to the LPP. One perspective on Proposition 1 is that it provides a particular class of concurrent filters that arise from specified models. However, so long as these models are mis-specified, the resulting concurrent filters will be sub-optimal. Therefore, it may be possible to improve performance by utilizing broader classes of concurrent filters that are not derived from a particular model. The Direct Filter Analysis (DFA) seeks a concurrent filter $\widehat{\Psi}(B)$ that optimizes the MSE in a given LPP. While DFA was originated to handle univariate time series, its multivariate generalization – Multivariate Direct Filter Analysis (MDFA) – is designed for the broader context of LPPs discussed in Chapter 2.

The entire class of concurrent filters corresponds to the collection of power series in L. Here we are interested in scalar targets given N input series, so the coefficient matrices of the concurrent filters are $1 \times n$. We may be interested in some subcollection $\mathcal{G}$ of all concurrent filters. For instance, $\mathcal{G}$ could be the optimal solutions to an LPP for a particular process, i.e., consist of all $\widehat{\Psi}(L)$ given in (2.8) for a particular $\Psi(L)$ and $\Theta(L)$. Or we might consider much broader classes of filters, that are described in terms of the rate of decay of the filter coefficients, e.g.,

$$\mathcal{G} = \{\Upsilon(L) : \Upsilon(e^{-i\omega}) \text{ is twice continuously differentiable at } \omega = 0\}.$$

Alternatively, $\mathcal{G}$ might consist of all VARMA filters of a particular AR and MA order, or might consist of all Zero-Pole Combination (ZPC) filters of a given specification (Wildi, 2008). The original univariate DFA of Wildi (2008) approached the LPP with $\mathcal{G}$ consisting of appropriately restricted ZPC filters.

For now, we shall suppose that the concurrent filters of $\mathcal{G}$ belong to some parametric family described by a parameter ϑ belonging to a parameter manifold. Because we seek elements of $\mathcal{G}$ that will solve an LPP, i.e., be a good concurrent approximation to $\Psi(L)$, we use the notation

$$\mathcal{G} = \{\widehat{\Psi}_\vartheta(L) : \vartheta \text{ belongs to a parameter space}\}. \tag{3.5}$$

Whereas the model-based approach to the LPP discussed in Chapter 2 involves minimizing a particular parametric form of the filter error MSE – namely the function $J_\Psi(\vartheta, G)$ for G corresponding either to the periodogram or true spectrum – a more direct approach is to minimize a general expression for the filter error MSE over a given set $\mathcal{G}$. The real-time estimation error is given in (2.7), which has mean zero and variance

$$\mathbb{E}[E_t^2] = \langle \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right] \widetilde{F} \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right]^* \rangle_0. \tag{3.6}$$

This suggests the criterion function $D_\Psi(\vartheta, G)$ for any Hermitian function G, defined as

$$D_\Psi(\vartheta, G) = \langle \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right] G \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right]^* \rangle_0. \tag{3.7}$$

This is the MDFA criterion function. An equivalent formula to (3.7) that can be useful for calculations is

$$D_\Psi(\vartheta, G) = \mathrm{tr}\{\langle G\, M_\vartheta \rangle_0\} \qquad M_\vartheta(z) = \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right]^* \left[\Psi(z) - \widehat{\Psi}_\vartheta(z)\right]. \tag{3.8}$$

Given a filter class $\mathcal{G}$, the best possible concurrent filter is given by $\widehat{\Psi}_{\vartheta(\widetilde{F})}$, where $\vartheta(\widetilde{F})$ is a minimizer of $D_\Psi(\vartheta, \widetilde{F})$. This $\vartheta(\widetilde{F})$ is the PTV for the filter parameter, in analogy with the terminology for model parameters. Clearly, if the set $\mathcal{G}$ is rendered sufficiently large to include the optimal concurrent filter for that particular LPP and process – as given in Proposition 1 – then there exists some $\widetilde{\vartheta}$ such that $\widehat{\Psi}_{\widetilde{\vartheta}}$ is identical with the optimal filter. However, if $\mathcal{G}$ is smaller, then the PTV $\vartheta(\widetilde{F})$ is as close as possible according to D_Ψ discrepancy to the optimal filter.

A case of interest arises from taking the broadest possible $\mathcal{G}$ for a sample of size T. Let $\mathcal{G}$ consist of all length T concurrent filters, with

$$\vartheta' = \left[\widehat{\psi}(0), \widehat{\psi}(1), \ldots, \widehat{\psi}(T-1)\right].$$

So ϑ is a column vector of length Tn. Then the criterion (3.7) can be rewritten as

$$D_\Psi(\vartheta, G) = \vartheta' B \vartheta - \vartheta' b - b^* \vartheta + \langle \Psi(z) G \Psi(z)^* \rangle_0,$$

where

$$b^* = \left[\langle \Psi(z) G \rangle_0, \langle \Psi(z) G \rangle_1, \ldots, \langle \Psi(z) G \rangle_{T-1} \right], \tag{3.9}$$

and B is a block matrix, where the jkth $n \times n$ block of is $\langle G \rangle_{k-j}$.

Proposition 4 *The minimizer of the MDFA criterion (3.7), given that $\mathcal{G}$ consist of all length T concurrent filters, is*

$$\vartheta = B^{-1} \Re(b),$$

where the jkth block of B is $\langle G \rangle_{k-j}$, and b is given by (3.9).

Proof of Proposition 4. The objective function is a quadratic in ϑ, and therefore the minimizer is obtained by computing the gradient and Hessian, whih are $-2\Re(b) + 2B\vartheta$ and $2B$ respectively, yielding the solution. □

This broad class $\mathcal{G}$ of filters will furnish concurrent filters that closely approximate those of Proposition 1 as $T \to \infty$.

Example 1: Multi-step Ahead Forecasting Suppose we consider the one-step ahead forecasting of stationary time series and $\mathcal{G}$ corresponds to all VMA filters of order q, where

$$\vartheta = \text{vec}[\widehat{\psi}(0)', \widehat{\psi}(1)', \ldots, \widehat{\psi}(q)'].$$

With $\Psi(L) = L^{-1}$ from (3.7) we have

$$\begin{aligned} D_\Psi(\vartheta, G) &= \langle \left[z^{-1} e_1' - \widehat{\Psi}_\vartheta(z) \right] G \left[z^{-1} e_1' - \widehat{\Psi}_\vartheta(z) \right]^* \rangle_0 \\ &= \langle \left[e_1' - \sum_{\ell=0}^{q} \widehat{\psi}(\ell)\, z^{\ell+1} \right] G \left[e_1' - \sum_{\ell=0}^{q} \widehat{\psi}(\ell)\, z^{\ell+1} \right]^* \rangle_0 \\ &= \langle G \rangle_0 - 2\,\vartheta' \langle G \rangle_{1:(q+1)}\, e_1 + \vartheta' \langle G \rangle_{0:q,0:q}\, \vartheta. \end{aligned}$$

Hence the optimizer is

$$\vartheta(G) = \langle G \rangle_{0:q,0:q}^{-1} \langle G \rangle_{1:(q+1)}\, e_1,$$

which is the first component of the solution to the Yule-Walker system of order $q+1$ determined by G. Therefore the MDFA solution is the same as the fit of a VAR$(q+1)$ using Proposition 1.

The empirical problem is solved by minimizing $D_\Psi(\vartheta, \widehat{F})$, yielding the estimator $\vartheta(\widehat{F})$. The empirical criterion can be simply computed using (3.8) and (2.6), namely

$$D_\Psi(\vartheta, \widehat{F}) = \sum_{|h|<T} \text{tr}\{\widehat{\Gamma}(h) \langle M_\vartheta \rangle_{-h}\}.$$

Filtering with $\widehat{\Psi}_{\vartheta(\widehat{F})}$ instead of $\widehat{\Psi}_{\vartheta(\widetilde{F})}$ involves some statistical error, which vanishes as $n \to \infty$ because $\vartheta(\widehat{F})$ is consistent for the PTV. We can quantify this additional error if we know the statistical properties of the estimate; under fairly broad conditions, it follows a central limit theorem. As in Chapter 2, we assume the HT conditions and that the Hessian $H(\vartheta) = \nabla\nabla' D_\Psi(\vartheta, \widetilde{F})$ of D_Ψ is positive definite at the PTV. The function M is defined in (3.8).

Theorem 2 *Suppose that $\vartheta(\widetilde{f})$ exists uniquely in the interior of the filter parameter space, and that $H(\vartheta(\widetilde{F}))$ is positive definite. Suppose that $\{X_t\}$ has finite fourth moments, conditions (HT1)-(HT6) of Taniguchi and Kakizawa (2000, pp.55-56) hold, and that the fourth order cumulant function of $\{X_t\}$ is zero. Then the estimator is consistent for the PTV, and*

$$\sqrt{T}\left(\vartheta(\widehat{F}) - \vartheta(\widetilde{F})\right) \stackrel{\mathcal{L}}{\Longrightarrow} \mathcal{N}\left(0, H(\vartheta(\widetilde{F}))^{-1} V(\vartheta(\widetilde{F})) H(\vartheta(\widetilde{F}))^{-1}\right)$$

as $T \to \infty$, where

$$V_{jk}(\vartheta) = tr\{\langle \partial_j M_\vartheta(z)\, \widetilde{F}\, \partial_k M_\vartheta(z)\, \widetilde{F}\rangle_0\}.$$

Proof of Theorem 2. This is proved in the same way as Theorem 1.

We designate the resulting prediction function $\widehat{\Psi}_{\widehat{\vartheta}}$ as a *Linear Prediction Filter* (LPF).

Illustration 1: VAR(1). Again consider a VAR(1) process, and suppose we wish to use MDFA to approximate the optimal LPP solution – even though we don't know the true dynamics. Let $\mathcal{G}$ denote the set of moving average filters of length T, and G is the spectral density of the VAR(1); the solution given by Proposition 4 can be compared to that of the LPP, which has the first T components given by

$$\varphi' = [\psi(0) + A_\Psi(\Phi), \psi(1), \ldots, \psi(T-1)].$$

This is an approximate solution to the system $\vartheta' B = \Re b'$, because $\varphi' B$ has $j+1$th component, for $0 \le j \le T-1$, equal to

$$\sum_{\ell=0}^{T-1} \psi(\ell)\, \langle G\rangle_{j-\ell} + A_\Psi(\Phi)\, \Gamma(j).$$

Noting that

$$A_\Psi(\Phi)\, \Gamma(j) = \sum_{\ell<0} \psi(-\ell)\, \Phi^{-\ell}\, \Gamma(j) = \sum_{\ell<0} \psi(-\ell)\, \Gamma(j-\ell),$$

because for a VAR(1) process $\Gamma(h) = \Phi^h\, \Gamma(0)$ when $h \ge 0$, we see that component $j+1$ of $\varphi' B$ is

$$\sum_{\ell \le T-1} \psi(\ell)\, \Gamma(j-\ell) = [\Re b']_{j+1} - \sum_{\ell \ge T} \psi(\ell)\, \Gamma(j-\ell).$$

As $T \to \infty$ the error term vanishes (for each j), indicating that $\varphi' B \approx \Re b'$, or $\vartheta \approx \varphi$.

Exercise 1: Correct VAR(1) LPP. Simulate a sample of size $T = 100$ from a bivariate VAR(1) process with

$$\Phi = \begin{bmatrix} 1 & .5 \\ -.2 & .3 \end{bmatrix}$$

and Σ equal to the identity. The eigenvalues are .8 and .5. Implement MDFA for this sample, using the moving average filters (Proposition 4) to approximate the optimal LPP solution, for the 2-step ahead forecasting LPP (Example 1) and the ideal low-pass LPP (Example 2) with $\mu = \pi/24$. Does the MDFA concurrent filter closely match the optimal LPP filter? Repeat for $T = 200$ and $T = 500$.

Exercise 2: Incorrect VAR(1) LPP. Simulate a sample of size $T = 100$ from a bivariate VMA(1) with

$$\Theta = \begin{bmatrix} 1 & 0 \\ .4 & 2 \end{bmatrix}$$

and Σ equal to the identity. Use the moving average filter MDFA (Proposition 4) to find the best concurrent filter, for the 2-step ahead forecasting LPP (Example 1) and the ideal low-pass LPP (Example 2) with $\mu = \pi/24$. Compare these results to the VAR(1) LPP filter previously obtained, based on the mis-specified VAR(1) model. Which filter, LPP or MDFA, more closely approximates the ideal filter? Repeat for $T = 200$ and $T = 500$, and explain your results.

Chapter 4

Classic Mean-Square Error (MSE) Perspective

4.1 Introduction

MDFA is a generic forecast and signalextraction paradigm with a richly parametrized user-interface allowing for sophisticated data analysis. In this chapter we emphasize mean-square performances: the corresponding default-parameters were introduced in section 1.2.4. Specifically, the parameters can be conveniently up-loaded by sourcing a corresponding R-file.

In section 4.2 we provide a brief 'fresh-up' of the antecedent DFA paradigm; section 4.3 generalizes the univariate (MSE-) case to a multivariate (MSE-) framework; section 4.4 presents a general matrix-notation which will allow for formal and convenient extensions of the univariate DFA to the MDFA as well as for suitable extensions of the classic MSE-norm; an alternative so-called grand-mean parametrization is discussed in section 4.5; the DFA is replicated by MDFA in section 4.6; finally, section 4.7 benchmarks a bivariate MDFA against the former DFA and evaluates performance gains by a leading-indicator design.

4.2 DFA Booster

We propose a brief survey or 're-fresher' of the main DFA-concepts. The interested reader is referred to DFA and to to McElroy and Wildi (2014) (DFA and Trilemma) for technical details, (R-)code and exercises on the topic.

4.2.1 Discrete Fourier Transform (DFT) and Periodogram

A time series x_t, $t = 1, ..., T$, of length T, can be mapped to the frequency-domain by the so-called DFT:

$$\Xi_{TX}(\omega) := \frac{1}{\sqrt{2\pi T}} \sum_{t=1}^{T} x_t \exp(-it\omega) \tag{4.1}$$

The DFT $\Xi_{TX}(\omega)$ is generally restricted to the discrete frequency-grid $\omega_k = \frac{k2\pi}{T}$, where $k = -T/2, ..., 0, ..., T/2$, for even T[1]. This (discrete grid) restriction can be justified by the fact that the data could be recovered from the DFT by applying the so-called *inverse* (DFT-) transformation

$$x_t = \frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} w_k \Xi_{TX}(\omega_k) \exp(it\omega_k) \tag{4.2}$$

$$= \frac{\sqrt{2\pi}}{\sqrt{T}} \left(\Xi_{TX}(0) + 2 \sum_{k=1}^{[T/2]} w_k \Re \left(\Xi_{TX}(\omega_k) \exp(it\omega_k) \right) \right) \tag{4.3}$$

where $[T/2] = \begin{cases} T/2 & T \text{ even} \\ (T-1)/2 & T \text{ odd} \end{cases}$ and where

$w_k = \begin{cases} 1 & , [-T/2] \leq k \leq [T/2] \text{ if } T \text{ is odd} \\ \begin{cases} 1 & |k| < T/2 \\ 1/2 & |k| = T/2 \end{cases} & \text{if } T \text{ is even} \end{cases}$. This results suggests that we can restrict the DFT to the discrete grid $\omega_k$since the corresponding (frequency-domain) information is equivalent to the original data sample $x_1, ..., x_T$. In practice, the weights w_k (not to be confounded with the frequencies ω_k) are negligible and can be omitted from formulas or code expressions.

The identity 4.2 is a tautological number identity: it applies to any sequence of numbers x_t, $t = 1, ..., T$ irrespective of (model-) assumptions. The identity suggests that the data x_t can be decomposed into a linear combination of sines and cosines as weighted by the DFT. The equation can be verified empirically, see DFA, section 2.2.1, exercise 2 (a proof is provided in the appendix).

The *periodogram* $I_{TX}(\omega_k)$ is defined by

$$I_{TX}(\omega_k) = |\Xi_{TX}(\omega_k)|^2 \tag{4.4}$$

The periodogram is the DFT of the *sample autocovariance function* $\hat{R}(k)$ of the data:

$$I_{TX}(\omega_k) = \begin{cases} \frac{1}{2\pi} \sum_{j=-(T-1)}^{T-1} \hat{R}(j) \exp(-ij\omega_k) & , \quad |k| = 1, ..., T/2 \\ \frac{T}{2\pi} \overline{x}^2 & , \quad k = 0 \end{cases} \tag{4.5}$$

where

$$\hat{R}(j) := \frac{1}{T} \sum_{t=1}^{T-|j|} x_t x_{t+|j|} \tag{4.6}$$

[1] For odd T one uses $T' = T - 1$ in these expressions instead of T.

is the sample autocovariance of a zero-mean stationary process. The periodogram can be interpreted as a decomposition of the sample variance:

$$\hat{R}(0) = \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} I_{TX}(\omega_k) = \frac{2\pi}{T} I_{TX}(0) + 2\frac{2\pi}{T} \sum_{k=1}^{[T/2]} I_{TX}(\omega_k) \tag{4.7}$$

The value $2\frac{2\pi}{T} I_{TX}(\omega_k), k > 0$ measures dynamic contributions of components with frequency ω_k[2] to the sample variance of the data.

In analogy to 4.2, the identity 4.5 is a tautological number-identity which holds irrespective of model assumptions about x_t, see DFA, section 2.3.1, exercise 1.

4.2.2 Filter Effects: Transfer- Amplitude- and Time-Shift Functions

Let y_t be the output of a general filter

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k}$$

In order to derive the important filter effect(s) we assume a particular (complex-valued) input series $x_t := \exp(it\omega)$. The output signal y_t then becomes

$$\begin{aligned} y_t &= \sum_{k=-\infty}^{\infty} \gamma_k \exp(i\omega(t-k)) && (4.8)\\ &= \exp(i\omega t) \sum_{k=-\infty}^{\infty} \gamma_k \exp(-ik\omega) && (4.9)\\ &= \exp(i\omega t)\Gamma(\omega) && (4.10) \end{aligned}$$

where the (generally complex-valued) function

$$\Gamma(\omega) := \sum_{k=-\infty}^{\infty} \gamma_k \exp(-ik\omega) \tag{4.11}$$

is called the *transfer function* of the filter. We can represent the complex number $\Gamma(\omega)$ in terms of polar coordinates:

$$\Gamma(\omega) = A(\omega)\exp(-i\Phi(\omega)) \tag{4.12}$$

where $A(\omega) = |\Gamma(\omega)|$ is called the *amplitude* of the filter and $\Phi(\omega)$ is its *phase*.

If the filter coefficients are real, then the real part of x_t is mapped to the real part of y_t. Therefore the cosine (real-part of the input) is mapped to

$$\begin{aligned} \cos(t\omega) &\rightarrow \Re(\exp(i\omega t)\Gamma(\omega)) && (4.13)\\ &= A(\omega)\left[\cos(t\omega)\cos(-\Phi(\omega)) - \sin(t\omega)\sin(-\Phi(\omega))\right] \\ &= A(\omega)\cos(t\omega - \Phi(\omega)) \\ &= A(\omega)\cos(\omega(t - \Phi(\omega)/\omega)) && (4.14) \end{aligned}$$

[2] More precisely: components with frequencies in the interval $[\omega_k - \pi/T, \omega_k + \pi/T]$.

The amplitude function $A(\omega)$ can be interpreted as the weight (damping if $A(\omega) < 1$, amplification if $A(\omega) > 1$) attributed by the filter to a sinusoidal input signal with frequency ω. The function

$$\phi(\omega) := \Phi(\omega)/\omega \tag{4.15}$$

can be interpreted as the *time shift* of the filter at frequency ω.

The transferfunction of a causal and stable ARMA-filter

$$y_t = \sum_{k=1}^{L'} a_k y_{t-k} + \sum_{j=0}^{L} b_j x_{t-j}$$

is obtained as

$$\Gamma(\omega) = \frac{\sum_{j=0}^{L} b_j \exp(-ij\omega)}{1 - \sum_{k=1}^{L'} a_k \exp(-ik\omega)}$$

Amplitude and time-shift functions of the ARMA-filter can be derived from this expression, see DFA, sections 3.2.2, 3.2.3 and 3.2.4 for comprehensive results and exercises on the topic.

4.2.3 Discrete Finite Sample Convolution

The transferfunction or, alternatively, the amplitude and the phase (or time-shift) functions, summarize and describe the effects of a filter as applied to an elementary (periodic and deterministic) trigonometric signal $x_t = \exp(it\omega)$:

$$y_t = \sum_{j=-\infty}^{\infty} \gamma_j x_{t-j} = \Gamma(\omega) x_t$$

An arbitrary sequence $x_1, ..., x_T$, neither periodic nor deterministic, can be decomposed into a weighted sum of trigonometric sinusoids

$$x_t = \frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} \Xi_{TX}(\omega_k) \exp(it\omega_k) \tag{4.16}$$

and similarly for y_t

$$y_t = \frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} \Xi_{TY}(\omega_k) \exp(it\omega_k) \tag{4.17}$$

recall the number-identity 4.2 (we omit the weights w_k). Therefore, when applying the filter to a general sequence $x_1, ..., x_T$ we might proceed as follows

$$\begin{aligned}
y_t &= \sum_{j=-\infty}^{\infty} \gamma_j x_{t-j} \\
&\approx \sum_{j=-\infty}^{\infty} \gamma_j \left(\frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} w_k \Xi_{TX}(\omega_k) \exp(i(t-j)\omega_k) \right) \qquad (4.18) \\
&= \frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} w_k \Xi_{TX}(\omega_k) \left(\sum_{j=-\infty}^{\infty} \gamma_j \exp(i(t-j)\omega_k) \right) \\
&= \frac{\sqrt{2\pi}}{\sqrt{T}} \sum_{k=-[T/2]}^{[T/2]} w_k \Xi_{TX}(\omega_k) \Gamma(\omega_k) \exp(it\omega_k) \qquad (4.19)
\end{aligned}$$

Comparing 4.17 and 4.19 suggests that the DFT $\Xi_{TY}(\omega_k)$ of the output signal is linked to the DFT $\Xi_{TX}(\omega_k)$ of the input signal via

$$\Xi_{TY}(\omega) \approx \Gamma(\omega)\Xi_{TX}(\omega) \tag{4.20}$$

This result is not a strict equality but from a practical point of view we can ignore the error [3]. A quantification of the finite sample error is provided in DFA, section 3.3.2. By definition, see 4.4, we then obtain

$$I_{TY}(\omega) \approx |\Gamma(\omega)|^2 I_{TX}(\omega) \tag{4.21}$$

4.2.4 Assembling the Puzzle: the Optimization Criterion

We assume a general target specification

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k} \tag{4.22}$$

Note, for example, that (classical one-step ahead) forecasting could be addressed by specifying $\gamma_{-1} = 1$, $\gamma_k = 0, k \neq -1$. We aim at finding filter coefficients b_k, $k = 0, ..., L-1$ such that the finite sample estimate

$$\hat{y}_t := \sum_{k=0}^{L-1} b_k x_{t-k} \tag{4.23}$$

is 'closest possible' to y_t in *mean-square*

$$E\left[(y_t - \hat{y}_t)^2\right] \to \min_{\mathbf{b}} \tag{4.24}$$

where $\mathbf{b} = (b_0, ..., b_{L-1})$.

As usual, in applications, the expectation is unknown and therefore we could try to replace 4.24 by its sample estimate

$$\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2 = \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} I_{T\Delta Y}(\omega_k) \tag{4.25}$$

where $I_{T\Delta Y}(\omega_k)$ is the periodogram of $\Delta y_t := y_t - \hat{y}_t$ and where the identity follows from 4.7. Unfortunately, the output y_t of the generally bi-infinite filter isn't observed and therefore $I_{T\Delta Y}(\omega_k)$ is unknown too. But we could try to approximate $I_{T\Delta Y}(\omega_k)$ by relying on the finite-sample discrete convolution 4.21:

$$I_{T\Delta Y}(\omega_k) \quad \approx \quad |\Delta\Gamma(\omega_k)|^2 I_{TX}(\omega_k)$$

[3] One can invoke a 'uniform super-consistency' argument for integrals or discrete sums, see Wildi (2005) and (2008).

where $\Delta\Gamma(\omega_k) = \Gamma(\omega_k) - \hat{\Gamma}(\omega_k) = \sum_{j=-\infty}^{\infty} \Delta\gamma_j \exp(-ij\omega_k)$ is the difference of target and real-time transfer functions. Then

$$\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2 = \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} I_{T\Delta Y}(\omega_k) \tag{4.26}$$

$$\approx \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} |\Delta\Gamma(\omega_k)|^2 I_{TX}(\omega_k) \to \min_{\mathbf{b}} \tag{4.27}$$

We refer the reader to section 4.1 in DFA for background on this derivation and in particular:

- for an extension of these concepts to integrated processes, see also chapter 16, and
- for the magnitude of the approximation error which is negligible in practice ('uniform superconsistency'[4]).

The optimization problem specified by 4.27 is the DFA-MSE criterion (Direct Filter Approach Mean-Square Error).

4.2.5 Exercises: 'Uniform Superconsistency' in a Finite Sample Number-Perspective

The empirical design of the following exercises is inspired from DFA, section 4.1.1, exercise 2. It is assumed that the reader is familiar with the concept of an ideal trend (definition, computation of symmetric coefficients) as well as with the DFA-booster proposed above. The series of exercises aims at illustrating the quality of the approximation 4.27 by putting the abstract 'uniform superconsistency' argument into a finite-sample number-perspective.

1. Generate realizations of three different stationary processes[5]

$$\left.\begin{aligned} x_t &= 0.9x_{t-1} + \epsilon_t \\ x_t &= 0.1x_{t-1} + \epsilon_t \\ x_t &= -0.9x_{t-1} + \epsilon_t \end{aligned}\right\} \tag{4.28}$$

   and apply an ideal trend with cutoff $\pi/6$ to each of them. The target series y_t should have length 120 (10 years of monthly data). Hint: since the ideal trend is a bi-infinite filter, we use a truncated finite sample approximation.

   - Generate realizations for all three processes. Hint: generate long time series of length 2000 in order to apply the symmetric target filter.

[4] Superconsistency means: the approximation error is asymptotically of smaller order than $1/\sqrt{T}$. Uniformity means: this claim remains valid after optimization. The magnitude of the approximation error is computed in DFA, section 3.3.2.

[5] The middle process $x_t = 0.1x_{t-1} + \epsilon_t$ is inspired from macro-economic applications since this model fits log-returns of INDPRO (industrial production in the US) quite well over a longer historical time span. The other two processes with coefficients $a_1 = -0.9$ and $a_1 = 0.9$ illustrate applications to strongly negatively and to strongly positively autocorrelated processes.

```
> # Generate series of length 2000
> lenh<-2000
> len<-120
> # Specify the AR-coefficients
> a_vec<-c(0.9,0.1,-0.9)
> xh<-matrix(nrow=lenh,ncol=length(a_vec))
> x<-matrix(nrow=len,ncol=length(a_vec))
> yhat<-x
> y<-x
> # Generate series for each AR(1)-process
> for (i in 1:length(a_vec))
+ {
+ # We want the same random-seed for each process
+   set.seed(10)
+   xh[,i]<-arima.sim(list(ar=a_vec[i]),n=lenh)
+ }
```

- Extract the data (series of length 120) and compute (truncated) ideal trends.

```
> # Extract 120 observations in the midddle of the longer series
> x<-xh[lenh/2+(-len/2):((len/2)-1),]
> # Compute the coefficients of the symmetric target filter
> cutoff<-pi/6
> # Order of approximation
> ord<-1000
> # Filter weights ideal trend (See DFA)
> gamma<-c(cutoff/pi,(1/pi)*sin(cutoff*1:ord)/(1:ord))
> # Compute the outputs yt of the (truncated) symmetric target filter
> for (i in 1:length(a_vec))
+ {
+   for (j in 1:120)
+   {
+     y[j,i]<-gamma[1:900]%*%xh[lenh/2+(-len/2)-1+(j:(j-899)),i]+
+     gamma[2:900]%*%xh[lenh/2+(-len/2)+(j:(j+898)),i]
+   }
+ }
```

Remark: the proposed simulation framework allows computation of the target series y_t which is generally unobservable. Therefore, we can obtain the sample mean-square filter error on the left-hand side of 4.26. In applications, the latter is generally *unobservable*.

2. For each of the above realizations: approximate the ideal trend y_t by a real-time[6] estimate

[6]So-called nowcasts, see section 5.3.1 for details. A nowcast is obtained by setting $Lag = 0$ in the head of the MDFA-function.

$\hat{y}_t$ based on a filter of length $L = 12$ and compute the criterion value (the right-hand side of 4.27).

- Use filters of length $L = 12$ and apply the MSE-DFA criterion 4.27 or, equivalently, the MSE-DFA function proposed in section 1.2.2 to the data samples of length 120 corresponding to $y_1, ..., y_{120}$.

```
> plot_T<-F
> periodogram<-matrix(ncol=3,nrow=len/2+1)
> trffkt<-periodogram
> perf_mat<-matrix(nrow=3,ncol=2)
> dimnames(perf_mat)[[2]]<-c("Criterion Value",
+                            "Mean-Square Sample Filter Error")
> dimnames(perf_mat)[[1]]<-c("a1=0.9","a1=0.1","a1=-0.9")
> # Filter length
> L<-12
> # Real-time design
> Lag<-0
> # Target ideal trend
> Gamma<-c(1,(1:(len/2))<len/12)
> b<-matrix(nrow=L,ncol=3)
> # Compute real-time filters
> for (i in 1:3)#i<-1
+ {
+ # Compute the periodogram based on the data (length 120)
+   periodogram[,i]<-per(x[,i],plot_T)$per
+ # Optimize filters
+   filt<-dfa_ms(L,periodogram[,i],Lag,Gamma)
+   trffkt[,i]<-filt$trffkt
+   b[,i]<-filt$b
+ # Compute real-time outputs (we can use the longer series in order
+ # to obtain estimates for time points t=1,...,11)
+   for (j in 1:len)
+     yhat[j,i]<-filt$b%*%xh[lenh/2+(-len/2)-1+j:(j-L+1),i]
+ }
```

- Compute criterion values (the right-hand side of 4.27) for each time series.

```
> for (i in 1:3)
+ {
+ # Compute criterion values
+   perf_mat[i,1]<-(2*pi/length(Gamma))*
+                   abs(Gamma-trffkt[,i])^2%*%periodogram[,i]
+ }
```

```
> perf_mat[,1]

    a1=0.9     a1=0.1    a1=-0.9
0.31523446 0.06013061 0.02791655
```

3. Compute sample mean-square filter errors: the left-hand side of 4.26.

```
> # Compute time-domain MSE
> mse<-apply(na.exclude((yhat-y))^2,2,mean)
> perf_mat[,2]<-mse
> round(perf_mat[,2],3)

 a1=0.9  a1=0.1 a1=-0.9
  0.321   0.060   0.028
```

4. Compare left-hand side of 4.26 and right-hand side of 4.27, see table 4.1. The alleged tightness of the approximation in 4.27 seems confirmed.

	Criterion Value	Mean-Square Sample Filter Error
a1=0.9	0.315	0.321
a1=0.1	0.060	0.060
a1=-0.9	0.028	0.028

Table 4.1: Criterion values vs. sample (mean-square) filter errors

5. Verify optimality of the estimated filter coefficients (left as an exercise to the reader).

4.2.6 Exercises: Explaining the Optimization Criterion

The DFA-criterion 4.27 corresponds to a weighted optimization: the real-time filter $\hat{\Gamma}(\cdot)$ should be close to the target $\Gamma(\cdot)$ in 'loaded' frequencies whereby the amount of loading is measured by the periodogram (alternative spectral estimates are analyzed in chapter 9). We here briefly illustrate this particular optimization concept by analyzing real-time filter outputs and filter characteristics (amplitude and time-shift functions).

1. Compare graphically target y_t and real-time estimate $\hat{y}_t$ for each realization of the previous exercise, see fig.4.1.

```
> file = paste("z_dfa_ar1_sym_output.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> par(mfrow=c(3,1))
> for (i in 1:3)   #i<-1
+ {
+   ymin<-min(min(y[,i]),min(na.exclude(yhat)[,i]))
+   ymax<-max(max(y[,i]),max(na.exclude(yhat)[,i]))
```

```
+   ts.plot(yhat[,i],main=paste("Time-domain MSE = ",
+   round(mse[i],3)," , Frequency-domain MSE = ",
+   round(perf_mat[i,1],3),", a1 = ",a_vec[i],sep=""),col="blue",
+         ylim=c(ymin,ymax),
+   gpars=list(xlab="", ylab=""))
+   lines(y[,i],col="red")
+   mtext("Real-time", side = 3, line = -1,at=len/2,col="blue")
+   mtext("target", side = 3, line = -2,at=len/2,col="red")
+ }
> invisible(dev.off())
```

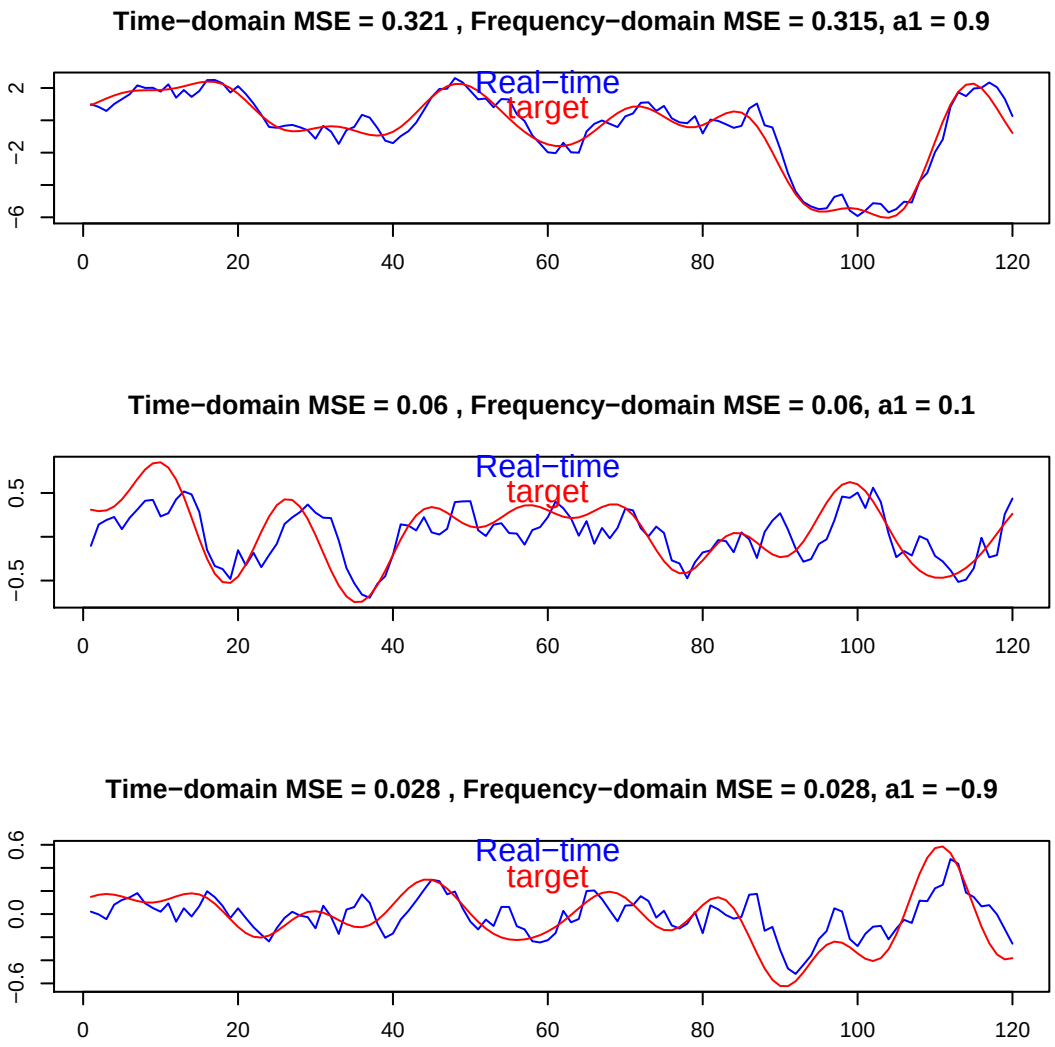


Figure 4.1: Real-time filter output (blue) vs. targets (red) for a1=0.9 (top), a1=0.1 (middle) and a1=-0.9 (bottom)

Visual inspection seems to conflict with MSE-performances: the real-time filter with the largest MSE (upper panel) appears to fit its target best. This conflict can be alleviated, to some extent, by adjusting for differences in scale of the time series[7]. But it is obvious that the task of the filter in the upper panel seems easier, in some way, than that of the bottom filter, whose output is much noisier than its target. A more refined analysis reveals, also, that the real-time estimates appear to be systematically shifted to the right: they are delayed. Once again, the upper filter seems least affected. In summary: the difficulty of the

[7] A relative (signal-to-noise) measure would seem more appropriate than the raw MSE, see later chapters.

estimation task seems to depend on the DGP as specified by the parameter a_1[8].

2. Compute and compare graphically amplitude and time-shift functions for all three realizations (processes), see fig.4.2[9].

```
> omega_k<-pi*0:(len/2)/(len/2)
> file = paste("z_dfa_ar1_amp_shift.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6,
+     height = 6)
> par(mfrow=c(2,2))
> amp<-abs(trffkt)
> shift<-Arg(trffkt)/omega_k
> plot(amp[,1],type="l",main="Amplitude functions",
+ axes=F,xlab="Frequency",ylab="Amplitude",col="black",ylim=c(0,1))
> lines(amp[,2],col="orange")
> lines(amp[,3],col="green")
> lines(Gamma,col="violet")
> mtext("Amplitude a1=0.9", side = 3, line = -1,at=len/4,col="black")
> mtext("Amplitude a1=0.1", side = 3, line = -2,at=len/4,col="orange")
> mtext("Amplitude a1=-0.9", side = 3, line = -3,at=len/4,col="green")
> mtext("Target", side = 3, line = -4,at=len/4,col="violet")
> axis(1,at=c(0,1:6*len/12+1),labels=c("0","pi/6","2pi/6","3pi/6",
+ "4pi/6","5pi/6","pi"))
> axis(2)
> box()
> plot(shift[,1],type="l",main="Time-shifts",
+ axes=F,xlab="Frequency",ylab="Shift",col="black",
+ ylim=c(0,max(na.exclude(shift[,3]))))
> lines(shift[,2],col="orange")
> lines(shift[,3],col="green")
> lines(rep(0,len/2+1),col="violet")
> mtext("Shift a1=0.9", side = 3, line = -1,at=len/4,col="black")
> mtext("Shift a1=0.1", side = 3, line = -2,at=len/4,col="orange")
> mtext("Shift a1=-0.9", side = 3, line = -3,at=len/4,col="green")
> mtext("Target", side = 3, line = -4,at=len/4,col="violet")
> axis(1,at=c(0,1:6*len/12+1),labels=c("0","pi/6","2pi/6","3pi/6",
+ "4pi/6","5pi/6","pi"))
> axis(2)
> box()
```

[8]Smaller a_1 correspond to noisier realizations x_t which, in turn, lead to noisier real-time estimates $\hat{y}_t$ (increasingly difficult estimation problems).

[9]Our plots are similar but not identical to the plots in DFA, section 4.1.1, exercise 1: here we use data in the middle of the long sample whereas in DFA the first 120 observations are used.

```
> plot(periodogram[,1],type="l",main="Periodograms",
+ axes=F,xlab="Frequency",ylab="Periodogram",col="black",
+ ylim=c(0,max(periodogram[,3])/6))
> lines(periodogram[,2],col="orange")
> lines(periodogram[,3],col="green")
> mtext("Periodogram a1=0.9", side = 3, line = -1,at=len/4,col="black")
> mtext("Periodogram a1=0.1", side = 3, line = -2,at=len/4,col="orange")
> mtext("Periodogram a1=-0.9", side = 3, line = -3,at=len/4,col="green")
> axis(1,at=c(0,1:6*len/12+1),labels=c("0","pi/6","2pi/6","3pi/6",
+ "4pi/6","5pi/6","pi"))
> axis(2)
> box()
> invisible(dev.off())
```

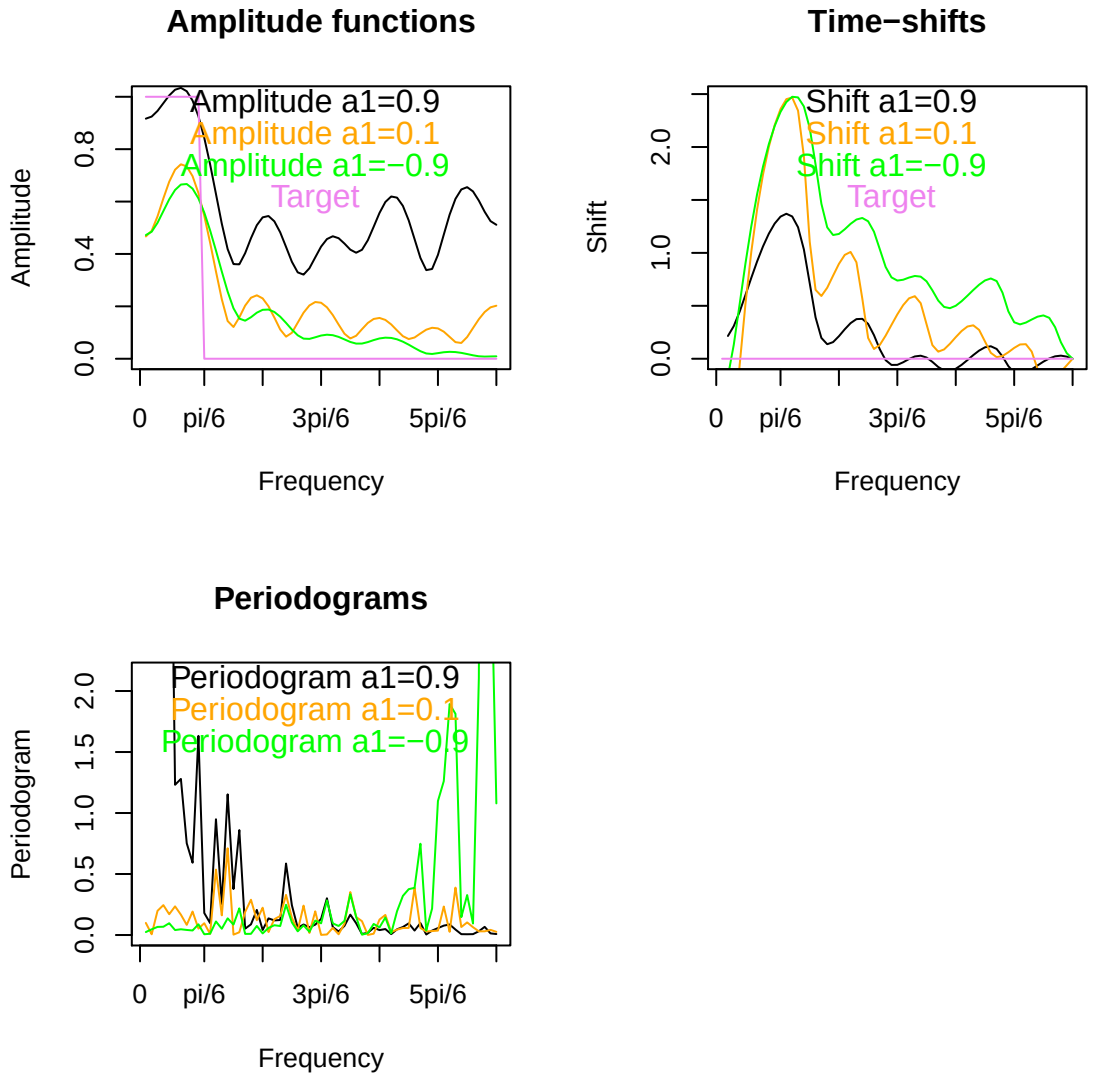


Figure 4.2: Amplitude (top left), time-shifts (top-right) and periodograms (bottom left) for a1=0.9 (black), a1=0.1 (orange) and a1=-0.9 (green)

- Noise and delay of the real-time estimates $\hat{y}_t$ previously observed in fig.4.1 are due to leaking amplitude functions (incomplete stop-band rejection) and to non-vanishing time-shift functions of the real-time filters, see fig.4.2. Chapter 7 proposes a more general optimization paradigm which will address these issues explicitly.
- The time-shift (top right panel) of the black filter ($a_1 = 0.9$) remains comparatively small. Its amplitude function (top left panel) is the farthest away from the target in the stop-band $\omega > \pi/6$ but it is closest to the target in the passband $\omega \leq \pi/6$: the optimization criterion seems to trade (poorer) high-frequency damping against (improved) passband properties. In summary: $\hat{\Gamma}(\cdot)$ tracks $\Gamma(\cdot)$ towards the loaded frequencies, as

measured by the periodogram (bottom panel) in 4.27. Similar findings apply to the other two processes.

4.3 MDFA: Problem-Structure and Target (MSE-Perspective)

4.3.1 Emphasizing the Filter Error

The previous (univariate) DFA has been generalized to a multivariate framework in Wildi (2008.2), theorem 7.1, and in McElroy-Wildi (2015) (MDFA-paper). We here briefly summarize the main results in the case of stationary processes (see chapters 16, 17 and 18 for generalizations to non-stationary processes).

Let the target y_t be defined by 4.22 and let x_t, w_{tj}, $t = 1, ..., T$ and $j = 1, ..., m$ be an $m+1$-dimensional set of explanatory variables[10]. Consider

$$\hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k) + \sum_{n=1}^{m} \hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k) \tag{4.29}$$

where

$$\hat{\Gamma}_X(\omega_k) = \sum_{j=0}^{L-1} b_{Xj} \exp(-ij\omega_k) \tag{4.30}$$

$$\hat{\Gamma}_{W_n}(\omega_k) = \sum_{j=0}^{L-1} b_{w_n j} \exp(-ij\omega_k) \tag{4.31}$$

are the (one-sided) transfer functions of the (real-time) filters, whose coefficients must be determined, and where $\Xi_{TX}(\omega_k)$, $\Xi_{TW_n}(\omega_k)$ are the corresponding DFTs of the data. The filter coefficients can be collected in a matrix $\mathbf{B} = (\mathbf{b}_X, \mathbf{b}_{w_1}, ..., \mathbf{b}_{w_m})$ where $\mathbf{b}_X = (b_{X0}, ..., b_{X,L-1})'$ and $\mathbf{b}_{w_n} = (b_{w_n 0}, ..., b_{w_n,L-1})'$ are the vectors of filter coefficients. Then the following multivariate MSE-criterion

$$\frac{2\pi}{T} \sum_{k=-T/2}^{T/2} \left| \left(\Gamma(\omega_k) - \hat{\Gamma}_X(\omega_k)\right) \Xi_{TX}(\omega_k) - \sum_{n=1}^{m} \hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k) \right|^2 \rightarrow \min_{\mathbf{B}} \tag{4.32}$$

generalizes the univariate DFA-MSE criterion 4.27, see theorem 7.1 in Wildi (2008.2) and McElroy-Wildi (2015)[11].

Remarks:

- Due to the explicit link between x_t and the target y_t, the former is generally informative about y_t. In some cases, however, we want to exclude x_t from the set of explanatory variables (for example if x_t is subject to large publication lags and/or large revisions, see chapter 14) and then we assume $\mathbf{b}_X = \mathbf{0}$ or, equivalently, $\hat{\Gamma}_X \equiv 0$.

[10] The explicit link between y_t and x_t, as defined by 4.22, justifies to distinguish x_t from the other explanatory series.

[11] MDFA-paper.

- In the absence of additional explanatory series (i.e. $m = 0$) the multivariate criterion 4.32 reduces to 4.27. The proposed (MSE-) MDFA-criterion thus generalizes the previous DFA.

4.3.2 One- and Multi-Step Ahead Forecast Criteria

The univariate and multivariate criteria 4.27 and 4.32 rely on a general target specification. The classic one-step ahead mean-square criterion could be replicated by specifying $\gamma_{-1} = 1, \gamma_k = 0, k \neq 0$ in 4.22:

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k} = x_{t+1}$$

In this case, criterion 4.32 becomes

$$\frac{2\pi}{T} \sum_{k=-T/2}^{T/2} \left| \left(\exp(i\omega_k) - \hat{\Gamma}_X(\omega_k) \right) \Xi_{TX}(\omega_k) - \sum_{n=1}^{m} \hat{\Gamma}_{W_n}(\omega_k) \Xi_{TW_n}(\omega_k) \right|^2 \rightarrow \min_{\mathbf{B}} \tag{4.33}$$

where the anticipative *allpass* target filter $\Gamma(\omega_k) := \exp(i\omega_k)$ rotates the DFT $\Xi_{TX}(\omega_k)$ in the frequency-domain (shifts the data in the time-domain). A direct link to classical (pseudo-) maximum likelihood approaches is provided in chapter 9 for details (replication of model-based performances by MDFA). To conclude, we note that h-step ahead forecasting could be obtained by specifying the allpass target $\Gamma_h(\omega_k) := \exp(ih\omega_k)$.

Remarks

- Typically, in the time-domain, h observations are lost when estimating the coefficients of a direct h-step ahead forecast equation. In contrast, the whole sample remains at disposal in the frequency-domain because time-shifts, of the data, are handled by rotations, of the (full-sample) DFTs.

- We here proposed forecasts of the *original* data. In section 5.3.1 we generalize this concept to forecasts, nowcasts and backcasts of arbitrary signals.

4.4 Matrix Notation and Generalized Least-Squares Solution

We here introduce a convenient matrix notation which will be useful when tackling filter constraints (see chapter 6) as well as more sophisticated optimization criteria (customization, regularization, mixed-frequency). We then derive the solution of the MSE-criterion 4.32 in closed-form.

4.4.1 Matrix Notation

Because of the symmetry of its summands around $\omega_0 = 0$, criterion 4.32 can be rewritten in a numerically more efficient form

$$\begin{aligned} & \frac{2\pi}{T}\left|\left(\Gamma(0)-\hat{\Gamma}_X(0)\right)\Xi_{TX}(0)-\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(0)\Xi_{TW_n}(0)\right|^2 \\ + \quad & 2\frac{2\pi}{T}\sum_{k>0}^{T/2}\left|\left(\Gamma(\omega_k)-\hat{\Gamma}_X(\omega_k)\right)\Xi_{TX}(\omega_k)-\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\right|^2 \rightarrow \min_{\mathbf{B}} \end{aligned} \tag{4.34}$$

Note that frequency zero is counted once, only, whereas all strictly positive frequencies are duplicated[12]. We now derive a more convenient vector notation for the above criterion.

Let $\mathbf{X}$ be a matrix whose k-th row $\mathbf{X}_k$ is defined as

$$\begin{aligned} \mathbf{X}_k \quad = \quad & \sqrt{1+I_{k>0}}\cdot \\ & \mathrm{Vec}_{\mathrm{row}}\begin{pmatrix} \Xi_{TX}(\omega_k) & \exp(-i\omega_k)\Xi_{TX}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TX}(\omega_k) \\ \Xi_{TW_1}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_1}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_1}(\omega_k) \\ \Xi_{TW_2}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_2}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_2}(\omega_k) \\ ... & ... & ... & ... \\ \Xi_{TW_m}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_m}(\omega_k & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_m}(\omega_k) \end{pmatrix} \end{aligned} \tag{4.35}$$

where the $\mathrm{Vec}_{\mathrm{row}}$-operator appends rows (we use this notation in order to avoid margin-overflow) and where the 'indicator' function $\sqrt{1+I_{k>0}} = \begin{cases} 1 & k=0 \\ \sqrt{2} & k=1,...,T/2 \end{cases}$ accounts for the fact that frequency zero occurs once only in the criterion[13]. The length of the k-th row is $(m+1)L$ and the dimension of the design-matrix $\mathbf{X}$ is $(T/2+1)*(m+1)L$. Next, define a coefficient vector $\mathbf{b}$ and a target vector $\mathbf{Y}$

$$\begin{aligned} \mathbf{b} = \mathrm{Vec}_{\mathrm{col}}(\mathbf{B}) \quad = \quad & \mathrm{Vec}_{\mathrm{col}}\begin{pmatrix} b_{X0} & b_{W_10} & b_{W_20} & ... & b_{W_m0} \\ b_{X1} & b_{W_11} & b_{W_21} & ... & b_{W_m1} \\ ... & ... & ... & ... & ... \\ b_{XL-1} & b_{W_1L-1} & b_{W_2L-1} & ... & b_{W_mL-1} \end{pmatrix} \\ \mathbf{Y} \quad = \quad & \begin{pmatrix} \Gamma(\omega_0)\Xi_{TX}(\omega_0) \\ \sqrt{2}\Gamma(\omega_1)\Xi_{TX}(\omega_1) \\ \sqrt{2}\Gamma(\omega_2)\Xi_{TX}(\omega_2) \\ . \\ \sqrt{2}\Gamma(\omega_{T/2})\Xi_{TX}(\omega_{T/2}) \end{pmatrix} \end{aligned}$$

where $\mathrm{Vec}_{\mathrm{col}}$ stacks the columns of the coefficient matrix $\mathbf{B}$. Note, once again, that all frequencies larger than zero are 'duplicated' (scaled by $\sqrt{2}$) in $\mathbf{Y}$. Criterion 4.32 or, equivalently 4.34, can now be expressed more conveniently in vector notation:

$$(\mathbf{Y}-\mathbf{X}\mathbf{b})'(\mathbf{Y}-\mathbf{X}\mathbf{b}) \rightarrow \min_{\mathbf{b}} \tag{4.36}$$

[12]The unequal weighting of frequency zero explains the modified DFT in exercise 1, section 4.6.

[13]Note that the square-root is required because we consider the DFT in 4.35 i.e. the square root of the expressions in 4.34.

where $(\mathbf{Y}-\mathbf{Xb})'$ is the Hermitian conjugate of $\mathbf{Y}-\mathbf{Xb}$ (transpose and complex conjugate)[14]. If all vectors and matrices were real (real numbers) then the solution to this minimization problem would be the well-known least-squares estimate

$$\hat{\mathbf{b}} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}$$

Unfortunately, the above vectors and matrices are complex-valued. Before presenting a correct least-squares estimate we propose to rotate all DFT's [15]: this transformation is not strictly necessary but it simplifies later expressions.

$$\begin{aligned}
& \frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\left|\left(\Gamma(\omega_k)-\hat{\Gamma}_X(\omega_k)\right)\Xi_{TX}(\omega_k)-\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\right|^2 \\
= & \frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\Big||\Gamma(\omega_k)\Xi_{TX}(\omega_k)|\exp\left(i*\arg\left(\Gamma(\omega_k)\Xi_{TX}(\omega_k)\right)\right)-\hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k) \\
& -\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\Big|^2 \\
= & \frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\Big||\Gamma(\omega_k)\Xi_{TX}(\omega_k)|-\hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k)\exp\left(-i*\arg\left(\Gamma(\omega_k)\Xi_{TX}(\omega_k)\right)\right) \\
& -\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\exp\left(-i*\arg\left(\Gamma(\omega_k)\Xi_{TX}(\omega_k)\right)\right)\Big|^2
\end{aligned} \tag{4.37}$$

where the arg-function corresponds to the phase (angle) of a complex number and where the function is applied component-by-component to the complex-valued vector. Let us define a rotated design-matrix $\mathbf{X}_{\text{rot}}$ and a rotated target vector $\mathbf{Y}_{\text{rot}}$

$$\mathbf{X}_{k,\text{rot}} = \mathbf{X}_k\exp\left(-i*\arg\left(\Gamma(\omega_k)\Xi_{TX}(\omega_k)\right)\right) \tag{4.38}$$

where $\mathbf{X}_{k,\text{rot}}$ designates the k-th row of $\mathbf{X}_{\text{rot}}$ and where

$$\mathbf{Y}_{\text{rot}} = |\mathbf{Y}| \tag{4.39}$$

is a (real) positive target vector.

4.4.2 Generalized Least Squares Solution

The optimization criterion becomes

$$(\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b})'(\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b}) \to \min_{\mathbf{b}} \tag{4.40}$$

The general (matrix derivative) formula for tackling this complex-valued minimization problem is[16]

$$\begin{aligned}
d/d\mathbf{b}\text{ Criterion} &= d/d\mathbf{b}\ (\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b})'(\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b}) \\
&= -(\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b})'\mathbf{X}_{\text{rot}}-(\mathbf{Y}_{\text{rot}}-\mathbf{X}_{\text{rot}}\mathbf{b})^{\mathbf{T}}\overline{\mathbf{X}_{\text{rot}}} \\
&= -2\mathbf{Y}'_{\text{rot}}\Re\left(\mathbf{X}_{\text{rot}}\right)+2\mathbf{b}'\Re(\mathbf{X}'_{\text{rot}}\mathbf{X}_{\text{rot}})
\end{aligned} \tag{4.41}$$

[14] For simplicity we omitted the normalization $\frac{2\pi}{T}$ which is irrelevant for optimization.
[15] The criterion 4.32 is invariant to such a transformation.
[16] A convenient survey of matrix derivatives is to be found on the wikipedia-site Matrix calculus.

where $\mathbf{X}_{\text{rot}}{}'$ is the Hermitian conjugate, $\mathbf{X}_{\text{rot}}^{\mathbf{T}}$ is the transposed and $\overline{\mathbf{X}_{\text{rot}}}$ is the complex conjugate matrix; $\Re(\cdot)$ means the real part of a complex number. The generalized least-squares estimate is obtained by equating the previous expression to zero

$$\hat{\mathbf{b}} \quad = \quad (\Re(\mathbf{X}'_{\text{rot}}\mathbf{X}_{\text{rot}}))^{-1}\,\Re(\mathbf{X}_{\text{rot}})'\mathbf{Y}_{\text{rot}} \tag{4.42}$$

The least-squares estimate $\hat{\mathbf{b}}$ is the solution of the MDFA-MSE signal extraction problem 4.32 (or 4.27 in the univariate case).

4.4.3 R-Code

DFA

The proposed matrix notation and the MSE (generalized least-squares) estimate $\hat{\mathbf{b}}$ can be traced-back in the R-code of the (MSE-) DFA proposed in section 1.2.2:

```
> dfa_ms

function (L, periodogram, Lag, Gamma)
{
    periodogram[1] <- periodogram[1]/2
    K <- length(periodogram) - 1
    X <- exp(-(0+1i) * Lag * pi * (0:(K))/(K)) * rep(1, K + 1) *
        sqrt(periodogram)
    X_y <- exp(-(0+1i) * Lag * pi * (0:(K))/(K)) * rep(1, K +
        1)
    for (l in 2:L) {
        X <- cbind(X, (cos((l - 1 - Lag) * pi * (0:(K))/(K)) +
            (0+1i) * sin((l - 1 - Lag) * pi * (0:(K))/(K))) *
            sqrt(periodogram))
        X_y <- cbind(X_y, (cos((l - 1 - Lag) * pi * (0:(K))/(K)) +
            (0+1i) * sin((l - 1 - Lag) * pi * (0:(K))/(K))))
    }
    xtx <- t(Re(X)) %*% Re(X) + t(Im(X)) %*% Im(X)
    b <- as.vector(solve(xtx) %*% (t(Re(X_y)) %*% (Gamma * periodogram)))
    trffkt <- 1:(K + 1)
    trffkt[1] <- sum(b)
    for (k in 1:(K)) {
        trffkt[k + 1] <- (b %*% exp((0+1i) * k * (0:(length(b) -
            1)) * pi/(K)))
    }
    return(list(b = b, trffkt = trffkt))
}
<bytecode: 0x0000000023820208>
<environment: namespace:MDFA>
```

The code replicates the above formulas up to the particular weighting of frequency zero (see exercise 1, section 4.6.1).

MDFA

The least-squares solution 4.42 is nested as a special case in the generic MDFA estimation routine introduced in section 1.2.3. The DFA is nested too, see section 4.6 (replication). Other nested solutions replicate classic model-based approaches, see chapter 9. Conceptually, 'nesting' is obtained by specifying hyperparameters in the head of the function call.

4.5 Grand-Mean Parametrization

This particular feature has been discontinued and has been substituted by the more powerful Regularization Troika, see chapter 13. Nevertheless, one may find vestiges in our R-package (and the functionality could still be activated). Therefore we briefly review the main concepts.

The term 'Grand-Mean' refers to a reparametrization or recoding of the filter coefficients whereby the original coefficients are expressed as deviations about a central (grand-mean) value. Specifically, let b_l^u, $l = 0, ..., L-1$, $u = 0, ..., m$ denote the lag-l coefficient assigned to series u, whereby $u = 0$ corresponds to x_t and $u = 1, ..., m$ stands for w_{tu}. The coefficients could be rewritten in the form

$$b_l^u = \begin{cases} b_l + \delta b_l^u, & u > 0 \\ b_l - \sum_{u=1}^m \delta b_l^u, & u = 0 \end{cases} \tag{4.43}$$

where b_l, $l = 0, ..., L-1$ is a central (grand-mean) vector of coefficients and where δb_l^u are the deviations about the central coefficient-vector. For obvious reasons we here assume $m > 0$ (truly multivariate design). Note that 4.43 imposes $\delta b_l^0 = -\sum_{u=1}^m \delta b_l^u$ which means that b_l is in the 'center' of the coefficients. Similarly, δb_l^u, $u > 0$ can be interpreted as series-specific 'effects'. We can express the above re-parametrization in terms of

$$\begin{aligned} \mathbf{b} &= \mathbf{A}\tilde{\mathbf{b}} \qquad (4.44) \\ \mathbf{A} &= \begin{pmatrix} \mathbf{Id} & -\mathbf{Id} & -\mathbf{Id} & ... & ... & -\mathbf{Id} \\ \mathbf{Id} & \mathbf{Id} & \mathbf{0} & \mathbf{0} & ... & \mathbf{0} \\ \mathbf{Id} & \mathbf{0} & \mathbf{Id} & \mathbf{0} & ... & \mathbf{0} \\ ::: & & & & & \\ \mathbf{Id} & \mathbf{0} & \mathbf{0} & ... & ... & \mathbf{Id} \end{pmatrix} \\ \mathbf{b}' &= \left(b_0^0, b_1^0, ..., b_{L-1}^0 \mid b_0^1, b_1^1, ..., b_{L-1}^1 \mid ... \mid ... \mid ... \mid b_0^m, b_1^m, ..., b_{L-1}^m\right)' \\ \tilde{\mathbf{b}}' &= \left(b_0, b_1, ..., b_{L-1} \mid \delta b_0^1, \delta b_1^1, ..., \delta b_{L-1}^1 \mid \delta b_0^2, \delta b_1^2, ..., \delta b_{L-1}^2 \mid ... \mid ... \mid ... \mid \delta b_0^m, \delta b_1^m, ..., \delta b_{L-1}^m\right)' \end{aligned}$$

where $\mathbf{Id}$ is an $L * L$ identity. In principle, a separation of the series-specific effects from the grand-mean would point towards an explicit control of the cross-sectional adaptivity of the multivariate filter (for example by imposing a zero-shrinkage of δb_l^u). In this context we may refer to chapter 13 for a more general approach and therefore we defer a more comprehensive discussion and treatment of the topic.

4.6 Replication of DFA by MDFA

We rely on the data in the previous exercises, see section 4.2.6, and *replicate* the obtained DFA-results (criterion 4.27) by MDFA (criterion 4.32).

4.6.1 Exercises

1. Define the data-matrix (target and explanatory variable) and compute the DFTs. For ease of exposition we consider the first AR(1)-process only ($a_1 = 0.9$).

```
> # Select the first process
> i_process<-1
> # Define the data-matrix:
> # The first column must be the target series.
> # Columns 2,3,... are the explanatory series. In a univariate setting
> # target and explanatory variable are identical
> data_matrix<-cbind(x[,i_process],x[,i_process])
> # Determine the in-sample period (fully in sample)
> insample<-nrow(data_matrix)
> # Compute the DFT by relying on the multivariate DFT-function:
> #   d=0 for stationary data (default settings)
> weight_func<-spec_comp(insample, data_matrix, d)$weight_func
```

2. Estimate optimal filter coefficients and compare DFA- and MDFA-estimates.

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Estimate filter coefficients:
> mdfa_obj<-mdfa_analytic(L, lambda, weight_func, Lag, Gamma, eta, cutoff, i1,i2, weight_con
> # Filter coefficients: compare MDFA and previous DFA
> b_mat<-cbind(mdfa_obj$b,b[,i_process])
> dimnames(b_mat)[[2]]<-c("MDFA","DFA")
> dimnames(b_mat)[[1]]<-paste("lag ",0:(L-1),sep="")
> as.matrix(round(b_mat,5))
```

```
           MDFA      DFA
lag 0   0.53821  0.53821
lag 1   0.10039  0.10039
lag 2   0.17419  0.17419
lag 3   0.11221  0.11221
lag 4   0.08075  0.08075
lag 5   0.01972  0.01972
lag 6   0.05718  0.05718
lag 7  -0.03330 -0.03330
```

```
lag 8  -0.04889 -0.04889
lag 9  -0.03821 -0.03821
lag 10 -0.08752 -0.08752
lag 11  0.04178  0.04178
```

The MDFA replicates DFA, as desired. Note that we could refer to the context-specific $MDFA\_mse$ function

```
> mdfa_obj_mse<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
```

which abbreviates the lengthy list of arguments of the generic *mdfa_analytic*-call to those required in a MSE-framework. As can be seen, estimated coefficients are identical:

```
> b_mat<-cbind(b_mat,mdfa_obj_mse$b)
> dimnames(b_mat)[[2]][3]<-"MDFA_mse"
> dimnames(b_mat)[[1]]<-paste("lag ",0:(L-1),sep="")
> head(as.matrix(round(b_mat,5)))
```

```
         MDFA     DFA MDFA_mse
lag 0 0.53821 0.53821  0.53821
lag 1 0.10039 0.10039  0.10039
lag 2 0.17419 0.17419  0.17419
lag 3 0.11221 0.11221  0.11221
lag 4 0.08075 0.08075  0.08075
lag 5 0.01972 0.01972  0.01972
```

3. The value of the multivariate criterion 4.32 is computed explicitly by the MDFA-function:

```
> # Criterion value
> criterion_mdfa<-mdfa_obj$MS_error
> # DFA-numbers are stored in perf_mat
> crit_mdfa<-matrix(c(criterion_mdfa,perf_mat[i_process,1],
+                     perf_mat[i_process,2]),ncol=1)
> dimnames(crit_mdfa)[[1]]<-c("MDFA criterion",
+                             "DFA criterion","sample MSE")
> dimnames(crit_mdfa)[[2]]<-"MSE estimates"
> t(round(crit_mdfa,3))
```

```
              MDFA criterion DFA criterion sample MSE
MSE estimates          0.314         0.315      0.321
```

4.7 Qualitative Easing by Leading Indicators: an Empirical Study

We attempt to quantify performance gains obtained by inclusion of a leading indicator into the univariate design of the previous section. Specifically, we construct a new explanatory series w_{1t}

$$w_{1t} = x_{t+\delta} + s \cdot \epsilon_{t+\delta} \tag{4.45}$$

where x_t is the data of the previous univariate design, ϵ_t is an idiosyncratic noise component (iid zero-mean Gaussian standardized), s is a scaling[17]) and δ is a time-shift. In the first exercise, section 4.7.1, we select $s = 0.1$ (a weak idiosyncratic component) and $\delta = 1$ (lead by one time unit). Alternative settings are analyzed in section 4.7.2.

4.7.1 Bivariate MDFA vs. Univariate DFA

1. Use the data in the previous section 4.6, construct the leading indicator 4.45 and specify the (3-dim) data-matrix.

```
> set.seed(12)
> # Select the AR(1)-process with coefficient 0.9
> i_process<-1
> # Scaling of the idiosyncratic noise
> scale_idiosyncratic<-0.1
> eps<-rnorm(nrow(xh))
> indicator<-xh[,i_process]+scale_idiosyncratic*eps
> # Data: first column=target, second column=x,
> #   third column=shifted (leading) indicator
> data_matrix<-cbind(xh[,i_process],xh[,i_process],c(indicator[2:nrow(xh)],NA))
> dimnames(data_matrix)[[2]]<-c("target","x","leading indicator")
> # Extract 120 observations from the long sample
> data_matrix_120<-data_matrix[lenh/2+(-len/2):((len/2)-1),]
> head(round(data_matrix_120,4))

     target      x leading indicator
[1,] 1.0687 1.0687            0.8137
[2,] 0.9497 0.9497            0.6150
[3,] 0.6628 0.6628            1.7381
[4,] 1.6465 1.6465            1.6748
[5,] 1.7680 1.7680            1.7033
[6,] 1.8317 1.8317            2.4977
```

The first two series are identical; the new third one leads by one-time unit and it is contaminated by noise.

[17] A larger s implies that the indicator is less informative about the target y_t.

2. Compute the DFTs of the data.

```
> # Fully in sample
> insample<-nrow(data_matrix_120)
> # d=0 for stationary series: see default settings
> weight_func<-spec_comp(insample, data_matrix_120, d)$weight_func
```

3. Estimate optimal (MSE-) filter coefficients.

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Estimate filter coefficients
> mdfa_obj<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
> # Filter coefficients
> b_mat<-mdfa_obj$b
> dimnames(b_mat)[[2]]<-c("x","leading indicator")
> dimnames(b_mat)[[1]]<-paste("Lag ",0:(L-1),sep="")#dim(b_mat)
> head(b_mat)
```

```
                x leading indicator
Lag 0 0.20556332        0.39969599
Lag 1 0.35970890       -0.08021796
Lag 2 0.21659593       -0.18695421
Lag 3 0.14359475       -0.06108555
Lag 4 0.13724690       -0.02475913
Lag 5 0.06399915       -0.09717151
```

The top-coefficient (the first six only are shown) in the second column assigns most weight to the last observation of the leading-indicator: this observation is particular because it leads the original data. In direct comparison with the first column, the other coefficients are generally smaller (in absolute value) because the leading-indicator is contaminated by noise (the coefficients in the first column must be shifted down in order to make meaningful cross-sectional comparisons).

4. Compute the minimal criterion value.

```
> # Criterion value
> round(mdfa_obj$MS_error,3)
```

```
[1] 0.145
```

The new criterion value is substantially smaller than the DFA (0.315, see previous exercise): efficiency gains exceed 50% (reduction of mean-square filter error).

5. Verify that the mean-square sample error is indeed smaller (recall that the sample MSE is generally not observable because it involves knowledge of the target y_t).

- Apply the (bivariate) filter to the data

```
> yhat_multivariate_leading_indicator<-rep(NA,len)
> for (j in 1:len)
+   yhat_multivariate_leading_indicator[j]<-sum(apply(b_mat*
+                   data_matrix[lenh/2+(-len/2)-1+j:(j-L+1),2:3],1,sum))
```

- Derive the (time-domain) sample mean-square error and compare performances with the DFA, above.

```
> y_target_leading_indicator<-y[,i_process]
> perf_mse<-matrix(c(mean(na.exclude((yhat_multivariate_leading_indicator-
+           y_target_leading_indicator))^2),
+           mean(na.exclude((yhat[,i_process]-
+           y_target_leading_indicator))^2)),nrow=1)
> dimnames(perf_mse)[[2]]<-c("bivariate MDFA","DFA")
> dimnames(perf_mse)[[1]]<-"Sample MSE"
> round(perf_mse,3)

           bivariate MDFA   DFA
Sample MSE          0.139 0.321
```

Sample MSE-performances of the bivariate design are substantially improved, as expected: the leading indicator is informative (this result depends on the magnitude of the idiosyncratic noise in the construction of the indicator, of course). Also, the criterion value 0.145 is fairly close to the sample MSE 0.139, as desired.

6. Plot and compare target y_t as well as DFA and MDFA real-time estimates $\hat{y}_t$.

```
> file = paste("z_mdfadfa_ar1_sym_output.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> i<-1
> ymin<-min(min(y[,i]),min(na.exclude(yhat)[,i]))
> ymax<-max(max(y[,i]),max(na.exclude(yhat)[,i]))
> ts.plot(yhat[,i],main=paste("Sample MSE MDFA: ",ylab="",
+ round(perf_mse[1],3),", DFA: ",round(perf_mse[2],3),sep=""),col="blue",
+       ylim=c(ymin,ymax))
> lines(y[,i],col="red")
> lines(yhat_multivariate_leading_indicator,col="green")
> mtext("DFA", side = 3, line = -2,at=len/2,col="blue")
> mtext("target", side = 3, line = -1,at=len/2,col="red")
> mtext("MDFA", side = 3, line = -3,at=len/2,col="green")
> invisible(dev.off())
```

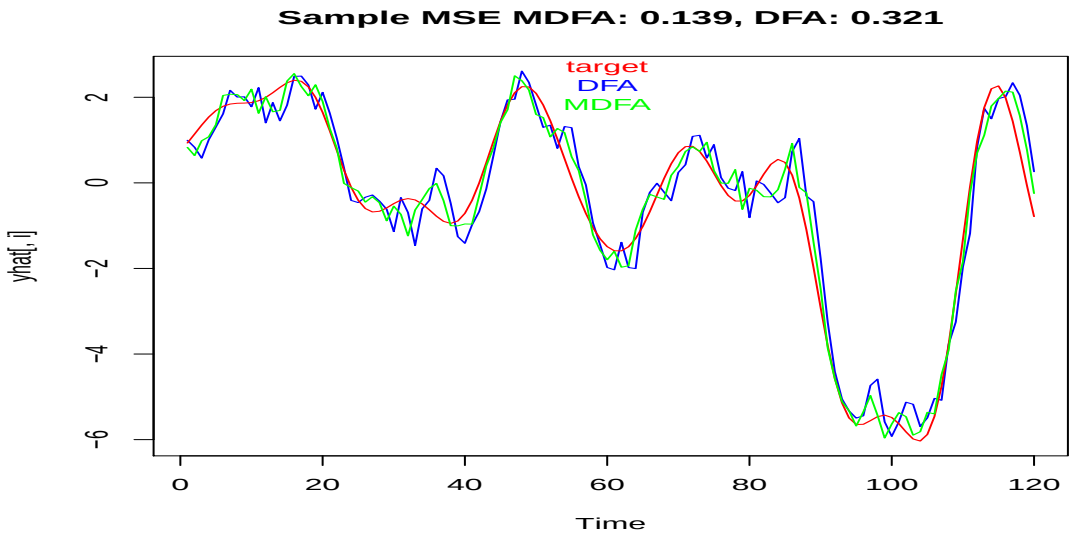


Figure 4.3: Target (red) vs. DFA (blue) and bivariate MDFA (green) for the first process (a1=0.9)

The MDFA-output (green) is both smoother and faster than the DFA (blue): noisy ripples are smaller in magnitude and turning points can be detected earlier (lead).

7. Verify that filter coefficients of the bivariate design are optimal: this is left as an exercise to the reader (any other set of bivariate filter coefficients, of length 12, should increase the criterion value 0.145 and/or the sample MSE 0.139).

4.7.2 Measuring Lead and Signal-to-Noise Effects of a Leading Indicator

Empirical evidence or experience suggest that 'lead' and 'signal-to-noise ratio' are conflicting requirements: faster indicators are generally noisier. We here attempt to disentangle and to quantify both effects, in terms of MSE-performances, by relying on our previous toy-model

$$w_{1t} = x_{t+\delta} + s \cdot \epsilon_{t+\delta}$$

For fixed x_t we can alter the shift δ and the (inverse) signal-to-noise ratio s. To be more specific, we want to analyze non-integer shifts $\delta_j = j/4, j = 0, 1, 2, 3, 4$[18] in order to be able to quantify *weekly* up-dating effects of a *monthly* indicator in the framework of a mixed-frequency approach, see chapter 15[19]. In the context of the following exercise we are interested in quantifying gains obtained by up-dating information on a weekly basis as a function of the signal-to-noise ratio of the high-frequency (weekly) indicator.

[18] Arbitrary δ can be implemented very easily in the frequency-domain because shifts become rotations of the data.

[19] A fractional lead of $\delta = 1/4$ corresponds to an anticipation by one week in an otherwise monthly framework: this lead by one week could be obtained by exploiting weekly up-dates of a 'high-frequency' weekly indicator correlating with the interesting monthly target series.

1. Specify candidate time-shifts $\delta_j = j/4, j = 0, ..., 4$ and (inverse) signal-to-noise ratios $\mathbf{s} = (0, 0.1, 0.5, 1, 2)/\sqrt{Var(x_t)}$

```
> # Inverse SNR: the variance of the standardized noise is one:
> #   we thus normalize by the standard deviation of the data x
> #   (second column of the data matrix)
> scale_idiosyncratic_vec<-c(0,0.1,0.5,1,2)/sqrt(var(data_matrix_120[,2]))
> # We select fractional leads: multiples of 0.25
> #   A fractional lead of 0.25 corresponds roughly to a week
> #   on a monthly time scale
> delta_vec<-0.25*0:4
```

2. Generate leading indicators for all combinations of (δ_j, s_i) and compute corresponding mean-square filter errors (criterion values).

```
> # Initialize the performance matrix
> lead_snr_mat<-matrix(ncol=length(scale_idiosyncratic_vec),
+                      nrow=length(delta_vec))
> dimnames(lead_snr_mat)[[2]]<-paste("1/SNR=",
+             sqrt(var(data_matrix_120[,1]))*scale_idiosyncratic_vec,paste="")
> dimnames(lead_snr_mat)[[2]][1]<-paste("Univ. design: ",
+             dimnames(lead_snr_mat)[[2]][1],sep="")
> dimnames(lead_snr_mat)[[1]]<-paste("Lead ",delta_vec,paste="")
> # Generate the idiosyncratic noise
> set.seed(20)
> eps<-rnorm(nrow(data_matrix_120))
> # Loop over all combinations of leads and SNR-ratios
> for (i in 1:length(scale_idiosyncratic_vec))#i<-1
+ {
+   for (j in 1:length(delta_vec))#j<-1
+   {
+ # Add the (suitably scaled) noise: no lead yet.
+     indicator<-data_matrix_120[,2]+scale_idiosyncratic_vec[i]*eps
+ # Overwrite the indicator column with the new time series
+     data_matrix_120[,3]<-indicator
+ # Compute the DFTs (full in-sample, for stationary series d=0)
+     insample<-nrow(data_matrix_120)
+     weight_func<-spec_comp(insample, data_matrix_120, d)$weight_func
+ # Compute the discrete frequency-grid omega_k: from zero to pi
+     omega_k<-(0:(nrow(weight_func)-1))*pi/(nrow(weight_func)-1)
+ # Introduce the fractional time-shift by rotation of the DFT
+ #   of the indicator (last column)
+     weight_func[,ncol(weight_func)]<-exp(-1.i*delta_vec[j]*omega_k)*
```

```
+                   weight_func[,ncol(weight_func)]
+ # If the idiosyncratic noise is zero, then we use a univariate design
+   if (i==1)
+      weight_func<-weight_func[,-2]
+ # Compute optimal filters and derive the (frequency-domain) MSE
+     mdfa_obj<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
+ # Store the MSE
+     lead_snr_mat[j,i]<-mdfa_obj$MS_error
+   }
+ }
```

3. Collect all MSEs (criterion values) in table 4.2 and analyse the obtained results.

	Univ. design: 1/SNR= 0	1/SNR= 0.1	1/SNR= 0.5	1/SNR= 1	1/SNR= 2
Lead 0	0.314	0.263	0.263	0.263	0.263
Lead 0.25	0.261	0.173	0.229	0.254	0.264
Lead 0.5	0.215	0.139	0.188	0.215	0.248
Lead 0.75	0.177	0.124	0.157	0.179	0.220
Lead 1	0.146	0.121	0.129	0.146	0.189

Table 4.2: Effect of lead and of (inverse) signal-to-noise ratio on filter MSE

Analysis: Design

- If the noise and the lead vanish ($s = \delta = 0$), then the design is singular because

$$w_{1t} = x_t + s\epsilon_t = x_t$$

  Therefore we have to skip one of the redundant explanatory variables in the DFT.

- If the idiosyncratic noise component vanishes but the lead does not ($s = 0, \delta > 0$) then, in principle, the data is not perfectly colinear, at least in the time-domain. In the frequency-domain, though, the two DFT-columns are linearly dependent (rotation) and therefore the proposed design is still singular[20]. Therefore, all results in the first column correspond to *univariate* designs, where the single explanatory variable is the *noise-free* leading indicator.

- Performances reported in columns 2-5 of the first row (0.263: $\delta = 0$) are identical because all designs are strictly equivalent in informational terms: one can subtract x_t (second data-column) from $w_{1t} = x_t + s\epsilon_t$ to obtain $s\epsilon_t$ i.e. the data-matrix (x_t, w_{1t}) could be substituted by (x_t, ϵ_t) for all $s \neq 0$.

- Since ϵ_t is independent of x_t (different random seed) one expects that performances of bivariate designs (columns 2-5) and of the univariate design (column 1) in the first

[20]The DFT assumes that the data is periodic: therefore original and shifted data are perfectly colinear. The singularity could be avoided by computing DFTs of original and shifted series explicitly (instead of rotating the DFT).

row ($\delta = 0$) should be identical, at least in theory. In practice, the spurious decrease 0.263−0.314=-0.05 of the bivariate designs is entirely due to *overfitting*, see chapter 13 for a comprehensive development of the topic.

Analysis: Results

- Column 1 in the above table corresponds to a *univariate* design; columns 2-5 are *bivariate* designs. The first column measures time-shift effects of a single *noise-free* leading indicator[21]. Columns 2-4 consider a classic bivariate design, where the original data is augmented by a noisy leading indicator.
- The top-left number (0.314) (univariate design without lead) replicates performances as reported in section 4.6.
- The MSE generally decreases with increasing lead (increasing δ) and/or with decreasing (inverse) signal-to-noise ratio (smaller s, except the degenerate case $s = 0$: overfitting). A stronger noise could be compensated, to some extent, by a larger lead, at least in a mean-square perspective. As an example, an increase of the (inverse) SNR from 0.1 to 1 could be compensated by a relative lead by half a month, see cells (3,2) and (5,4) in the above table.
- Leads larger than one time unit (one month) do not seem to add significant value if the noise is weak (second column). If the noise is strong (last column) larger leads may be required in order to compensate for the augmented noise[22].
- Ignoring weekly up-dates of the filter-output $\hat{y}_t$, within the running month, could be costly in terms of performance-losses: if the noise component is weak (second column) then a lead of 0.25 instead of 0 (up-date at the week following the monthly release) would reduce MSE from 0.263 to 0.173 under the above experimental setting. If noise and signal are equally strong (4-th column) then an up-date of $\hat{y}_t$ in the middle of the month (lead 0.5 instead of 0) would reduce the MSE from 0.263 to 0.215.

These results suggests pertinence of a mixed-frequency approach for which the filter output is aligned on an in-flowing high-frequency data-stream and continuously up-dated in real-time, see chapter 15.

4.8 Summary

- We emphasized MSE-performances of unconstrained, unregularized and stationary designs.
- We provided a DFA-reminder and confirmed empirical pertinence of the abstract uniform superconsistency argument. We illustrated and interpreted the univariate DFA-criterion.

[21] The DFA cannot be used to replicate these results if the explanatory variable is shifted because target and explanatory variables wouldn't be identical.

[22] Stronger noise suppression by the filter induces larger delays of the output signal which must be compensated by an increasing lead of the series.

- We generalized the DFA-criterion to a multivariate framework and derived a closed-form solution.
- We replicated the DFA by the more general MDFA and we quantified efficiency gains obtained by a bivariate leading-indicator design (over the univariate approach).
- Our results suggested pertinence and practical relevance of a mixed-frequency approach, for which filter-outputs would be up-dated along a high-frequency data-stream.

For ease of exposition, all examples emphasized in-sample and single realization experiments. Empirical distributions of in-sample as well as of out-of-sample performances are to be found in chapters 7 and following.

Chapter 5

Optimal Time-Dependent Filter Sequences

5.1 Introduction

Until yet we analyzed real-time *fixed* filters

$$\hat{y}_t = \sum_{k=0}^{L-1} b_k x_{t-k} \ , \ t = L, ..., T$$

whose coefficients b_k did not dependent on time t. In practice, however, it is not uncommon to rely on *sequences* of time-dependent filters, whereby each filter in the sequence is optimized for a particular time point t in the sample. In such a case, adding a new observation x_{T+1} generally affects all (or part of the) previous estimates $\hat{y}_t$, for $t = L, ..., T$, because x_{T+1} might provide new evidences about the 'true' signal y_t in $t = L, ..., T$. Preliminary estimates $\hat{y}_t$, $t = L, ..., T$ are revised as new information x_{T+1} becomes available so that $\hat{y}_t|_{\{x_1,...,x_T\}} \neq \hat{y}_t|_{\{x_1,...,x_{T+1}\}}$, in general. In this chapter we explore optimal time-dependent filter-sequences and their associated revision-sequences. Section 5.2 provides a brief digression about data revisions and nails the topic of the chapter; the main concepts such as nowcasting, backcasting, revisions, filter-vintages and tentacle plots are introduced in section 5.3; finally, section 5.4 applies the novel findings to a bivariate leading-indicator design.

5.2 Data Revisions

In the above brief introduction we implicitly assumed the data $x_1, ..., x_T$ to be fixed or error-free. However, in practice, many important economic aggregates are revised over time. As an example, table 5.1 provides a snapshot of GDP towards the great recession[1]:

[1]The data can be accessed via the Philadelphia FED: `http://www.philadelphiafed.org/research-and-data/real-time-center/real-time-data/data-files/ROUTPUT/` or via Quandl.

```
> # Data is included in MDFA-package
> #US_GDP<-read.csv(paste(path.dat,"US_GDP.csv",sep=""),header=T)
> #US_GDP_wp<-read.csv(paste(path.dat,"US_GDP_wp.csv",sep=""),header=T,sep=";")
```

	US.GDP	X09Q1	X10Q1	X11Q1	X12Q1	X13Q1
1	2007:Q4	-0.04%	0.53%	0.72%	0.42%	0.42%
2	2008:Q1	0.22%	-0.18%	-0.18%	-0.44%	-0.44%
3	2008:Q2	0.70%	0.36%	0.15%	0.33%	0.33%
4	2008:Q3	-0.13%	-0.68%	-1.01%	-0.93%	-0.93%
5	2008:Q4	-0.96%	-1.37%	-1.74%	-2.30%	-2.30%
6	2009:Q1		-1.65%	-1.24%	-1.71%	-1.34%
7	2009:Q2		-0.18%	-0.18%	-0.17%	-0.08%
8	2009:Q3		0.55%	0.40%	0.42%	0.36%
9	2009:Q4		1.40%	1.23%	0.94%	0.99%
10	2010:Q1			0.92%	0.97%	0.58%
11	2010:Q2			0.43%	0.93%	0.56%
12	2010:Q3			0.63%	0.62%	0.64%
13	2010:Q4			0.78%	0.58%	0.59%
14	2011:Q1				0.09%	0.02%
15	2011:Q2				0.33%	0.61%
16	2011:Q3				0.45%	0.32%
17	2011:Q4				0.68%	1.01%
18	2012:Q1					0.49%
19	2012:Q2					0.31%
20	2012:Q3					0.77%
21	2012:Q4					-0.04%

Table 5.1: US-GDP: yearly vintages starting in Q1 2009 and ending in Q1 2013

Columns correspond to publication dates: the first column was published in the first quarter 2009 and the last column was published in the first quarter 2013. Rows correspond to historical time. The fifth row (2008:Q4) addresses GDP in the last quarter 2008: the initial estimate in the first column, namely -0.96%, is successively revised accross columns 2-5, as new information in subsequent years becomes available. Three years after the first estimate was released (fourth column), the figure stabilized at -2.3%[2]. Evidently, this data-specific error source generally affects the quality of real-time filter estimates. However, in order to avoid confusions, we shall assume the data to be error-free (fixed) in the remainder of this chapter. Accordingly, we focus on revisions solely imputable to (optimal) time-dependent filter-sequences. A comprehensive analysis of optimal filtering in the presence of data revisions is proposed in chapter 14.

[2]The reader is refered to the BEA, Bureau of Economic Analysis, for a comprehensive analysis of magnitude and source of the revisions.

5.3 Optimal Filter Sequences

5.3.1 Forecasting, Nowcasting and Smoothing

For simplicity of exposition and ease of notation we adopt a univariate framework; extensions to the multivariate case are straightforward. Let the target y_t be specified by 4.22

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k}$$

Until now we seeked filter coefficients $b_0, ..., b_{L-1}$ such that $\hat{y}_t = \sum_{k=0}^{L-1} b_k x_{t-k}$ is close to y_t in mean-square (a so-called nowcast). But we could have targeted y_{t+1} (forecast) or y_{t-1} (backcast), instead; more generally, we could be interested in estimating y_{t+h} by relying on data $x_t, ..., x_{t-L-1}$ where $h \in \mathbb{Z}$. In this more general perspective, we aim at finding filter coefficients b_{kh}, $k = h, ..., L-1+h$ such that the finite sample estimate

$$\hat{y}_t^h := \sum_{k=h}^{L-1+h} b_{kh} x_{t-k} \tag{5.1}$$

is 'closest possible' to y_t, $h \in \mathbb{Z}$, in *mean-square*

$$E\left[(y_t - \hat{y}_t^h)^2\right] \to \min_{\mathbf{b}_h}$$

where $\mathbf{b}_h = (b_{hh}, ..., b_{L-1+h,h})$.

- If $h = 0$ we use data $x_t, ..., x_{t-(L-1)}$ for estimating y_t: $\hat{y}_t^0$ is a *nowcast* of y_t and b_{k0}, $k = 0, ..., L-1$ is a real-time filter.

- If $h = 1$ we use data $x_{t-1}, ..., x_{t-L}$ for estimating y_t: $\hat{y}_t^1$ is a *forecast* of y_t and $b_{k,1}$, $k = 1, ..., L$ is a forecast filter.

- If $h = -1$ we use data $x_{t+1}, ..., x_{t-(L-2)}$ for estimating y_t: $\hat{y}_t^{-1}$ is a *backcast* of y_t and $b_{k,-1}$, $k = -1, ..., L-2$ is a smoother.

In contrast to classical one-step ahead forecasting which emphasize the original data, see section 4.3.2, we here extend the concept to general signal specifications: we forecast, nowcast or backcast the output y_t of a possibly bi-infinite filter $\Gamma(\cdot)$. In this more general perspective the proposed nowcast- and backcast-problems are non-trivial estimation tasks.

Intuitively, a backcast should improve (the filter-MSE should decrease) with decreasing horizon $h < 0$ because future information $x_{t+1}, ..., x_{t-h}$ becomes available for tracking the historical target y_t. We now analyze these effects: section 5.3.2 emphasizes filter characteristics (amplitude and time-shifts) as a function of h; filter vintages and the revision error are discussed in section 5.3.3; section 5.3.4 proposes a convenient graphical summary, the so called tentacle plot.

5.3.2 Backcasting: Analysis of Filter Characteristics

We here analyze amplitude and time-shift functions of univariate DFA-filters as a function of $h \leq 0$. For this purpose we rely on the empirical design introduced in section 4.2.6. Specifically, we estimate optimal filters for $h = 0, -1, ..., -6$ for the three stationary processes

$$\left.\begin{array}{rcl} x_t &=& 0.9x_{t-1} + \epsilon_t \\ x_t &=& 0.1x_{t-1} + \epsilon_t \\ x_t &=& -0.9x_{t-1} + \epsilon_t \end{array}\right\} \tag{5.2}$$

The target is the ideal (bi-infinite) lowpass filter with cutoff $\pi/6$. The horizon parameter $h \leq 0$ corresponds to the *Lag*-variable in the head of the DFA-function call

```
> head(dfa_ms)

1 function (L, periodogram, Lag, Gamma)
2 {
3     periodogram[1] <- periodogram[1]/2
4     K <- length(periodogram) - 1
5     X <- exp(-(0+1i) * Lag * pi * (0:(K))/(K)) * rep(1, K + 1) *
6         sqrt(periodogram)
```

Note however that $Lag = -h$ by convention i.e. a positive *Lag* means a backcast.

1. Compute optimal finite sample filters for $Lag = 0, ..., (L-1)/2$ for the above three processes. Hint: we set $L = 13$ in order to obtain a symmetric filter in $Lag = 6$

```
> L<-13
> yhat_Lag<-array(dim=c(len,3,L/2+2))
> trffkt<-array(dim=c(len/2+1,3,L/2+2))
> b<-array(dim=c(L,3,L/2+2))
> # Compute real-time filters for Lag=0,...,L/2 and for the
> #   above three AR-processes
> for (i in 1:3)
+ {
+   periodogram[,i]<-per(x[,i],plot_T)$per
+   for (Lag in 0:((L/2)+1))
+   {
+ # Optimize filters
+     filt<-dfa_ms(L,periodogram[,i],Lag,Gamma)
+     trffkt[,i,Lag+1]<-filt$trffkt
+     b[,i,Lag+1]<-filt$b
+ # Compute outputs
+     for (j in L:len)
+       yhat_Lag[j,i,Lag+1]<-filt$b%*%x[j:(j-L+1),i]
```

```
+   }
+ }
```

2. Focus on the second process ($a_1 = 0.1$) and analyze the outcome as a function of Lag, see fig.5.1.

```
> # Discrete frequency grid
> omega_k<-pi*0:(len/2)/(len/2)
> colo<-rainbow(L/2+2)
> file = paste("z_dfa_ar1_amp_shift_Lag_0.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> par(mfrow=c(2,2))
> amp<-abs(trffkt)
> shift<-Arg(trffkt)/omega_k
> for (i in 2:2)
+ {
+   ymin<-min(amp[,i,],na.rm=T)
+   ymax<-max(amp[,i,],na.rm=T)
+   plot(amp[,i,1],type="l",main=paste("Amplitude functions, a1 = ",a_vec[i],sep=""),
+   axes=F,xlab="Frequency",ylab="Amplitude",col=colo[1],ylim=c(ymin,ymax))
+   mtext("Lag=0", side = 3, line = -1,at=len/4,col=colo[1])
+   for (j in 2:(L/2+2))
+   {
+     lines(amp[,i,j],col=colo[j])
+     mtext(paste("Lag=",j-1,sep=""), side = 3, line = -j,at=len/4,col=colo[j])
+   }
+   axis(1,at=c(0,1:6*len/12+1),labels=c("0","pi/6","2pi/6","3pi/6",
+   "4pi/6","5pi/6","pi"))
+   axis(2)
+   box()
+   ymin<-min(shift[,i,],na.rm=T)
+   ymax<-max(shift[,i,],na.rm=T)
+   plot(shift[,i,1],type="l",main=paste("Time-Shifts, a1 = ",a_vec[i],sep=""),
+   axes=F,xlab="Frequency",ylab="Shift",col=colo[1],ylim=c(ymin,ymax))
+   mtext("Lag=0", side = 3, line = -1,at=len/4,col=colo[1])
+   for (j in 2:(L/2+2))
+   {
+     lines(shift[,i,j],col=colo[j])
+     mtext(paste("Lag=",j-1,sep=""), side = 3, line = -j,at=len/4,col=colo[j])
+   }
+   axis(1,at=c(0,1:6*len/12+1),labels=c("0","pi/6","2pi/6","3pi/6",
+   "4pi/6","5pi/6","pi"))
```

```
+   axis(2)
+   box()
+   ymin<-min(b[,i,],na.rm=T)
+   ymax<-max(b[,i,],na.rm=T)
+   plot(b[,i,1],col=colo[1],ylim=c(ymin,ymax),main=paste("Filter coefficients"),
+   ylab="Output",xlab="lag",axes=F,typ="l")
+   mtext("Lag=0", side = 3, line = -1,at=L/2,col=colo[1])
+   for (j in 2:(L/2+2))
+   {
+     lines(b[,i,j],col=colo[j],type="l")
+     mtext(paste("Lag=",j-1,sep=""), side = 3, line = -j,at=L/2,col=colo[j])
+   }
+   axis(1,at=1:L,labels=-1+1:L)
+   axis(2)
+   box()
+
+   ymin<-min(yhat_Lag[,i,],na.rm=T)
+   ymax<-max(yhat_Lag[,i,],na.rm=T)
+   ts.plot(yhat_Lag[,i,1],col=colo[1],ylim=c(ymin,ymax),
+   main=paste("Output series"),ylab="Output")
+   mtext("Lag=0", side = 3, line = -1,at=len/2,col=colo[1])
+   for (j in 2:(L/2+2))
+   {
+     lines(yhat_Lag[,i,j],col=colo[j])
+     mtext(paste("Lag=",j-1,sep=""), side = 3, line = -j,at=len/2,col=colo[j])
+   }
+
+ }
> invisible(dev.off())
```

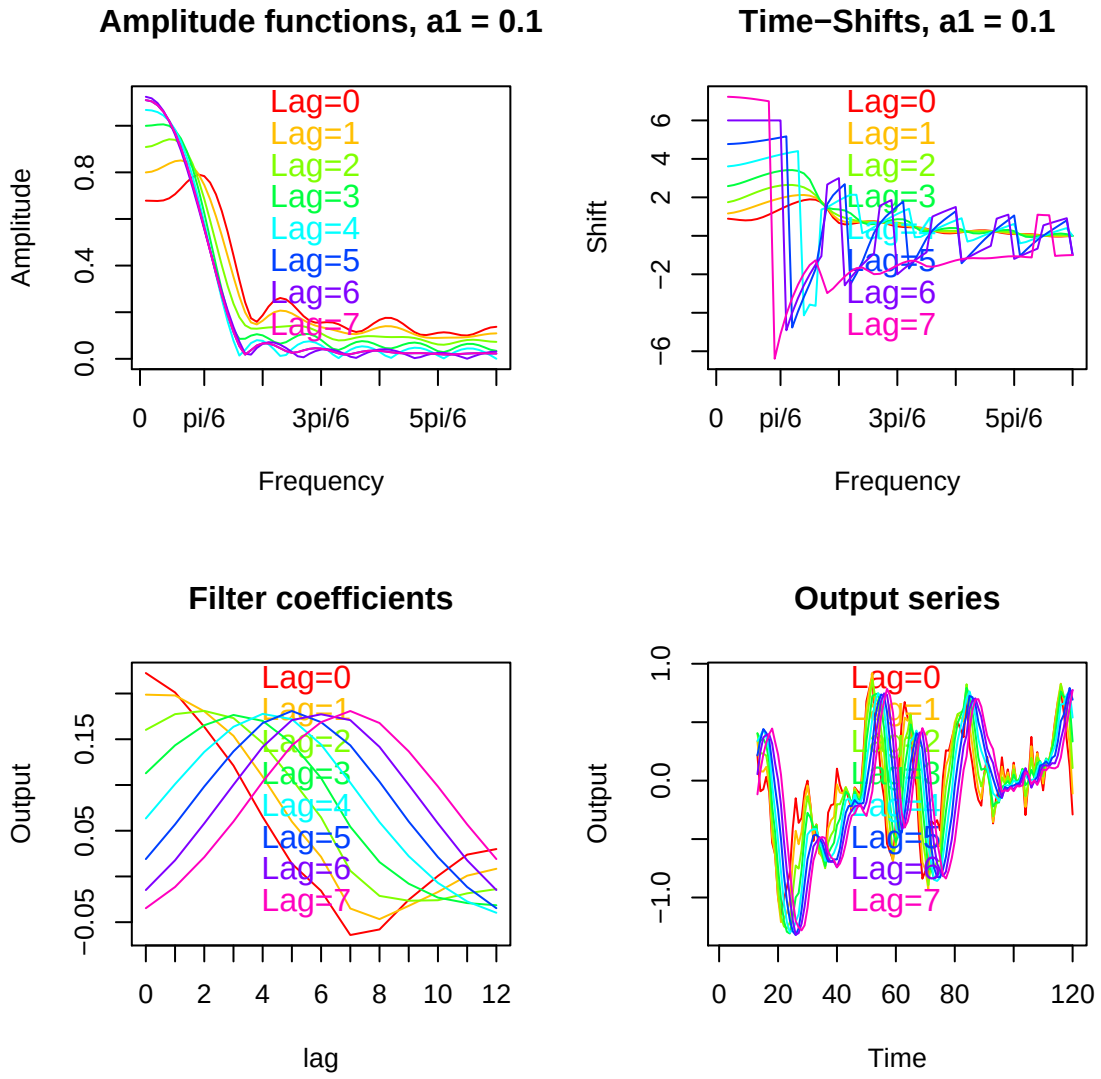


Figure 5.1: Amplitude (left) and time-shift (right) functions as a function of Lag (rainbow colors) for the second process (a1=0.1)

As expected, the time-shift (top-right) increases with increasing Lag (decreasing h). For $Lag = (L-1)/2 = 6$ the filter is symmetric (see bottom left graph) and therefore the corresponding time-shift (violet line) is constant[3]. In contrast to the symmetric target filter, which is centered about x_t, the time-shift of the symmetric $Lag = 6$-filter does not vanish because the filter is causal: its coefficients are centered about x_{t-6}. We can see that the output series (bottom-right panel) are shifted accordingly: larger shifts are associated to stronger noise rejection (amplitude closer to zero in the stop-band) and to smoother series.

[3]The shift is constant (flat line) in the passband. The variable shift in the stopband is an artifact of the Arg-function: the physical shift must be constant since the filter weights (violet line bottom left panel) are symmetric.

5.3.3 Filter Vintages

In the GDP table 5.1 historical releases are up-dated (revised) when new information becomes available. We could proceed analogously for the above filter-designs:

- Assume that in time point t a new observation x_t becomes available. Therefore we can compute a nowcast $\hat{y}_t^0$ of y_t based on the real-time filter $Lag = 0$.
- But we can also improve our previous estimate $\hat{y}_{t-1}^0$ of y_{t-1} if the new observation x_t is informative about y_{t-1}. For this purpose we can rely on the $Lag = 1$ filter and obtain a potentially better estimate $\hat{y}_{t-1}^{-1}$.
- We do similarly for $\hat{y}_{t-Lag}^{-Lag}$, $Lag > 1$: all historical estimates can be up-dated[4].

Assume now that we have a sample of length T and define a filter-vintage according to

$$\hat{y}_{T-t}^{-t}, t = 0, ..., T-1$$

In each time point $T-t$, the data point $\hat{y}_{T-t}^{-t}$ is the last observation of the output of the $Lag = t$-filter. The series $\hat{y}_{T-t}^{-t}, t = 0, ..., T-1$ is called a *filter-vintage*. Filter vintages can be arranged in a so-called *filter-vintage triangle*.

Exercises

1. Compute a filter-vintage triangle for each of the above AR(1)-processes. Specifically, use the filters for $Lag = 0, ..., 6$. Note that our maximal Lag-value is six: therefore, $\hat{y}_{T-t}^{-t}, t > 6$ is identified with $\hat{y}_{T-t}^{-6}$ i.e. the filter-vintage becomes

$$\hat{y}_{T-t}^{-\min(6,t)}, t = 0, ..., T-1$$

```
> vintage<-array(dim=c(len,3,len))
> # For each of the three AR(1)-processes We compute the vintage series
> for (i in 1:3)
+ {
+   for (j in L:len)#j<-L
+   {
+     vintage[(j-as.integer(L/2)):j,i,j]<-yhat_Lag[j,i,(as.integer(L/2)+1):1]
+     vintage[1:(j-as.integer(L/2)-1),i,j]<-
+     yhat_Lag[(as.integer(L/2)+1):(j-1),i,as.integer(L/2)+1]
+   }
+ }
> number_vint<-6
```

2. Compute the last 7 vintages (last 7 columns and rows) for the third process ($a_1 = -0.9$), see table 5.2.

[4]The new observation x_t is potentially informative because the target filter is bi-infinite.

```
> # We select the third DGP with a1=-0.9
> i<-3
> vintage_triangle<-vintage[,i,]
> dimnames(vintage_triangle)[[2]]<-paste("Publ. ",1:len,sep="")
> dimnames(vintage_triangle)[[1]]<-paste("Target ",1:len,sep="")
```

	Publ. 114	Publ. 115	Publ. 116	Publ. 117	Publ. 118	Publ. 119	Publ. 120
Target 114	-0.200	-0.185	-0.117	-0.158	-0.111	-0.118	-0.124
Target 115		-0.117	-0.042	-0.090	-0.026	-0.028	-0.046
Target 116			0.024	-0.028	0.050	0.058	0.028
Target 117				0.021	0.106	0.129	0.088
Target 118					0.137	0.172	0.123
Target 119						0.185	0.132
Target 120							0.116

Table 5.2: Last few vintages for the AR(1)-process with a1=-0.9: columns correspond to vintages and are indexed by corresponding publication dates; rows correspond to revisions of estimates for a fixed historical target date; diagonals correspond to releases: the lowest diagonal corresponds to the first release (real-time filter or nowcast).

The last column collects the last vintage $\hat{y}^{-t}_{120-t}$ for $t = 0, 1, ..., 6$: $\hat{y}^{0}_{120} = 0.116$ is the real-time estimate (first release) of the target y_{120} based on data $x_1, ..., x_{120}$; $y^{-1}_{119} = 0.132$ is the second release of the target y_{119} based on data $x_1, ..., x_{120}$ (the output of the $Lag = 1$ filter) and so on. In analogy to table 5.1, the column-date in table 5.2 refers to the *publication date* and the row-date refers to the *target time* i.e. the index t of y_t. The initial release in the first column (publication date 114) is -0.2; the second release of the same target value y_{114} in the second column is -0.185; the third release in the third column is -0.117, and so on. The differences between the various releases are due to filter-revisions: previous estimates of y_{114} are up-dated when new information becomes available. The initial release -0.2 is up-dated until it reaches its *final* value -0.124 in the last column[5].

Remarks

- Recall that the data x_t is error-free (fixed). Revisions are solely due to the deployment of the (optimal MSE) filter-sequence.
- The diagonals of the filter-vintage triangle are the releases: the initial release (first diagonal), the second release (second diagonal), and so on. A diagonal is generated by a fixed filter. Therefore diagonals are stationary time series (assuming stationarity of the data x_t).
- A filter-vintage (a column) is a non-stationary series: the DGP changes because different observations are generated by different filters (at least until the final values are obtained). As

[5]Recall that $\hat{y}^{-t}_{T-t} = \hat{y}^{-6}_{T-t}$ if $t > 6$ i.e. the estimate of y_{114} won't be revised anymore in later vintages ($T > 120$).

an example, the last observation of a vintage is generally 'noisier' than earlier observations (because it is based on a causal real-time filter $Lag = 0$).

- Fitting classic models to a vintage series would conflict with model assumptions because the series is non-stationary: the data-quality (noise/delay) worsens towards the sample end due to asymmetry of the underlying filters.

5.3.4 Tentacle Plot

All vintages can be collected and plotted in a single graph, called *tentacle plot*. The purpose of a tentacle plot is to reveal the quality of early releases of the filter-vintage by juxtaposing smoothed and real-time estimates.

1. Plot the vintages obtained for the three AR(1)-processes 7.22 in the previous section, see fig.5.2.

```
> colo<-rainbow(len)
> file = paste("z_vintages.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> par(mfrow=c(3,1))
> for (i in 1:3)
+ {
+   ymin<-min(vintage[,i,],na.rm=T)
+   ymax<-max(vintage[,i,],na.rm=T)
+   ts.plot(vintage[,i,L],col=colo[1],ylim=c(ymin,ymax),
+   main=paste("Tentacle plot: vintages and full revision sequence,
+   a1 = ",a_vec[i],sep=""),ylab="Vintages")
+   for (j in (L+1):len)
+   {
+     lines(vintage[,i,j],col=colo[j])
+   }
+   lines(vintage[,i,len],col="red",lwd=2)
+ }
> invisible(dev.off())
```

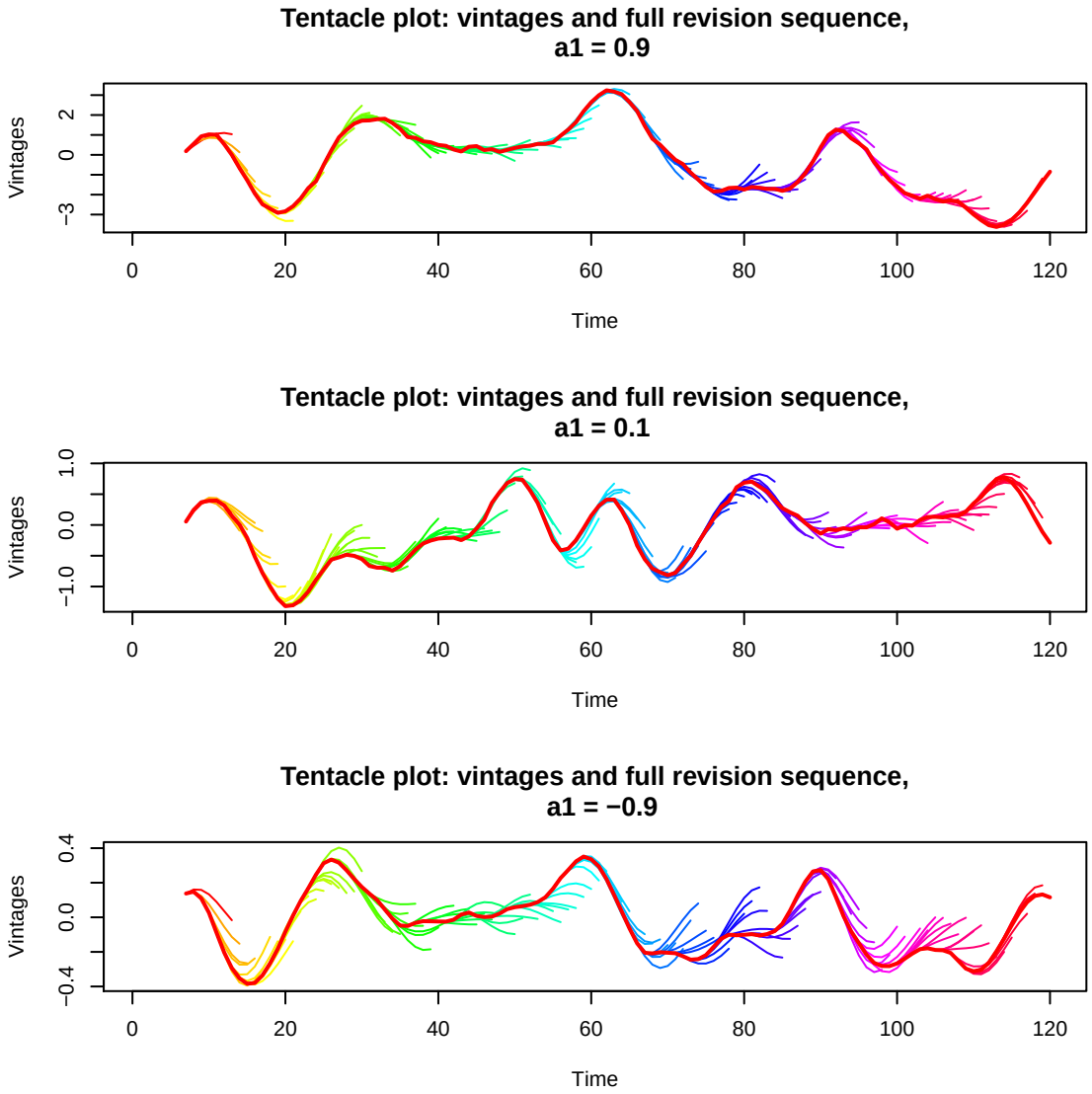


Figure 5.2: Tentacle plot: full historical revision sequence for a1=0.9 (top), a1=0.1 (middle) and a1=-0.9 (bottom). Final release is emphasized in bold red

The rate of convergence of early releases to their final values and the dynamic pattern of the revisions, in particular in the vicinity of turning points, can be summarized and analyzed conveniently by this graphical tool which provides a backtest of the otherwise unobserved real-time performances of the filter-*sequence*. The plot summarizes intrinsic properties of the filter-*sequence* which are not addressed by previous diagnostic tools (MSEs, amplitude and time-shift functions). A direct comparison of the three tentacle plots confirms, once again, that the signal-extraction task seems less demanding when the data is positively autocorrelated (top graph).

2. we focus attention on the second process ($a_1 = 0.1$) and emphasize final and initial releases, see fig.5.3.

```
> file = paste("z_vintages_2.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> par(mfrow=c(2,1))
> i<-2
> ymin<-min(vintage[,i,],na.rm=T)
> ymax<-max(vintage[,i,],na.rm=T)
> ts.plot(vintage[,i,L],col=colo[1],ylim=c(ymin,ymax),
+ main="Vintages: full revision sequence and final release (black)",ylab="Vintages")
> for (j in (L+1):len)
+ {
+   lines(vintage[,i,j],col=colo[j])
+ }
> lines(vintage[,i,len],col="black",lwd=2)
> i<-2
> ymin<-min(vintage[,i,],na.rm=T)
> ymax<-max(vintage[,i,],na.rm=T)
> ts.plot(vintage[,i,L],col=colo[1],ylim=c(ymin,ymax),
+ main="Vintages: full revision sequence and real-time initial release (black)",
+ ylab="Vintages")
> for (j in (L+1):len)
+ {
+   lines(vintage[,i,j],col=colo[j])
+ }
> lines(yhat_Lag[,i,1],col="black",lty=1)
> invisible(dev.off())
```

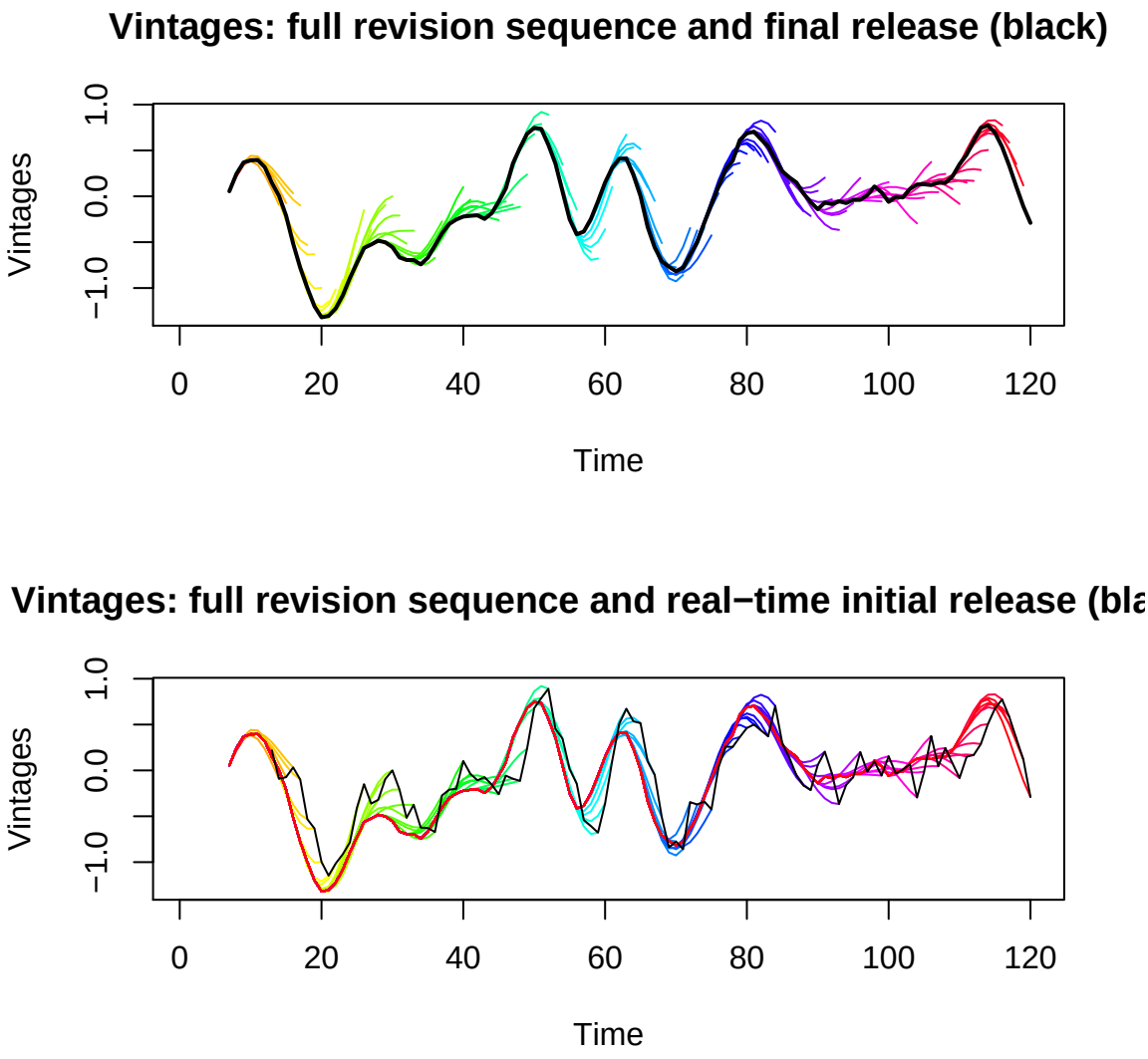


Figure 5.3: Vintages: full historical revision sequence in the case of the second process (a1=0.1)

Convergence to the final values is achieved after $(L-1)/2 = 6$ time steps in this particular example. Top and bottom graphs stress final (symmetric filter) and first (concurrent filter) releases, respectively. Linking the end-points of the tentacles in the bottom graph highlights the noisy real-time dynamics as well as the inherent delay towards the sample-end of the filter-vintage: these effects were illustrated in fig.5.1, in terms of amplitude and time-shift functions of the one-sided filter. The spreading tentacles at the turning points suggest that univariate MSE-designs are not ideally suited for inferring growth-breaks in *real time*. A better bivariate filter is proposed in the following section and chapters 7 and 8 propose a generic criterion for tackling 'timeliness' and 'smoothness' of real-time designs (ATS-trilemma).

5.4 Tentacle Plot of the Bivariate Leading Indicator Design

We here apply the leading indicator design introduced in section 4.7. For simplicity we restrict the analysis to the second process ($a_1 = 0.1$).

1. Generate a leading indicator and define the corresponding (3-dim) data matrix.

```
> set.seed(12)
> # Select the AR(1)-process with coefficient 0.1
> i_process<-2
> # Scaling of the idiosyncratic noise
> scale_idiosyncratic<-0.1
> eps<-rnorm(nrow(x))
> indicator<-x[,i_process]+scale_idiosyncratic*eps
> # Data: first column=target, second column=x,
> #   third column=shifted (leading) indicator
> data_matrix_120<-cbind(x[,i_process],x[,i_process],
+                        c(indicator[2:nrow(x)],indicator[nrow(x)]))
> dimnames(data_matrix_120)[[2]]<-c("target","x","leading indicator")
> head(data_matrix_120)

         target          x leading indicator
[1,]  0.2207327  0.2207327         0.5695845
[2,]  0.4118676  0.4118676        -1.2625639
[3,] -1.1668894 -1.1668894        -0.5723655
[4,] -0.4803650 -0.4803650        -1.8744734
[5,] -1.6747092 -1.6747092        -0.4511789
[6,] -0.4239493 -0.4239493         1.0278497
```

2. Compute the DFTs.

```
> # Fully in sample
> insample<-nrow(data_matrix_120)
> # d=0 for stationary series: see default settings
> weight_func<-spec_comp(insample, data_matrix_120, d)$weight_func
```

3. Estimate optimal (MSE-) filter coefficients as a function of Lag=0,...,6

```
> yhat_Lag_mdfa<-matrix(nrow=len,ncol=L/2+2)
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Estimate filter coefficients
> for (Lag in 0:((L/2)))
+ {
+   mdfa_obj<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
```

```
+
+   print(paste("Lag=",Lag," Criterion=",round(mdfa_obj$MS_error,4),sep=""))
+ # Filter coefficients
+   b_mat<-mdfa_obj$b
+ # Compute outputs
+   for (j in L:len)
+     yhat_Lag_mdfa[j,Lag+1]<-sum(apply(b_mat*data_matrix_120[j:(j-L+1),2:3],1,sum))
+ }
```

```
[1] "Lag=0 Criterion=0.0507"
[1] "Lag=1 Criterion=0.0294"
[1] "Lag=2 Criterion=0.0182"
[1] "Lag=3 Criterion=0.0148"
[1] "Lag=4 Criterion=0.0154"
[1] "Lag=5 Criterion=0.0166"
[1] "Lag=6 Criterion=0.0166"
```

The criterion is minimal at Lag=3 which suggests a rapid convergence of early releases (short tentacles).

4. Define the vintage triangle.

```
> vintage_mdfa<-matrix(nrow=len,ncol=len)
> # For each of the three AR(1)-processes We compute the vintage series
> for (j in L:len)#j<-len
+ {
+   vintage_mdfa[(j-as.integer(L/2)):j,j]<-yhat_Lag_mdfa[j,(as.integer(L/2)+1):1]
+   vintage_mdfa[1:(j-as.integer(L/2)-1),j]<-
+   yhat_Lag_mdfa[(as.integer(L/2)+1):(j-1),as.integer(L/2)+1]
+ }
```

5. Generate a tentacle plot, see fig5.4.

```
> file = paste("z_vintages_mdfa.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> par(mfrow=c(2,1))
> ymin<-min(vintage_mdfa,na.rm=T)
> ymax<-max(vintage_mdfa,na.rm=T)
> ts.plot(vintage_mdfa[,L],col=colo[1],ylim=c(ymin,ymax),
+ main="Vintages: full revision sequence and final release (black)",ylab="Vintages")
> for (j in (L+1):len)
+ {
+   lines(vintage_mdfa[,j],col=colo[j])
+ }
```

```
> lines(vintage_mdfa[,len],col="black",lwd=2)
> ymin<-min(vintage_mdfa,na.rm=T)
> ymax<-max(vintage_mdfa,na.rm=T)
> ts.plot(vintage_mdfa[,L],col=colo[1],ylim=c(ymin,ymax),
+ main="Vintages: full revision sequence and final release (black)",ylab="Vintages")
> for (j in (L+1):len)
+ {
+   lines(vintage_mdfa[,j],col=colo[j])
+ }
> lines(yhat_Lag_mdfa[,1],col="black",lwd=1)
> invisible(dev.off())
```

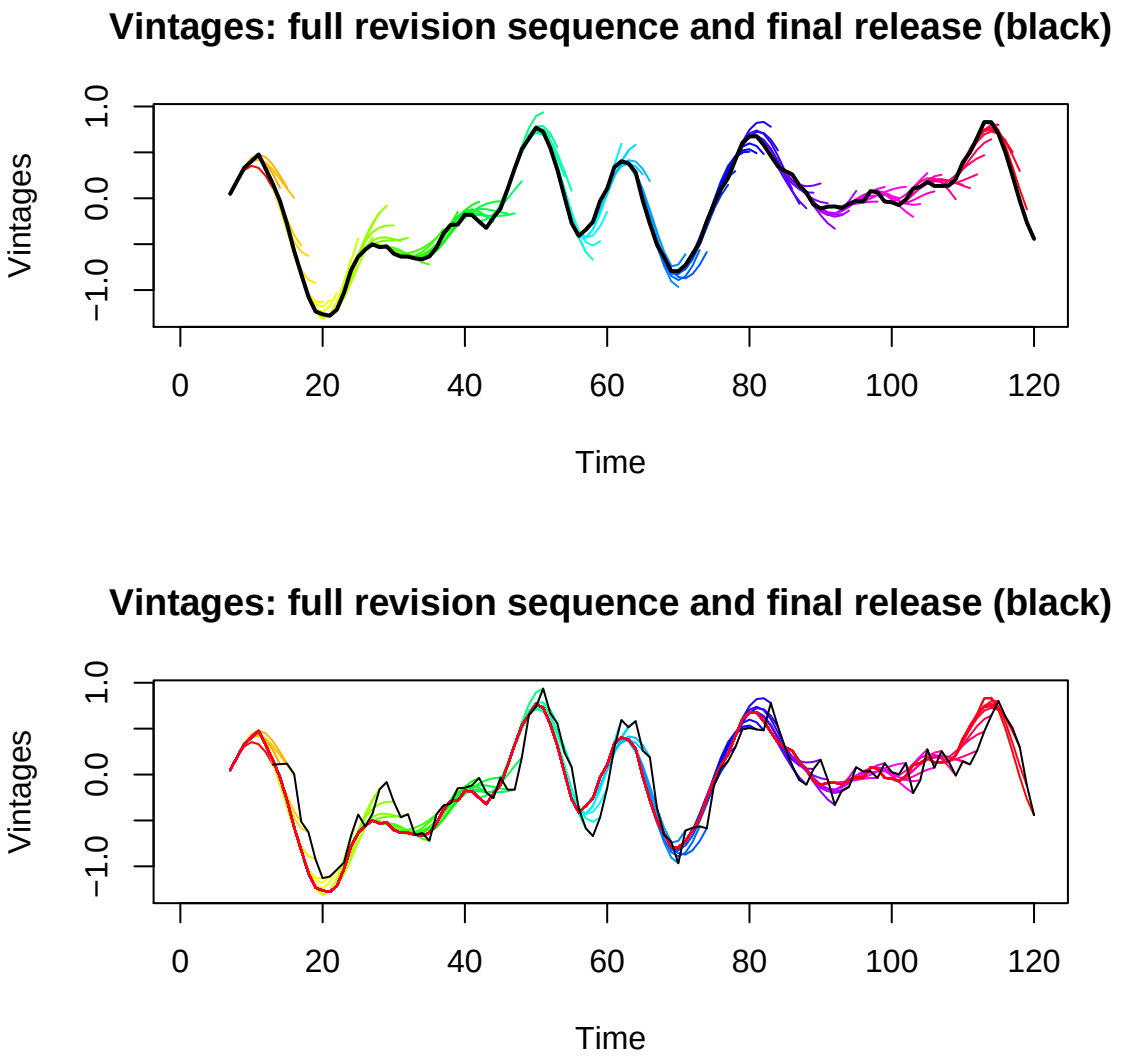


Figure 5.4: Vintages: full historical revision sequence in the case of the second process (a1=0.1)

Visual inspections and comparisons of figs.5.4 and 5.3 (univariate DFA) are more eloquent than a convoluted comparative analysis.

Final Remark

- When applying a real-time filter ($Lag = 0$) of length L to the data, the output $\hat{y}_t^0, t = 1, ..., L-1$ corresponding to the first $(L-1)$-observations is missing (because the sample starts in $t = 1$ i.e. $x_0, x_{-1}, x_{-2}, ...$ are not observed). These missing 'initial' values could be obtained in terms of backcasts $\hat{y}_t^{-L+t}, t = 1, ..., L-1$ (verification of this claim is left as an exercise to the reader).

5.5 Summary

- We analyzed optimal (time-dependent) filter sequences for tracking a generic target signal at each time point $t = 1, ..., T$ of a finite sample of length T.
- We distinguished forecast ($h > 0$), nowcast ($h = 0$) and backcast ($h < 0$) of a generic target: the corresponding estimation problems can be handled by the parameter $Lag = -h$ in the relevant function-calls of DFA and MDFA.
- All results were based on the assumption of error-free data. Optimal filter design in the presence of data revisions is addressed in chapter 14.
- We proposed a useful diagnostic tool, the so-called tentacle plot, and benchmarked the bivariate leading-indicator design against the univariate DFA.
- Pertinence and practical relevance of the proposed filter-sequence is closely linked to the MSE-perspective because the optimization criterion adresses the (mean-square) filter error i.e. the revision error is minimized explicitly. More sophisticated criteria proposed in chapters 7 and 8 will emphasize more specifically *nowcast* and *forecast* applications, instead.

Chapter 6

Filter Constraints

We propose and analyze a set of filter constraints which are deemed to be relevant in applications. Formally, the constraints ensure finiteness of the mean-square filter error in the case of integrated processes[1]. Section 6.1 sets-up the context; a link to integrated processes is proposed in section 6.2; a general matrix notation is proposed in section 6.3; section 6.4 extends the former (unconstrained) MDFA-criterion to the constrained case; finally, section 6.5 illustrates effects (of the constraints) on characteristics of real-time filters.

The constraints proposed in this chapter do not address cross-sectional links, such as required in the presence of cointegration, for example. The relevant multivariate aspects and extensions are proposed in chapter 17.

6.1 Level and Time-Shift Constraints

6.1.1 Level: i1==T

A first-order (level-) restriction of the coefficients $b_0, ..., b_{L-1}$ of a filter with transfer function $\hat{\Gamma}(\cdot)$ is specified as

$$\hat{\Gamma}(0) = w$$

or, equivalently

$$b_0 + b_1 + ... + b_{L-1} = w \tag{6.1}$$

where w is a constant. For $w := \Gamma(0)$ the constraint implies that $\hat{\Gamma}(\cdot)$ fits the target $\Gamma(\cdot)$ exactly in frequency zero: typically, then $w = 0$ (bandpass or highpass) or $w = 1$ (lowpass). If x_t is integrated of order one with a single unit-root in frequency zero, then the proposed restriction ensures a finite mean-square filter error (assuming some mild regularity conditions, see McElroy and Wildi (2014) DFA and Wildi (2005)). Figuratively, $\hat{y}_t$ tracks the non-stationary level of the

[1]The constraints address filter characteristics in *frequency zero*. Arbitrary frequencies could be tackled but we did not find any relevant practical application so far.

target y_t. Formally, $\hat{y}_t$ and y_t are cointegrated, with cointegration vector $(1, -1)$.

R Code

Consider the head of the main MDFA estimation function

```
> head(mdfa_analytic)
```

```
1 function (L, lambda, weight_func, Lag, Gamma, eta, cutoff, i1,
2     i2, weight_constraint, lambda_cross, lambda_decay, lambda_smooth,
3     lin_eta, shift_constraint, grand_mean, b0_H0, c_eta, weight_structure,
4     white_noise, synchronicity, lag_mat, troikaner)
5 {
6     lambda <- abs(lambda)
```

The Boolean $i1$ in the function call of MDFA determines imposition (or not) of the level constraint. The constant w can be set in a vector called *weight_constraint*. In a multivariate design one must specify multiple constraints $w^u, u = 1, ..., m + 1$. The restrictions are independent: each time series receives its own weight (cross-sectional dependencies among the constraints are proposed in chapter 17).

6.1.2 Time-shift: i2==T

The time-shift $\hat{\phi}(\omega) = \hat{\Phi}(\omega)/\omega$ of a filter is subject to a singularity in frequency zero. However, the limiting value, as $\omega \to 0$, could be obtained, see section 3.2.3 in DFA:

$$\begin{aligned}
\hat{\phi}(0) &= \lim_{\omega \to 0} \frac{\hat{\Phi}(\omega)}{\omega} \\
&= \frac{\frac{d}{d\omega}\hat{\Phi}(\omega)\Big|_{\omega=0}}{1} \\
&= \frac{\frac{d}{d\omega}\hat{\Gamma}(\omega)\Big|_{\omega=0}}{-i\hat{A}(0)} \\
&= \frac{\sum_{j=0}^{L-1} j b_j}{\sum_{j=0}^{L-1} b_j}
\end{aligned} \tag{6.2}$$

Second and third equalities are obtained from

$$\begin{aligned}
-i\sum_{j=0}^{L-1} j b_j &= \frac{d}{d\omega}\hat{\Gamma}(\omega)\Big|_{\omega=0} \\
&= \frac{d}{d\omega}\hat{A}(\omega)\Big|_{\omega=0} \exp(-i\hat{\Phi}(0)) - i\hat{A}(0)\exp(-i\hat{\Phi}(0))\ \frac{d}{d\omega}\hat{\Phi}(\omega)\Big|_{\omega=0} \\
&= -i\hat{A}(0)\ \frac{d}{d\omega}\hat{\Phi}(\omega)\Big|_{\omega=0}
\end{aligned}$$

The derivative of the amplitude vanishes in zero because the amplitude is a continuous even function i.e. $\hat{A}(-\omega) = \hat{A}(\omega)$. Note that we implicitly assumed $\hat{\Gamma}(0) > 0$ such that $\hat{A}(0) = \hat{\Gamma}(0) =$

$\sum_{j=0}^{L-1} b_j > 0$ and thus 6.2 applies and is well defined[2].

We are now in a position to formulate a time-shift restriction in frequency zero:

$$\frac{\sum_{j=0}^{L-1} jb_j}{\sum_{j=0}^{L-1} b_j} = s$$

This expression can be rewritten as

$$\sum_{j=0}^{L-1}(j-s)b_j = 0 \tag{6.3}$$

In practice, a vanishing time-shift $s = 0$ in frequency zero, i.e.

$$\sum_{j=1}^{L-1} jb_j = 0$$

is often a desirable 'feature' because turning-points (of the trend) should not be delayed too much. Also, a vanishing time-shift is necessary in order to ensure finiteness of the mean-square filter error if the data is integrated of order two, I(2).

R Code

The Boolean $i2$ in the function call of MDFA determines imposition (or not) of the time-shift constraint. The constant s can be set in a vector called $shift\_constraint$. In a multivariate design one must specify multiple constraints $s^u, u = 1, ..., m+1$. The restrictions are independent: each time series receives its own weight (cross-sectional dependencies among the constraints are proposed in chapter 17).

Since this chapter emphasizes constrained MSE-designs, without customization or regularization, the simpler context-specific $MDFA\_mse\_constraint$ function

```
> head(MDFA_mse_constraint)

1 function (L, weight_func, Lag, Gamma, i1, i2, weight_constraint,
2     shift_constraint)
3 {
4     cutoff <- pi
5     lin_eta <- F
6     lambda <- 0
```

can be used instead of the generic (lengthy) $mdfa\_analytic$-call.

[2] As we shall see, the case $\hat{\Gamma}(0) = 0$ (highpass or bandpass) does not involve the time-shift in frequency zero. Moreover, the case $\hat{\Gamma}(0) < 0$ is generally not practically relevant.

6.1.3 Level and Time-Shift: i1==T,i2==T

Both constraints can be imposed simultaneously by solving the above expressions for b_{L-1} and b_{L-2}:

$$b_{L-2} = (L-1-s)w-(L-1)b_0-(L-2)b_1-...-2b_{L-3} \quad (6.4)$$

$$b_{L-1} = (2+s-L)w+(L-2)b_0+(L-3)b_1+(L-4)b_2+...+b_{L-3} \quad (6.5)$$

6.1.4 Exercises: Checking the Formulas

1. We verify the filter constraints by a simple number experiment.

   - Define arbitrary filter coefficients $b_0, ..., b_3$ and constants w, s and derive b_4 and b_5 according to 6.4 and 6.5

     ```
     > # Filter length
     > L<-5
     > set.seed(1)
     > # The first three coefficients are random numbers
     > b<-rnorm(1:(L-2))
     > # Define the constants: the following are classical
     > #   restrictions (amplitude is 1 and time shift is zero)
     > w<-1
     > s<-0
     > b<-c(b,(L-1-s)*w-(L-1:(L-2))%*%b,(2+s-L)*w+(L-2:(L-1))%*%b)
     ```

   - Verify pertinence of the filter constraints.

     ```
     > # Level constraint
     > sum(b)-w
     [1] 8.881784e-16
     > # time-shift constraint
     > sum(b*0:(L-1))/sum(b)-s
     [1] 5.051515e-15
     ```

     Both expressions are (virtually) zero, as desired.

2. To be filled: we verify the effects of the filter constraints on amplitude and time-shift functions

   - univariate: (problem: DFA is cheating (explain why) i.e. one must use MDFA univariate)
   - recompute exercise previous section with i1 and/or i2 imposed with various constraints

6.2 Filter Constraints, Pseudo-DFT and Pseudo-Periodogram

We briefly link the aforementioned filter restrictions to unit roots of the DGP and we illustrate tracking of the target signals of non-stationary integrated processes by the real time designs (finite

mean-square filter error). The treatment is informal, see chapter 16 and Wildi (2005), McElroy and Wildi (2015: DFA paper) for details.

6.2.1 Pseudo-DFT and Pseudo-Periodogram

Assume that x_t is a realization of an I(1)-process with a single unit-root in frequency zero (typical spectral shape of many economic time series). Then

$$\tilde{x}_t \quad = \quad x_t - x_{t-1} = (1-B)x_t$$

is a stationary process. According to 4.20 (finite sample convolution) we expect that the approximation

$$\Xi_{\tilde{X}T}(\omega) \quad \simeq \quad (1-\exp(-i\omega))\Xi_{TX}(\omega)$$

should be 'tight'. Unfortunately, one can show that the gap between left and right-hand sides of the above approximation does not close asymptotically[3] and therefore the periodogram of x_t is a biased spectral estimate, see Wildi (2005). For an I(1)-process we consider the so-called pseudo-DFT

$$\Xi_{TX}^{pseudo}(\omega) := \frac{\Xi_{\tilde{X}T}(\omega)}{1-\exp(-i\omega)},\ \omega \neq 0$$

with associated pseudo-periodogram

$$I_{TX}^{pseudo}(\omega) = \left|\Xi_{TX}^{pseudo}(\omega)\right|^2,\ \omega \neq 0$$

where $\Xi_{\tilde{X}T}(\omega)$ relies on first differences $\tilde{x}_t$ of the data. The pseudo-periodogram is an unbiased estimate of the pseudo-spectral density. The pseudo-DFT (pseudo-periodogram) is defined in all frequencies except $\omega = 0$.

6.2.2 First Order Constraint

Consider an extension of the stationary (univariate) DFA-criterion 4.27

$$\begin{aligned} & \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} |\Delta\Gamma(\omega_k)|^2\, I_{TX}^{pseudo}(\omega_k)\Big|_{\Gamma(0)=\hat{\Gamma}(0)} \\ = \quad & \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} \frac{|\Delta\Gamma(\omega_k)|^2}{|1-\exp(-i\omega_k)|^2} I_{T\tilde{X}}(\omega_k)\Big|_{\Gamma(0)=\hat{\Gamma}(0)} \rightarrow \min_{\mathbf{b}} \end{aligned} \tag{6.6}$$

where the original periodogram has been replaced by the unbiased pseudo-periodogram. Interestingly, the singularity in frequency zero can be adressed by imposing the level constraint

$$\Delta\Gamma(0) = \Gamma(0) - \hat{\Gamma}(0) = 0$$

[3]The realization of an I(1)-process is not mean-reverting (it is 'far from periodic') which conflicts with the intrinsic assumption of the DFT. Fortunately, a very simple linear adjustment of the data, which does not affect its information content, re-establishes perfect equality between both terms in all frequencies (except zero), see Wildi (2005).

i.e. the limit

$$\lim_{\omega\to 0}\frac{|\Delta\Gamma(\omega)|^2}{|1-\exp(-i\omega)|^2}I_{T\tilde{X}}(\omega)\Big|_{\Gamma(0)=\hat{\Gamma}(0)}$$

exists. More precisely, assuming a mild set of regularity assumptions, one can show that the coefficients of the filter

$$\tilde{\Gamma}(\omega):=\frac{\Delta\Gamma(\omega)}{1-\exp(-i\omega)}\Big|_{\Gamma(0)=\hat{\Gamma}(0)} \tag{6.7}$$

are well-defined and converge at a suitable rate towards zero with increasing/decreasing lags, see Wildi (2005). Therefore, consider

$$\Gamma(\omega)-\hat{\Gamma}(\omega)=\tilde{\Gamma}(\omega)(1-\exp(-i\omega))$$

We infer that the difference of target and real-time filters, on the left side, can be factored into a well-defined MA-filter $\tilde{\Gamma}(\omega)$ and a first difference operator. Our estimation problem can then be stated in (at least) two equivalent ways:

- Apply the filter with transfer function $\Gamma(\omega)-\hat{\Gamma}(\omega)$ to the non-stationary data x_t and determine 'optimal' filter coefficients of $\hat{\Gamma}(\omega)$.
- Apply the filter with transfer function $\tilde{\Gamma}(\omega)$ to the stationary data $\tilde{x}_t$ and determine 'optimal' filter coefficients of $\hat{\Gamma}(\omega)$ (these coefficients enter into $\tilde{\Gamma}(\omega)$ via 6.7).

In the latter case, we are back to the stationary case discussed in chapter 4 (except that we have to impose a filter constraint). In particular

$$\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\left|\tilde{\Gamma}(\omega_k)\right|^2 I_{T\tilde{X}}(\omega_k)\Big|_{\Gamma(0)=\hat{\Gamma}(0)}$$

is a superconsistent estimate of the (sample) mean-square filter error and therefore criterion 6.6 is pertinent.

We conclude that the proposed level-constraint $\sum_{k=0}^{L-1}b_k=\Gamma(0)$ ensures existence and pertinence of the optimization criterion 6.6. Moreover, the filter error $y_t-\hat{y}_t$ is stationary i.e. the mean-square filter error is finite[4]. If $\Gamma(0)>0$ and if $\Gamma(\omega)$ is sufficiently regular in a vicinity of $\omega=0$, then the non-stationary target y_t and the non-stationary finite sample estimate $\hat{y}_t$ are cointegrated with cointegration vector $(1,-1)$. If $\Gamma(0)=0$ and if $\Gamma(\omega)$ is sufficiently regular in a vicinity of $\omega=0$, then both the target y_t as well as the finite sample estimate $\hat{y}_t$ are stationary.

6.2.3 Second Order Constraint

We here assume that x_t is a realization of an I(2)-process. Then the pseudo-DFT (-periodogram) becomes

$$\begin{aligned}\Xi_{TX}^{pseudo}(\omega) &:= \frac{\Xi_{\tilde{X}T}(\omega)}{(1-\exp(-i\omega))^2},\ \omega\neq 0\\ I_{TX}^{pseudo}(\omega) &= \left|\Xi_{TX}^{pseudo}(\omega)\right|^2,\ \omega\neq 0\end{aligned} \tag{6.8}$$

[4] An extension to for/backcasting is straightforward by imposing $\sum_{k=h}^{L-1+h}b_{kh}=\Gamma(0)$.

where

$$\tilde{x}_t \quad = \quad (1-B)^2 x_t$$

are the stationary second-order differences of the data. Consider the following optimization criterion

$$\begin{aligned} & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} |\Delta\Gamma(\omega_k)|^2 \, I_{TX}^{pseudo}(\omega_k) \Big|_{\Delta\Gamma(0)=\Delta\Gamma^{(1)}(0)=0} \\ = & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \frac{|\Delta\Gamma(\omega_k)|^2}{|1-\exp(-i\omega_k)|^4} I_{T\tilde{X}}(\omega_k) \Big|_{\Delta\Gamma(0)=\Delta\Gamma^{(1)}(0)=0} \to \min_{\mathbf{b}} \end{aligned} \tag{6.9}$$

where we impose first- and second-order constraints

$$\begin{aligned} \Delta\Gamma(0) &= 0 \\ \Delta\Gamma^{(1)}(0) &= \frac{d(\Gamma(\omega) - \hat{\Gamma}(\omega))}{d\omega}\Big|_{\omega=0} = 0 \end{aligned}$$

Note that $d\Gamma(\omega)/d\omega|_{\omega=0} = 0$ in the second order-constraint, by symmetry of the target filter, (even function) so that we must impose a vanishing derivative of $\hat{\Gamma}(\cdot)$ in $\omega = 0$ for the constraint to hold. We first consider the case $|\Gamma(0)| > 0$ (lowpass target). Let

$$\hat{\Gamma}(\omega) = \hat{A}(\omega) \exp(-i\hat{\Phi}(\omega))$$

so that

$$\frac{d\hat{\Gamma}(\omega)}{d\omega}\Big|_{\omega=0} = \frac{d\hat{A}(\omega)}{d\omega}\Big|_{\omega=0} \exp(-i\hat{\Phi}(0)) - i\hat{A}(0)\exp(-i\hat{\Phi}(0)) \, \frac{d\hat{\Phi}(\omega)}{d\omega}\Big|_{\omega=0} \tag{6.10}$$

$$= -i\hat{A}(0) \, \frac{d\hat{\Phi}(\omega)}{d\omega}\Big|_{\omega=0} \tag{6.11}$$

where $d\hat{A}(\omega)/d\omega\Big|_{\omega=0} = 0$ by symmetry of the amplitude function[5] (even function). Since $\hat{A}(0) = |\Gamma(0)| > 0$ (first order constraint), the derivative in (6.11) vanishes in $\omega = 0$ if and only if

$$\frac{d\hat{\Phi}(\omega)}{d\omega}\Big|_{\omega=0} = \lim_{h\to 0} \frac{\hat{\Phi}(h) - \hat{\Phi}(0)}{h} = \hat{\phi}(0) = 0 \tag{6.12}$$

where we used the fact that the phase function vanishes in $\omega = 0$. We infer that both the first order level as well as the second-order time-shift constraints are necessary in order to cancel the second-order singularity in 6.9 if $|\Gamma(0)| > 0$[6]. On the other hand, if $\Gamma(0) = 0$ (bandpass, highpass) then $\hat{A}(0) = 0$ (first order level constraint). Therefore the right-hand side of 6.10 vanishes iff $d\hat{A}(\omega)/d\omega\Big|_{\omega=0} = 0$. However, the last equality together with $\hat{A}(0) = 0$ implies that the zero of the amplitude function must be of order two in frequency zero, because $\hat{A}(\cdot)$ is an even function[7].

[5] Note that $\hat{A}(0) > 0$ (level constraint) and therefore the derivative exists.

[6] Intuitively the vanishing time-shift is necessary because the slope (first difference) of the data diverges asymptotically.

[7] A graphical illustration of the double-zero in frequency zero is obtained in fig.9.21, chapter 9: the bottom-right panel (double-zero in frequency-zero) is to be contrasted with the bottom-left panel (single zero in frequency zero).

The same argument applies to $\Gamma(\cdot)$ which is an even function, too. Therefore, both filter outputs y_t and $\hat{y}_t$ are stationary and therefore the filter-error is stationary too i.e. the mean-square filter error is finite, irrespective of the time-shift of the real-time filter.

In analogy to the previous section, one can show that the coefficients of the filter

$$\tilde{\Gamma}(\omega) := \left.\frac{\Delta\Gamma(\omega)}{(1-\exp(-i\omega))^2}\right|_{\Delta\Gamma(0)=\Delta\Gamma^{(1)}(0)=0} \tag{6.13}$$

are well defined and converge at a suitable rate towards zero with increasing lag, see Wildi (2005). As a consequence, our estimation problem can be stated in (at least) two equivalent ways:

- Apply the filter with transfer function $\Gamma(\omega)-\hat{\Gamma}(\omega)$ to the non-stationary data x_t and determine 'optimal' filter coefficients of $\hat{\Gamma}(\omega)$.
- Apply the filter with transfer function $\tilde{\Gamma}(\omega)$ to the stationary data $\tilde{x}_t$ and determine 'optimal' filter coefficients of $\hat{\Gamma}(\omega)$ (these coefficients enter into $\tilde{\Gamma}(\omega)$ via 6.13).

In the latter case, we are back to the stationary case discussed in chapter 4 (except that we have to impose a double filter constraint). In particular

$$\left.\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\left|\tilde{\Gamma}(\omega_k)\right|^2 I_{T\tilde{X}}(\omega_k)\right|_{\Delta\Gamma(0)=\Delta\Gamma^{(1)}(0)=0}$$

is a superconsistent estimate of the (sample) mean-square filter error and therefore criterion 6.9 is pertinent. In particular, the filter error $y_t-\hat{y}_t$ is stationary. If $\Gamma(0)>0$ and if $\Gamma(\omega)$ is sufficiently regular in $\omega=0$ then the non-stationary target y_t and the non-stationary estimate $\hat{y}_t$ are cointegrated with cointegration vector $(1,-1)$. If $\Gamma(0)=0$ and if $\Gamma(\omega)$ is sufficiently regular in $\omega=0$ then both the target y_t and the estimate $\hat{y}_t$ are stationary[8].

6.3 General Parametrization

6.3.1 Caveats

The particular parametrization of the filter constraints in terms of b_{L-1} and b_{L-2} in 6.4 and 6.5 is to some extent arbitrary. Indeed, we could have selected b_0 and b_1, instead. Of course, our particular choice does not affect the estimation result, at least as long as the coefficients are otherwise freely determined. Unfortunately, the regularization features to be introduced in chapter 13 conflict with this assumption. Therefore we here propose a more general approach.

6.3.2 Nowcasting, Forecasting and Smoothing

We want to estimate y_{T+h} for $h>0$ (forecasting), $h=0$ (nowcasting) or $h<0$ (backcasting), given data $x_T, x_{T-1}, ..., x_{T-(L-1)}$, see section 5.3.1. If $h=0$ (nowcasting) then imposing a vanishing

[8] Assuming mild regularity restrictions, the zero of $\Gamma(\omega)$ must be of second order because the function is even (symmetric around zero).

time shift ($s = 0$) means that $\hat{y}_t$ and y_t are 'synchronized' (at least in frequency zero). If $h \neq 0$, then the synchronization should apply between $\hat{y}^h_{t-h}$ and y_t i.e. the vanishing time-shift should apply to the *shifted* output signal. Relying on 6.2 we obtain:

$$\hat{\phi}(0) \quad = \quad \frac{\sum_{j=h}^{L-1+h} j b_{jh}}{\sum_{j=h}^{L-1+h} b_{jh}}$$

and the time-shift constraint 6.3 becomes

$$\sum_{j=h}^{L-1+h} (j-s) b_{jh} = 0 \tag{6.14}$$

The level constraint is 'as usual'

$$b_{hh} + b_{h+1,h} + ... + b_{h+(L-1),h} = w \tag{6.15}$$

6.3.3 Implementing the Filter Constraints

As discussed, any single coefficient b_{jh} (single constraint) or any pair of coefficients (both constraints imposed) could be isolated out of the above equations for implementing the constraint(s). In order to avoid conflicts with later regularization features, see chapter 13, we here select $b_{max(0,h),h}$ (single constraint) or $b_{max(0,h),h}, b_{max(0,h)+1,h}$ (both constraints imposed) as *natural candidate(s)*. For $h \geq 0$ (nowcast/forecast) these are $b_{hh}, b_{h+1,h}$ i.e. the coefficients assigned to the last data points x_T, x_{T-1}, which are likely to be most informative for y_{T+h}. For $h < 0$ (backcast) these are b_{0h}, b_{1h} i.e the coefficient assigned to $x_{T-|h|}$ and $x_{T-|h|-1}$ which are likely to be most informative for $y_{T-|h|}$ (recall that the backcast filter is 'centered' about $x_{T-|h|}$).

6.3.4 Matrix Notation

For notational ease we now drop the index h from the subscripts i.e. we write b_j instead of b_{jh}. The proposed filter constraints can then be re-written in the generic form

$$\mathbf{b} \quad = \quad \mathbf{R}\mathbf{b_f} + \mathbf{c} \tag{6.16}$$

where $\mathbf{b_f}$ is the vector of freely determined coefficients. We now specify the right-hand side of this equation for each of the three relevant cases: i1=T, i2=F (simple level-constraint), i1=F, i2=T (simple time-shift constraint) and i1=i2=T (both constraints imposed).

The case $i1 < -T$, $i2 < -F$

We consider a general multivariate framework with $m+1$ explanatory variables $(x_t, w_{1t}, ..., w_{mt})$. In this case we obtain

$$b^u_{\max(0,h)} = w^u - \sum_{k=h, k\neq 0}^{h+L-1} b^u_k$$

where the index $u = 0, ..., m$ runs across series ($u = 0$ corresponds to x_t).

The entries in 13.12 become

$$\mathbf{R} = \begin{pmatrix} \mathbf{C} & 0 & ... & 0 \\ 0 & \mathbf{C} & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & \mathbf{C} \end{pmatrix} \tag{6.17}$$

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & ... & 0 & 0 \\ 0 & 1 & 0 & ... & 0 & 0 \\ ::: & & & & & \\ -1 & -1 & -1 & ... & -1 & -1 \\ ::: & & & & & \\ 0 & 0 & 0 & ... & 1 & 0 \\ 0 & 0 & 0 & ... & 0 & 1 \end{pmatrix} \tag{6.18}$$

$$\mathbf{c}' = (0, ..., 0, w^0, 0, ..., 0 \;||\; 0, ..., 0, w^1, 0, ..., 0 \;||\; ... \;||0, ..., 0, w^m, 0, ..., 0)$$

$$\mathbf{b_f}' = (b_h^0, ..., b_{-1}^0, b_1^0, ..., b_{h+L-1}^0 \;||\; b_h^1, ..., b_{-1}^1, b_1^1, ..., b_{h+L-1}^1 \;||\; ... \;||\; b_h^m, ..., b_{-1}^m, b_1^m, ..., b_{h+L-1}^m)$$

The vector of -1's in $\mathbf{C}$ is in row-position $\max(0, -h)+1$ whereas the constant w^u $(u = 0, ..., m)$ in $\mathbf{c}$ is in position $u*(L-1)+\max(0, -h)+1$. The vector $\mathbf{b_f'}$ collects all freely determined parameters (thus b_0^u is missing) whereas $\mathbf{b}$ collects all coefficients: the former vector is used for optimization and the latter is required for filtering.

The case $i1 < -F$, $i2 < -T$

We consider a simple time-shift constraint without level requirement[9] and we first assume $s \neq 0$ and $h < 0$ (backcast). From 6.14 we obtain

$$-sb_0^u = -(h-s)b_h^u - (h+1-s)b_{h+1}^u - ... - (-1-s)b_{-1}^u - (1-s)b_1 - (2-s)b_2^u - ... - (h+L-1-s)b_{h+L-1}^u$$

or equivalently

$$b_0^u = \frac{(h-s)b_h^u + (h+1-s)b_{h+1}^u + ... + (-1-s)b_{-1}^u + (1-s)b_1 + (2-s)b_2^u + ... + (h+L-1-s)b_{h+L-1}^u}{s}$$

Such that

$$\mathbf{C}_{h<0} = \begin{pmatrix} 1 & 0 & 0 & ... & ... & ... & ... & 0 & 0 \\ 0 & 1 & 0 & ... & ... & ... & ... & 0 & 0 \\ ::: & & & & & & & & \\ \frac{h-s}{s} & \frac{h+1-s}{s} & \frac{h+2-s}{s} & ... & \frac{-1-s}{s} & \frac{1-s}{s} & \frac{2-s}{s} & ... & \frac{h+L-1-s}{s} \\ ::: & & & & & & & & \\ 0 & 0 & 0 & ... & ... & ... & ... & 1 & 0 \\ 0 & 0 & 0 & ... & ... & ... & ... & 0 & 1 \end{pmatrix} \tag{6.19}$$

$$\mathbf{c}'_{h<0} = \mathbf{0} \tag{6.20}$$

$$\mathbf{b_f}' = (b_h^0, ..., b_{-1}^0, b_1^0, ..., b_{h+L-1}^0 \;||\; b_h^1, ..., b_{-1}^1, b_1^1, ..., b_{h+L-1}^1 \;||\; ... \;||\; b_h^m, ..., b_{-1}^m, b_1^m, ..., b_{h+L-1}^m)$$

[9] This case would be quite unusual in a classic model-based approach because imposition of the second-order constraint would require the first-order constraint as a prerequisite.

The vector $\left(\frac{h-s}{s}, ..., \frac{h+L-1-s}{s}\right)$ in $\mathbf{C}_{h<0}$ is in row-position $-h+1$. The vector $\mathbf{b_f}$ collects the freely determined coefficients: all coefficients except b_0^u.

The case $h = 0$ (nowcast) is handled by

$$\begin{aligned}\mathbf{C}_{h=0} &= \begin{pmatrix} \frac{1-s}{s} & \frac{2-s}{s} & ... & \frac{L-1-s}{s} \\ 1 & 0 & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & 1 \end{pmatrix} \\ \mathbf{c}'_{h=0} &= \mathbf{0}\end{aligned}$$

$$\mathbf{b_f}' = (b_1^0, ..., b_{L-1}^0 \mid\mid b_1^1, ..., b_{L-1}^1 \mid\mid ... \mid\mid b_1^m, ..., b_{L-1}^m)$$

and the case $h > 0$ (forecast) corresponds to

$$(h-s)b_h^u = -(h+1-s)b_{h+1}^u - (h+2-s)b_{h+2}^u - ... - (h+L-1-s)b_{h+L-1}^u$$

or, equivalently

$$b_h^u = \frac{-(h+1-s)b_{h+1}^u - (h+2-s)b_{h+2}^u - ... - (h+L-1-s)b_{h+L-1}^u}{h-s}$$

We obtain

$$\begin{aligned}\mathbf{C}_{h>0} &= \begin{pmatrix} -\frac{h+1-s}{h-s} & -\frac{h+2-s}{h-s} & ... & -\frac{h+L-1-s}{h-s} \\ 1 & 0 & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & 1 \end{pmatrix} \\ \mathbf{c}'_{h>0} &= \mathbf{0}\end{aligned}$$

$$\mathbf{b_f}' = (b_{h+1}^0, ..., b_{h+L-1}^0 \mid\mid b_{h+1}^1, ..., b_{h+L-1}^1 \mid\mid ... \mid\mid b_{h+1}^m, ..., b_{h+L-1}^m)$$

Note that we implicitly assumed $h - s \neq 0$ in the above derivation. Otherwise we would have to isolate b_{h+1} instead of b_h (left as an exercise to the reader). Recall, also, that we assumed $s \neq 0$ in the case $h \leq 0$: if $s = 0$ then we isolate b_1, instead of b_0, in the above expressions[10] (left as an exercise to the reader).

The case $i1 < -T$, $i2 < -T$

As in the previous section, we first tackle the backcast-problem: $h < 0$. Solving for b_0^u and b_1^u in 6.14 and 6.15 leads to

$$\begin{aligned} b_1^u &= s^u w^u - h b_h^u - (h+1)b_{h+1}^u - ... - (-1)b_{-1}^u - 0 - 2b_2^u - 3b_3^u - ... - (h+L-1)b_{h+L-1}^u \\ b_0^u &= w^u(1-s^u) + (h-1)b_h^u + h b_{h+1}^u + ... + (-2)b_{-1}^u + b_2^u + 2b_3^u + ... + (L-2+h)b_{L-1+h}^u \end{aligned}$$

[10] If $s = 0$ then, obviously, $1 - s \neq 0$ and therefore the quotients will be defined.

We obtain

$$\mathbf{C}_{h<0} = \begin{pmatrix} 1 & 0 & 0 & ... & & & 0 & 0 \\ 0 & 1 & 0 & ... & & & 0 & 0 \\ ::: & & & & & & & \\ h-1 & h & h+1 & ... & -2 & 1 & 2 & ... & (L-2+h) \\ -h & -(h+1) & -(h+2) & ... & 1 & -2 & -3 & ... & -(h+L-1) \\ ::: & & & & & & & \\ 0 & 0 & 0 & ... & & & 1 & 0 \\ 0 & 0 & 0 & ... & & & 0 & 1 \end{pmatrix} \tag{6.21}$$

$$\mathbf{c}'_{h<0} = (0,...,0,w^0(1-s^0),s^0w^0,0,...,0 \;||\; 0,...,0,w^1(1-s^1),s^1w^1,0,...,0 \;||\; ... \;||\; 0,...,0,w^m(1-s^m),s^mw^m,0,...,0) \tag{6.22}$$

$$\mathbf{b_f}' = (b^0_{-h},...,b^0_{-1},b^0_2,...,b^0_{L-1-h} \;||\; b^1_{-h},...,b^1_{-1},b^1_2,...,b^1_{L-1-h} \;||\; ... \;||\; b^m_{-h},...,b^m_{-1},b^m_2,b^m_{L-1-h}) \tag{6.23}$$

The non-trivial weighting-vectors in $\mathbf{C}_{h<0}$ are located in row-positions $-h+1$ and $-h+2$ whereas the constants $w^u(1-s^u)$ and s^uw^u in $\mathbf{c}_{h<0}$ are to be found in positions $u*(L-2)-h+1$ and $u*(L-2)-h+2$, respectively. Note that both b^u_0 and b^u_1 are now missing in the vector of freely determined coefficients $\mathbf{b_f}$.

For $h=0$ we obtain

$$\mathbf{C}_{h=0} = \begin{pmatrix} 1 & 2 & ... & (L-2) \\ -2 & -3 & ... & -(L-1) \\ 1 & & 0 & 0 \\ ::: & & & \\ 0 & ... & 0 & 1 \end{pmatrix} \tag{6.24}$$

$$\mathbf{c}'_{h=0} = w^0(1-s^0),s^0w^0,0,...,0 \;||\; w^1(1-s^1),s^1w^1,0,...,0 \;||\; ... \;||w^m(1-s^m),s^mw^m,0,...,0)$$

$$\mathbf{b_f}' = (b^0_2,...,b^0_{L-1} \;||\; b^1_2,...,b^1_{L-1} \;||\; ... \;||\; b^m_2,...,b^m_{L-1})$$

Finally, for $h>0$ the constraints become

$$\begin{aligned} (h+1)b^u_{h+1} &= s^uw^u-(h+2)b^u_{h+2}-(h+3)b_{h+3}-...-(h+L-1)b^u_{h+L-1} \\ hb^u_h &= w^u(1-s^u)+(h+1)b^u_{h+2}+(h+2)b^u_{h+3}+...+(h+L-2)b^u_{h+L-1} \end{aligned}$$

or, equivalently

$$\begin{aligned} b^u_{h+1} &= \frac{s^uw^u-(h+2)b^u_{h+2}-(h+3)b_{h+3}-...-(h+L-1)b^u_{h+L-1}}{h+1} \\ b^u_h &= \frac{w^u(1-s^u)+(h+1)b^u_{h+2}+(h+2)b^u_{h+3}+...+(h+L-2)b^u_{h+L-1}}{h} \end{aligned}$$

so that

$$\begin{aligned}
\mathbf{C}_{h>0} &= \begin{pmatrix} \frac{h+1}{h} & \frac{h+2}{h} & \frac{3}{h} & ... & \frac{h+L-2}{h} \\ -\frac{h+2}{h+1} & -\frac{h+3}{h+1} & -\frac{h+4}{h+1} & ... & -\frac{h+L-1}{h+1} \\ 1 & 0 & 0 & ... & 0 \\ 0 & 1 & 0 & ... & 0 \\ ::: & & & & \\ 0 & 0 & 0 & ... & 1 \end{pmatrix} \\
\mathbf{c}'_{h>0} &= (\frac{w^0(1-s^0)}{h}, \frac{s^0 w^0}{h+1}, 0, ..., 0 \,||\, \frac{w^1(1-s^1)}{h}, \frac{s^1 w^1}{h+1}, 0, ..., 0 \,||\, ... \,|| \\
&\quad \frac{w^m(1-s^m)}{h}, \frac{s^m w^m}{h+1}, 0, ..., 0) \\
\mathbf{b_f}' &= (b^0_{-h}, ..., b^0_{-1}, b^0_2, ..., b^0_{L-1-h} \,||\, b^1_{-h}, ..., b^1_{-1}, b^1_2, ..., b^1_{L-1-h} \,||\, ... \,||\, b^m_{-h}, ..., b^m_{-1}, b^m_2, ..., b^m_{L-1-h})
\end{aligned}$$

Obviously, all expressions on the right-hand side of 13.12 depend on i1, i2 as well as on h but for notational simplicity we refrain from attaching a cumbersome triple index to them.

6.4 Constrained Optimization

6.4.1 Generalized Criterion

Optimal constrained filter coefficients can be obtained by plugging 13.12 into 4.40 and by taking derivatives

$$\begin{aligned}
d/d\mathbf{b_f} \text{ Criterion} &= d/d\mathbf{b_f}\ (\mathbf{Y}_{\text{rot}} - \mathbf{X}_{\text{rot}}\,(\mathbf{R}\mathbf{b_f} + \mathbf{c}))'(\mathbf{Y}_{\text{rot}} - \mathbf{X}_{\text{rot}}\,(\mathbf{R}\mathbf{b_f} + \mathbf{c})) \\
&= -2(\mathbf{Y}_{\text{rot}})'\Re\,(\mathbf{X}_{\text{rot}})\,\mathbf{R} - \mathbf{2}\Re\Big\{(\mathbf{X}_{\text{rot}}\mathbf{c})'(\mathbf{X}_{\text{rot}}\mathbf{R})\Big\} + \mathbf{2b_f'}\Re\Big\{(\mathbf{X}_{\text{rot}}\mathbf{R})'\mathbf{X}_{\text{rot}}\mathbf{R}\Big\}
\end{aligned}$$

The *constrained* solution is obtained by equating this expression to zero

$$\begin{aligned}
\hat{\mathbf{b}}_f^{\text{Const}}(i1, i2) &= \Big\{\Re\Big[(\mathbf{X}_{\text{rot}}\mathbf{R})'\mathbf{X}_{\text{rot}}\mathbf{R}\Big]\Big\}^{-1}\Big((\Re(\mathbf{X}_{\text{rot}})\mathbf{R})'\mathbf{Y}_{\text{rot}} + \Re\Big\{(\mathbf{X}_{\text{rot}}\mathbf{R})'\mathbf{X}_{\text{rot}}\mathbf{c}\Big\}\Big) \\
&= \Big\{\Re\Big[(\mathbf{X}_{\text{rot}}\mathbf{R})'\mathbf{X}_{\text{rot}}\mathbf{R}\Big]\Big\}^{-1}\Big((\Re(\mathbf{X}_{\text{rot}})\mathbf{R})'\mathbf{Y}_{\text{rot}} + \mathbf{Level}\Big) \qquad (6.25)
\end{aligned}$$

where the vector

$$\mathbf{Level} := \Re\Big\{(\mathbf{X}_{\text{rot}}\mathbf{R})'\mathbf{X}_{\text{rot}}\mathbf{c}\Big\}$$

can be interpreted as a generalized level term[11]. A comparison with 4.42 illustrates that constrained and unconstrained solutions are similar up to the presence of the transformation $\mathbf{R}$ and the occurrence of the new level-term **Level**. In particular $\mathbf{R} = \mathbf{Id}$ and $\mathbf{c} = \mathbf{0}$ replicates 4.42, as expected (no constraints imposed). The 'full-coefficient' vector $\mathbf{b}$, indispensable for filtering, is obtained by plugging the obtained constrained solution $\hat{\mathbf{b}}_f^{\text{Const}}(i1, i2)$ into 13.12.

[11] If $w^u = 0$ for all $u = 0, ..., m$ then $\mathbf{Level} = \mathbf{0}$.

6.5 Exercises: Implementing Constraints in MDFA

We rely on the bivariate leading indicator design proposed in chapter 4 and compute amplitude and time-shift functions for all possible combinations of the Boolean $(i1, i2)$.

1. Rely on the data of section 4.7.1 (leading indicator) and use the default settings ($Lag = 0$, $i1 = i2 = F$: unconstrained design) for estimating filter coefficients for filters of length $L = 13$. Use the third series ($a_1 = -0.9$) and compute and plot amplitude and time-shift functions, see fig.6.1.

```
        target         x leading indicator
[1,] -2.926789 -2.926789          3.082815
[2,]  2.925098  2.925098         -3.965857
[3,] -3.870182 -3.870182          2.934987
[4,]  3.026988  3.026988         -3.754376
[5,] -3.554612 -3.554612          3.512037
[6,]  3.539266  3.539266         -2.150498
```

```
> # Filter length
> L<-13
> # Fully in sample
> insample<-nrow(data_matrix_120)
> # d=0 for stationary series: see default settings
> weight_func<-spec_comp(insample, data_matrix_120, d)$weight_func
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Estimate filter coefficients
> mdfa_obj<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
> file = paste("z_mdfa_ar1_amp_shift_Lag_0_iF_i2F.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> K<-nrow(weight_func)-1
> par(mfrow = c(2, 1))
> # amplitude functions
> mplot <- abs(mdfa_obj$trffkt)
> # x-axis
> freq_axe <- rep(NA, K + 1)
> freq_axe[1] <- 0
> freq_axe[1 + (1 : 6) * K / 6] <- c(paste0(c("", 2 : 5), "pi/6"), "pi")
> ax <- freq_axe
> # colors, title and additional titles
> insamp <- 1.e+90
> colo <- NULL
```

```
> plot_title <- "Amplitude Functions"
> title_more <- colnames(x[, -1])
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> # time-shift
> mplot <- Arg(t(sign(apply(mdfa_obj$b, 2, sum)) * t(mdfa_obj$trffkt))) /
+       ((0 : (nrow(mdfa_obj$trffkt) - 1)) * pi / (nrow(mdfa_obj$trffkt) - 1))
> # We use the exact formula for the time-shift in frequency zero
> mplot[1, ] <- apply(mdfa_obj$b * ((0 : (L - 1))), 2, sum) /
+       apply(mdfa_obj$b, 2, sum)
> plot_title <- "Time-Shift"
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
```

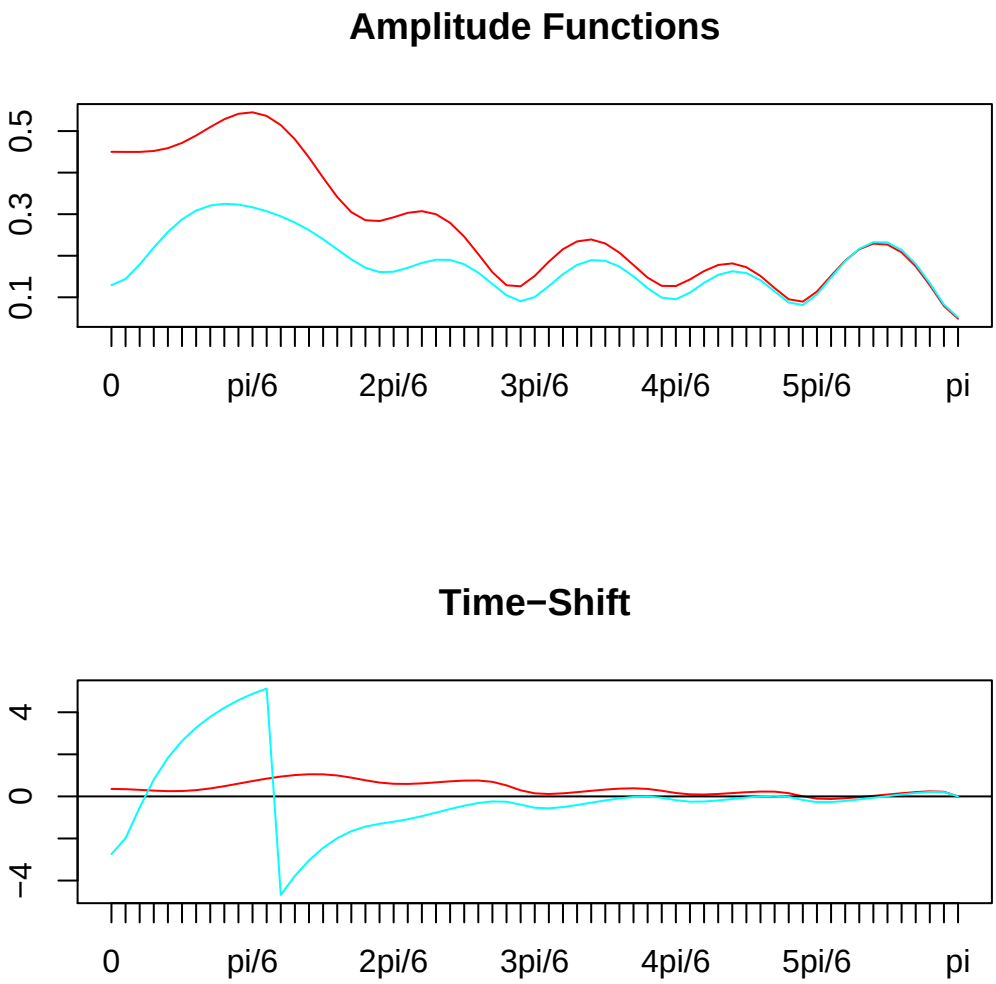


Figure 6.1: Amplitude (top) and time-shift (bottom) functions: unconstrained i1=I2=F

No constraints are imposed in frequency zero.

2. Same as above but impose a simple level constraint: $i1 = T, i2 = F$ and $w^0 = \frac{1+\sqrt{5}}{2}, w^1 = -\sqrt{2}$. Plot and compare the resulting amplitude functions, see fig.6.2. Hint: we here use the call based on the context-specific function $MDFA_mse_constraint$ whose additional arguments account for the relevant constraints.

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Impose level constraint
> i1<-T
```

```
> # Constraints: for series 1 and 2
> weight_constraint<-c((1+sqrt(5))/2,-sqrt(2))
> # Call to the context-specific MDFA_mse_constraint function
> mdfa_obj<-MDFA_mse_constraint(L,weight_func,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
> file = paste("z_mdfa_ar1_amp_shift_Lag_0_iT_i2F.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> K<-nrow(weight_func)-1
> par(mfrow = c(2, 1))
> # amplitude functions
> mplot <- abs(mdfa_obj$trffkt)
> # x-axis
> freq_axe <- rep(NA, K + 1)
> freq_axe[1] <- 0
> freq_axe[1 + (1 : 6) * K / 6] <- c(paste0(c("", 2 : 5), "pi/6"), "pi")
> ax <- freq_axe
> # colors, title and additional titles
> insamp <- 1.e+90
> colo <- NULL
> plot_title <- "Amplitude Functions"
> title_more <- colnames(x[, -1])
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> # time-shift
> mplot <- Arg(t(sign(apply(mdfa_obj$b, 2, sum)) * t(mdfa_obj$trffkt))) /
+       ((0 : (nrow(mdfa_obj$trffkt) - 1)) * pi / (nrow(mdfa_obj$trffkt) - 1))
> # We use the exact formula for the time-shift in frequency zero
> mplot[1, ] <- apply(mdfa_obj$b*(0:(L-1)),2,sum)/apply(mdfa_obj$b, 2, sum)
> plot_title <- "Time-Shift"
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
> print(c("Level restrictions", round(apply(mdfa_obj$b,2,sum),4)))
```

```
[1] "Level restrictions" "1.618"              "-1.4142"
```

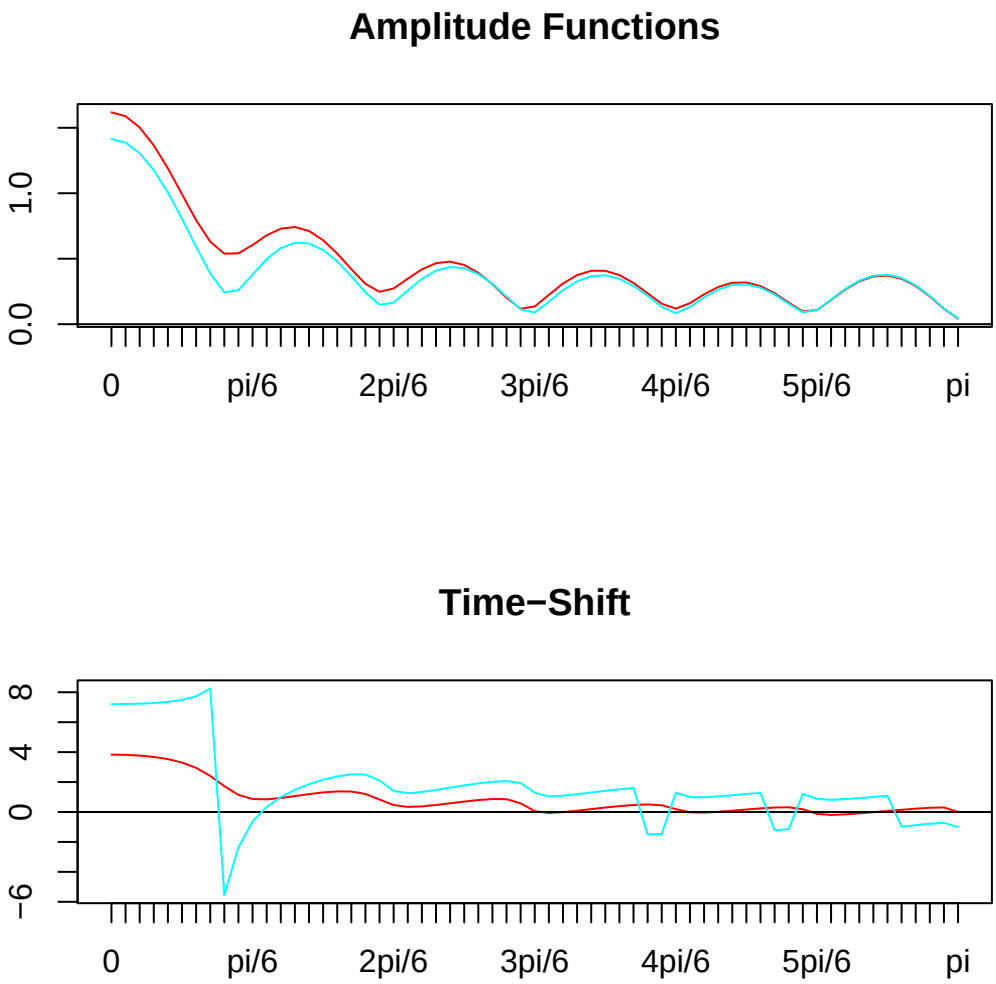


Figure 6.2: Amplitude (top) and time-shift (bottom) functions: i1=T,i2=F

As expected, the bivariate transfer function satisfies the imposed first order restrictions $w^0 = \frac{1+\sqrt{5}}{2}, w^1 = -\sqrt{2}$ in frequency zero. Consider that the resulting filter is not obtained by a simple scaling of its coefficients; instead, it is the best (MSE) filter among all those which satisfy the restriction (a simple scaling could not pretend to optimality, in general).

3. Same as above but impose a simple time-shift restriction, $i1 = F, i2 = T$, and select $s^0 = e = \exp(1), s^1 = -\pi$. Compute and compare the time-shift functions, see fig.6.3.

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))#b0_H0
> # Estimate filter coefficients: note that i1<-F by sourcing
```

```
> #   the default parameters
> i2<-T
> shift_constraint<-c(exp(1),-pi)
> mdfa_obj<-MDFA_mse_constraint(L,weight_func,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
> file = paste("z_mdfa_ar1_amp_shift_Lag_0_iF_i2T.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> K<-nrow(weight_func)-1
> par(mfrow = c(2, 1))
> # amplitude functions
> mplot <- abs(mdfa_obj$trffkt)
> # x-axis
> freq_axe <- rep(NA, K + 1)
> freq_axe[1] <- 0
> freq_axe[1 + (1 : 6) * K / 6] <- c(paste0(c("", 2 : 5), "pi/6"), "pi")
> ax <- freq_axe
> # colors, title and additional titles
> insamp <- 1.e+90
> colo <- NULL
> plot_title <- "Amplitude Functions"
> title_more <- colnames(x[, -1])
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> # time-shift
> mplot <- Arg(t(sign(apply(mdfa_obj$b, 2, sum)) * t(mdfa_obj$trffkt))) /
+       ((0 : (nrow(mdfa_obj$trffkt) - 1)) * pi / (nrow(mdfa_obj$trffkt) - 1))
> # We use the exact formula for the time-shift in frequency zero
> mplot[1, ] <- apply(mdfa_obj$b*(0:(L-1)),2,sum)/apply(mdfa_obj$b, 2, sum)
> plot_title <- "Time-Shift"
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
> print(c("Time-shift restrictions",
+     round(apply(mdfa_obj$b*(0:(L-1)),2,sum)/apply(mdfa_obj$b, 2, sum),4)))
```

```
[1] "Time-shift restrictions" "2.7183"                  "-3.1416"
```

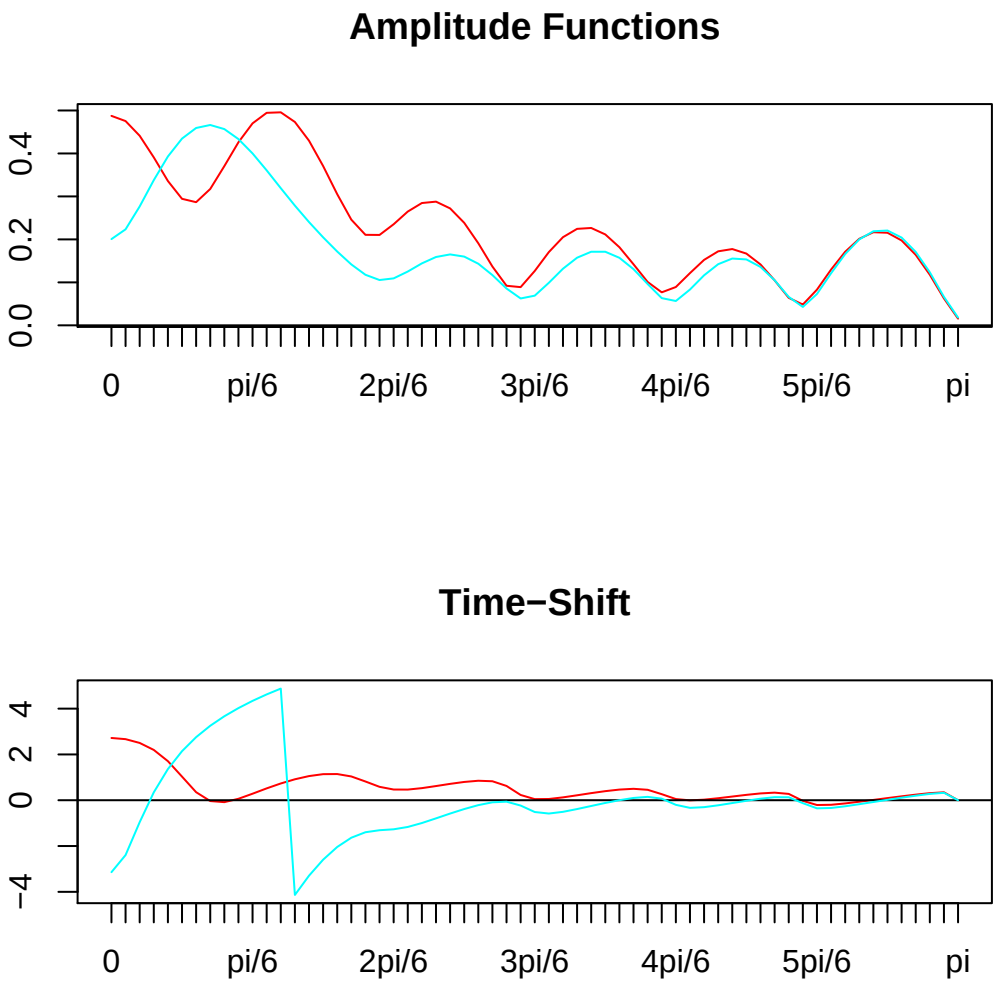


Figure 6.3: Amplitude (top) and time-shift (bottom) functions: i1=F,i2=T

The time-shifts agree with our constraints, $s^0 = e = \exp(1), s^1 = -\pi$, in frequency zero.

4. Impose both constraints simultaneously, see fig.6.4.

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Impose both constraints
> i1<-T
> i2<-T
> # Specify the constraints
> weight_constraint<-c((1+sqrt(5))/2,-sqrt(2))
```

```
> shift_constraint<-c(exp(1),-pi)
> mdfa_obj<-MDFA_mse_constraint(L,weight_func,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
> file = paste("z_mdfa_ar1_amp_shift_Lag_0_iT_i2T.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> K<-nrow(weight_func)-1
> par(mfrow = c(2, 1))
> # amplitude functions
> mplot <- abs(mdfa_obj$trffkt)
> # x-axis
> freq_axe <- rep(NA, K + 1)
> freq_axe[1] <- 0
> freq_axe[1 + (1 : 6) * K / 6] <- c(paste0(c("", 2 : 5), "pi/6"), "pi")
> ax <- freq_axe
> # colors, title and additional titles
> insamp <- 1.e+90
> colo <- NULL
> plot_title <- "Amplitude Functions"
> title_more <- colnames(x[, -1])
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> # time-shift
> mplot <- Arg(t(sign(apply(mdfa_obj$b, 2, sum)) * t(mdfa_obj$trffkt))) /
+       ((0 : (nrow(mdfa_obj$trffkt) - 1)) * pi / (nrow(mdfa_obj$trffkt) - 1))
> # We use the exact formula for the time-shift in frequency zero
> mplot[1, ] <- apply(mdfa_obj$b*(0:(L-1)),2,sum)/apply(mdfa_obj$b, 2, sum)
> plot_title <- "Time-Shift"
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
> print(c("Level restrictions", round(apply(mdfa_obj$b,2,sum),4)))


[1] "Level restrictions" "1.618"              "-1.4142"


> print(c("Time-shift restrictions",
+ round(apply(mdfa_obj$b*(Lag:(L-1+Lag)),2,sum)/apply(mdfa_obj$b, 2, sum),4)))


[1] "Time-shift restrictions" "2.7183"                  "-3.1416"
```

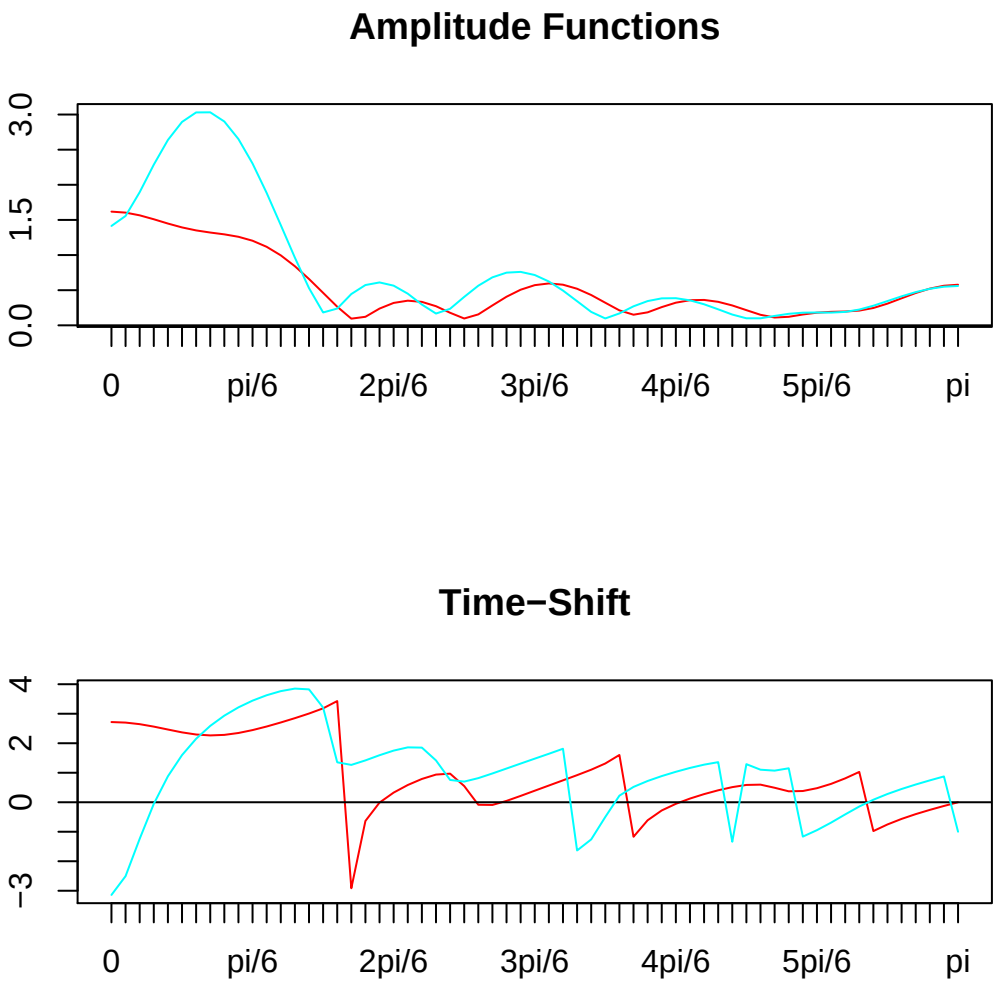


Figure 6.4: Amplitude (top) and time-shift (bottom) functions: i1=i2=T

Both functions satisfy the restrictions $w^0 = \frac{1+\sqrt{5}}{2}, w^1 = -\sqrt{2}$ and $s^0 = e = \exp(1), s^1 = -\pi$, as desired.

5. To conclude, we briefly analyze the effect of $Lag = -h \neq 0$. Specifically we address a two-step ahead forecast of the target signal and analyze synchronization effects obtained by the time-shift constraint.

   - We rely on the following example: $i1 = F$, $i2 = T$ (time-shift constraint only[12]), $s^0 = 1, s^1 = 2$ and we set $Lag = -2$ (two-step ahead forecast of the signal). This

[12]This setting could not be replicated by classic model-based approaches because the I(2)-shift constraint would assume the I(1)-level constraint as a prerequisite.

particular example is not selected for its practical relevance; instead we want to illustrate handling of potential singularities by our code (note that $Lag + s^1 = 0$).

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))#b0_H0
> # Estimate filter coefficients: note that i1<-F by sourcing the default parameters
> i2<-T
> Lag<--2
> shift_constraint<-c(0,2)
> mdfa_obj<-MDFA_mse_constraint(L,weight_func,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
> b_mat<-mdfa_obj$b
> trffkt_mdfa<-mdfa_obj$trffkt
> file = paste("z_mdfa_ar1_amp_shift_Lag_0_iF_i2T_Lag.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> K<-nrow(weight_func)-1
> par(mfrow = c(2, 1))
> # amplitude functions
> mplot <- abs(mdfa_obj$trffkt)
> # x-axis
> freq_axe <- rep(NA, K + 1)
> freq_axe[1] <- 0
> freq_axe[1 + (1 : 6) * K / 6] <- c(paste0(c("", 2 : 5), "pi/6"), "pi")
> ax <- freq_axe
> # colors, title and additional titles
> insamp <- 1.e+90
> colo <- NULL
> plot_title <- "Amplitude Functions"
> title_more <- colnames(x[, -1])
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> # time-shift
> mplot <- Arg(t(sign(apply(mdfa_obj$b, 2, sum)) * t(mdfa_obj$trffkt))) /
+       ((0 : (nrow(mdfa_obj$trffkt) - 1)) * pi / (nrow(mdfa_obj$trffkt) - 1))
> # We use the exact formula for the time-shift in frequency zero
> mplot[1, ] <- apply(mdfa_obj$b*(0:(L-1)),2,sum)/apply(mdfa_obj$b, 2, sum)
> plot_title <- "Time-Shift"
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
> print(c("Time-shift restrictions",
+         round(apply(mdfa_obj$b*(0:(L-1)),2,sum)/
+           apply(mdfa_obj$b, 2, sum),4)))
```

```
[1] "Time-shift restrictions" "-2"                    "0.01"
```

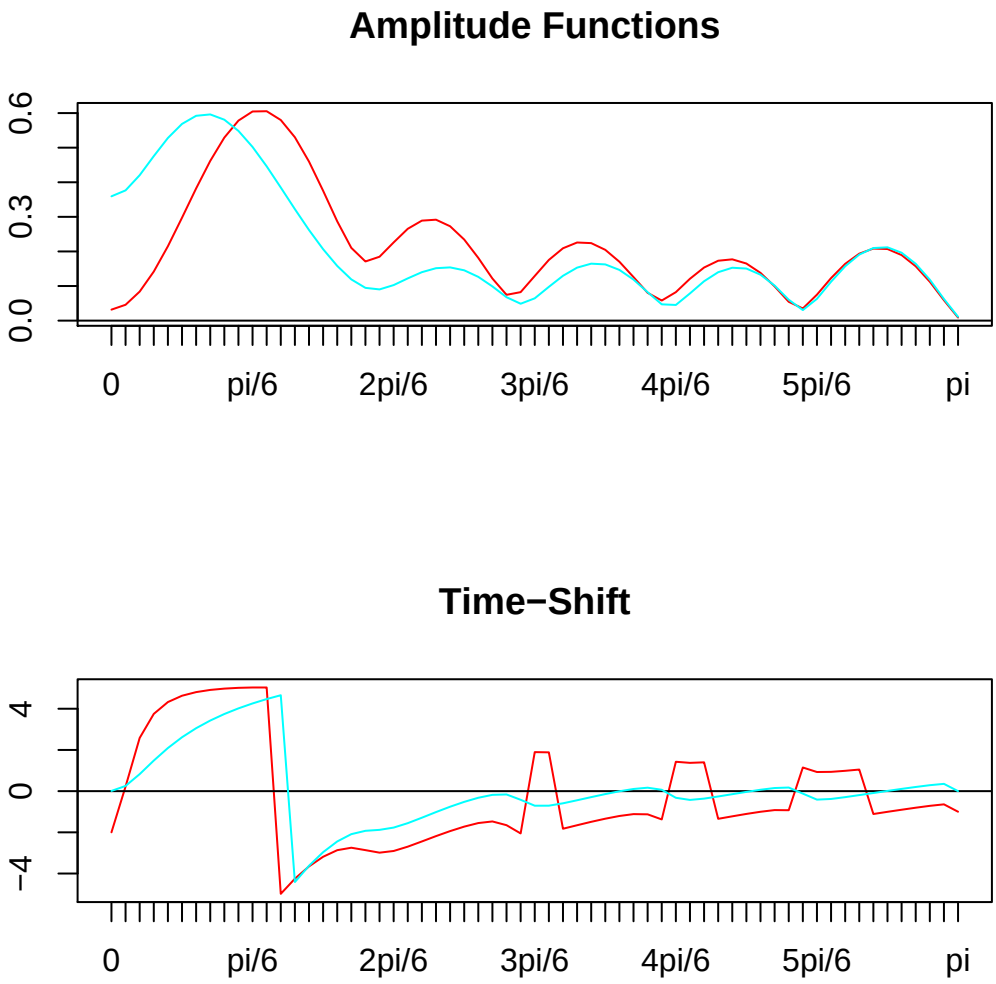


Figure 6.5: Amplitude (top) and time-shift (bottom) functions: i1=F,i2=T, Lag=-2

The first time-shift agrees with our constraints when shifted by Lag: $s^0 + Lag = 0 - 2 = -2$. The second time-shift should be $s^1 + Lag = 2 - 2 = 0$ which would generate a singularity (dividing by zero); in order to circumvent the problem we 'shift the shift' by a small constant (0.01) in our code: the resulting effect should be negligible by all practical means and the trick allows a much simpler implementation of the constraints in the considered case (i1=F,i2=T).

- Feed a signal to the filters of the the bivariate design and verify the time-shift constraints. Hints: we generate two filter outputs, one for each filter; since the constraints apply in frequency zero, our input signals are linear trends (signals of frequency zero);

since our amplitude functions are not normalized to one, in frequency zero, we need to normalize the output signals (by the inverse of the transfer functions in frequency zero).

```
> len_t<-100
> # Generate two linear trends (they are shifted by a constant
> # in order to be distinguished)
> trend_data<-cbind(1:len_t,0.5+1:len_t)
> # Compute both output series: normalize by the inverse
> #   transferfunctions in frequency zero
> yhat_multivariate<-cbind(rep(NA,len_t),rep(NA,len_t))
> for (i in 1:ncol(yhat_multivariate))
+   for (j in L:len_t)
+ #  The transfer function in frequency zero is a real number:
+ #   R computes a complex number with vanishing imaginary part.
+ #   We have to extract the real part because otherwise the complex
+ #   series would not be plotted...
+     yhat_multivariate[j,i]<-sum(b_mat[,i]*trend_data[j:(j-L+1),i])/
+       Re(trffkt_mdfa[1,i])
> mplot<-cbind(trend_data,yhat_multivariate)
> dimnames(mplot)[[2]]<-c("Input 1","Input 2","Output 1","Output 2")
> # Display the last observations
> tail(mplot)
```

	Input 1	Input 2	Output 1	Output 2
[95,]	95	95.5	97	95.49
[96,]	96	96.5	98	96.49
[97,]	97	97.5	99	97.49
[98,]	98	98.5	100	98.49
[99,]	99	99.5	101	99.49
[100,]	100	100.5	102	100.49

We observe that 'Output 1' leads 'Input 1' by two time units, as desired. 'Output 2' is almost synchronized with 'Input 2' up to the small correction by 0.01 (to avoid singular expressions in the R-code).

The chosen real-numbered constraints in the above exercises illustrate that arbitrary (real or integer-numbered) restrictions can be imposed. Whether they are useful, or not, depends on the particular application (integration order of the DGP) as well as on the particular research priorities of the user (MSE or turning points).

6.6 Summary

- We proposed a set of filter restrictions which affect the level and the time-shift of the filter outputs.

- We extended the stationary DFA-criterion to non-stationary integrated processes. A generalization to multivariate filters is proposed in chapter 17.
- Given a non-stationary I(1)- or I(2)-input signal x_t and a generic target specification $\Gamma(\omega)$, the output $\hat{y}_t$ of the constrained forecast-, nowcast- or backcast-filter $\hat{\Gamma}(\omega)$ is able to track the signal in the sense that the filter error remains stationary (cointegration).
- In practice, one is often interested in a vanishing time-shift whether the data is integrated or not.
- The proposed restrictions apply independently to each time series of a multivariate design. Cross-sectional links (cointegration) are proposed and discussed in chapter 17.
- We proposed a formal matrix implementation and derived a generalized optimization criterion. The unconstrained case, proposed in the previous chapters, is nested in the proposed solution.

Chapter 7

ATS-Trilemma: the Univariate Case

We propose an extension of the classic MSE-paradigm which addresses nowcast and forecast applications[1]. Specifically, we split the original MSE-norm into three components, identified as Accuracy, Timeliness and Smoothness. The resulting two-dimensional trade-off, controlled by the parameter-pair λ, η in the head of the main function call, is called Accuracy-Timeliness-Smoothness Trilemma, or ATS-Trilemma for short, see Wildi (2005), McElroy and Wildi (2015). We derive a generic optimization principle, called Customization, which nests the classic MSE-approach. We show that the ATS-trilemma collapses to a one-dimensional trade off, the so-called AT-dilemma, in the case of forecasting. We infer that classic (pseudo-maximum likelihood) one-step ahead forecast approaches are incapable of addressing Timeliness and Smoothness, simultaneously. Efficiency gains of customized designs are quantified in a series of empirical examples.

The ATS-trilemma and the generic customization principle are introduced in section 7.1; section 7.2 highlights the classic dichotomic forecast paradigm; quadratic optimization criteria and closed-form solutions are presented in section 7.3; an application of customization is proposed in section 7.4; performance measures are presented in section 7.5; finally, sections 7.6 and 7.7 assess performances of customized designs when benchmarked against classic MSE-approaches.

7.1 Signal Extraction and the ATS-Trilemma

We address the univariate dfa-case and generalize criterion 4.27. Our treatment follows DFA, section 4.3, and McElroy and Wildi (2015).

[1] Backcasts were discussed in chapter 5.

7.1.1 Decomposition of the MSE-Norm

The DFA emphasizes the so-called transferfunction-error $\Gamma(\cdot) - \hat{\Gamma}(\cdot)$. The following geometric identity holds in general (law of cosine in the complex plane):

$$\begin{aligned} |\Gamma(\omega) - \hat{\Gamma}(\omega)|^2 &= A(\omega)^2 + \hat{A}(\omega)^2 - 2A(\omega)\hat{A}(\omega)\cos\left(\hat{\Phi}(\omega) - \Phi(\omega)\right) \\ &= (A(\omega) - \hat{A}(\omega))^2 \\ &\quad +2A(\omega)\hat{A}(\omega)\left[1 - \cos\left(\hat{\Phi}(\omega) - \Phi(\omega)\right)\right] \end{aligned} \tag{7.1}$$

If Γ is symmetric and positive, then $\Phi(\omega) \equiv 0$ so that we can omit $\Phi(\omega)$ from our notation[2]. By inserting 7.1 into 4.27 and using $1 - \cos(\hat{\Phi}(\omega)) = 2\sin(\hat{\Phi}(\omega)/2)^2$ we obtain

$$\begin{aligned} &\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\left|\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)\right|^2 I_{TX}(\omega_k) \\ = \quad &\frac{2\pi}{T}\sum_{k=-T/2}^{T/2}(A(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \qquad (7.2) \\ &+\frac{2\pi}{T}\sum_{k=-T/2}^{T/2} 4A(\omega_k)\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \qquad (7.3) \end{aligned}$$

The first summand 7.2 is the distinctive part of the total mean-square filter error which is attributable to the amplitude function of the real-time filter (the MS-amplitude error). The second summand 7.3 measures the distinctive contribution of the phase or time-shift to the total mean-square error (the MS-time-shift error). Note that the product $A(\omega_k)\hat{A}(\omega_k)I_{TX}(\omega_k)$ in 7.3 maps the scales of the filter outputs $y_t, \hat{y}_t$ to the frequency-domain (the phase function alone would be dimensionless).

7.1.2 Accuracy, Timeliness and Smoothness (ATS-) Error-Components

The previous decomposition can be refined by splitting each term into contributions from *pass-* and *stopbands*. For ease of exposition we assume an ideal lowpass target

$$\Gamma(\omega) = \begin{cases} 1\ , & |\omega| \leq \text{cutoff} \\ 0\ , & \text{otherwise} \end{cases}$$

The set $\{\omega_k | |\omega_k| \leq \text{cutoff}\}$ corresponds to the passband and its complement $\{\omega_k | |\omega_k| > \text{cutoff}\}$ is the stop- or rejection-band. We now define

$$\left.\begin{array}{rcl} \text{A(ccuracy)} & := & \frac{2\pi}{T}\sum_{\text{Passband}}(A(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \\ \text{S(moothness)} & := & \frac{2\pi}{T}\sum_{\text{Stopband}}(A(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \\ \text{T(imeliness)} & := & \frac{2\pi}{T}\sum_{\text{Passband}} 4A(\omega_k)\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \\ \text{R(esidual)} & := & \frac{2\pi}{T}\sum_{\text{Stopband}} 4A(\omega_k)\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \end{array}\right\} \tag{7.4}$$

[2]The ideal trend, classical filters (HP, CF, henderson) or model-based filters are typically positive (symmetric) filters. It should be clear that $\Phi(\omega)$ cannot be omitted if the condition is unfulfilled.

- *Residual* is that part of the MSE which is contributed by the time-shift in the stopband: it *vanishes* exactly in the case of the ideal trend. Even if it does not vanish, it is generally negligible because the product $A(\omega_k)\hat{A}(\omega_k)$ is small in the stopband. We now ignore this error-term (it is dropped from subsequent notation).
- *Accuracy* measures the contribution to the MSE-norm which would be obtained by ignoring time-shift (passband) and noise suppression (stop-band) issues[3].
- *Smoothness* measures the contribution to total MSE of the undesirable high-frequency noise, leaking from the imperfect amplitude fit in the stopband (convolution theorem 4.20). This measure is tightly linked to classical time-domain *curvature* (smoothness) measures.
- *Timeliness* measures the MSE-contribution generated by the time-shift. This measure is closely linked to the well-known (time-domain) *peak-correlation*, as referenced against the target signal.

We are now able to address each single or each *pairwise* combination of error-terms by splitting the MSE-criterion accordingly. Typically, speed (Timeliness) and noise suppression (Smoothness) are important properties of a real-time filter in prospective estimation problems (nowcast, forecast)

7.1.3 Generic Customized Criterion

We are in a position to generalize the mean-square optimization criterion 4.27 by assigning weights to the ATS-components (recall that the Residual either vanishes exactly or is negligible):

$$\text{MSE-Cust}(\lambda_1, \lambda_2) = \text{Accuracy} + (1+\lambda_1)\text{Timeliness} + (1+\lambda_2)\text{Smoothness} \rightarrow \min_{\mathbf{b}} \tag{7.5}$$

Selecting $\lambda_1 > 0$ results, ceteris paribus, in a smaller Timeliness component; as we shall see the resulting filter will be faster: turning-points can be detected earlier (smaller peak-correlation against the target signal). Selecting $\lambda_2 > 0$ magnifies, ceteris paribus, the Smoothness term: the output of the resulting filter is smoother or, stated otherwise, undesirable high-frequency noise is damped more effectively. Finally, selecting $\lambda_1 > 0, \lambda_2 > 0$ emphasizes both attributes: the resulting filter-output is able to gain both in terms of 'speed' as well as in terms of 'noise suppression'. The ATS-trilemma confers the user the possibility to adjust trade-offs according to research priorities.

The schematic criterion 7.5 can be generalized by substituting weighting functions $W_1(\omega_k) \geq 0$, $W_2(\omega_k) \geq 0$ for the fixed weights λ_1, λ_2:

$$\begin{aligned}
&\frac{2\pi}{T} \sum_{\text{Passband}} (A(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \\
&+\frac{2\pi}{T} \sum_{\text{Stopband}} (A(\omega_k) - \hat{A}(\omega_k))^2 W_2(\omega_k) I_{TX}(\omega_k) \\
&+\frac{2\pi}{T} \sum_{\text{Passband}} 4A(\omega_k)\hat{A}(\omega_k) \sin(\hat{\Phi}(\omega_k)/2)^2 W_1(\omega_k) I_{TX}(\omega_k) \rightarrow \min_{\mathbf{b}}
\end{aligned}$$

[3]It is the performance of a symmetric filter (no time-shift) with perfect noise suppression ($\hat{A}(\cdot) = \Gamma(0) = 0$ in the stopband) and with the same amplitude as $\hat{\Gamma}(\cdot)$ in the passband. Such a design could be easily constructed by the techniques presented in DFA, section 1.

Criterion 7.5 can be replicated by selecting

$$\begin{aligned} W_1(\omega_k, \lambda_1) &= 1 + \lambda_1 \\ W_2(\omega_k, \lambda_2) &= 1 + \lambda_2 \end{aligned}$$

Note, however, that W_2 would introduce an undesirable discontinuity in the cutoff-frequency, if $\lambda_2 \neq 0$. Therefore, we propose the following weighting-scheme

$$\left.\begin{aligned} W_1(\omega_k, \lambda) &= 1 + \lambda \\ W_2(\omega_k, \eta) &= (1 + |\omega_k| - \text{cutoff})^\eta \end{aligned}\right\} \tag{7.6}$$

The new function W_2 allows for a continuous transition from pass- to stopband and its monotonic shape overemphasizes undesirable high-frequency components[4]. We re-labelled λ_1, λ_2 by λ, η in order to distinguish symbolically both effects: $\lambda > 0$ emphasizes Timeliness and $\eta > 0$ addresses Smoothness. The criterion can be re-written more compactly as

$$\begin{aligned} &\frac{2\pi}{T} \sum_{\text{All Frequencies}} (A(\omega_k) - \hat{A}(\omega_k))^2 W(\omega_k, \eta) I_{TX}(\omega_k) \\ &+(1+\lambda)\frac{2\pi}{T} \sum_{\text{Passband}} 4A(\omega_k)\hat{A}(\omega_k) \sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \to \min_{\mathbf{b}} \end{aligned} \tag{7.7}$$

where

$$W(\omega_k, \eta, \text{cutoff}) = \left\{ \begin{array}{c} 1 \text{ , if } |\omega_k| < \text{cutoff} \\ (1 + |\omega_k| - \text{cutoff})^\eta \text{ , otherwise} \end{array} \right. \tag{7.8}$$

The optimization principle 7.7 is called a *customized* criterion, see Wildi (2005) and McElroy and Wildi (2015). Obviously, the customized criterion nests the MSE-criterion 4.27 (select $\lambda = \eta = 0$).

7.2 Forecasting and the AT-Dilemma

In the previous section we assumed a particular signal as specified by a lowpass target. Here we analyze (anticipative) *allpass* targets which appear in classical forecast applications, recall section 4.3.2.

7.2.1 ATS-Trilemma Collapses to AT-Dilemma

An application of the MDFA-MSE criterion 4.32 to ordinary one- (and multi-) step head forecasting was proposed in section 4.3.2. Specifically, in the case of the classic one-step ahead criterion, 4.27 becomes

$$\frac{2\pi}{T} \sum_{k=-T/2}^{T/2} \left| \exp(i\omega_k) - \hat{\Gamma}_X(\omega_k) \right|^2 I_{TX}(\omega_k) \to \min_{\mathbf{b}} \tag{7.9}$$

Note that the target filter $\Gamma(\omega_k) := \exp(i\omega_k)$ is an allpass since $|\Gamma(\omega_k)| = 1$ for all frequencies. As a consequence, both the Residual- as well as the Smoothness-term vanish – they are missing – and (the schematic) criterion 7.5 simplifies to

$$\text{MSE-Cust}(\lambda_1) = \text{Accuracy} + (1 + \lambda_1)\text{Timeliness} \to \min_{\mathbf{b}}$$

[4]The 'nuisance' of an undesirable high-frequency component, as measured by its derivative (or its first-order difference) is proportional to ω (or to $|1 - \exp(-i\omega)|$).

The signal-extraction ATS-trilemma collapses to a forecast AT-dilemma. We may infer that the classic time series paradigm (pseudo-maximum-likelihood), which relies on (mean-square one-step ahead) forecast performances is immanently unable to break-up the tie between Timeliness and Smoothness in real-time signal extraction.

7.2.2 Illustration: White Noise and Random-Walk Processes

LDP-Filter and Arithmetic Mean

Let $\Gamma(\omega_k) = \exp(i\omega_k)$ be the one-step ahead target and consider the following two polar filter designs:

1. Let $b_0 = 1, b_j = 0, j = 1, ..., L-1$: the filter assigns full weight $b_0 = 1$ to x_T in the data-sample $x_1, ..., x_T$. Let's call it 'Last-Data-Point' (LDP-) filter. Its transfer function is

$$\hat{\Gamma}(\omega_k) = 1$$

2. Let $b_j = 1/L, j = 0, ..., L-1$: the filter assigns equal weight $1/L$ to $x_T, ..., x_{T-(L-1)}$. We now set $L := T$ so that the equal-weight filter corresponds to the arithmetic mean. The transfer function of the arithmetic mean is

$$\hat{\Gamma}(\omega_k) = \frac{1}{T}\sum_{j=0}^{T-1}\exp(-ij\omega_k)$$

The LDP-filter is an allpass without time-shift; the arithmetic mean is a lowpass with a narrow passband and a large time-shift.

Random-Walk Process

Let $x_1, ..., x_T$ be a realization of a random-walk process. In this case, the LDP-filter is optimal, in a mean-square one-step ahead perspective: the best forecast of x_{T+1} is x_T. Since Smoothness is missing (no stop-band), the (schematic) customized criterion 7.5 simplifies – degenerates – to

$$\begin{aligned}
\text{MSE-Cust}(\lambda_1) &= \text{Accuracy} + (1+\lambda_1)\text{Timeliness} \quad \left(\rightarrow \min_{\mathbf{b}}\right) && (7.10)\\
&= \frac{2\pi}{T}\sum_{\text{All Frequencies}}(A(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k)\\
&\quad +(1+\lambda_1)\frac{2\pi}{T}\sum_{\text{All Frequencies}} 4A(\omega_k)\hat{A}(\omega_k)\sin\left(\frac{\Phi(\omega_k) - \hat{\Phi}(\omega_k)}{2}\right)^2 I_{TX}(\omega_k)\\
&= (1+\lambda_1)\frac{2\pi}{T}\sum_{\text{All Frequencies}} 4\sin\left(\omega_k/2\right)^2 I_{TX}(\omega_k) \quad \left(\rightarrow \min_{\mathbf{b}}\right) && (7.11)
\end{aligned}$$

The last equality follows from $A(\omega_k) = \hat{A}(\omega_k) = 1$ (both the target as well as the LDP-filter are allpass designs), $\Phi(\omega_k) = \text{Arg}(\exp(i\omega_k)) = \omega_k$ (the target filter is anticipative) and $\hat{\Phi}(\omega_k) = 0$. We infer that MSE is entirely attributable to Timeliness (Accuracy vanishes); indeed, the output x_T of $\hat{\Gamma}(\cdot)$ is shifted by one time-unit relative to the target x_{T+1}: no other action or effect is

supported by the LDP-filter. In contrast, the arithmetic mean would strongly smooth the filter output: Accuracy would inflate considerably in 7.10.

The (true) pseudo-spectrum of the process is given by

$$h_X(\omega) = \frac{\sigma^2}{2\pi|1-\exp(-i\omega)|^2} \tag{7.12}$$

where σ^2 is the variance of the (stationary) white noise $\tilde{x}_t = x_t - x_{t-1}$ and $\frac{\sigma^2}{2\pi}$ is its (flat) spectral-density; $\frac{1}{1-\exp(-i\omega)}$ is the transfer function of the (non-stationary) AR(1)-filter linking the noise $\tilde{x}_t$ to the random-walk x_t, see also section 6.2. We could now plug the true pseudo-spectrum into 7.11 and set $\lambda_1 = 0$ to obtain a so-called 'true' MSE

$$\begin{aligned} \text{true MSE} &= \frac{2\pi}{T} \sum_{\text{All Frequencies}} 4\sin\left(\omega_k/2\right)^2 \frac{\sigma^2}{2\pi|1-\exp(-i\omega_k)|^2} \\ &= \frac{2\sigma^2}{T} \sum_{\text{All Frequencies}} \frac{1-\cos(\Phi(\omega_k))}{|1-\exp(-i\omega_k)|^2} \\ &= \frac{2\sigma^2}{T} \sum_{\text{All Frequencies}} \frac{1}{2} \\ &= \sigma^2 \end{aligned} \tag{7.13}$$

where we used $|1-\exp(-i\omega_k)|^2 = (1-\cos(\omega_k))^2 + \sin(\omega_k)^2 = 2 - 2\cos(\omega_k)$[5]. The 'true MSE' is the mean-square one-step ahead forecast error, as expected.

White Noise

Assume x_t is a white noise process and consider, once again, the LDP-filter. From the above we have

$$\text{MSE-Cust}(\lambda_1) = (1+\lambda_1)\frac{2\pi}{T} \sum_{\text{All Frequencies}} 4\sin\left(\omega_k/2\right)^2 I_{TX}(\omega_k) \quad \left(\rightarrow \min_{\mathbf{b}}\right) \tag{7.14}$$

Setting $\lambda_1 = 0$ and plugging the (flat) spectral density $\frac{\sigma^2}{2\pi}$ of the process into this expression we obtain

$$\begin{aligned} \text{true MSE} &= \frac{2\pi}{T} \sum_{\text{All Frequencies}} 4\sin(\omega_k/2)^2 \frac{\sigma^2}{2\pi} \\ &= \frac{\sigma^2}{T} \sum_{\text{All Frequencies}} 4\sin(\omega_k/2)^2 \\ &= \frac{2\sigma^2}{T} \sum_{\text{All Frequencies}} (1-\cos(\omega_k)) \\ &= 2\sigma^2 \end{aligned} \tag{7.15}$$

[5] For notational simplicity we did not discriminate the case of odd and even T: in the latter case additional weights w_k would be necessary, see section 4.2.1.

where we made use of $\sum_{\text{All Frequencies}} \cos(\omega_k) = 0$, see corollary 1 in the Appendix . Indeed, the (true) MSE of the LDP-filter is $Var(x_{T+1} - x_T) = 2\sigma^2$. Comparing 7.15 and 7.13 reveals that high-frequency components contribute disproportionate to Timeliness since $1 - \cos(\omega_k)$ is monotonically increasing. We now substitute the arithmetic mean for the LDP-filter and obtain:

$$\begin{aligned}
\text{true MSE} &= \text{Accuracy+Timeliness} \\
&= \frac{2\pi}{T} \sum_{\text{All Frequencies}} (A(\omega_k) - \hat{A}(\omega_k))^2 \frac{\sigma^2}{2\pi} \\
&\quad + \frac{2\pi}{T} \sum_{\text{All Frequencies}} 4A(\omega_k)\hat{A}(\omega_k) \sin\left(\frac{\Phi(\omega_k) - \hat{\Phi}(\omega_k)}{2}\right)^2 \frac{\sigma^2}{2\pi} \\
&= \frac{2\pi}{T} \sum_{\text{All Frequencies}} \left(1 - \left|\frac{1}{T}\sum_{j=0}^{T-1} \exp(-ij\omega_k)\right|\right)^2 \frac{\sigma^2}{2\pi} \\
&\quad + \frac{2\pi}{T} \sum_{\text{All Frequencies}} 4\left|\frac{1}{T}\sum_{j=0}^{T-1} \exp(-ij\omega_k)\right| \sin\left(\frac{-\omega_k - \frac{T}{2}\omega_k}{2}\right)^2 \frac{\sigma^2}{2\pi}
\end{aligned}$$

where we inserted $\Phi(\omega_k) = -Arg(\exp(i\omega_k)) = -\omega_k$ (anticipative allpass target) and $\hat{\Phi}(\omega_k) = \hat{\phi}(\omega_k)\omega_k = \frac{T}{2}\omega_k$[6]. Using $\frac{1}{T}\sum_{j=0}^{T-1} \exp(-ij\omega_k) = 0, \omega_k \neq 0$, see proposition 5 in the appendix Appendix , and the fact that the sine function vanishes at $\omega_0 = 0$, the expression for the (true) MSE simplifies to

$$\begin{aligned}
\text{MSE} &= \frac{2\pi}{T} \sum_{\text{All Frequencies}} \left(1 - \left|\frac{1}{T}\sum_{j=0}^{T-1} \exp(-ij\omega_k)\right|\right)^2 \frac{\sigma^2}{2\pi} \\
&= \sigma^2 \frac{T-1}{T} \qquad (7.16)
\end{aligned}$$

Timeliness vanishes completely because $\hat{A}(\omega_k) = \left|\frac{1}{T}\sum_{j=0}^{T-1} \exp(-ij\omega_k)\right| = 0$ if $\omega_k \neq 0$ and because $\sin\left(\frac{\omega_0 - \frac{T}{2}\omega_0}{2}\right) = 0$ for $\omega_0 = 0$. Also, the summand of the Accuracy term vanishes in ω_0; therefore Accuracy sums to $\frac{T-1}{T}\sigma^2$. The asymptotically negligible correction $\frac{T-1}{T}$ accounts for the fact that the arithmetic mean is an in-sample estimate i.e. we observe a tiny bit of overfitting here (the fit in frequency zero is 'too good'). We conclude that the (optimal) arithmetic mean trades the entire Timeliness term against a full-blown Accuracy term[7]. The net result is that MSE decreases from $2\sigma^2$, in the case of the LDP filter, to $\approx \sigma^2$ (ignoring the finite sample correction).

The fact that the arithmetic mean, a strong smoothing filter with a large time-shift, reduces Timeliness is not without a touch of irony[8]; even more so when considering that the decrease in Timeliness must overcompensate the inflated Accuracy; and exceedingly so when considering that Smoothness does not even appear in criterion 7.10 since the forecast-target is an *allpass*.

[6] The equally-weighted (arithmetic mean) filter shifts all components by half its filter-length $L/2$ and we selected $L = T$. Therefore $\hat{\phi}(\omega_k) = T/2$.

[7] Figuratively, the arithmetic mean squeezes the data to a flat line which is free of time-shifts.

[8] This is because Timeliness is not a scale-invariant measure i.e. the contribution of the time-shift to MSE depends on the magnitude of the signals. See section 7.5 for a corresponding scale-invariant metric.

Hopefully, the proposed counter-intuitive examples shed some light on the rich(er) structure of the ATS-trilemma or, for that matter, of its degenerate sibling, the forecast AT-dilemma.

7.3 Quadratic Criterion*

7.3.1 I-DFA

The mean-square error criterion 4.27 is a quadratic function of the filter coefficients $\mathbf{b}$ and can be solved in closed-form, see 4.42: the solution is unique and numerical computations are fast. Unfortunately, 7.7 is no longer quadratic in $\mathbf{b}$, if $\lambda > 0$. We here propose an alternative expression which is quadratic irrespective of λ. Interestingly, the new quadratic criterion matches closely 7.7, even for 'large' λ (virtuous feedback loop[9]). Our treatment is general i.e. the target $\Gamma(\cdot)$ does not need to be positive and real. Moreover, the proposed formalism lends itself for a straightforward generalization to the multivariate case, to be developed in chapter 8. Consider

$$\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big|\big|\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big| - \Big\{\Re\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big\}\big)\Big] \\ +i\sqrt{1+\lambda|\Gamma(\omega_k)|}\Im\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big]\Big\}\Big|^2 W(\omega_k,\eta) \to \min_{\mathbf{b}} \tag{7.17}$$

where $\Re(\cdot)$ and $\Im(\cdot)$ denote real and imaginary parts and $i^2 = -1$ is the imaginary unit. We call the resulting approach I-DFA[10]. The above expression is quadratic in the filter coefficients because real and imaginary parts are linear in the coefficients. In analogy to 7.7, the weighting function $W(\omega_k,\eta)$ emphasizes the fit in the stop band. The term $\lambda|\Gamma(\omega_k)|$, under the square-root, emphasizes the imaginary part of the real-time filter in the pass band: for $\lambda > 0$ the imaginary part is artificially inflated and therefore we expect the phase to shrink, see below for details. If $\Gamma(\cdot)$ is a real and positive function (ideal trend, for example), then the quadratic criterion 7.17 simplifies to

$$\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big|\Gamma(\omega_k) - \Big\{\Re\Big(\hat{\Gamma}(\omega_k)\Big) + i\sqrt{1+\lambda\Gamma(\omega_k)}\Im\Big(\hat{\Gamma}(\omega_k)\Big)\Big\}\Big|^2 W(\omega_k,\eta)I_{TX}(\omega_k) \to \min_{\mathbf{b}} \tag{7.18}$$

Expression 7.17 is intentionally more complex, than strictly necessary, because it will allow for a straightforward derivation of the closed-form solution, in the next chapter. The following devel-

[9] Larger λ tend to linearize the problem because $\hat{\Phi}(\omega) - \Phi(\omega)$ will be small, see section 7.3.2.

[10] The capital I in the acronym stands for the imaginary part which is emphasized by λ.

opment allows for a direct comparison of 7.7 and 7.17:

$$\begin{aligned}
&\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big|\big|\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big| - \Big\{\Re\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big\}\big)\Big]\\
&+i\sqrt{1+\lambda|\Gamma(\omega_k)|}\Im\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k))\Xi_{TX}(\omega_k)\big\}\big)\Big]\Big\}\Big|^2 W(\omega_k,\eta) \quad (7.19)\\
=\ &\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big\{\Big(\big|\Gamma(\omega_k)\big| - \Re\Big[\hat{\Gamma}(\omega_k)\exp\big(-i\arg(\Gamma(\omega_k))\big)\Big]\Big)^2\\
&+\Im\Big[\hat{\Gamma}(\omega_k)\exp\big(-i\arg(\Gamma(\omega_k))\big)\Big]^2\Big\}W(\omega_k,\eta)I_{TX}(\omega_k)\\
&+\lambda\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}A(\omega_k)\Im\Big[\hat{\Gamma}(\omega_k)\exp\big(-i\arg(\Gamma(\omega_k))\big)\Big]^2 W(\omega_k,\eta)I_{TX}(\omega_k)\\
=\ &\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big|\Gamma(\omega_k)-\hat{\Gamma}(\omega_k)\Big|^2 W(\omega_k,\eta)I_{TX}(\omega_k)\\
&+\lambda\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}A(\omega_k)\hat{A}(\omega_k)^2\sin\Big(\hat{\Phi}(\omega_k)-\Phi(\omega_k)\Big)^2 W(\omega_k,\eta)I_{TX}(\omega_k)\\
=\ &\begin{cases}
\dfrac{2\pi}{T}\displaystyle\sum_{\text{All Frequencies}}(A(\omega_k)-\hat{A}(\omega_k))^2 W(\omega_k,\eta)I_{TX}(\omega_k)\\
+\dfrac{2\pi}{T}\displaystyle\sum_{\text{Passband}}4A(\omega_k)\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k)\\
+\lambda\dfrac{2\pi}{T}\displaystyle\sum_{\text{Passband}}A(\omega_k)\hat{A}(\omega_k)^2\sin\Big(\hat{\Phi}(\omega_k)-\Phi(\omega_k)\Big)^2 I_{TX}(\omega_k)
\end{cases} \quad (7.20)
\end{aligned}$$

where we assumed that $W(\omega_k,\eta)=1$ for ω_k in the passband. If $\Phi(\omega_k)=0$, then a direct comparison of 7.7 and 7.20 reveals that $\hat{\Phi}(\omega_k)/2$, in the former, is replaced by $\hat{\Phi}(\omega_k)$ in the term weighted by λ in 7.20; also, a supernumerary weighting-term $\hat{A}(\omega_k)$ appears in this expression and the constant scaling-term 4 is missing. Before discussing the quality of the approximation of 7.7 by 7.20 we note that the latter expression is quadratic in the filter coefficients because both sums are quadratic: while this statement is obvious for the first sum, it applies to the second too because $\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)-\Phi(\omega_k))$ is the imaginary part of $\hat{\Gamma}(\omega_k)\exp(-i\Phi(\omega_k))$ which is linear in the filter coefficients. Therefore, minimization of 7.20 can be solved in closed form. For $\lambda=\eta=0$ the original (DFA) mean-square criterion 4.27 is obtained. Overemphasizing the imaginary part of the real-time filter in the pass-band, by $\lambda>0$, achieves a smaller time-shift and increasing η magnifies stopband rejection, as desired. If $\Phi(\omega_k)\neq 0$[11] then a larger λ generates a smaller phase-error i.e. $|\hat{\Phi}(\omega_k)-\Phi(\omega_k)|$ shrinks in the passband, as desired.

Remark:

- Although 7.20 is a simpler, more elegant and intuitively more appealing form of the equivalent criterion 7.17, we prefer the latter because the solution can be derived more easily, in closed-form, see chapter 8. Also, 7.17 matches the implementation in our R-code: the interested reader can track theory in code.

[11] A classic (allpass) one-step ahead target implies $\Phi(\omega_k)=\omega_k$, see section 4.3.2.

7.3.2 Tightness of Approximation

If $\lambda > 0$ then 7.7 is no more quadratic in the filter coefficients. Therefore, we might be tempted to deduce that 7.20 and 7.7 differ in proportion to λ: larger λ should magnify discrepancies between both criteria. Interestingly, a larger λ implies a smaller phase-error which, in turn, 'linearizes' the approximation problem as we now show – virtuous feedback-loop – (for notational convenience we assume that $\Phi(\omega_k) = 0$):

1. For small $\hat{\Phi}(\omega_k)$ the following approximations apply (first order Taylor)

$$4\sin(\hat{\Phi}(\omega_k)/2)^2 \approx 4\hat{\Phi}(\omega_k)^2/4 \approx \sin(\hat{\Phi}(\omega_k))^2 \tag{7.21}$$

   We deduce that the phase terms in 7.7 and 7.20 are interchangeable; also, the additional scaling constant 4, in 7.7, is replicated exactly by 7.20.

2. In general, $\hat{A}(\omega_k)$ is nearly constant in the passband because Accuracy ensures that $\hat{A}(\omega_k) \approx |\Gamma(\omega_k)| = 1$. Obviously, in such a case, the supernumerary amplitude term in 7.20 can be ignored.

3. Sometimes, imposing a strong customization weakens Accuracy, see for example section 7.4.3. If $\hat{A}(\omega_k) \not\approx 1$, then the effect of the supernumerary amplitude term in 7.20 cannot be ignored, anymore. However, in such a case the distortion induced by the additional amplitude term in 7.20 could be compensated by a simple re-adjusting or re-scaling of λ, as discussed in section 7.4.3 below.

We conclude that the quadratic customized criterion 7.20 is identical to 7.7, when $\lambda = 0$; if $\lambda > 0$ is small, then both criteria are nearly identical; if $\lambda > 0$ is large, then $\hat{\Phi}(\omega_k) - \Phi(\omega_k)$ tends to be small (because λ emphasizes the time-shift) and therefore the approximation problem is linearized, see 7.21.

7.3.3 R-code

The R-function *dfa_analytic*() proposed in DFA, section 4.3.5, implements a closed-form solution of criterion 7.17.

```
> head(dfa_analytic)

1 function (L, lambda, periodogram, Lag, Gamma, eta, cutoff, i1,
2     i2)
3 {
4     periodogram[1] <- periodogram[1]/2
5     lambda <- abs(lambda)
6     eta <- abs(eta)
```

The entries L, $weight\_func$, Lag, $Gamma$, $i1$ and $i2$ retain their original meanings, as discussed in previous chapters. The new customization parameters λ, η and $cutoff$ refer to the weighting functions W_1 and W_2 defined in 7.6 (in general cutoff coincides with the specification of the target $Gamma$). The function returns coefficients as well as transfer function of the filter:

```
> tail(dfa_analytic)
```

```
55     for (k in 1:(K)) {
56         trffkt[k + 1] <- (b %*% exp((0+1i) * k * (0:(length(b) -
57             1)) * pi/(K)))
58     }
59     return(list(b = b, trffkt = trffkt))
60 }
```

Remark

- For historical reasons the level-constraint (i1=T) was not implemented in closed (exact) form6[12]. We therefore recommend usage of the generic MDFA-code *mdfa_analytic* instead. The latter relies on exact formulas, as derived in chapter 6.

7.4 ATS-Components: a Worked-Out Example

We here rely on the simulation framework of section 4.2.5, taken over from McElroy and Wildi (2015). Specifically, we consider the three AR(1)-processes

$$\left.\begin{array}{rcl} x_t &=& 0.9x_{t-1} + \epsilon_t \\ x_t &=& 0.1x_{t-1} + \epsilon_t \\ x_t &=& -0.9x_{t-1} + \epsilon_t \end{array}\right\} \tag{7.22}$$

and we generate a single realization of length $T = 120$ (10 years of monthly data) for each process. Our target is an ideal trend with cutoff $\pi/12$ and we rely on real-time DFA-filters of length $L = 24$[13]. We then assess customization effects obtained by λ, η in (the quadratic) criterion 7.17 or, equivalently, 7.20. For this purpose we compute and report ATS-components, total MSE, amplitude and time-shift functions, filter coefficients as well as filter outputs. In order to save space, we report results for the third process, $a_1 = 0.9$, only (the other results are reported in the appendix). The estimation routine is based on *dfa_analytic* as introduced in DFA and briefly presented in chapter 1. In-sample and out-of-sample distributions of performances of competing designs, based on multiple realizations of the above processes, will be analyzed in sections 7.6 and 7.7, further below.

7.4.1 Timeliness Only: $\lambda \geq 0$, $\eta = 0$ Fixed

1. We source the relevant R-file

   ```
   > source(file=paste(path.pgm,"functions_trilemma.r",sep=""))
   ```

[12] In order to improve the fit of the transfer functions in frequency zero ($\hat{\Gamma}(0) \equiv \Gamma(0)$) a large artificial value is assigned to $I_{TX}(0)$.

[13] The target signal eliminates components whose duration is shorter than $2\pi/(\pi/12) = 24$ (two years of monthly data). This design is inspired from real-time 'business-cycle' applications. The chosen filter-length $L = 24$ reflects the maximal duration of components in the stopband of the target filter (two years).

The functions in this file generate the data, estimate filter coefficients, compute ATS-components as well as alternative performance measures (Peak Correlation and Curvature, see section 7.5 below) and amplitude and time-shift-functions.

2. We specify the empirical design and run the code. Specifically, we select $\lambda = 0$ (MSE) and $\lambda = 2^k, k = 0, ..., 7$ (emphasize Timeliness) and we fix $\eta = 0$ (no emphasis of Smoothness).

```
> #rm(list=ls())
> # Specify the processes: ar(1) with coefficients -0.9,0.1 and 0.9
> a_vec<-c(0.9,0.1,-0.9)
> # Specify the lambdas
> lambda_vec<-c(0,2^(0:7))
> # Specify the fixed eta
> eta_vec<-rep(0,length(lambda_vec))
> # Specify filter length
> L<-24
> # Length of estimation sample
> len<-120
> # cutoff
> cutoff<-pi/12
> # Nowcast
> Lag<-0
> # No filter constraints
> i1<-i2<-F

> # Proceed to estimation
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,
+               Lag,i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

3. ATS-components and MSE for the first process ($a_1 = 0.9$) are summarized in table 7.1. Let us briefly explain how the numbers in the above table were obtained (the same proceeding

	Accuracy	Timeliness	Smoothness	Residual	Total MSE
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639
Lambda=1, eta=0	0.005776	0.008884	0.030273	0.000000	0.044933
Lambda=2, eta=0	0.007474	0.005849	0.035979	0.000000	0.049301
Lambda=4, eta=0	0.009792	0.003298	0.043450	0.000000	0.056540
Lambda=8, eta=0	0.012483	0.001653	0.051885	0.000000	0.066022
Lambda=16, eta=0	0.015247	0.000755	0.060728	0.000000	0.076729
Lambda=32, eta=0	0.017804	0.000306	0.069476	0.000000	0.087586
Lambda=64, eta=0	0.019880	0.000109	0.077027	0.000000	0.097016
Lambda=128, eta=0	0.021362	0.000036	0.082459	0.000000	0.103856

Table 7.1: ATS-Components as a function of lambda (eta=0 fixed)

applies to all subsequent examples):

- For each combination of λ, η we obtain a corresponding filter from criterion 7.17.
- ATS-components are then obtained by plugging the resulting amplitude and time-shift functions into 7.4.
- MSE is obtained as the sum of these ATS-components. It is an estimate of the true (unknown) mean-square filter error, recall section 4.2.5.
- This MSE-number does not correspond to the criterion value 7.17 because the latter is 'distorted' by the customization weights $\lambda, \eta \neq 0$.

Table 7.1 as well as the following graphs will be analyzed all at once, at the end of the exercise.

4. Amplitude and time-shift functions for the first process are plotted in fig.7.1.

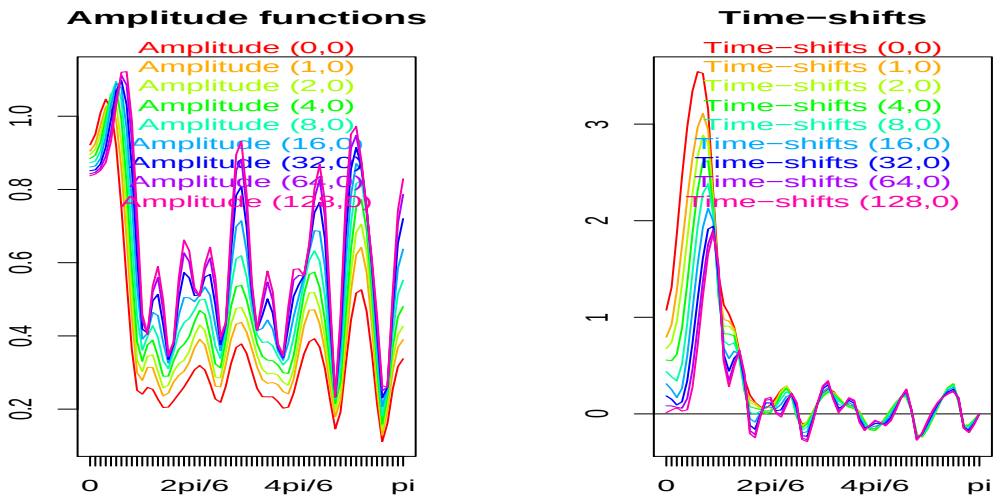


Figure 7.1: Amplitude (left) and time-shift functions (right) as a function of lambda for fixed eta=0

5. Filter-outputs are plotted in fig.7.2: series are standardized in order to facilitate visual inspection.

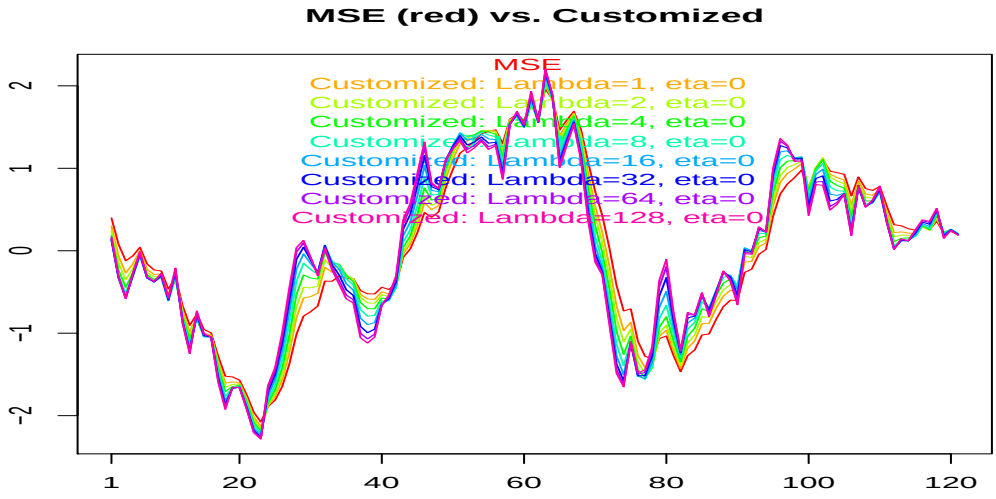


Figure 7.2: Filter outputs: MSE (red) vs. customized , a1=0.9

Analysis

- Accuracy, Timeliness and Smoothness sum-up to total MSE in table 7.1. Residual vanishes because the target filter vanishes in the stopband.

- Timeliness decreases with increasing λ, as desired. Accuracy and Smoothness as well as total MSE increase, as a consequence.

- Amplitude and time-shift functions in fig.7.1 provide more insights. Timeliness decreases because the time-shift is uniformly decreasing in the passband, as a function of λ. Accuracy increases with increasing λ because the amplitude function departs from the target ($\Gamma(\omega)$ in the stopband: leakage is increasing with λ. As a consequence, Smoothness deteriorates.

- Outputs of the customized filter in fig.7.2 tend to lie to the left (they are 'faster') and the series are becoming increasingly noisy, as λ increases.

7.4.2 Smoothness Only: $\eta \geq 0$, $\lambda = 0$ Fixed

1. We fix $\lambda = 0$ (no emphasis of Timeliness) and let $\eta = k * 0.3, k = 0, ..., 6$ (Smoothness is emphasized).

```
> # Specify the etas
> eta_vec<-0.3*0:6
> # Specify the fixed lambda
> lambda_vec<-rep(0,length(eta_vec))
```

```
> # Proceed to estimation
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,
+                Lag,i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

2. ATS-components and MSE for the first process ($a_1 = 0.9$) are summarized in table 7.2.

	Accuracy	Timeliness	Smoothness	Residual	Total MSE
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639
Lambda=0, eta=0.3	0.005263	0.023179	0.016156	0.000000	0.044597
Lambda=0, eta=0.6	0.007721	0.031747	0.012262	0.000000	0.051730
Lambda=0, eta=0.9	0.011083	0.041832	0.009555	0.000000	0.062470
Lambda=0, eta=1.2	0.015746	0.053695	0.007473	0.000000	0.076914
Lambda=0, eta=1.5	0.022155	0.067872	0.005682	0.000000	0.095709
Lambda=0, eta=1.8	0.030550	0.084845	0.004081	0.000000	0.119477

Table 7.2: ATS-Components as a function of eta (lambda=0 fixed)

3. Amplitude and time-shift functions for the first process are plotted in fig.7.3.

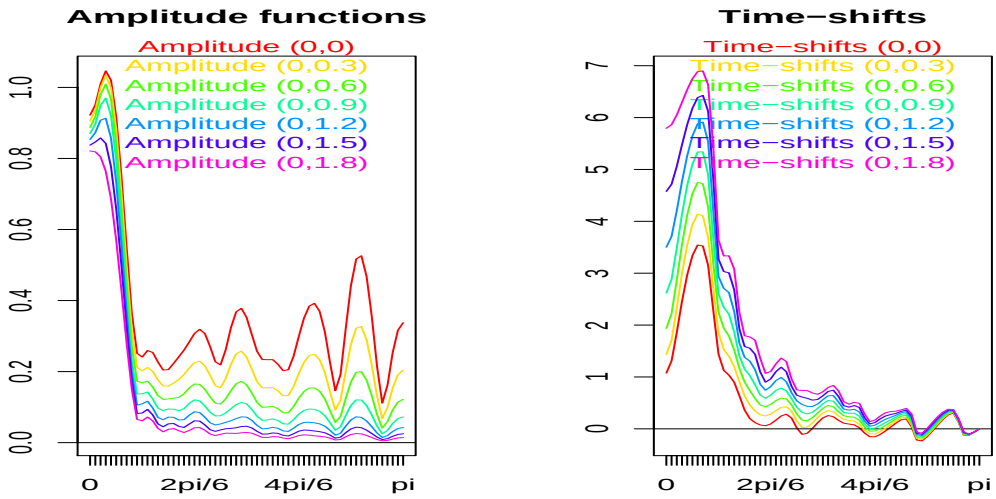


Figure 7.3: Amplitude (left) and time-shift functions (right) as a function of eta (lambda=0 fixed)

4. Filter coefficients can be seen in fig.7.4.

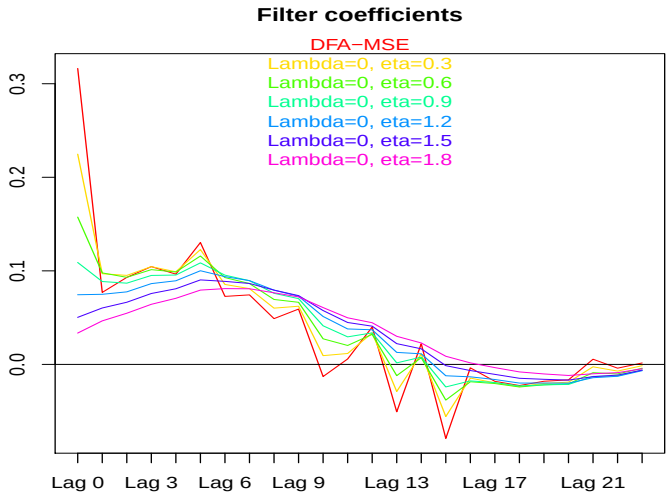


Figure 7.4: Filter coefficients as a function of eta (lambda=0 fixed)

5. Filter-outputs are plotted in fig.7.5: the series are standardized for ease of visual inspection.

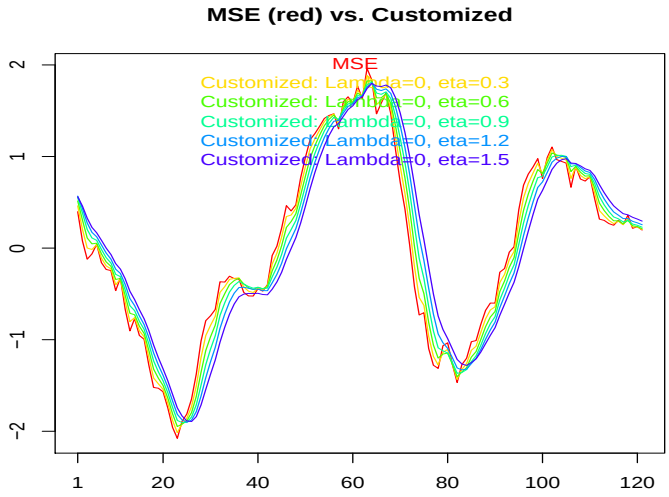


Figure 7.5: Filter outputs MSE (red) vs. customized, a1=0.9

Analysis

- Smoothness in table 7.2 decreases with increasing η, as required. Accuracy, Timeliness as well as total MSE increase, as a consequence.
- Amplitude and time-shift functions in fig.7.3 offer a more detailed picture: for increasing

η the amplitude functions are approaching zero in the stopband at costs of the time-shifts which grow substantially in the passband.

- Fig.7.4 reveals that filter coefficients become smoother for increasing η. This desirable side-effect is obtained by imposing stronger shrinkage of the amplitude function in the (wide) stopband: degrees of freedom are implicitly freezed; in particular, we expect overfitting to be contained.

- As η increases, outputs of the customized filter in fig.7.5 tend to lie to the right (delay) and the series appear to be increasingly smooth.

7.4.3 Emphasizing Timeliness and Smoothness: $\lambda, \eta \geq 0$

1. We compare the MSE-design $\lambda = \eta = 0$ with the strongly customized filter $\lambda = 128, \eta = 1.8$ which emphasizes Timeliness and Smoothness simultaneously.

```
> # Specify the etas
> eta_vec<-c(0,1.8)
> # Specify the fixed lambda
> lambda_vec<-c(0,128)
```

```
> # Proceed to estimation
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,
+                Lag,i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

2. ATS-components and MSE for the first process ($a_1 = 0.9$) are summarized in table 7.3.

	Accuracy	Timeliness	Smoothness	Residual	Total MSE
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639
Lambda=128, eta=1.8	0.595139	0.000650	0.002427	0.000000	0.598216

Table 7.3: ATS-Components as a function of lambda and eta, a1=0.9

3. Amplitude and time-shift functions for the first process are plotted in fig.7.6.

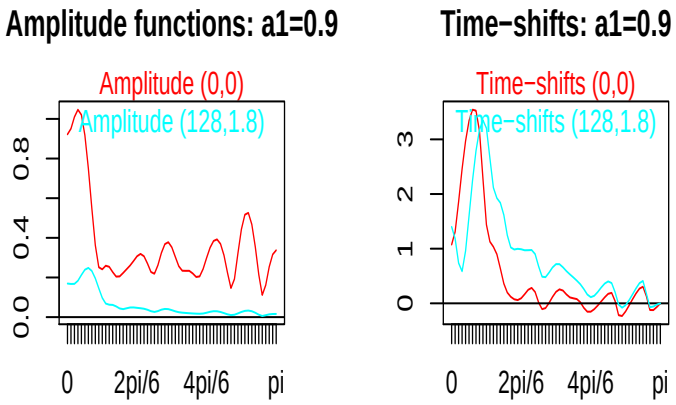


Figure 7.6: Amplitude (left) and time-shift functions (right) as a function of lambda and eta

4. Filter-outputs of MSE and customized filters are plotted in fig.7.7 (series are standardized for ease of visual inspection).

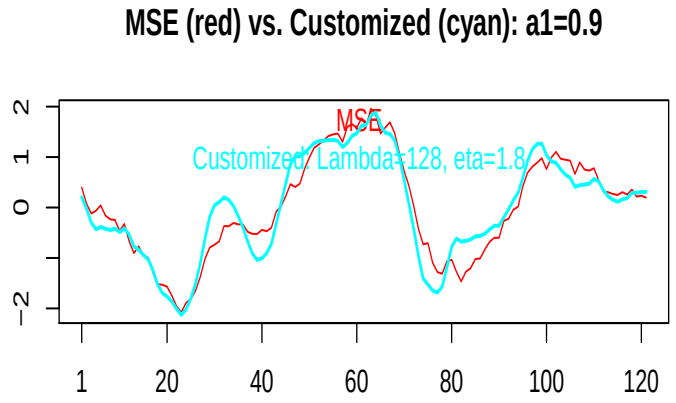


Figure 7.7: Filter outputs MSE (red) vs. customized (cyan), a1=0.9

Analysis

- As confirmed in table 7.3, the customized design improves in terms of Timeliness as well as of Smoothness, simultaneously at costs of a remarkable deterioration of Accuracy and, as a

a consequence, of total MSE.

- The time-shift of the customized design in fig.7.6 is generally smaller in the passband (except at frequency zero). Its amplitude function is close to zero in the passband, as desired; but the amplitude of the customized filter is also pulled towards zero in the passband (zero-shrinkage of the coefficients) which explains the excess Accuracy- and MSE-losses.

- The output of the customized filter in fig.7.7 tends to lie to the left (faster) and it is smoother, as desired (recall that the series are standardized).

In order to grasp the observed zero-shrinkage of the customized filter we observe that the scale-dependent Timeliness and Smoothness measures vanish if the amplitude function shrinks to zero. Accuracy alone lifts the amplitude away from zero (at least in the passband). Therefore, emphasizing heavily Smoothness and Timeliness, as we did in our example, exhausts to some extent Accuracy's action[14]. Fortunately, this problem could be addressed by a simple re-scaling of the (customized) filter output[15], as we briefly explore below.

```
> # We allow for a re-calibration (by the inverse amplitude function in the passband)
> scaled_ATS<-T
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,Lag,i1,
+                          i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

Specifically, we scale the filter coefficients by a constant which is inversely proportional to the amplitude function in the passband[16]

$$\text{Calibration term} := \frac{\text{Length of passband}}{\sum_{\text{Passband}} \hat{A}(\omega_k)}$$

Fig. 7.8 and table 7.4 confirm that the previous zero-shrinkage is withdrawn.

[14] The supernumerary amplitude-term in criterion 7.20 is counterproductive too, see section 7.3.
[15] Recall that the series in fig.7.7 were standardized.
[16] Alternatively one could fit an optimal MSE-normalization.

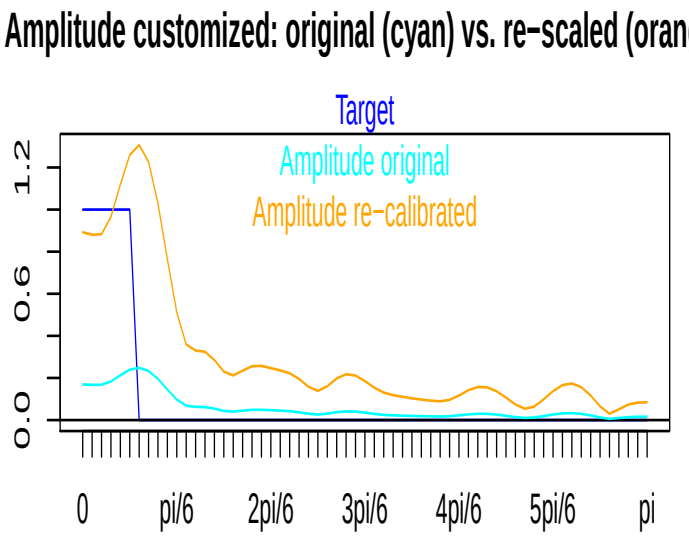


Figure 7.8: Amplitude functions of original (cyan) and scaled customized filter (orange)

In particular, Accuracy and MSE are relaxed, this time at costs of Smoothness and Timeliness. We retain from the above examples that both the time-shift (delay) as well as the noise-suppression

	Accuracy	Timeliness	Smoothness	Residual	Total MSE
Customized unscaled	0.595139	0.000650	0.002427	0.000000	0.598216
Customized re-scaled	0.010997	0.003422	0.067254	0.000000	0.081673

Table 7.4: ATS-Components of customized design: unscaled vs. scaled filter, a1=0.9

of a (real-time) filter can be addressed by a decomposition of the MSE-norm into ATS-components. However, The fact that Timeliness and Smoothness are scale-dependent measures hampers interpretability. Therefore we now propose an alternative set of scale-invariant measures which will allow for a straightforward evaluation of customized designs.

7.5 Curvature and Peak-Correlation

7.5.1 Definition

Define

$$\text{Curvature} := \frac{E\left[\left((1-B)^2\widehat{y}_t\right)^2\right]}{\text{var}(\widehat{y}_t)} \tag{7.23}$$

$$\text{Peak-Correlation} := \text{Arg}\left(\max_j(cor(y_t, \widehat{y}_{t+j}))\right) \tag{7.24}$$

where $(1-B)^2$ is the second-order difference operator (curvature) and where $\text{Arg}\left(\max_j(cor(y_t, \widehat{y}_{t+j}))\right)$ means the lead or lag j_0 at which the correlation between the target y_t and the estimate $\hat{y}_{t+j_0}$ is maximized[17]. Note also that the covariance could be substituted to the correlation without altering the corresponding outcome. Curvature and Peak Correlation are scale-invariant measures. As we shall show, they are linked to Smoothness and Timeliness. Empirical measures can be obtained by substituting sample-estimates for unknown expectations in the above expressions.

7.5.2 Examples

We apply the above statistics to our previous example and complete the former tables 7.1 (emphasizing Timeliness only), 7.2 (emphasizing Smoothness only) and 7.3 (emphasizing Timeliness and Smoothness).

	Accuracy	Timeliness	Smoothness	Residual	Total MSE	Curv-in	Peak-Cor-in
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639	0.053264	2.000000
Lambda=1, eta=0	0.005776	0.008884	0.030273	0.000000	0.044933	0.076929	2.000000
Lambda=2, eta=0	0.007474	0.005849	0.035979	0.000000	0.049301	0.093906	1.000000
Lambda=4, eta=0	0.009792	0.003298	0.043450	0.000000	0.056540	0.117728	1.000000
Lambda=8, eta=0	0.012483	0.001653	0.051885	0.000000	0.066022	0.147909	0.000000
Lambda=16, eta=0	0.015247	0.000755	0.060728	0.000000	0.076729	0.183068	0.000000
Lambda=32, eta=0	0.017804	0.000306	0.069476	0.000000	0.087586	0.219062	0.000000
Lambda=64, eta=0	0.019880	0.000109	0.077027	0.000000	0.097016	0.249550	0.000000
Lambda=128, eta=0	0.021362	0.000036	0.082459	0.000000	0.103856	0.270883	0.000000

Table 7.5: ATS-Components, Peak-Correlation and Curvature: emphasizing Timeliness only, a1=0.9

- In table 7.5 Timeliness as well as Peak-Correlation decrease with increasing λ, as desired. The Peak Correlation is readily interpretable: in particular target series and real-time estimate appear to be synchronized for $\lambda \geq 8$ (Peak Correlation vanishes).

[17] In applications, y_t is generally unobserved and must be approximated by symmetric filters of finite order. Fortunately, the Peak Correlation concept is fairly robust against such approximations because the effective value of the correlation is irrelevant since we are solely interested in the lead/lag at which the correlation peaks.

	Accuracy	Timeliness	Smoothness	Residual	Total MSE	Curv-in	Peak-Cor-in
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639	0.053264	2.000000
Lambda=0, eta=0.3	0.005263	0.023179	0.016156	0.000000	0.044597	0.023755	3.000000
Lambda=0, eta=0.6	0.007721	0.031747	0.012262	0.000000	0.051730	0.010688	3.000000
Lambda=0, eta=0.9	0.011083	0.041832	0.009555	0.000000	0.062470	0.004966	4.000000
Lambda=0, eta=1.2	0.015746	0.053695	0.007473	0.000000	0.076914	0.002446	5.000000
Lambda=0, eta=1.5	0.022155	0.067872	0.005682	0.000000	0.095709	0.001301	5.000000
Lambda=0, eta=1.8	0.030550	0.084845	0.004081	0.000000	0.119477	0.000748	6.000000

Table 7.6: ATS-Components, Peak-Correlation and Curvature: emphasizing Smoothness only, a1=0.9

- In table 7.6 Smoothness and Curvature decline with increasing η, as desired. The latter is readily interpretable in terms of normalized (inverse) signal-to-noise ratio. As a trade off, the Peak Correlation increases substantially.

	Accuracy	Timeliness	Smoothness	Residual	Total MSE	Curv-in	Peak-Cor-in
DFA-MSE	0.003458	0.016112	0.022069	0.000000	0.041639	0.053264	2.000000
Lambda=128, eta=1.8	0.595139	0.000650	0.002427	0.000000	0.598216	0.016443	0.000000
Scaled customized	0.010997	0.003422	0.067254	0.000000	0.081673	0.016443	0.000000

Table 7.7: ATS-Components, Peak-Correlation and Curvature: emphasizing Timeliness and Smoothness, a1=0.9

- In table 7.3 the double-score of the customized design is evidenced by Peak Correlation and Curvature which remain unaffected by the normalization (last row), in contrast to Timeliness and Smoothness.

We deduce from the above examples that the new Peak Correlation and Curvature statistics are simple to implement, simple to formalize and simple to interpret.

7.5.3 Linking ATS-Components, Peak Correlation and Curvature

Consider the second-order differences in the numerator of the Curvature statistic 7.23

$$\begin{aligned} E\left[\left((1-B)^2\widehat{y}_t\right)^2\right] &\approx \frac{1}{T}\sum_{t=1}^{T}\left\{(1-B)^2\widehat{y}_t\right\}^2 \\ &= \frac{2\pi}{T}\sum_{\text{All frequencies}} I_{T\Delta^2\hat{Y}}(\omega_k) \\ &\approx \frac{2\pi}{T}\sum_{\text{All frequencies}} |1-\exp(-i\omega_k)|^4 I_{T\hat{Y}} \\ &\approx \frac{2\pi}{T}\sum_{\text{All frequencies}} |1-\exp(-i\omega_k)|^4 \hat{A}^2(\omega_k) I_{TX} \end{aligned}$$

where we assumed that $\hat{y}_{-1}, \hat{y}_0$ were available and where $I_{T\Delta^2\hat{Y}}(\omega_k)$ is the periodogram of $(1-B)^2\widehat{y}_t$. The first equality follows from 4.7 and the subsequent approximations follow from 4.21. We can compare this expression to Smoothness

$$S = \frac{2\pi}{T} \sum_{\text{Stopband}} \hat{A}(\omega_k)^2 W(\omega_k, \eta) I_{TX}(\omega_k)$$

which is defined in the stopband, only. Assimilating $|1-\exp(-i\omega_k)|^4$ with $W(\omega_k, \eta)$ and ignoring the passband establishes a link between Curvature and Smoothness[18].

In order to relate Peak Correlation to Timeliness we consider the following development[19]

$$\begin{aligned}
cov(y_t, \widehat{y}_{t+j}) &\approx \frac{1}{T}\sum_{t=1}^{T} y_t \hat{y}_{t+j} \\
&\approx \frac{2\pi}{T} \sum_{\text{All frequencies}} \Re\left(\Xi_{TY}(\omega_k)\overline{\exp(ij\omega_k)\Xi_{T\hat{Y}}(\omega_k)}\right) \\
&\approx \frac{2\pi}{T} \sum_{\text{All frequencies}} \Re\left(\Gamma(\omega_k)\Xi_{TX}(\omega_k)\overline{\exp(ij\omega_k)\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)}\right) \\
&= \frac{2\pi}{T} \sum_{\text{Passband}} A(\omega_k)\hat{A}(\omega_k)\Re\left(\overline{\exp(ij\omega_k - i\hat{\Phi}(\omega_k))}\right) I_{TX}(\omega_k) \\
&= \frac{2\pi}{T} \sum_{\text{Passband}} A(\omega_k)\hat{A}(\omega_k)\cos(j\omega_k - \hat{\Phi}(\omega_k)) I_{TX}(\omega_k)
\end{aligned}$$

The second approximation is a consequence of proposition 6 in Appendix . Let's assume, for a while, that j is not a fixed integer but a general real-valued function of ω_k. Then the last term in the above development would be maximized, as a function of $j(\omega_k)$, if $j(\omega_k) = \hat{\Phi}(\omega_k)/\omega_k = \hat{\phi}(\omega_k)$ (up to multiples of 2π). Therefore, reducing the time-shift $\hat{\phi}(\omega_k)$ in the passband, by emphasizing Timeliness, will decrease the lag at which the covariance or the correlation are peaking, which formalizes the link between Peak Correlation and Timeliness.

7.6 Double Score Against the MSE Paradigm

7.6.1 Introduction

We here illustrate the flexibility of the ATS-trilemma by benchmarking empirical real-time designs (nowcasts), optimized in view of particular research priorities (speed/reliability), against the best theoretical *MSE*-filter, assuming knowledge of the *true DGP*. The empirical framework leans on the previous sections, see McElroy and Wildi (2015), and the whole R-code is carried over. In contrast to the previous sections, where we emphasized *single* realization *in-sample* results, we now consider *multiple* realizations and compute in-sample as well as *out-of-sample* distributions of performance measures of competing designs. Specifically, we analyze performances based on

[18] Note also that we ignored the denominator $\text{var}(\widehat{y}_t)$ in the Curvature expression: the latter ensures scale-invariance of the measure.

[19] Recall that the covariance could be substituted to the correlation without affecting numbers i.e. Peak Correlation and Peak Covariance are identical statistics.

the classic (total) MSE-norm as well as on the scale-invariant Peak Correlation and Curvature measures introduced in the previous section.

Our contenders in this study will be the best theoretical MSE-estimate based on knowledge of the true DGP (benchmark), as well as the following empirical designs

- DFA-MSE: $\lambda = \eta = 0$
- Strong noise suppression: $\lambda = 0, \eta = 1.5$
- Balanced (fast and smooth): $\lambda = 30, \eta = 1$
- Very fast: $\lambda = 500, \eta = 0.3$

7.6.2 Simulation Run

Coefficients of the benchmark MSE-filters can be obtained by substituting the true spectral density

$$\left| \frac{\sigma^2}{1 - a_1 \exp(-i\omega)} \right|^2 \ , \ a_1 = -0.9, 0.1, 0.9$$

for the periodogram $I_{TX}(\omega)$ in 7.7, setting $\lambda = \eta = 0$, see chapter 9 for further background[20]. Note that benchmark filters do not depend on data, at all. Our target filter is the ideal trend with cutoff $\pi/12$. For the empirical DFA-filters we generate 100 realizations for each process and compute real-time (nowcast) filters of length $L = 24$. The code leans on McElroy and Wildi (2015)[21]:

1. Source the R-file

   ```
   > source(file=paste(path.pgm,"functions_trilemma.r",sep=""))
   ```

   The functions in this file generate the data, estimate filter coefficients, compute performance measures (in- and out-of-sample), ATS-components as well as amplitude and time-shift-functions.

2. Specify the empirical design (same as in the previous section except for the number of realizations) and run the code.

   ```
   > # Number of realizations
   > anzsim<-100

   > # Proceed to simulation run
   > for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,Lag,
   +                          i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
   ```

[20] We implement an alternative time-domain solution in our R-code, see chapter 9.

[21] It is a faster-running 'debugged' version based on reconciliation with a multivariate extension proposed in the next section.

7.6.3 Analysis

Performance Measures

In order to save space we emphasize the second process $a_1 = 0.1$[22]. Fig.7.9 shows box-plots of in-sample (left) and out-of-sample (right) Curvature and Peak Correlation performances and fig.7.10 shows the corresponding sample MSEs: the latter are *effective* time-domain measures

$$\frac{1}{120}\sum_{t=1}^{120}(y_t - \hat{y}_t)^2$$

where the target signal y_t is obtained by applying a high-order (finite) symmetric trend filter to (very) long time series. Plots of the other two processes are shown in section A.3 in the Appendix Appendix .

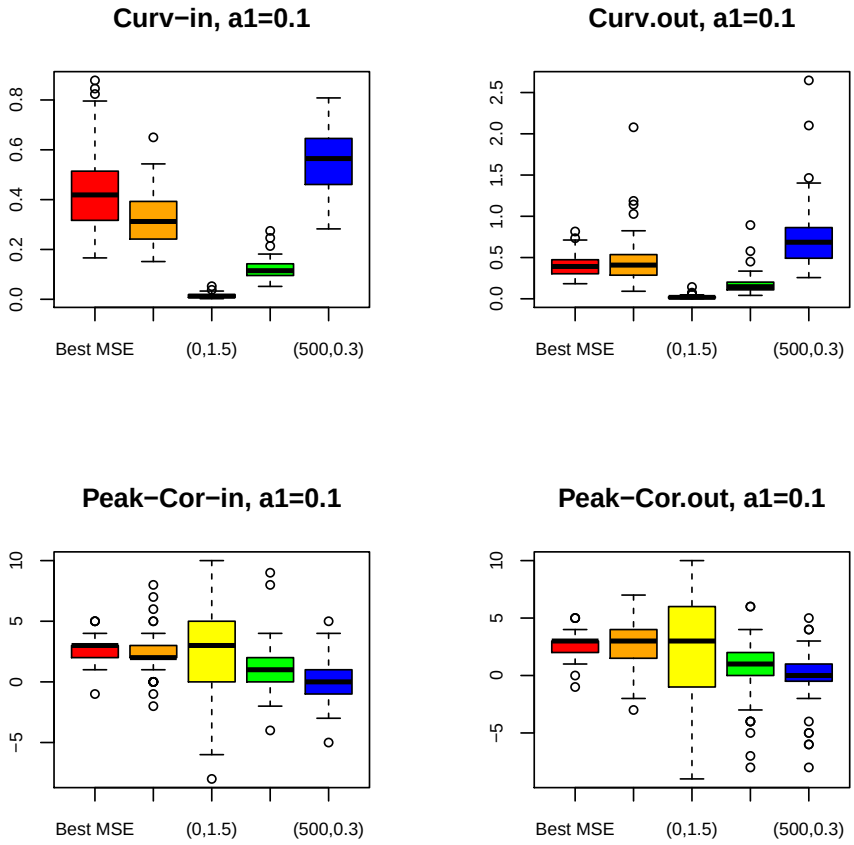


Figure 7.9: Empirical distributions of Curvature and Peak-Correlation of best theoretical MSE (red), empirical MSE (orange), strong noise suppression (yellow), balanced fast and smooth (green) and very fast (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=0.1

[22]Recall that this model fits log-returns of INDPRO quite well over a longer historical time span.

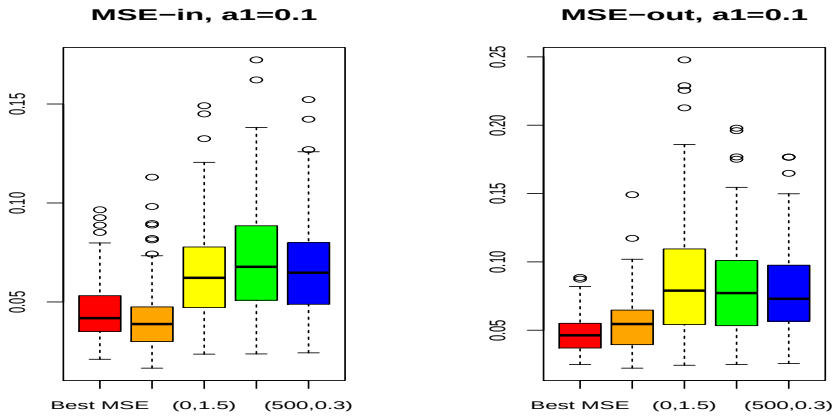


Figure 7.10: Empirical distributions of Sample MSEs of best theoretical MSE (red), empirical MSE (orange), strong noise suppression (yellow), balanced fast and speed (green) and very fast (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=0.1

Findings

- The customized designs perform as expected. In particular the balanced design (green) outperforms the benchmark MSE-filter both in terms of Curvature as well as Peak-Correlation, in- and out-of-sample.
- In- and out-of-sample performances are congruent[23].
- A comparison of in- and out-of-sample MSE-performances of benchmark (red) and empirical (orange) filters is indicative for overfitting: the orange filter performs slightly better in-sample but it is slightly outperformed out-of-sample. Interestingly, overfitting is moderate[24] even in the case of richly parametrized designs (L =24 coefficients are estimated in samples of length T =120).
- The customized designs are systematically outperformed in terms of MSE-performances in- and out-of-sample, as expected.

Real-Time Filter Outputs

Benchmark-MSE (red) and customized-balanced (green) filter-outputs are compared in fig.7.11 (both time series are standardized for ease of visual inspection). The chosen time-span covers in-sample as well as out-of-sample periods.

[23]The distributions of the integer-valued Peak Correlations are almost identical due, in part, to discretization effects.

[24]In contrast to classic forecast approaches, which target an allpass filter, our examples target a lowpass filter with a wide stop-band. A close fit of the target in the stopband mimics, in some ways, classic shrinkage approaches or, stated otherwise, degrees of freedom are implicitly controlled by the DFA/MDFA-criteria.

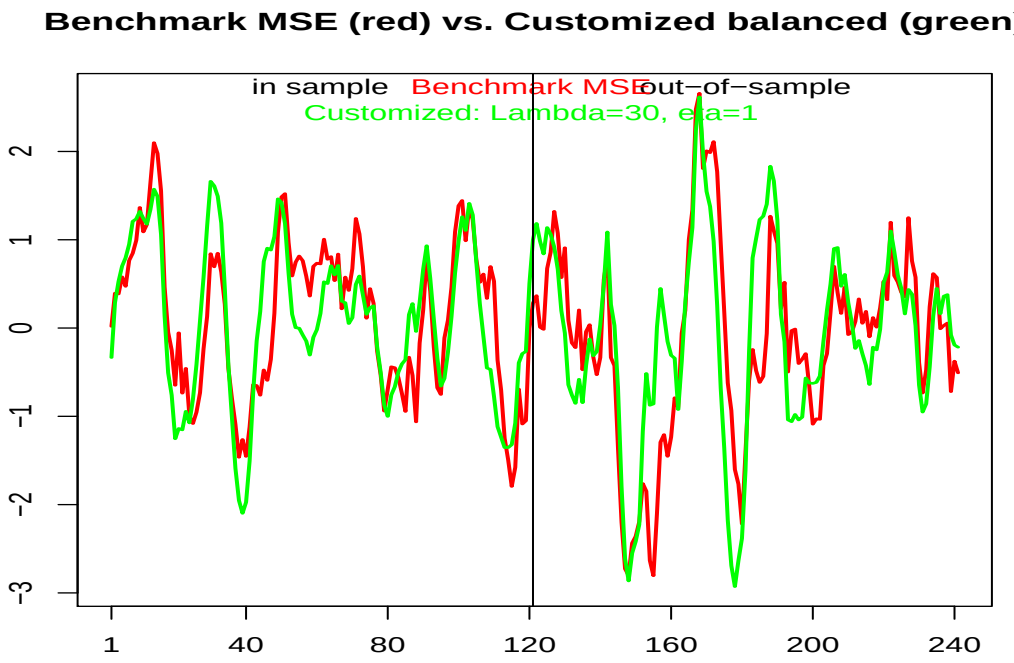


Figure 7.11: Outputs of benchmark MSE (red) and customized balanced (green) filters: a1=0.1. In-sample (left half) and out-of-sample (right half)

The customized filter outperforms the benchmark-MSE design on both accounts: it lies on the left (faster) and it is smoother both in-sample as well as out-of-sample.

7.7 Univariate Customized Design vs. Bivariate MSE Leading-Indicator

In this section we stiffen the competition by benchmarking the univariate customized filter against a bivariate MSE-design with a leading indicator. The empirical design is otherwise identical to the previous section.

Data and Contenders

We emphasize performances in the case of the second process ($a_1 = 0.1$) and compare the univariate MSE-DFA $\lambda = \eta = 0$ (orange), the univariate balanced customized DFA $\lambda = 30$,$\eta = 1$ (green)) and the bivariate MDFA-MSE design proposed in section 4.7.1.

```
> # Second process
> a1<-0.1
> # Customization settings DFA
```

```
> lambda_vec<-c(0,30)
> eta_vec<-c(0,1)

> # Run the competition: the new function handles the multivariate case
> cust_leading_obj<-mdfa_mse_leading_indicator_vs_dfa_customized(anzsim,
+                    a1,cutoff,L,lambda_vec,eta_vec,len1,len,i1,i2,Lag,
+                    lambda_mdfa,eta_mdfa,troikaner)
```

7.7.1 Performances: Curvature

Box-plots of in-sample and out-of-sample Curvature-scores are depicted in fig.7.12.

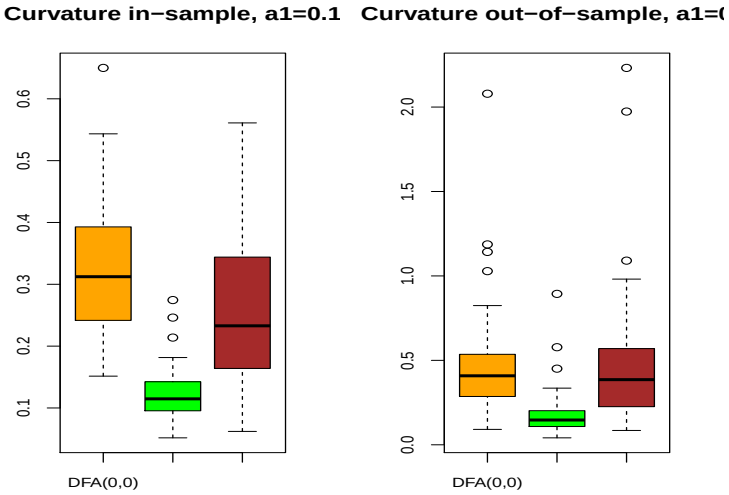


Figure 7.12: Curvature: mean-square DFA (orange), customized DFA (green) and MSE-MDFA leading indicator (brown): a1=0.1 In-sample (left) and out-of-sample (right).

The empirical distributions of the univariate designs coincide with those reported in the previous section (same random-seed). The customized design (green) outperforms both contenders in-sample as well as out-of-sample. In- and out-of-sample figures are remarkably similar considering that the bivariate design (brown) relies on 48 estimated coefficients, in realizations of length T =120, only.

7.7.2 Performances: Peak-Correlation

Box-plots of in-sample and out-of-sample Peak-Correlation scores are shown in fig.7.13.

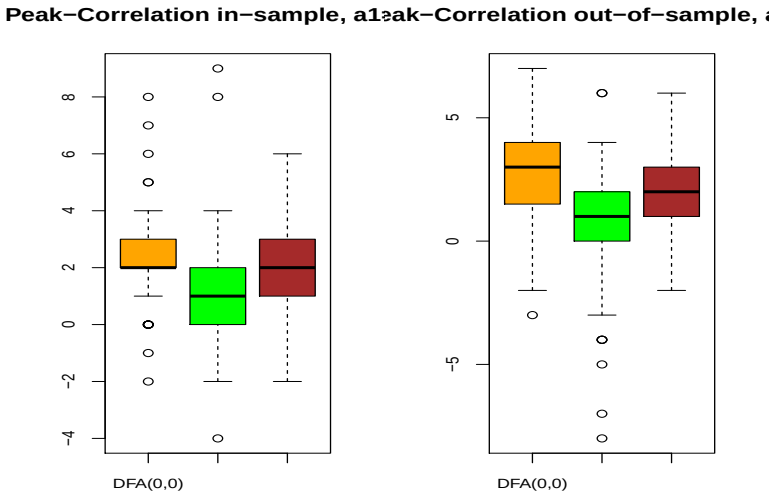


Figure 7.13: Peak Correlation: mean-square DFA (orange), customized DFA (green) and MSE-MDFA leading indicator (brown): a1=0.1 In-sample (left) and out-of-sample (right).

The customized design (green) substantially outperforms both contenders: it anticipates the DFA-MSE (orange) by -1 time-units in the median; remarkably, it also outperforms the leading-indicator design (brown) by -1 time-unit[25]. As expected, the univariate MSE-DFA (orange) is lagging the MSE-leading-indicator design (brown) by 0 time-unit which reflects the lead-time provided by the additional leading indicator.

7.7.3 Performances: MSE

Box-plots of in-sample and out-of-sample MSE-scores are shown in fig.7.14.

[25]Setting $\lambda = 100$ would result in an anticipation of the customized design by 2 time-units, at costs of Accuracy and Smoothness.

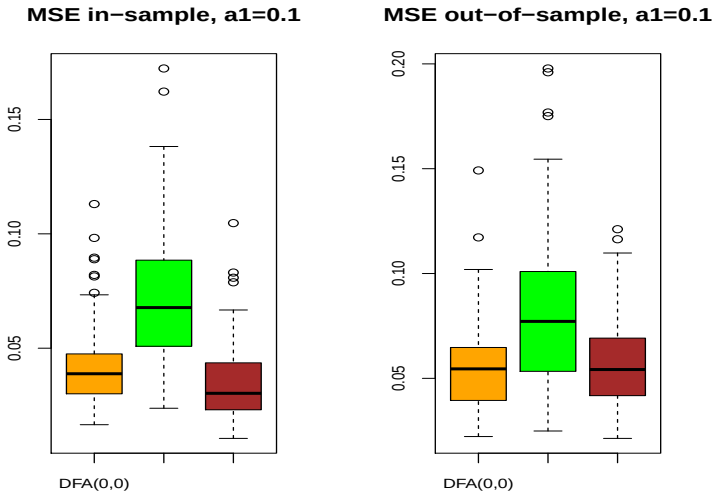


Figure 7.14: MSE: mean-square DFA (orange), customized DFA (green) and MSE-MDFA leading indicator (brown): a1=0.1 In-sample (left) and out-of-sample (right).

The customized design (green) is outperformed by the MSE-designs, as expected, in-sample as well as out-of-sample: re-scaling of filter coefficients would mitigate, at least to some extent, the observed losses, see section 7.4.3. The customized design seems to be less sensitive to overfitting: an explanation has been provided in section 7.4.2, recall fig.7.4.

7.7.4 Filter Outputs

The filter outputs corresponding to the last realization of the process are plotted in fig.7.15(the series are scaled for ease of visual inspection).

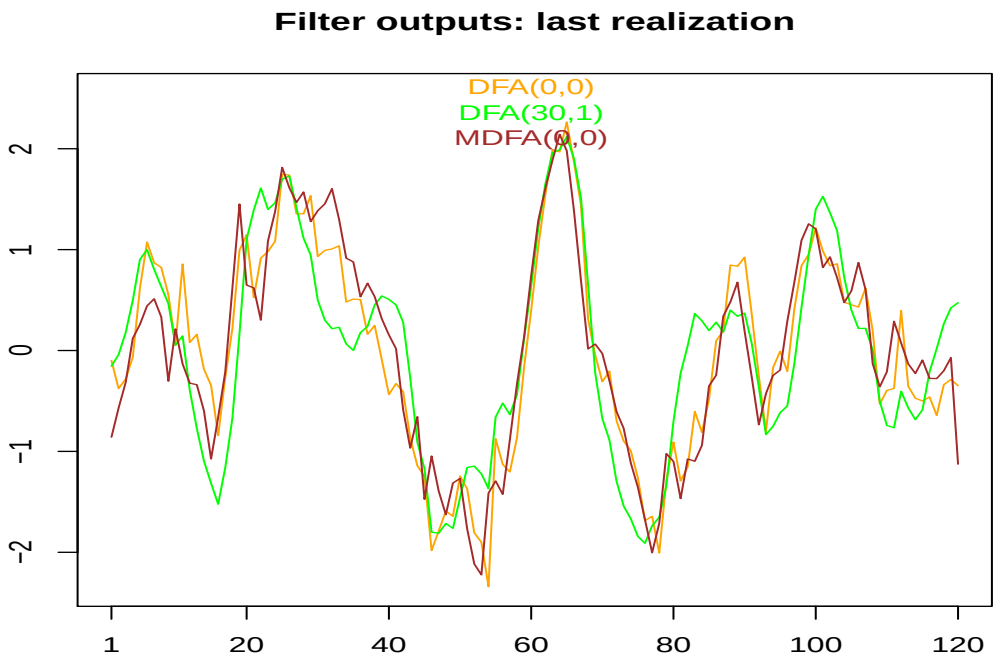


Figure 7.15: Scaled outputs of DFA-MSE (orange), DFA-balanced (green) and bivariate MDFA-MSE (brown): a1=0.9

The customized DFA (green) appears faster and smoother, as expected. Note that the leading-indicator design (brown) is systematically faster than DFA-MSE (orange) in the turning-points.

We retain from the above example that a suitably customized *univariate* design can outperform a classic MSE bivariate leading-indicator design, in-sample and out-of-sample, in terms of Peak Correlation and Curvature. A direct comparison of univariate and bivariate MSE-designs illustrates that the latter outperforms the former merely in terms of Peak Correlation (due to the leading indicator); gains in terms of MSE- or Curvature are moderate, at least out-of-sample.

7.8 An Extended Customization Triplet

Overemphasizing the Timeliness contribution to the MSE-norm, in the schematic criterion 7.5 or in criterion 7.7, provides a direct control of the magnitude of the time-shift of a real-time filter in the passband: as a result the filter-output shifts to the left, as confirmed by the Peak-Correlation statistic. As a possible alternative, the time-shift contribution to the MSE-norm can be overemphasized by targeting a lead $l > 0$ of the signal

$$\exp(il\omega)\Gamma(\omega) \tag{7.25}$$

where $\Gamma(\omega)$ designates the original target. Note that l corresponds to $-Lag$ in our R-code. In order to simplify the exposition we assume that $\Gamma(\omega) \geq 0$ is the transfer function of the ideal trend. Then

$$\Phi(\omega) = -Arg(\exp(il\omega)\Gamma(\omega)) = -l\omega$$

and the mean-square error becomes

$$\begin{aligned} MSE &= \frac{2\pi}{T} \sum_{\text{Passband}} (\Gamma(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \\ &+ \frac{2\pi}{T} \sum_{\text{Stopband}} (\Gamma(\omega_k) - \hat{A}(\omega_k))^2 I_{TX}(\omega_k) \\ &+ \frac{2\pi}{T} \sum_{\text{Passband}} 4\Gamma(\omega_k)\hat{A}(\omega_k) \sin\left(\frac{\hat{\Phi}(\omega_k) + l\omega_k}{2}\right)^2 I_{TX}(\omega_k) \end{aligned} \tag{7.26}$$

where we used the fact that $\Gamma(\omega) = A(\omega)$ since the transfer function is positive everywhere. Since our target is the ideal lowpass, we expect that $\hat{\Phi}(\omega_k) > 0$ in the passband: the output is shifted to the right with respect to the target. Therefore

$$\sin\left(\frac{\hat{\Phi}(\omega_k) + l\omega_k}{2}\right)^2 > \sin\left(\frac{\hat{\Phi}(\omega_k)}{2}\right)^2$$

if $l > 0$. As a result, the Timeliness term 7.26 inflates, in contrast to Accuracy and Smoothness, which remain unaffected by the lead-time. The resulting effect must be similar, though not identical, to a specific magnification of Timeliness in the ATS-trilemma and therefore we expect the output of the resulting *forecast* filter ($Lag = -l < 0$) to lie to the left of the corresponding nowcast filter ($Lag = 0$).

Instead of interpreting the forecast lead as a possible (indirect) alternative for controlling Timeliness, in lieu of λ, say, we envisage a combination of l and of the customization parameters: we consider the triplet (l, λ, η) as a natural extension of the original customization pair (λ, η). The additional 'customization' parameter $l > 0$ is helpful and effective if the time-shift of a particular real-time design is small already, in which case Timeliness cannot be substantially reduced anymore by selecting $\lambda > 0$: a corresponding example is provided in section 9.7.3 (customization of the Christiano Fitzgerald bandpass design). More generally and more formally, Timeliness is a measure of the phase error (the contribution of the phase to MSE)

$$\frac{2\pi}{T} \sum_{\text{Passband}} 4A(\omega_k)\hat{A}(\omega_k) \sin\left(\frac{\hat{\Phi}(\omega_k) - \Phi(\omega_k)}{2}\right)^2 I_{TX}(\omega_k)$$

where $\Phi(\omega_k) = -l\omega_k$. Selecting $\lambda > 0$, in criteria 7.5 or 7.7, shrinks the Timeliness term by pulling $\hat{\Phi}(\omega_k)$ towards $-l\omega_k$ or, equivalently, by attracting the time-shift $\hat{\phi}(\omega_k) = \hat{\Phi}(\omega_k)/\omega_k$ towards $-l$. In contrast to the original customization pair (λ, η), the extended triplet (l, λ, η) allows to design filters with more general lead or lag properties. A worked-out example is provided in chapter 9, section 9.7.3.

7.9 Summary

- The MSE-norm can be split into Accuracy, Timeliness and Smoothness components[26].
- The MSE-paradigm is replicated by weighting ATS-components equally. Equal-weighting reflects one particular 'diffuse' research priority.
- A strict MSE-approach is unable, per definition, to address Timeliness and Smoothness, either separately or all together.
- The ATS-trilemma degenerates to a AT-dilemma in the case of classic (allpass) forecasting. Stated otherwise: classic (quasi maximum likelihood) forecast approaches have a blind spot.
- Curvature and Peak-Correlation performances can be addressed simultaneously by customized designs.
- The DFA and the resulting ATS-trilemma are generic concepts: they allow for arbitrary target signals as well as arbitrary spectral estimates. Model-based targets and (pseudo) spectral densities are addressed in chapter 9.
- The generic customization triplet (l, λ, η) extends the action of the ATS-trilemma to arbitrary real-valued lead or lag specifications.
- The DFA/MDFA optimization criteria appear to be surprisingly robust against overfitting. Emphasizing Smoothness reinforces this statement.

[26] The Residual either vanishes or is negligible in practice.

Chapter 8

ATS-Trilemma: the Multivariate Case

In this chapter, the ATS-trilemma and the customized optimization principle are generalized to a multivariate MDFA-framework. Section 8.1 briefly reviews previously available criteria and derives all relevant concepts: multivariate ATS-trilemma, customized criterion, quadratic criterion and the corresponding closed-form solution; section 8.2 analyzes the effects obtained by emphasizing Timeliness and/or Smoothness in the new multivariate framework; finally, section 8.3 concludes by an empirical contest, comparing performances of univariate and multivariate MSE- and customized designs.

8.1 MDFA: ATS-Trilemma and Customization

We briefly list previous optimization criteria and we derive a generalization of the ATS-trilemma to the MDFA. A quadratic approximation of the estimation problem is proposed with a corresponding unique closed-form solution.

8.1.1 A Review of Previous Optimization Criteria

DFA-MSE

$$\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\left|\Gamma(\omega_k)-\hat{\Gamma}(\omega_k)\right|^2 I_{TX}(\omega_k)\rightarrow\min_{\mathbf{b}} \tag{8.1}$$

see 4.27.

MDFA-MSE

$$\frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\left|\left(\Gamma(\omega_k)-\hat{\Gamma}_X(\omega_k)\right)\Xi_{TX}(\omega_k)-\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\right|^2\rightarrow\min_{\mathbf{B}} \tag{8.2}$$

see 4.32. In matrix-notation we obtain

$$(\mathbf{Y}_{\text{rot}} - \mathbf{X}_{\text{rot}}\mathbf{b})'(\mathbf{Y}_{\text{rot}} - \mathbf{X}_{\text{rot}}\mathbf{b}) \rightarrow \min_{\mathbf{b}} \tag{8.3}$$

where $\mathbf{b} := \text{Vec}(\mathbf{B})$ is the vector of stacked columns of $\mathbf{B}$. The solution is

$$\hat{\mathbf{b}} \quad = \quad (\Re(\mathbf{X}'_{\text{rot}}\mathbf{X}_{\text{rot}}))^{-\mathbf{1}}\,\Re(\mathbf{X}_{\text{rot}})'\mathbf{Y}_{\text{rot}} \tag{8.4}$$

see section 4.4.2. The solution of the constrained optimization problem was proposed in 6.25.

Customized DFA

$$\begin{aligned}&\frac{2\pi}{T}\sum_{\text{All Frequencies}}(A(\omega_k) - \hat{A}(\omega_k))^2 W(\omega_k,\eta) I_{TX}(\omega_k)\\ &+(1+\lambda)\frac{2\pi}{T}\sum_{\text{Passband}} 4A(\omega_k)\hat{A}(\omega_k)\sin(\hat{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \rightarrow \min_{\mathbf{b}}\end{aligned} \tag{8.5}$$

see 7.7. The analytically tractable quadratic expression is

$$\begin{aligned}&\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Big|\big|\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big| - \Big\{\Re\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big\}\big)\Big]\\ &+i\sqrt{1+\lambda|\Gamma(\omega_k)|}\Im\Big[\hat{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big]\Big\}\Big|^2 W(\omega_k,\eta) \rightarrow \min_{\mathbf{b}}\end{aligned} \tag{8.6}$$

see 7.17.

8.1.2 ATS-Trilemma and Multivariate Customization

A generalization of the customized criterion 8.5 to the multivariate case can be gleaned from the timely ordering of criteria, by applying a simple, but effective, notational trick to the MSE-criterion 8.2:

$$\begin{aligned}&\frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\left|\Gamma(\omega_k)\Xi_{TX}(\omega_k) - \hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k) - \sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\right|^2\\ =\;&\frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\left|\Gamma(\omega_k) - \hat{\Gamma}_X(\omega_k) - \sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\frac{\Xi_{TW_n}(\omega_k)}{\Xi_{TX}(\omega_k)}\right|^2 |\Xi_{TX}(\omega_k)|^2\\ =\;&\frac{2\pi}{T}\sum_{k=-T/2}^{T/2}\left|\Gamma(\omega_k) - \tilde{\Gamma}(\omega_k)\right|^2 I_{TX}(\omega_k)\end{aligned} \tag{8.7}$$

where

$$\tilde{\Gamma}(\omega_k) := \hat{\Gamma}_X(\omega_k) + \sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\frac{\Xi_{TW_n}(\omega_k)}{\Xi_{TX}(\omega_k)} \tag{8.8}$$

Literally, expression 8.7 'looks' very much like 8.1 (up to the fact that the former involves multiple time series and a richly parametrized multivariate filter). Therefore, we could re-write the ATS-components in 7.4 as

$$\left.\begin{array}{rcl} \text{A(ccuracy)} & := & \frac{2\pi}{T}\sum_{\text{Passband}}(A(\omega_k)-\tilde{A}(\omega_k))^2 I_{TX}(\omega_k) \\ \text{S(moothness)} & := & \frac{2\pi}{T}\sum_{\text{Stopband}}(A(\omega_k)-\tilde{A}(\omega_k))^2 I_{TX}(\omega_k) \\ \text{T(imeliness)} & := & \frac{2\pi}{T}\sum_{\text{Passband}} 4A(\omega_k)\tilde{A}(\omega_k)\sin(\tilde{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \\ \text{R(esidual)} & := & \frac{2\pi}{T}\sum_{\text{Stopband}} 4A(\omega_k)\tilde{A}(\omega_k)\sin(\tilde{\Phi}(\omega_k)/2)^2 I_{TX}(\omega_k) \end{array}\right\} \tag{8.9}$$

where $\tilde{A}(\omega_k)$ and $\tilde{\Phi}(\omega_k)$ are amplitude and phase functions of the aggregate transfer function $\tilde{\Gamma}(\omega_k)$ defined in 8.8. The generalized ATS-Trilemma is obtained by splitting the MSE-norm into ATS-contributions (Residual is ignored)

$$\text{MSE} = \text{Accuracy} + \text{Timeliness} + \text{Smoothness}$$

and a schematic (simplified) customized criterion is obtained by assigning weights to its constituents

$$\text{MSE-Cust}(\lambda_1,\lambda_2) = \text{Accuracy} + (1+\lambda_1)\text{Timeliness} + (1+\lambda_2)\text{Smoothness} \rightarrow \min_{\mathbf{B}} \tag{8.10}$$

where $\mathbf{B}$ is the matrix of filter coefficients. Using the weighting-functions introduced in section 7.1.3 the multivariate customized criterion becomes

$$\begin{aligned} &\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} |\Gamma(\omega_k)-\tilde{\Gamma}(\omega_k)|^2 W(\omega_k,\eta) I_{TX}(\omega_k) \\ &+4\lambda\frac{2\pi}{T}\sum_{k=-T/2}^{T/2} A(\omega_k)\tilde{A}(\omega_k)\sin(\tilde{\Phi}(\omega_k)/2)^2 W(\omega_k,\eta) I_{TX}(\omega_k) \rightarrow \min_{\mathbf{B}} \end{aligned} \tag{8.11}$$

Instead of customizing all sub-filters individually, the proposed generalization of 8.5 addresses the aggregate multivariate filter $\tilde{\Gamma}(\omega_k)$ 'directly': a single customization pair (λ,η) addresses Timeliness and Smoothness of the compound filter output.

8.1.3 Quadratic Criterion

The univariate quadratic customized criterion 8.6 can be rewritten in the multivariate case as

$$\begin{aligned} &\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]} \Big|\big|\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big| - \Big\{\Re\Big[\tilde{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big\}\big)\Big] \\ &+i\sqrt{1+\lambda|\Gamma(\omega_k)|}\Im\Big[\tilde{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big]\Big\}\Big|^2 W(\omega_k,\eta) \rightarrow \min_{\mathbf{b}} \end{aligned} \tag{8.12}$$

where $\tilde{\Gamma}(\omega_k)$ is the aggregated multivariate filter 8.8. This criterion can be solved in closed-form, see below. Moreover, λ and η address the relevant Timeliness – speed – and Smoothness – noise suppression – issues. Finally, all previous criteria are nested in 8.12. In order to derive a

corresponding solution we explode the former expression into contributions by all subfilters:

$$\begin{aligned}
& \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Bigg|\big|\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big| - \Bigg\{\Re\Big[\tilde{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\big\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\big\}\big)\Big] \\
& +i\sqrt{1+\lambda|\Gamma(\omega_k)|}\Im\Big[\tilde{\Gamma}(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big]\Bigg\}\Bigg|^2 W(\omega_k,\eta) \\
= \; & \frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\Bigg||\Gamma(\omega_k)\Xi_{TX}(\omega_k)| - \Bigg\{\Re\Big[\hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big) \\
& +\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big] \\
& +i\sqrt{1+\lambda\Gamma(\omega_k)}\Im\Big[\hat{\Gamma}_X(\omega_k)\Xi_{TX}(\omega_k)\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big) \\
& +\sum_{n=1}^{m}\hat{\Gamma}_{W_n}(\omega_k)\Xi_{TW_n}(\omega_k))\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\Big]\Bigg\}\Bigg|^2 W(\omega_k,\eta) \\
& \to \min_{\mathbf{B}}
\end{aligned} \tag{8.13}$$

Being a quadratic function of filter coefficients, criterion 8.13 can be solved in closed-form, see below. If $\lambda = 0$ then 8.13 and 8.11 are identical; indeed both criteria coincide with the MSE-criterion 8.2. For $\lambda > 0$, 8.13 and 8.11 are not strictly identical, anymore. However, as shown in section 7.3.2, the former (quadratic criterion) approximates the latter (customized criterion) tightly, even if $\lambda \geq 0$ is large[1].

8.1.4 Least-Squares Customized Solution

We now generalize the MSE-expressions 4.38 and 4.39 of design matrix and target to the proposed customization-framework. Define

$$\begin{aligned}
\mathbf{X}_k^{\text{Cust}}(\lambda,\eta) &:= \Big\{\Re(\mathbf{X}_k(\lambda,\eta)) + i\sqrt{1+\lambda\Gamma(\omega_k)}\Im(\mathbf{X}_k(\lambda,\eta))\Big\}\sqrt{W(\omega_k,\eta)} \\
\mathbf{Y}^{\text{Cust}}(\eta) &:= \begin{pmatrix}
|\Gamma(\omega_0)\Xi_{TX}(\omega_0)|\sqrt{W(\omega_0,\eta)} \\
\sqrt{2}|\Gamma(\omega_1)\Xi_{TX}(\omega_1)|\sqrt{W(\omega_1,\eta)} \\
\sqrt{2}|\Gamma(\omega_2)\Xi_{TX}(\omega_2)|\sqrt{W(\omega_2,\eta)} \\
\cdot \\
\sqrt{2}|\Gamma(\omega_{T/2})\Xi_{TX}(\omega_{T/2})|\sqrt{W(\omega_{T/2},\eta)}
\end{pmatrix}
\end{aligned}$$

where the target $\mathbf{Y}^{\text{Cust}}(\eta)$ depends on η, through the weighting function $W(\omega_k,\eta)$, and where

$$\begin{aligned}
\mathbf{X}_k(\lambda,\eta)' &= \sqrt{1+I_{k>0}}\exp\big(-i\arg\{\Gamma(\omega_k)\Xi_{TX}(\omega_k)\}\big)\cdot \\
& \qquad \text{Vec}_{\text{row}}\begin{pmatrix}
\Xi_{TX}(\omega_k) & \exp(-i\omega_k)\Xi_{TX}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TX}(\omega_k) \\
\Xi_{TW_1}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_1}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_1}(\omega_k) \\
\Xi_{TW_2}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_2}(\omega_k) & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_2}(\omega_k) \\
... & ... & ... & ... \\
\Xi_{TW_m}(\omega_k) & \exp(-i\omega_k)\Xi_{TW_m}(\omega_k & ... & \exp(-i(L-1)\omega_k)\Xi_{TW_m}(\omega_k)
\end{pmatrix}
\end{aligned}$$

[1] Because phase-differences between target and aggregate multivariate output are small (linearization).

whereby Vec_{row} appends rows of a matrix to build a single long row-vector of dimension $(m+1)L$, recall section 4.4.1. Finally, define $\mathbf{X}^{\text{Cust}}(\lambda,\eta)$ to be the customized design matrix whose k-th row is $\mathbf{X}_k^{\text{Cust}}(\lambda,\eta)'$, as defined above. Since the target vector is already real and positive we can save the rotation in section 4.4.1 and criterion 8.13 can be rewritten more conveniently as

$$(\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b})'(\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b})\rightarrow\min_{\mathbf{b}} \tag{8.14}$$

where $\mathbf{b}:=\text{Vec}(\mathbf{B})$ is the vector of stacked columns of $\mathbf{B}$. Accordingly, the customized coefficient estimate is obtained as

$$\hat{\mathbf{b}}^{\text{Cust}}(\lambda,\eta) \;=\; \left(\Re\left\{(\mathbf{X}^{\text{Cust}}(\lambda,\eta))'\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right\}\right)^{-\mathbf{1}}\Re(\mathbf{X}^{\text{Cust}}(\lambda,\eta))'\mathbf{Y}^{\text{Cust}}(\eta) \tag{8.15}$$

see section 4.4.2. In order to avoid unnecessary overloading of notations in subsequent chapters we drop the parameters (λ,η) in all customized expressions; we retain the superscript 'Cust' to distinguish the customized estimate from the 'plain vanilla' MSE-estimate 4.42, in the case $\lambda=\eta=0$. Formally, it is an easy matter to account for additional I(1)- ($i1=T$) and/or I(2)-constraints ($i2=T$) proposed in chapter 6: just add the superscript Cust to all terms in the solution 6.25 (with all accompanying modifications). The generic MDFA-function

```
> head(mdfa_analytic)

1 function (L, lambda, weight_func, Lag, Gamma, eta, cutoff, i1,
2     i2, weight_constraint, lambda_cross, lambda_decay, lambda_smooth,
3     lin_eta, shift_constraint, grand_mean, b0_H0, c_eta, weight_structure,
4     white_noise, synchronicity, lag_mat, troikaner)
5 {
6     lambda <- abs(lambda)
```

handles the general case (and more, to come) whereas the context-specific variants

```
> head(MDFA_cust)

1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta)
2 {
3     lin_eta <- F
4     weight_constraint <- rep(1/(ncol(weight_func) - 1), ncol(weight_func) -
5         1)
6     lambda_cross <- lambda_smooth <- 0
```

```
> head(MDFA_cust_constraint)

1 function (L, weight_func, Lag, Gamma, cutoff, lambda, eta, i1,
2     i2, weight_constraint, shift_constraint)
3 {
4     lin_eta <- F
5     lambda_cross <- lambda_smooth <- 0
6     lambda_decay <- c(0, 0)
```

handle multivariate customization with or without constraints imposed. We now analyze performances of multivariate customized designs.

8.2 The Multivariate ATS-Trilemma in Action

8.2.1 Empirical Framework

We rely on the bivariate leading indicator design proposed in section 4.7. Specifically, we analyze the second process, $a_1 = 0.1$, and we target an ideal trend with cutoff $\pi/6$ by filters of length $L = 12$. This design slightly departs from the previous chapter, where we selected cutoff $\pi/12$ and length $L = 24$, because we do not want overfitting to interfer with our results in the richer multivariate framework[2].

8.2.2 Emphasizing Smoothness: $\eta \geq 0$, $\lambda = 0$ Fixed

We fix $\lambda = 0$ and let $\eta_k = 0.3 * k, k = 0, ..., 6$.

Run the Code

```
> # Second process
> a1<-0.1
> # Customization settings MDFA
> eta_mdfa<-0.3*(0:6)
> # Fix lambda=0
> lambda_mdfa<-rep(0,length(eta_mdfa))
> # Customization settings DFA
> lambda_vec<-lambda_vec
> eta_vec<-eta_vec
> # target
> cutoff<-pi/6
> # Filter length
> L<-12
> # Real-time design
> Lag<-0
> # No constraints
> i1<-i2<-F
> # Sample length 120
> len<-120

> cust_leading_obj<-mdfa_mse_leading_indicator_vs_dfa_customized(anzsim,a1,
+     cutoff,L,lambda_vec,eta_vec,len1,len,i1,i2,Lag,lambda_mdfa,eta_mdfa,troikaner)
```

[2]Recall that a minimal filter-length of $L = 2\pi/(\pi/6) = 12$ is required in order to suppress frequency $\pi/6$, as required by the cutoff, and that $2 * L = 24$ filter coefficients are estimated for the bivariate filters in samples of length $T = 120$ (which corresponds to ten years of monthly data).

The above function generates artificial data, applies our general DFA (*dfa_analytic*) and MDFA (*mdfa_analytic*) estimation routines, computes amplitude and time-shift-functions, filter coefficients, ATS-components and filter-outputs as well as Peak Correlation and Curvature statistics.

ATS-Components

ATS-components of the aggregate filter, as defined in section 8.1.2, are calculated explicitly and returned by the multivariate MDFA-function

```
> tail(mdfa_analytic)
```

```
187     else {
188         return(list(b = b, trffkt = trffkt, rever = rever, Accuracy = Accuracy,
189             Smoothness = Smoothness, Timeliness = Timeliness,
190             MS_error = MS_error))
191     }
192 }
```

The results are summarized in table 8.1. Smoothness decreases monotonically as a function of η,

	Accuracy	Timeliness	Smoothness	Residual	MSE
MDFA(0,0)	0.020167	0.032431	0.052677	0.000000	0.105276
MDFA(0,0.3)	0.032263	0.046041	0.034387	0.000000	0.112692
MDFA(0,0.6)	0.049511	0.060152	0.022365	0.000000	0.132028
MDFA(0,0.9)	0.071750	0.074737	0.013860	0.000000	0.160348
MDFA(0,1.2)	0.097173	0.089669	0.007933	0.000000	0.194775
MDFA(0,1.5)	0.123658	0.102905	0.004286	0.000000	0.230849
MDFA(0,1.8)	0.151625	0.110037	0.002373	0.000000	0.264036

Table 8.1: ATS-Components as a function of eta (lambda=0 fixed)

as desired.

Amplitude-Functions, Time-Shift Functions and Filter Coefficients

The effect of the Smoothness-customization on amplitude- and time-shift functions of the subfilters, as well as on their coefficients, is visualized in fig.8.1. Prior to interpreting these results let us emphasize that customization does not address the subfilters but the *aggregate* filter $\tilde{\Gamma}(\cdot)$ defined in 8.8. Therefore, the transformation of the sub-filters, analyzed here, is an 'indirect' effect of customization.

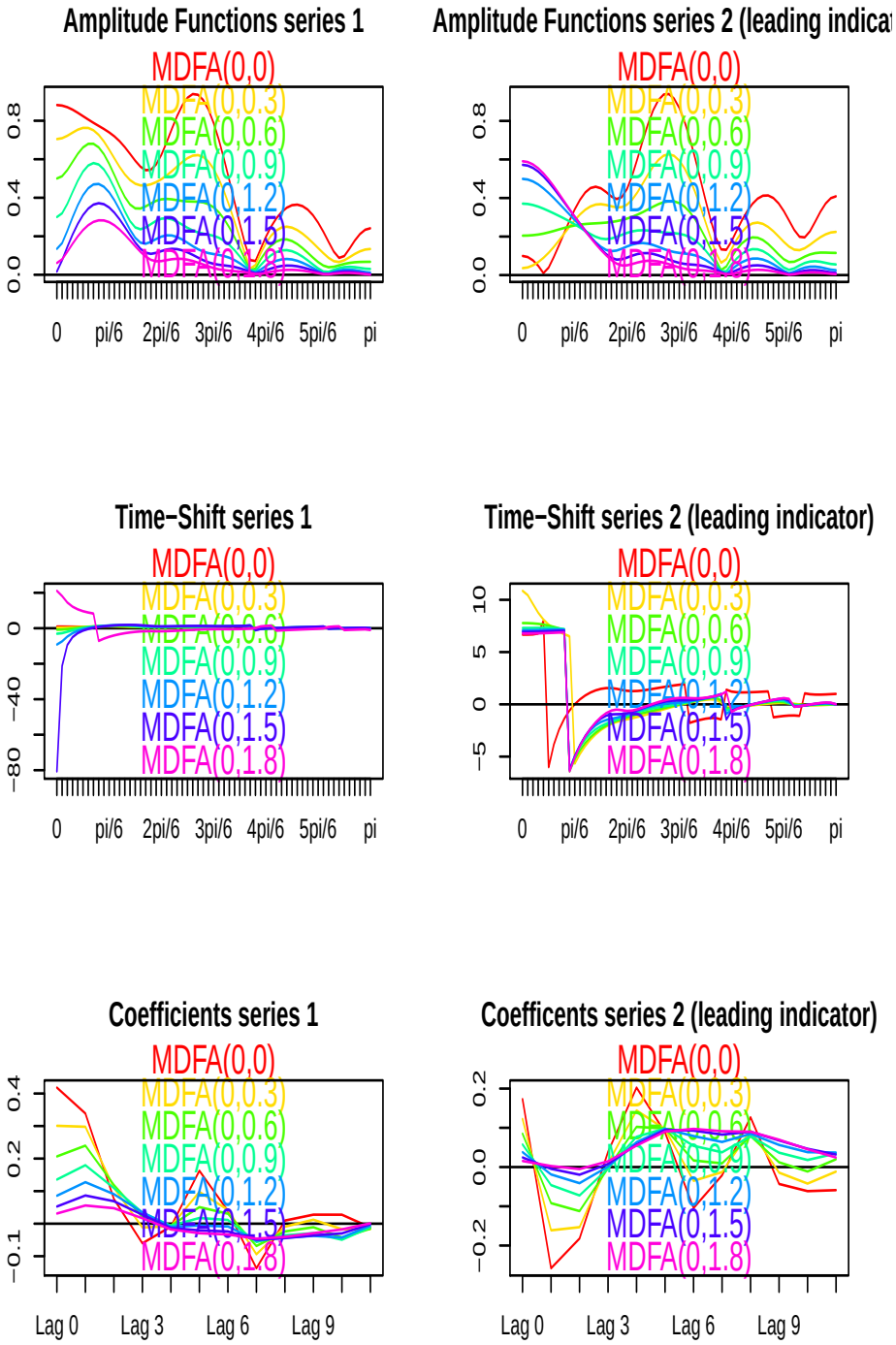


Figure 8.1: Amplitude (top), time-shift functions (middle) and filter coefficients (bottom): series 1 (left) and series 2 (right) for customized MDFA-leading indicator (emphasizing Smoothness): a1=0.1

Analysis

- Emphasizing Smoothness of the aggregate filter $\tilde{\Gamma}(\cdot)$, by η, exerts effects on amplitude and time-shift functions of the subfilters $\hat{\Gamma}_X(\cdot), \hat{\Gamma}_{W_1}$ of the bivariate design which conform with expectations. In particular, the amplitude functions are shrinking in the stopband (stronger noise suppression) and the time-shifts are increasing in the passband, as η increases[3].
- As for the DFA in the previous chapter, emphasizing Smoothness, by η, seems to smooth filter coefficients too. This phenomenon, which is mainly due to the more or less intensive amplitude-shrinkage in the stop-band (steeper transfer function), implies that suitably customized filters are less prone to overfitting than classic MSE-designs.

Effects on Aggregated Filter

Amplitude and time-shift functions of the aggregated filter $\tilde{\Gamma}(\cdot)$, defined in 8.8, are plotted in fig. 8.2.

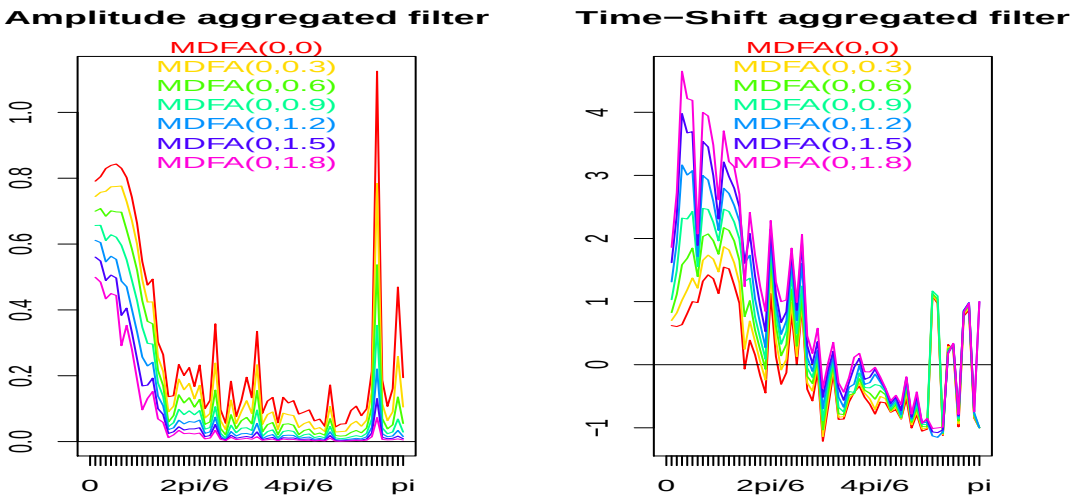


Figure 8.2: Amplitude (left) and time-shift functions (right) of aggregated filter when emphasizing Smoothness: a1=0.1

Note that the ratio of DFTs in 8.8 is uninformative in frequency zero and therefore we skipped the corresponding numbers. Both functions, amplitude and time-shift, are noisy because the DFT-ratio is noisy, too. Nevertheless, we can observe that the effect intended by η is achieved by the aggregate: the amplitude function shrinks to zero in the stopband, as desired[4] and the time-shift in the passband increases.

[3]Note that the results obtained here are not directly comparable with those in section 4.7 because the processes are different: $a_1 = 0.1$ (here) vs. $a_1 = 0.9$ (there).

[4]The amplitude is also slightly shrunken in the passband, as expected, but shrinkage in the stopband is much stronger.

8.2.3 Emphasizing Timeliness: $\lambda \geq 0$, $\eta = 0$ Fixed

We fix $\eta = 0$ and let $\lambda_k = (0, 2^k), k = 0, ..., 7$.

Run the Code

```
> # Customization settings DFA
> lambda_mdfa<-c(0,2^(0:7))
> # Fix eta=0
> eta_mdfa<-rep(0,length(lambda_mdfa))
```

```
> cust_leading_obj<-mdfa_mse_leading_indicator_vs_dfa_customized(anzsim,a1,
+     cutoff,L,lambda_vec,eta_vec,len1,len,i1,i2,Lag,lambda_mdfa,eta_mdfa,troikaner)
```

ATS-Components

ATS-components are summarized in table 8.2. Timeliness decreases monotonically as a function

	Accuracy	Timeliness	Smoothness	Residual	MSE
MDFA(0,0)	0.020167	0.032431	0.052677	0.000000	0.105276
MDFA(1,0)	0.031299	0.017587	0.060892	0.000000	0.109778
MDFA(2,0)	0.038297	0.011085	0.066862	0.000000	0.116244
MDFA(4,0)	0.046356	0.005646	0.074515	0.000000	0.126518
MDFA(8,0)	0.053683	0.002395	0.082194	0.000000	0.138272
MDFA(16,0)	0.059287	0.000931	0.088355	0.000000	0.148573
MDFA(32,0)	0.063452	0.000364	0.092600	0.000000	0.156417
MDFA(64,0)	0.066853	0.000148	0.095282	0.000000	0.162283
MDFA(128,0)	0.069854	0.000063	0.096888	0.000000	0.166806

Table 8.2: ATS-Components as a function of lambda (eta=0 fixed)

of λ, as required. We here skip the sub-filters and go straight to the interesting aggregate.

Effects on Aggregated Filter

Amplitude and time-shift functions of the aggregated filter $\tilde{\Gamma}(\cdot)$, defined in 8.8, are plotted in fig. 8.3.

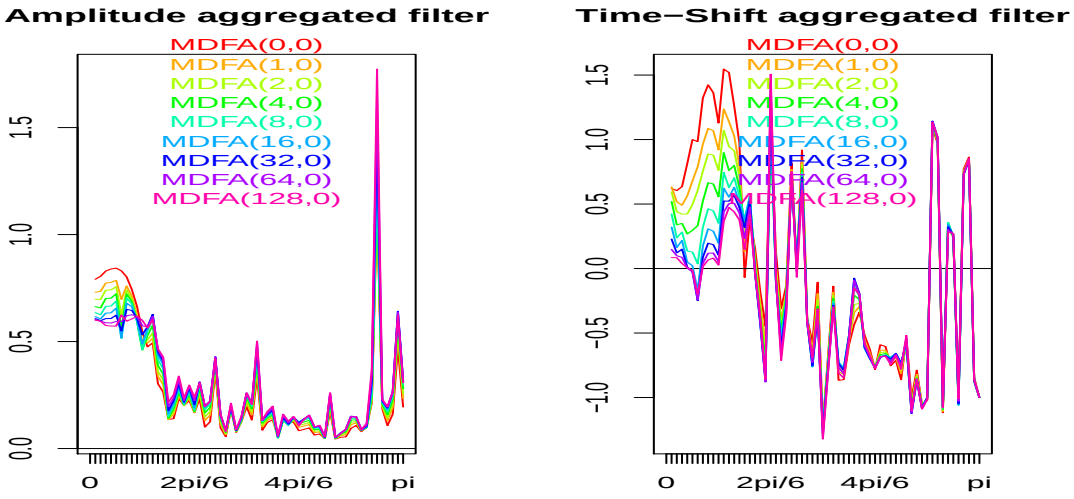


Figure 8.3: Amplitude (left) and time-shift functions (right) of aggregated filter when emphasizing Timeliness: a1=0.1

We can observe that the effect intended by emphasizing Timeliness is achieved: the time-shift of $\tilde{\Gamma}(\cdot)$ approaches zero in the passband as λ increases.

8.2.4 Emphasizing Timeliness and Smoothness: : $\lambda > 0$, $\eta > 0$

We compare MSE and balanced customized $\lambda = 30, \eta = 1$ designs: the latter refers to the univariate balanced DFA-design in section 7.6.

Run the Code

```
> # Customization settings DFA
> lambda_mdfa<-c(0,30)
> # Fix eta=0
> eta_mdfa<-c(0,1)
> cust_leading_obj<-mdfa_mse_leading_indicator_vs_dfa_customized(anzsim,a1,
+     cutoff,L,lambda_vec,eta_vec,len1,len,i1,i2,Lag,lambda_mdfa,eta_mdfa,troikaner)
```

ATS-Components

ATS-components are summarized in table 8.3. Timeliness and Smoothness decrease with increasing (λ, η)-pair, as required.

Effects on Aggregated Filter

Amplitude and time-shift functions of the aggregated filter $\tilde{\Gamma}(\cdot)$, defined in 8.8, are plotted in fig. 8.4: note that we re-scaled the customized filter because of zero-shrinkage, recall section 7.4.3.

	Accuracy	Timeliness	Smoothness	Residual	MSE
MDFA(0,0)	0.020167	0.032431	0.052677	0.000000	0.105276
MDFA(30,1)	0.262417	0.002423	0.008149	0.000000	0.272988

Table 8.3: ATS-Components as a function of lambda and eta

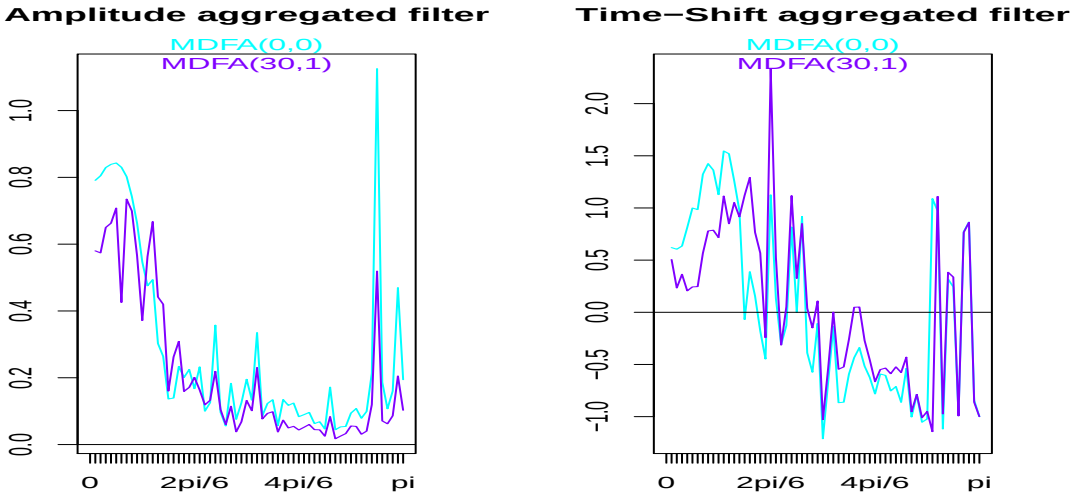


Figure 8.4: Amplitude (left) and time-shift functions (right) of aggregated filter when emphasizing Timeliness: a1=0.1

The effects intended by emphasizing Timeliness and Smoothness are achieved: the amplitude of the customized $\tilde{\Gamma}(\cdot)$ is closer to zero in the stopband and its time-shift is closer to zero in the passband. We now measure the relevant impact of customization on Curvature and Peak Correlation statistics.

8.3 Empirical Contest: MDFA vs. DFA

In contrast to the previous section we here address multiple realizations of the process and derive in-sample as well as out-of-sample distributions of Peak Correlation, Curvature and MSE performance measures. The empirical design is the same as in the previous section: we target an ideal trend with cutoff $\pi/6$, based on filters of length $L = 12$. The contenders in our study are univariate (DFA) and bivariate (MDFA) *MSE*-designs ($\lambda = \eta = 0$) and *customized* DFA and MDFA 'balanced' designs $\lambda = 30, \eta = 1$: four competitors, one of each kind.

```
> # Customization MDFA: MSE and balanced designs
> lambda_mdfa<-c(0,30)
> eta_mdfa<-c(0,1.)
> # Customization DFA: MSE and balanced designs
```

```
> lambda_vec<-c(0,30)
> eta_vec<-c(0,1)
> # Second process
> a1<-0.1
> # target
> cutoff<-pi/6
> # Filter length
> L<-12
> # Real-time design
> Lag<-0
> # No constraints
> i1<-i2<-F
> # Sample length 120
> len<-120
> # Number of replications
> anzsim<-100
```

```
> cust_leading_obj<-mdfa_mse_leading_indicator_vs_dfa_customized(anzsim,a1,
+     cutoff,L,lambda_vec,eta_vec,len1,len,i1,i2,Lag,lambda_mdfa,eta_mdfa,troikaner)
```

Box-plots of in-sample and out-of-sample Curvature, Peak Correlation and MSE-scores are depicted in fig.8.5.

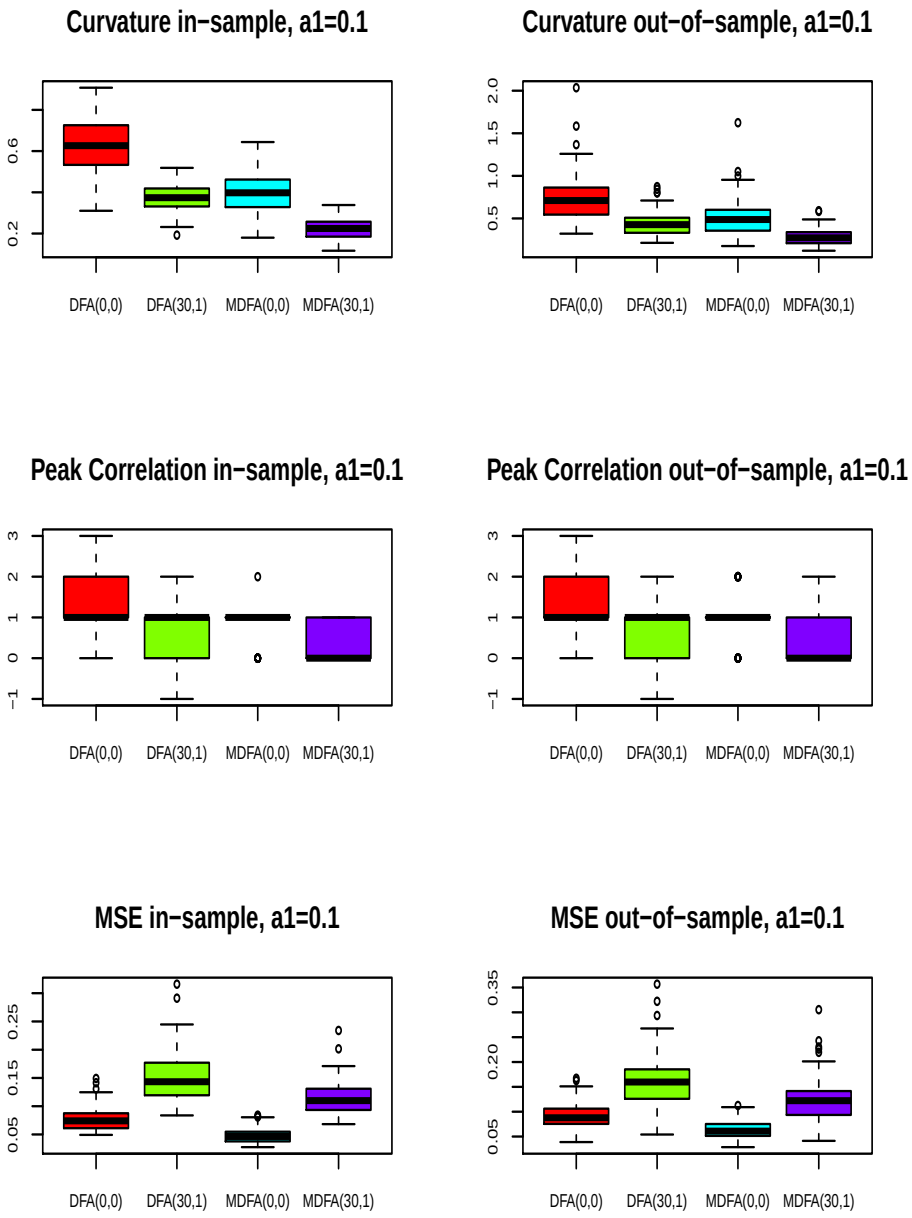


Figure 8.5: Curvature (top), Peak Correlation (middle) and MSE (bottom), in-sample (left) and out-of-sample (right) for MSE-DFA (red), balanced DFA (green), MSE-MDFA (cyan) and balanced MDFA (violet): a1=0.1

Analysis

- Recall that the empirical design is different than in section 7.6 so that performances cannot be compared directly.

- Both MSE-designs (red,cyan) are outperformed in terms of Curvature and Peak Correlation, by the balanced customized designs, in-sample and out-of-sample; but the latter loose in terms of MSE.
- The univariate balanced design (green) is outperformed by the multivariate balanced design (violet) on all accounts; in-sample as well as out-of-sample. Similarly, DFA-MSE (red) is outperformed by MDFA-MSE (cyan), in-sample and out-of-sample.
- In-sample and out-of-sample performances are congruent which suggests that overfitting is avoided[5].

8.4 Summary

- The concepts in the previous chapter – trilemma, customized criterion, quadratic criterion – could be generalized to a multivariate framework by emphasizing the aggregate filter (output). Instead of tackling each sub-filter of the multivariate design separately, this proceeding addresses the problem in its entirety, by a single set of customization parameters (λ, η).
- Empirical examples illustrate efficiency gains of MDFA vs. DFA in the context of the bivariate leading-indicator design. Multivariate designs (MSE/balanced customized) outperform their univariate siblings on all accounts: Timeliness, Curvature and MSE, in-sample as well as out-of-sample.
- All previous optimization criteria are nested in the proposed multivariate closed-form solution.

[5]The almost identical in- and out-of-sample distributions of Peak Correlation is due to discretization effects, since the Peak Correlation statistic is integer-valued.

Chapter 9

Replicating and Customizing Classic Model-Based Approaches

9.1 Introduction

The DFA is a generic signal extraction and forecast paradigm whose optimization criterion can be configurated by the user in order to align research priorities, problem structure and estimation principle. This chapter addresses the spectrum interface. In the generic DFA, the spectrum $h(\cdot)$ assigns a frequency-dependent weight to the approximation of $\Gamma(\cdot)$ by $\hat{\Gamma}(\cdot)$

$$\frac{2\pi}{T}\sum_{k=-[T/2]}^{[T/2]}\left|\Gamma(\omega_k)-\hat{\Gamma}(\omega_k)\right|^2 h(\omega_k)$$

If $h(\omega_k)$ is 'large', in relative terms, then $\hat{\Gamma}(\omega_k)$ should match $\Gamma(\omega_k)$ closely in frequency ω_k. Interestingly, classic model-based ARIMA or state-space approaches as well as classic filter designs, such as Hodrick Prescott, Christiano Fitzgerald or Henderson, for example, can be replicated up to arbitrary precision by specifying and inserting appropriate model-based or implied (pseudo-) spectral densities for $h(\omega_k)$ in the above expression. Once replicated, nothing stands in the way of customization: Timeliness and Smoothness or, equivalently, Peak Correlation and Curvature of classic time series approaches can be tackled at once. A correspondingly customized model-based approach is a *hybrid*: whereas the spectrum relies on (pseudo-) maximum likelihood, the filter output is obtained by emphasizing the filter error and by abandoning the mean-square perspective in the expanded scope of the ATS-trilemma.

Section 9.2 emphasizes ARIMA-based approaches: an equivalence between (MSE-) DFA and model-based approaches (MBA) is established by linking time-domain and frequency-domain mean-square approaches; customization of the MBA by the DFA is discussed in section 9.3; section 9.4 addresses replication of unobserved components – state space – models whereby the DGP is factored into sub-processes (structural approach); section 9.5 handles customization of the latter; finally, section 9.6 proposes replication and customization of classic filter designs, specifically HP

and CF-filters.

9.2 Replication of ARIMA-Based Approaches by the DFA

We show that forecast and signal-extraction performances of the classic ARIMA-based paradigm can be replicated by the DFA.

9.2.1 Framework

In order to keep exposition simple and short we emphasize AR(1)-processes, our 'fil rouge' introduced in section 4.2.5. We assume that the DGP is known and we assume, also, that the target is an ideal trend with *cutoff* $\pi/12$

$$\Gamma(\omega) = \begin{cases} 1 & |\omega| \leq \pi/12 \\ 0 & \text{otherwise} \end{cases}$$

This is a typical target specification for monthly data in the context of business-cycle analysis. The target is supposed to be approximated by *real-time* designs: $Lag = 0$. Note, however, that the scope of our analysis remains general: any ARIMA-process[1] and/or any target signal[2] and/or any *Lag* could be entertained by straightforward modifications. We now distinguish approaches according to whether (mean-square) solutions are obtained in the time domain or in the frequency domain.

9.2.2 Time-domain

The target signal

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k} \tag{9.1}$$

assumes knowledge of data $x_{t-k}, k < 0$ that is unobserved at t, the time of inference. An optimal (mean-square) estimate $\hat{y}_T$ of y_T in $t = T$ could be obtained by inserting optimal forecasts (and/or backcasts) of the unobservable data:

$$\begin{aligned} \hat{y}_T &= \overbrace{\sum_{k=-\infty}^{\infty} \gamma_k x_{T-k}} \\ &= \sum_{k=-\infty}^{-1} \gamma_k \hat{x}_{T-k} + \sum_{k=0}^{T-1} \gamma_k x_{T-k} + \sum_{k=T}^{\infty} \gamma_k \hat{x}_{T-k} \end{aligned}$$

[1] Non-stationary integrated processes are analyzed in chapter 16.

[2] Model-based or non model-based (see further below); trend, cycle, seasonal adjustment or any combination thereof.

where $x_1, ..., x_T$ is the observed data sample. Inserting optimal forecasts and backcasts of the data, based on our AR(1)-specification, we obtain

$$\begin{aligned}
\hat{y}_T &= \sum_{k=-\infty}^{-1} \gamma_k \hat{x}_{T-k} + \sum_{k=0}^{T-1} \gamma_k x_{T-k} + \sum_{k=T}^{\infty} \gamma_k \hat{x}_{T-k} \\
&= \sum_{k=-\infty}^{-1} \gamma_k a_1^{|k|} x_T + \sum_{k=0}^{T-1} \gamma_k x_{T-k} + \sum_{k=T}^{\infty} \gamma_k a_1^{k-(T-1)} x_1 \qquad (9.2) \\
&= \left(\sum_{k=-\infty}^{0} \gamma_k a_1^{|k|} \right) x_T + \sum_{k=1}^{T-2} \gamma_k x_{T-k} + \left(\sum_{k=T-1}^{\infty} \gamma_k a_1^{k-(T-1)} \right) x_1
\end{aligned}$$

We infer that the coefficients $b_j, j = 0, ..., T-1$ of the real-time filter $\hat{\Gamma}(\cdot)$ are obtained as

$$b_j = \begin{cases} \sum_{k=-\infty}^{0} \gamma_k a_1^{|k|} , & j = 0 \\ \gamma_j , & j = 1, ..., T-2 \\ \sum_{k=T-1}^{\infty} \gamma_k a_1^{k-(T-1)} , & j = T-1 \end{cases} \qquad (9.3)$$

Remarks

- Frequently, backcasts can be neglected because the filter coefficients γ_k decay sufficiently rapidly. For ease of exposition we now ignore the rightmost term in 9.2 (the simplification is mainly due to convenience and hence does not preclude generality).
- The proposed solution 9.3 assumes $L = T$. Smaller L are analyzed below.

We briefly compute the resulting filter coefficients, which will then be replicated by the DFA. We use the three AR(1)-processes introduced in section 4.2.5: $a_1 = 0.9, 0.1, -0.9$:

```
> # Sample length
> len<-120
> # Specify lowpass target
> cutoff<-pi/12
> # Order of approximation of bi-infinite target by finite symmetric filter
> ord<-120
> # Compute coefficients gamma of symmetric filter
> gamma_k<-c(cutoff/pi,(1/pi)*sin(cutoff*1:ord)/(1:ord))
> # AR(1)-coefficient
> a_vec<-c(0.9,0.1,-0.9)
> # Initialize matrix of best MSE coefficients
> gamma_mba_rt<-matrix(ncol=length(a_vec),nrow=ord)
> for (k in 1:length(a_vec))
+ {
+ # Model-based filter: backcasts are ignored
+   gamma_0<-gamma_k%*%a_vec[k]^(0:ord)
+   gamma_mba_rt[,k]<-c(gamma_0,gamma_k[2:ord])
+ }
```

9.2.3 Frequency-Domain

An alternative solution of the mean-square filter approximation problem is obtained in the frequency domain

$$\int_{-\pi}^{\pi} |\Gamma(\omega) - \hat{\Gamma}(\omega)|^2 \frac{1}{|1 - a_1 \exp(-i\omega)|^2} d\omega \rightarrow \min_{\mathbf{b}}$$

where $\frac{1}{|1 - a_1 \exp(-i\omega)|^2}$ is the spectral density of the AR(1)-DGP. The DFA MSE-criterion 4.27 stands for a particular empirical implementation of this theoretical criterion, whereby the continuous integral is replaced by a discrete sum and the periodogram is substituted for the true spectral density. In order to make the frequency-domain criterion operable we discretize the integral

$$\sum_{k=-M}^{M} |\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)|^2 \frac{1}{|1 - a_1 \exp(-i\omega_k)|^2} \rightarrow \min_{\mathbf{b}} \tag{9.4}$$

where $\omega_k = k\pi/M, k = -M, ..., M,\ M > 0$ is a discrete frequency-grid (the normalization constant can be omitted for optimization). The denseness of the frequency-grid, as specified by M, determines the tightness of the approximation of the exact solution by the discrete estimate[3]. In order to keep things simple we select $L = T = 120$, as above, and $M = 10 * L = 1200$.

```
> # Frequency resolution: higher means tighter approximation
> #   but computationally more intensive
> M<-10*len
> # Filter length
> L<-len
> # MSE-filter
> lambda<-0
> eta<-0
> # Real-time design
> Lag<-0
> # Unconstrained filter
> i1<-F
> i2<-F
> # Target in frequency-domain
> Gamma<-(0:M)<as.integer(cutoff*M/pi)+1
> omega_k<-(0:M)*pi/M
> # Coefficients of optimal MSE filter
> b_rt<-matrix(ncol=length(a_vec),nrow=ord)
> for (k in 1:length(a_vec))
+ {
+ # true spectral density
+   weight_func<-1/(abs(1-a_vec[k]*exp(1.i*omega_k))^2*2*pi)
```

[3]The quality of the approximation could be improved, to some extent, by allowing for a non-equidistant frequency-grid, modulating the density of the grid according to the steepness of the spectrum.

```
+ # Estimate filter coefficients
+   dfa_ar1<-dfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,i1,i2)
+   b_rt[,k]<-dfa_ar1$b
+ }
> benchmark<-cbind(b_rt,gamma_mba_rt)
> dimnames(benchmark)[[2]]<-c(paste("MBA ",a_vec,sep=""),paste("DFA ",
+                                        a_vec,sep=""))
> dimnames(benchmark)[[1]]<-paste("lag ",0:(L-1),sep="")
```

We now briefly compare the previous time-domain (MBA) estimate to the latter frequency-domain (DFA) estimate:

```
> head(benchmark)
```

```
         MBA 0.9    MBA 0.1   MBA -0.9    DFA 0.9    DFA 0.1   DFA -0.9
lag 0 0.41976653 0.09197094 0.04364398 0.42060902 0.09245016 0.04387223
lag 1 0.08212978 0.08200236 0.08198302 0.08238466 0.08238466 0.08238466
lag 2 0.07906881 0.07919623 0.07921557 0.07957747 0.07957747 0.07957747
lag 3 0.07487898 0.07475157 0.07473222 0.07502636 0.07502636 0.07502636
lag 4 0.06855951 0.06868693 0.06870627 0.06891611 0.06891611 0.06891611
lag 5 0.06153166 0.06140425 0.06138490 0.06149275 0.06149275 0.06149275
```

For each process, both estimates are close, as expected. The approximation could be tightened arbitrarily by selecting a denser frequency-grid (for the DFA) and by accounting for backcasts in the MBA. The DFA replicates the MBA, if the spectral density of the latter is used in the former. Since the process is assumed to be known, neither filter depends on data. In practice, a model must be identified and unknown parameters must be estimated. Of course, these intermediary steps do not affect the replication of the MBA by the DFA: just plug the estimated ARMA-parameters $\hat{b}_1, ..., \hat{b}_{\hat{q}}, \hat{a}_1, ..., \hat{a}_{\hat{p}}$ of the former in the empirical spectral-density of the latter:

$$\hat{h}(\omega_k) := \left| \frac{1 + \sum_{j=1}^{\hat{q}} \hat{b}_j \exp(-ij\omega_k)}{1 - \sum_{j=1}^{\hat{p}} \hat{a}_j \exp(-ij\omega_k)} \right|^2$$

and plug the estimate into 9.4. An extension to integrated processes is provided in chapter 16.

9.2.4 Forecasting

Since our target specification 9.1 is general, the forecast-problem is actually nested in the previous Signal-Extraction formalism: replace the (ideal trend) target by the anticipative $h-$steps ahead allpass filter

$$\Gamma_h(\omega) = \exp(ih\omega) \ , \ h > 0$$

or, equivalently, set $\gamma_k = \begin{cases} 1 & , \ k = -h \\ 0 & , \ \text{otherwise} \end{cases}$ in 9.1.

9.3 Customization of MBA by DFA: AR(1)-Process

9.3.1 Known DGP

Having achieved replication of the MBA by the DFA, the next step addresses customization of the MBA by means of the replicating DFA. This natural extension allows to tackle Timeliness and Smoothness of (suitably customized) model-based designs. We first assume knowledge of the true DGP (a_1 is known). Our contenders are the classic MSE-filter $\lambda = \eta = 0$, the balanced customized design $\lambda = 30, \eta = 1$, a strong smoothness filter $\lambda = 0, \eta = 1$ and a 'fast-only' filter $\lambda = 100, \eta = 0$. The filter-length is fixed at $L = 24$ and the denseness of the frequency-grid is preset at $M = T/2 = 60$[4].

```
> # Specify the processes: ar(1) with coefficients -0.9,0.1 and 0.9
> a_vec<-c(0.9,0.1,-0.9)
> # Specify the lambdas
> lambda_vec<-c(0,30,0,100)
> # Specify the fixed eta
> eta_vec<-c(0,1,1,0)
> # Length of model-based filters
> L<-24
> # Length of estimation sample (not used yet since we rely on the true model)
> len<-120
> # Denseness frequency-grid
> M<-len/2
> # cutoff
> cutoff<-pi/12
> # Nowcast
> Lag<-0
> # No filter constraints
> i1<-i2<-F
> # Use model-based spectrum
> mba<-T
```

In the following simulation-run we generate 100 replications of the three AR(1)-processes and we use the true spectral densities i.e we do not estimate the AR(1)-parameter. We derive amplitude and time-shift functions, ATS-components, Peak Correlation and Curvature statistics as well as filter outputs.

```
> # Use true spectral density
> estim_MBA<-F
```

[4]Our selection $L = 24, M = 60$ is a fairly good compromise ensuring statistical accuracy as well as computational speed. Note that $L = 24$ is derived from the selected cutoff $\pi/12$: shorter filters would not be able to eliminate a component with frequency $\pi/12$. Note also that larger filter lengths L might require denser frequency-grids (in order to avoid overfitting at the grid-points) which, in turn, would affect numerical speed.

```
> anzsim<-100
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,
+                  Lag,i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

In order to save space we here emphasize the second process ($a_1 = 0.1$). Results for the other two processes are to be found in the appendix.

1. Amplitude and time-shift functions of the competing designs for the second process ($a_1 = 0.1$) are to be seen in fig.9.1.

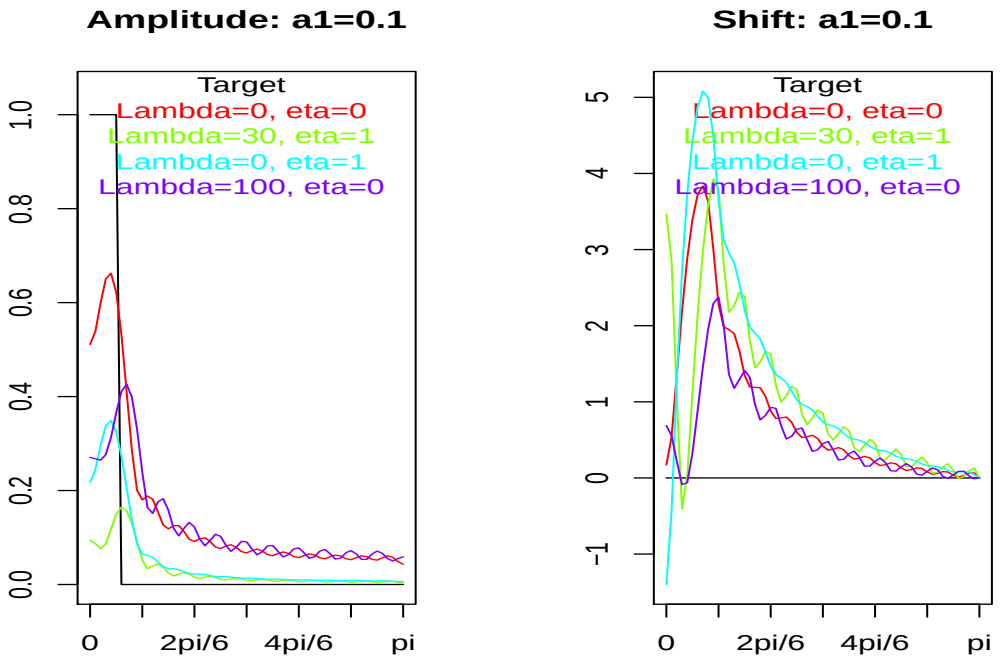


Figure 9.1: Amplitude (left) and time-shift functions (right) of classic MSE model-based filter (red) vs. balanced model-based (green), smooth model-based (cyan) and fast model-based (violet); a1=0.1

2. The ATS-components of the competing designs, for the second process ($a_1 = 0.1$), are summarized in table 9.1.

3. The empirical distributions of Curvature, Peak-Correlation and MSE are plotted in fig.9.2.

	Accuracy	Timeliness	Smoothness	Residual	Total MSE
MBA-MSE	0.020364	0.016004	0.018656	0.000000	0.055025
Lambda=30, eta=1	0.099369	0.000327	0.001835	0.000000	0.101531
Lambda=0, eta=1	0.061509	0.013489	0.003390	0.000000	0.078388
Lambda=100, eta=0	0.061714	0.000064	0.021031	0.000000	0.082809

Table 9.1: ATS-Components: classic model-based vs. customized model-based

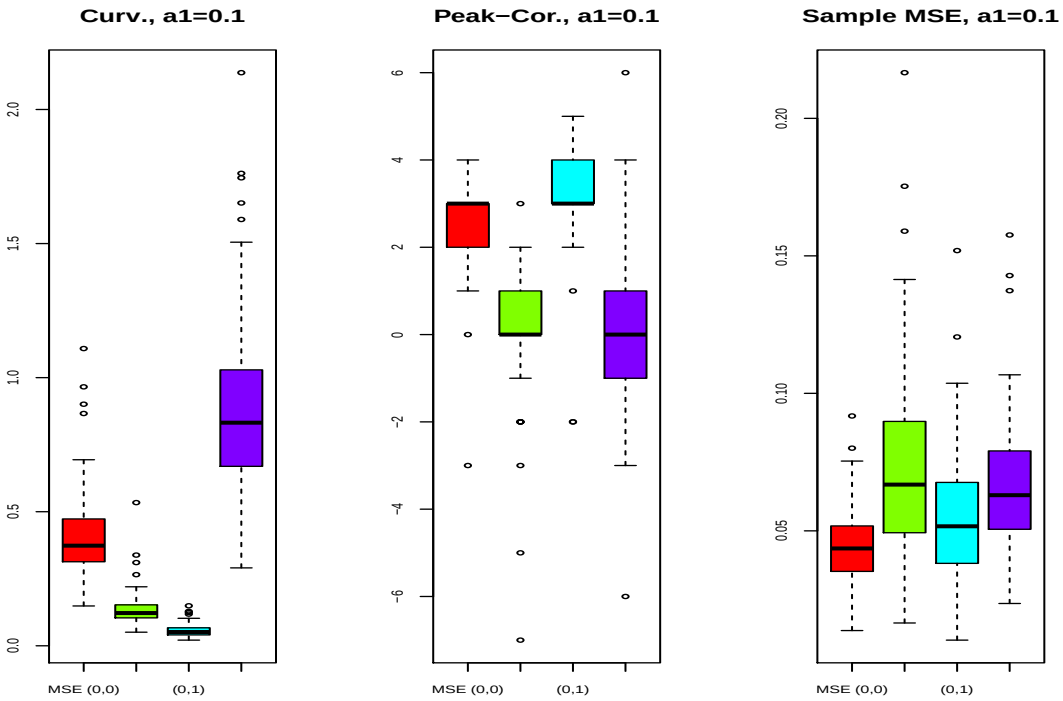


Figure 9.2: Curvature (left), Peak Correlation (middle) and MSE distributions (right) of classic MSE model-based filter (red) vs. balanced model-based (green), smooth model-based (cyan) and fast model-based (violet); a1=0.1

4. A comparison of filter-outputs of classic model-based (red) and balanced model-based (green) is shown in fig.9.3 (we selected the first realization of the process).

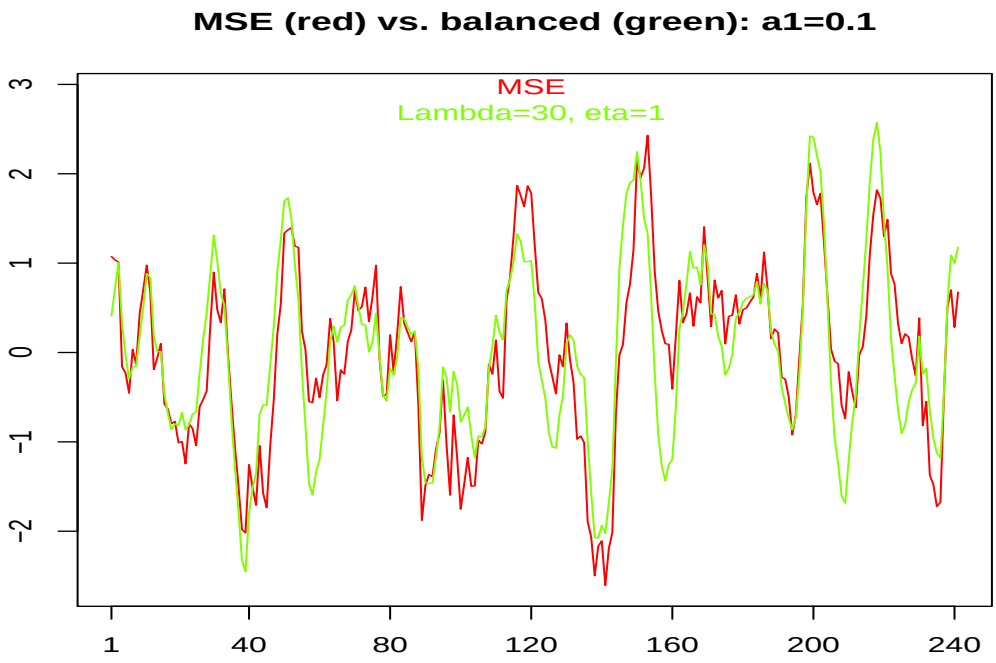


Figure 9.3: Filter-outputs: classic MSE model-based filter (red) vs. balanced customized model-based (green); a1=0.1

Findings

- The time-shifts in fig.9.1 suggest that balanced (green) and fast (violet) filters should be faster than the classic MBA (red), even if the balanced filter has a larger shift in frequency zero[5]; the smooth filter (cyan) is outperformed by the MBA. Amplitude functions are more difficult to interpret because of zero-shrinkage, recall section 7.4.3.

- ATS-components in table 9.1 behave as expected but interpretation is hampered by the fact that the statistics are not scale-invariant.

- The scale-invariant Peak Correlation and Curvature statistics in fig.9.2 reveal more effectively efficiency gains: the balanced model-based design (green) outperforms the classic MBA (red) on both accounts; but it is outperformed in terms of MSE-performances.

- Finally, a comparison of filter outputs in fig.9.3 (first realization of the process) confirms the previous picture: the customized design (green line) is smoother and it tends to anticipate turning points.

[5]The spectral content in frequency zero does not dominate the dynamics of the second process (in contrast to a random-walk, for example).

9.3.2 Empirical AR(1)-Spectrum

We assume that the true model-order – AR(1) – is known (no identification), but a_1 is unknown and must be estimated. The empirical spectral estimate is based on a fitted AR(1)-model and estimation relies on unconditional maximum likelihood, such as implemented in the R-function $arima$[6]; estimates rely on samples of length $T = 120$. In contrast to the previous section we here distinguish in-sample and out-of-sample performances of real-time (nowcast) filters.

```
> # Estimate the AR(1) coefficient
> estim_MBA<-T
> anzsim<-100
> for_sim_obj<-for_sim_out(a_vec,len1,len,cutoff,L,mba,estim_MBA,L_sym,
+               Lag,i1,i2,scaled_ATS,lambda_vec,eta_vec,anzsim,M,dif)
```

As in the previous section we emphasize the second process ($a_1 = 0.1$). In- and out-of-sample distributions of Peak Correlation and Curvature statistics are shown in fig.9.4; corresponding MSE-distributions are to be found in fig.9.5.

[6]Unconditional (full) maximum likelihood is obtained from a state-space model representation.

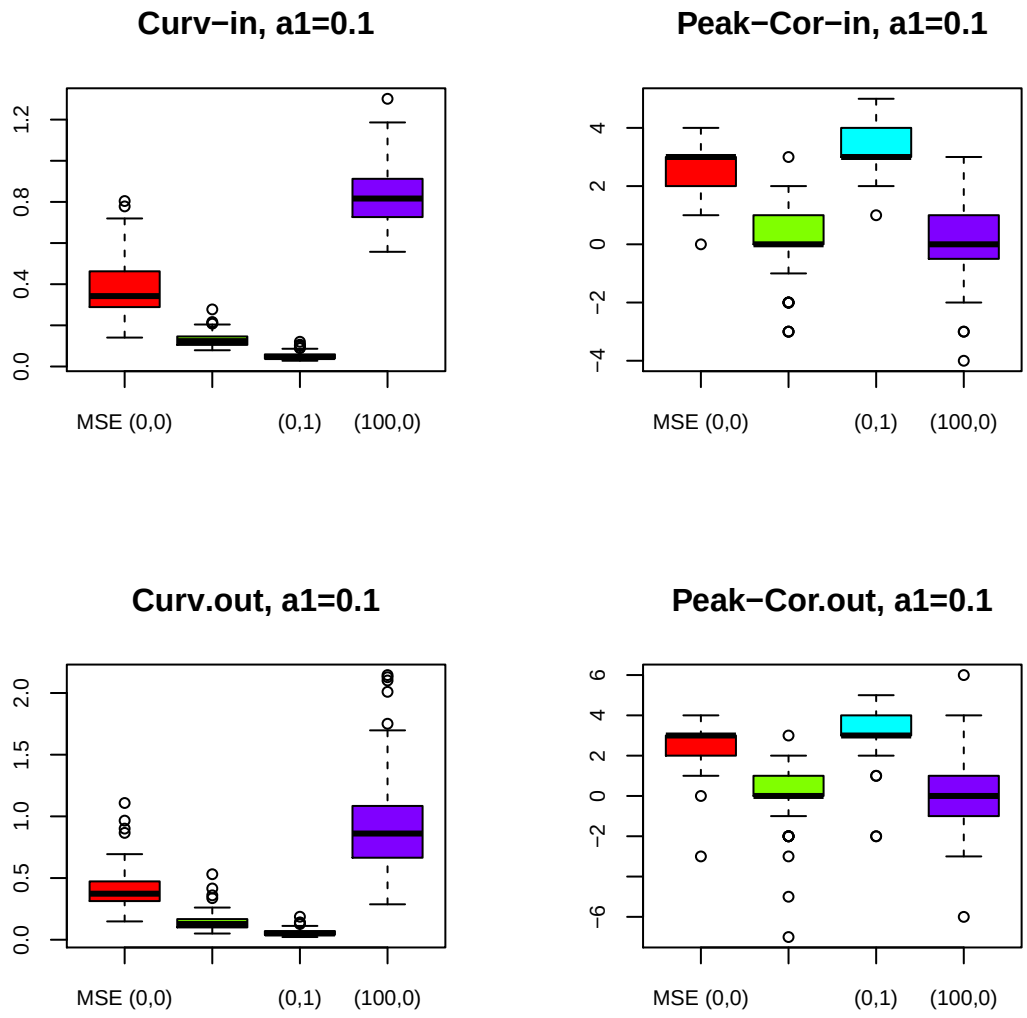


Figure 9.4: Curvature (left) and Peak Correlation (right) in-sample (top) and out-of-sample (bottom): classic MSE model-based filter (red) vs. balanced model-based (green), smooth model-based (cyan) and fast model-based (violet); a1=0.1

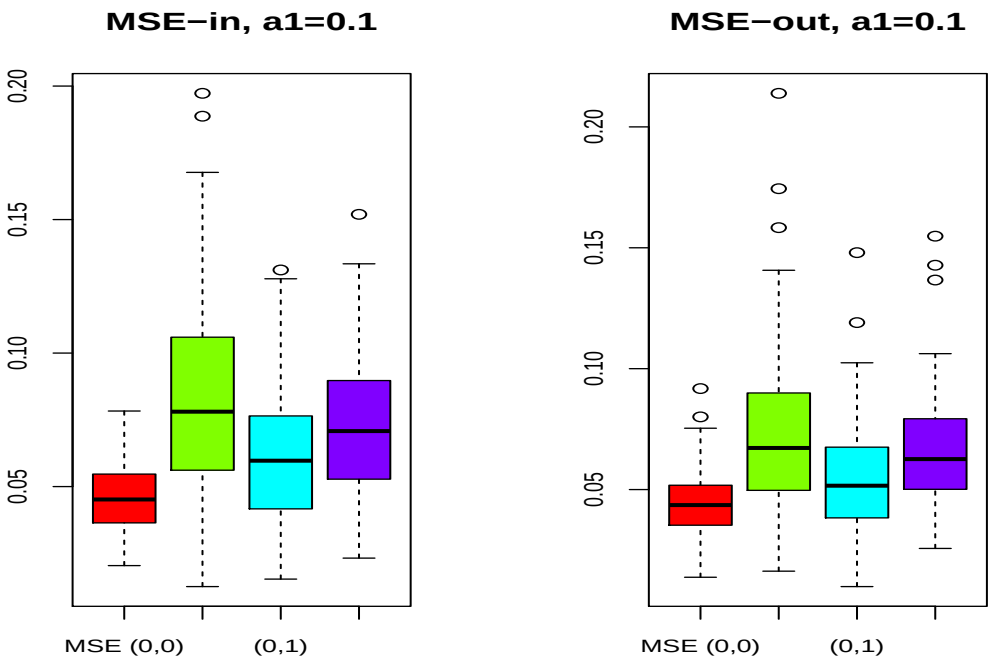


Figure 9.5: MSE in-sample and out-of-sample: classic MSE model-based filter (red) vs. balanced model-based (green), smooth model-based (cyan) and fast model-based (violet); a1=0.1

Findings

- The above results confirm our previous analysis: the balanced model-based design (green) outperforms the MBA in-sample as well as out-of-sample in terms of Curvature and Peak Correlation. The MBA performs best in terms of MSE, as expected.

9.4 Replication of Unobserved-Components (UC-) Models

In contrast to ARIMA-models, which model the entire DGP by a single (reduced-form) equation, an unobserved-components (UC-) model disaggregates the DGP into interesting 'components', typically trend T_t, cycle C_t, seasonality S_t and irregular I_t

$$x_t = T_t + C_t + S_t + I_t$$

Specific sub-models are assigned to the components which could be linked (dependency) or not (orthogonality) and the DGP of the data is obtained by aggregation of the components (structural form). We here review classic trend-cycle models for seasonally-adjusted log-transformed real US-GDP.

9.4.1 The Data: (Log) Real US GDP

US (log) real GDP is plotted in fig.9.6: the shaded areas correspond to recessions as declared by the NBER. The data is downloaded from the Quandl-website, as illustrated in the following piece of code:

```
> # Post WWII data
> start_year<-1947
> start_date=paste(start_year,"-01-01",sep="")
> # Last data point
> end_date<-format(Sys.time(), "%Y-%m-%d")
> end_year<-as.double(substr(end_date,1,4))
> # Load Real GDP
> #Title:               Real Gross Domestic Product, 3 Decimal
> #Series ID:          GDPC96
> #Source:             US. Bureau of Economic Analysis
> #Release:            Gross Domestic Product
> #Seasonal Adjustment: Seasonally Adjusted Annual Rate
> #Frequency:          Quarterly
> #Units:              Billions of Chained 2009 Dollars
> #Date Range:         1947-01-01 to 2014-07-01
> #Last Updated:       2014-11-25 7:56 AM CST
> #Notes:              A Guide to the National Income and
> #                    Product Accounts of the United States
> #                    (NIPA) -
> #   (http://www.bea.gov/national/pdf/nipaguid.pdf)
> if (load_from_quandl)
+ {
+   mydata<-Quandl(c("FRED/GDPC96"),start_date=start_date,
+                  end_date=end_date,type="xts")
+   save(mydata,file=paste(path.dat,"US_GDP.Rdata",sep=""))
+ } else
+ {
+ # Data is included in MDFA-package
+ #  load(file=paste(path.dat,"US_GDP.Rdata",sep=""))
+ }

NULL

> tail(mydata)

            [,1]
2015 Q2 16374.18
2015 Q3 16454.88
```

```
2015 Q4 16490.68
2016 Q1 16524.99
2016 Q2 16583.15
2016 Q3 16712.53
```

```
> lgdp <- ts(100*log(mydata),start=start_year,frequency=4)
> nobs <- length(lgdp)
> # Annualized sharpe of GDP series
> sharpe_GDP<-sqrt(4)*mean(diff(lgdp))/sqrt(var(diff(lgdp)))
```

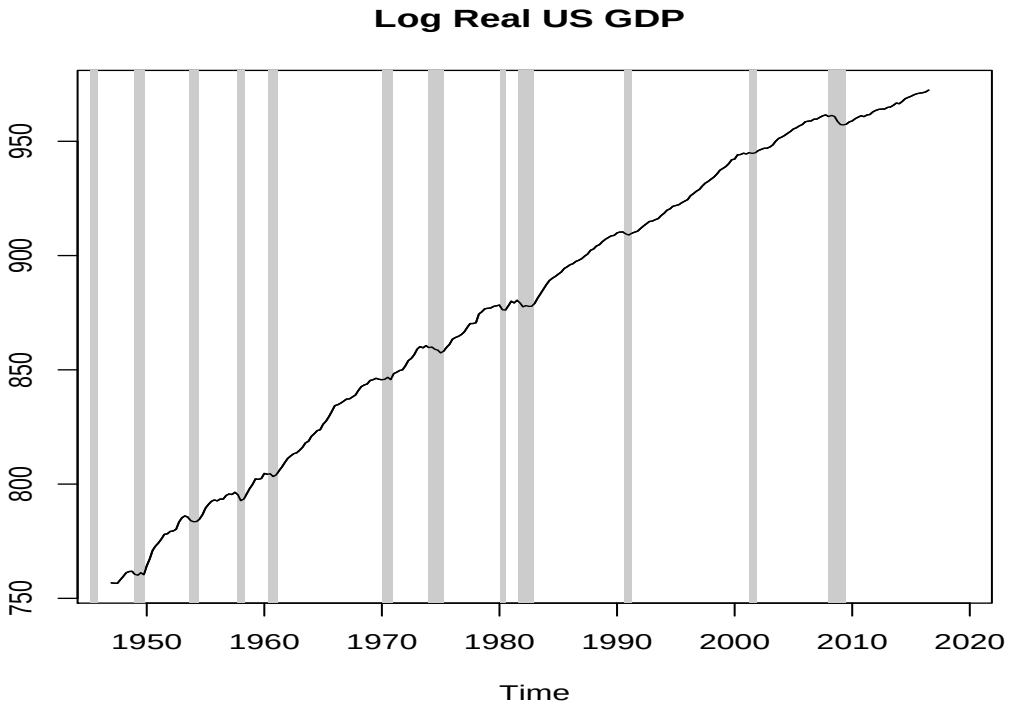


Figure 9.6: Log US Real GDP

Except for the (shaded) recession episodes, the series grows steadily: post-WWII annualized sharpe ratio is 1.64 and the drift is regular but slightly decreasing towards the sample end.

9.4.2 UC-Model

In order to fit the data, Morley, Nelson, and Zivot (2003), MNZ (2003), for short, propose the following Unobserved-Components (UC-) model

$$\left.\begin{array}{rcl} GDP_t & = & (1,1,0)\cdot\mathbf{S}_t + v_t \\ \mathbf{S}_t & = & \begin{pmatrix}\mu\\0\\0\end{pmatrix} + \begin{pmatrix}1&0&0\\0&a_1&1\\0&a_2&0\end{pmatrix}\mathbf{S}_{t-1} + \mathbf{w_t} \end{array}\right\} \tag{9.5}$$

where v_t and $\mathbf{w}_t' = (w_{1t}, w_{2t}, 0)$ are mutually uncorrelated iid random variables with variances σ_v^2, $\mathbf{\Sigma_w}^2 = \begin{pmatrix}\sigma_{w,11}^2&0&0\\0&\sigma_{w,22}^2&0\\0&0&0\end{pmatrix}$ and where μ is a constant drift. $\mathbf{S}_t$ collects trend (first component) and cycle (second component). If $\sigma_v^2 = 0$, as in MNZ (2003), then the data can be modeled by the sum of a random-walk with drift (trend) and a stationary AR(2) (cycle) whereby both processes are mutually independent. Given that the drift looses momentun after the dot-com recession (2001)[7], we complement the above fixed trend-growth model by an additional stochastic drift-equation in the system of state-equations:

$$\left.\begin{array}{rcl} GDP_t & = & (1,0,1,0)\cdot\mathbf{S}_t + v_t \\ \mathbf{S}_t & = & \begin{pmatrix}1&1&0&0\\0&1&0&0\\0&0&a_1&1\\0&0&a_2&0\end{pmatrix}\mathbf{S}_{t-1} + \mathbf{w_t} \end{array}\right\} \tag{9.6}$$

where $\mathbf{w}_t' = (w_{1t}, w_{2t}, w_{3t}, 0)$ and $\mathbf{\Sigma_w}^2 = \begin{pmatrix}\sigma_{w,11}^2&0&0&0\\0&\sigma_{w,22}^2&0&0\\0&0&\sigma_{w,33}^2&0\\0&0&0&0\end{pmatrix}$. The first model is nested in this more general specification: set the second diagonal element $\sigma_{w,22}^2$ of $\mathbf{\Sigma_w}^2$, corresponding to the adaptive drift, to zero and initialize the second element of $\mathbf{S}_0$ (drift) in $t = 0$ by μ. Note that if $\sigma_{w,22}^2 > 0$, then the DGP is integrated of order two (double unit root in frequency zero). Let $\boldsymbol{\theta}' = (a_1, a_2, \sigma_{w,11}^2, \sigma_{w,22}^2, \sigma_{w,33}^2)$ designate the vector of unknown parameters. The latter can be estimated by maximizing the likelihood function, assuming that the noise terms are all Gaussian and mutually uncorrelated.

9.4.3 I(1)-Model MNZ (2003)

We here rely on the R-package *dlm*

```
> library(dlm)
```

and fit model 9.5[8] to the GDP data, assuming three different time spans:

[7] MNZ (2003) use data up to 1998.

[8] More precisely, we fit 9.6 with nesting constraints.

- Jan-1947 to Feb-1998: replicate results in MNZ (2003)
- Jan-1947 to Dec-2007: validate the model on data prior to the great recession
- Jan-1947 to Dec-2014: evaluate impact of great recession on model parameters.

Let us emphasize that the Kalman-filter equations are (heavily) non-linear in the unknown variance parameters and therefore numerical optimization is challenging. The variances are crucial for determining the long-term dynamics (the integration order) of the DGP: if $\sigma^2_{w,22} = \sigma^2_{w,11} = 0$ then the process is stationary (assuming stationarity of the cycle-AR(2)); if $\sigma^2_{w,22} = 0$ and $\sigma^2_{w,11} > 0$ then the process is integrated of order one I(1); if $\sigma^2_{w,22} > 0$ then the process is integrated of order two I(2).

Data: Jan-1947 to Feb-1998

In order to illustrate handling of the *dlm*-package we attempt to replicate results in MNZ (2003).

1. Data:

```
> # Post WWII data
> start_year<-1947
> start_date=paste(start_year,"-01-01",sep="")
> # Data up to Feb-1998 as
> end_date<-"1998-02-01"
> end_year<-as.double(substr(end_date,1,4))
> # Select data prior to end_year
> data_sample<-mydata[paste("/",end_date,sep="")]
> lgdp <- ts(100*log(data_sample),start=start_year,frequency=4)
> nobs <- length(lgdp)
```

2. Estimation: we replicate 9.5 by 9.6, imposing $\sigma^2_{w,22} = 0$. We refer the reader to the *dlm*-vignette for details of model implementation[9]. In order to exclude numerical issues, we used the solution in MNZ (2003) as initial value for our own optimization.

```
> # We specify the model: sigma_{w,22} is vanishing and sigma_v=sqrt(1e-7)
> #   is nearly vanishing (slightly positive is needed for numerical
> #   stability because otherwise
> #   quotients can vanish in the Kalman-Filter).
> # Note also that all variances are parametrized as positive
> #   constants (squares).
> # This parametrization replicates the model in MNZ (2003).
> ssm2 <- function(parm){
+   dlm <- dlmModPoly(2,dV=1e-7,dW=c(parm[4]^2,0)) +
```

[9] We adapted code from Fossati (2013). In particular we parametrized the model in such a way that variances are always positive. Additionally, we can pre-specify and impose the argument of the roots of the AR(2) polynomial. This parameter is frequently (but erroneously) identified with the cycle-frequency, see section 9.4.5 for details.

```
+     dlmModARMA(ar=c(parm[1],parm[2]), ma=NULL, sigma2=parm[3]^2)
+         # get distribution variance of initial state
+         tmp0 <- matrix(c(parm[1],parm[2],1,0),nr=2)
+         tmp1 <- matrix(c(parm[3]^2,0,0,0),nc=1)
+         tmp <- solve(diag(4)-tmp0%x%tmp0)%*%tmp1
+         dlm$C0[3:4,3:4] <- matrix(tmp,nr=2)
+         return( dlm )
+ }
> # Estimate parameters: we use the estimates in
> #   Morley, Nelson and Zivot (2003)
> # for initialization
> fit2_98 <- dlmMLE(y=lgdp,parm=c(1.5303,-.6097,.6199,.6893),build=ssm2,
+                   hessian=T)
```

The optimization affected slightly our initial estimates and therefore the final estimates in table 9.2 do not coincide with those reported in MNZ (2003), table 1. In particular, the

	Criterion Value	AR(1)	AR(2)	Sigma_w1	Sigma_w2
Initial values (MNZ 2003)		1.53	-0.61	0.62	0.69
Final estimates	109.69	1.51	-0.58	0.66	0.62
Standard errors		0.03	0.05	0.05	0.06

Table 9.2: Estimates with standard errors: data from 1947 to 1997

arguments of the complex roots of the AR(2)-polynomial differ

```
> Arg(polyroot(c(1,-1.5303,.6097)))

[1]  0.2007606 -0.2007606

> Arg(polyroot(c(1,-fit2_98$par[1:2])))

[1]  0.1222615 -0.1222615
```

Whereas our estimate corresponds to an 'implied' cycle-length of 12.85 (years[10]), MNZ (2003) find 7.82 (years). Note, however, that our AR(2)-estimates are not significantly different from those in MNZ (2003). A more comprehensive discussion of the topic is provided below as well as in section 9.4.5.

3. Unobserved components, trend and cycle, are plotted in fig.9.7.

[10]The cycle-length in quarters is obtained as $2 * \pi/\phi$ whereby ϕ is the argument of the AR(2)-polynomial. Divide this number by four to obtain a duration in years.

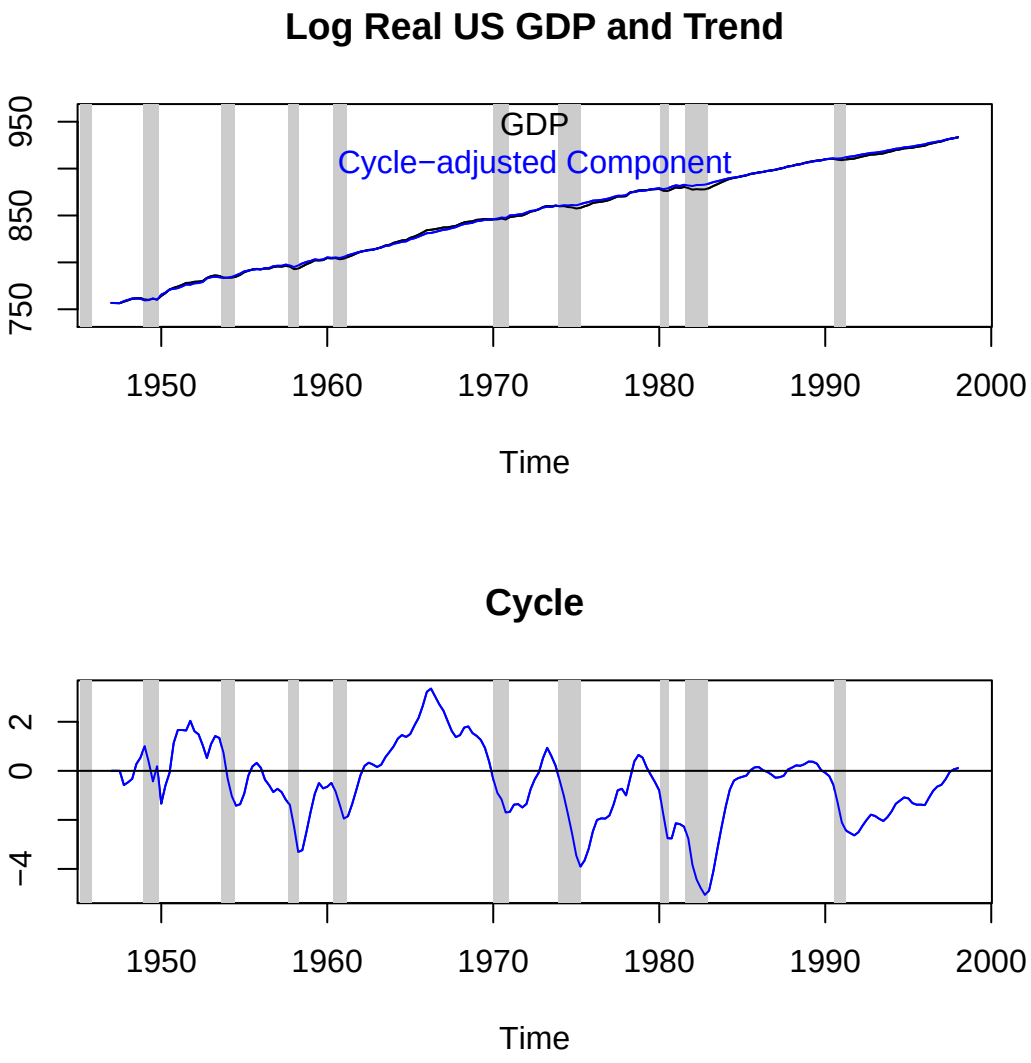


Figure 9.7: US Real GDP: Trend (top) and cycle (bottom): data ends in Feb-1998

The cycle is similar, but not identical, to the estimate in MNZ (2003), fig.1.

Remark

Typically, in applications, the mean duration of the cycle is identified with the argument of the roots of the AR(2)-polynomial: this way we inferred a duration of 12.85 years and, similarly, MNZ (2003) inferred an 'implied period' of 7.82 years, see their table 1. However, as we shall see in section 9.4.5, this alleged link between the AR-roots and the effective cycle-length does not apply, in general, because of interference phenomena: the combined effect of both (complex conjugate) roots of the AR-polynomial precludes such an interpretation. Let us foreclose that the spectral densities of *both* AR(2)-cycles (ours and MNZ (2003)) peak in frequency zero, see fig.9.9

which suggests a periodicity of infinite length: neither 12.85 nor 7.82 are therefore pertinent cycle-periodicities. In the following we distinguish 'implied' and 'effective' cycle lengths: the former is the argument of the root of the AR(2)-polynomial and the latter is the peak-frequency of the resulting spectral density.

9.4.4 Adding Recent Data: Validation, Great Recession

We try to validate the former model by including data up to the onset of the great recession, Dec-2007. Then we assess the impact of the great recession by considering observations up to Nov-2014.

1. Data:

```
> # Data up to 2008
> end_date<-"2007-12-31"

> # Data up to Dec 2014
> end_date<-"2014-11-30"
```

2. Estimation:

```
> fit2_07 <- dlmMLE(y=lgdp_07,parm=c(1.5303,-.6097,sqrt(.6199),
+                              sqrt(.6893)),build=ssm2,hessian=T)

> fit2_14 <- dlmMLE(y=lgdp_14,parm=c(1.5303,-.6097,sqrt(.6199),
+                              sqrt(.6893)),build=ssm2,hessian=T)
```

Estimates are reported in table 9.3. We infer that the model is remarkably stable up to

	Criterion Value	AR(1)	AR(2)	Sigma_w1	Sigma_w2	'Implied' length (years)
1947-1998	109.69	1.51	-0.58	0.66	0.62	12.85
1947-2007	112.68	1.52	-0.59	0.61	0.59	13.71
1947-2014	121.12	1.54	-0.54	0.64	0.56	Inf

Table 9.3: Estimates for three different time spans: 1947-1998, 1947-2007, 1947-2014

the onset of the great recession. Afterwards, the AR(2)-cycle becomes non-stationary (the roots of the AR(2)-polynomial lie on both sides of the unit-circle); accordingly, the 'implied' cycle-length is infinite.

3. cycle estimates are plotted in fig.9.8.

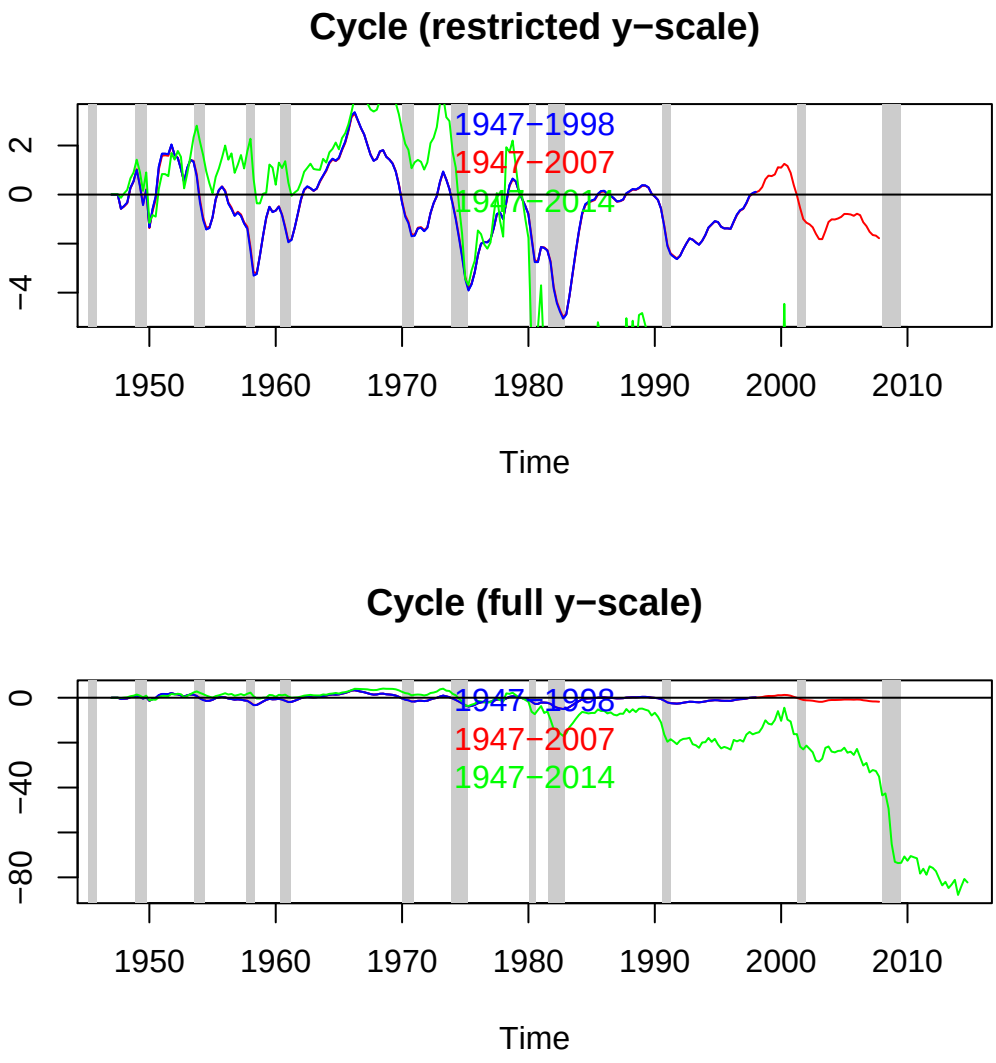


Figure 9.8: Cycles: data up to Feb-1998 (blue), Dec-2007 (red), Dec-2014 (green): restricted scale (top) vs. full scale (bottom)

The 'explosive' dynamics of the non-stationary cycle are eminently visible in the bottom panel.

Findings

- Our two estimates of the AR(2)-cycle, calibrated prior to the great recession, have similar 'implied' lengths (roughly 13 years) and the extracted cycle-components are almost indistinguishable, by eye (top panel).
- By including the great recession in the estimation span, the cycle-component becomes non-stationary.

Summary

- Numerical optimization of UC- (state space) models is tedious.

- Short-term forecast performances (maximum likelihood principle) are not well-suited for resolving and discriminating mid-term (cycle) from long-term (trend) dynamics.

- We were unable to replicate – exactly – the results in MNZ (2003) by the $dlm-$package. In particular the 'implied' cycle-lengths, as specified by the roots of the AR(2)-polynomial, differ noticeably. However, both AR(2)-models are essentially similar in the sense that coefficients are not statistically different; moreover, both spectral densities peak in frequency zero, see fig.9.9.

- Model estimates are fairly stable for data prior to the great recession.

- The great recession jumbles components and the cycle becomes non-stationary.

9.4.5 Implied vs. Effective Cycle-Lengths

In the case of MNZ (2003), the cycle-component is

$$c_t = 1.5303c_{t-1} - 0.6097c_{t-2} + \epsilon_t$$

with complex conjugate roots 0.765-0.156i, 0.765+0.156i. The common (absolute) argument of the roots is 0.201 and the duration of a cycle corresponding to this frequency is $\frac{2\pi}{4 \cdot 0.201} = 7.82$ (years). We argue that this number, the 'implied' cycle-length as derived from the above calculus, is not a pertinent descriptive statistic of c_t, in general. For that purpose we compute the amplitude functions of the two AR(2)-filters specified in table 9.2, see fig.9.9.

```
> K<-1000
> omega_k<-pi*(0:K)/K
> trffkt_ar2_MNZ<-1/abs(1-1.5303*exp(1.i*omega_k)+0.6097*exp(1.i*2*omega_k))^2
> trffkt_ar2_98<-1/abs(1-fit2_98$par[1]*exp(1.i*omega_k)-fit2_98$par[2]*
+                          exp(1.i*2*omega_k))^2
```

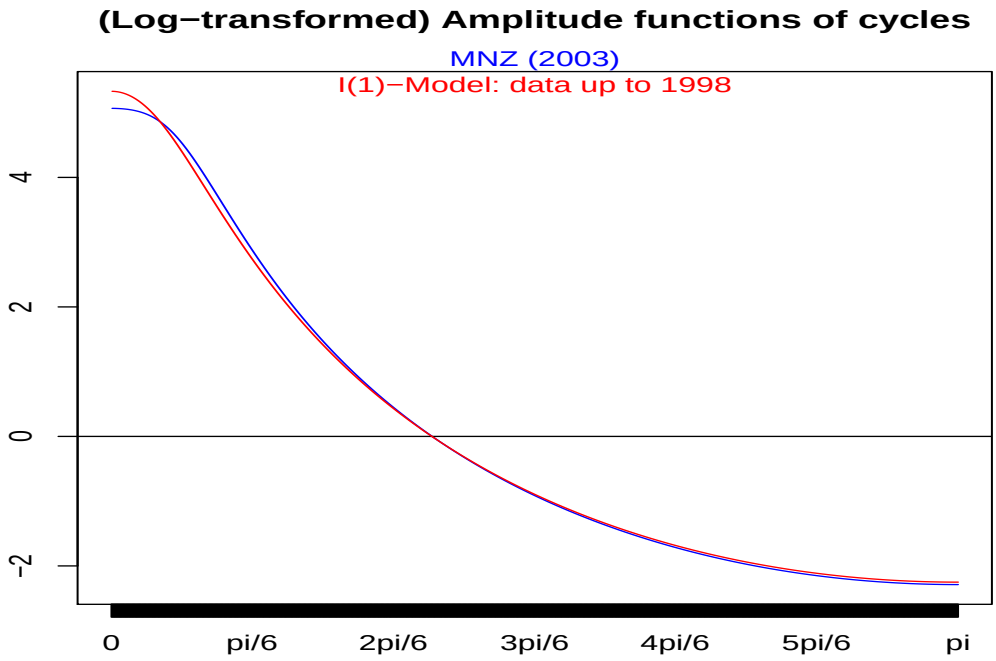


Figure 9.9: (Log-transformed) Amplitude functions of AR(2)-cycles with 'implied' cycle-lengths of 8 (blue) and 13 (red) years

Both amplitude functions peak in frequency zero[11]: there are no peaks at the alleged 'cycle-frequencies' of 8 and 13 years. Accordingly, neither AR(2)-filter or AR(2)-model generates 'cycles'.

9.4.6 I(2)-Models: Unconstrained vs. Constrained Cycle-Frequency

We extend the previous results by fitting model 9.6 to the data. Since unconstrained optimization of the likelihood for the general model is an even more challenging task, we here propose four alternative designs:

1. An unconstrained design: 'implied' cycle-length and innovation variances are determined freely.
2. The same model but we impose an 'implied' cycle-length in accordance with the mean duration of business-cycles which is 5.58 years (from 1947 to 2014). To make things simple we impose an implied length of 6 years (24 quarters).
3. Impose a vanishing level-innovation $\sigma^2_{w,11} = 0$.
4. Impose both constraints: 'implied' cycle-length of 6 years and $\sigma^2_{w,11} = 0$.

We consider data up to the onset of the great recession only (because the model is not robust).

[11] The two complex conjugate roots interfer.

Generate Empirical Results

1. Read the data:

```
> # Data up to great recession
> end_date<-"2007-12-31"
```

2. Estimate constrained and unconstrained I(2)-models.

```
> source(file=paste(path.pgm,"state_space_trend_cycle_gdp.r",sep=""))
```

3. Estimates and criterion values are to be found in table 9.4. The columns s_11, s_22, s_33

	Neg.log-lik	AIC	Duration	s_33	s_11	s_22
Unconstrained	112.15	122.15	13.78	0.63	0.57	0.01
Length=6	112.61	120.61	6.00	0.57	0.61	0.01
s_11=0	115.09	123.09	Inf	0.91	0.00	0.00
s_11=0, length=6	119.22	125.22	6.00	0.88	0.00	0.02

Table 9.4: Estimates of I(2)-models: data from 1947 to the onset of the great recession (Dec-2007)

refer to the unknown innovation variances. The AIC-column is obtained by adding twice the number of estimated parameters to the first column (negative log-likelihood): as an example we estimate five parameters in the unconstrained model (first row) which leads to an AIC of $112.146 + 2 * 5 = 122.146$. The resulting information criterion is 'informative' about the added-value (pertinence) of freely-determined parameters or of imposed constraints (it is not a formal statistical test, though).

4. Extracted cycle- and drift-components are plotted in fig.9.10

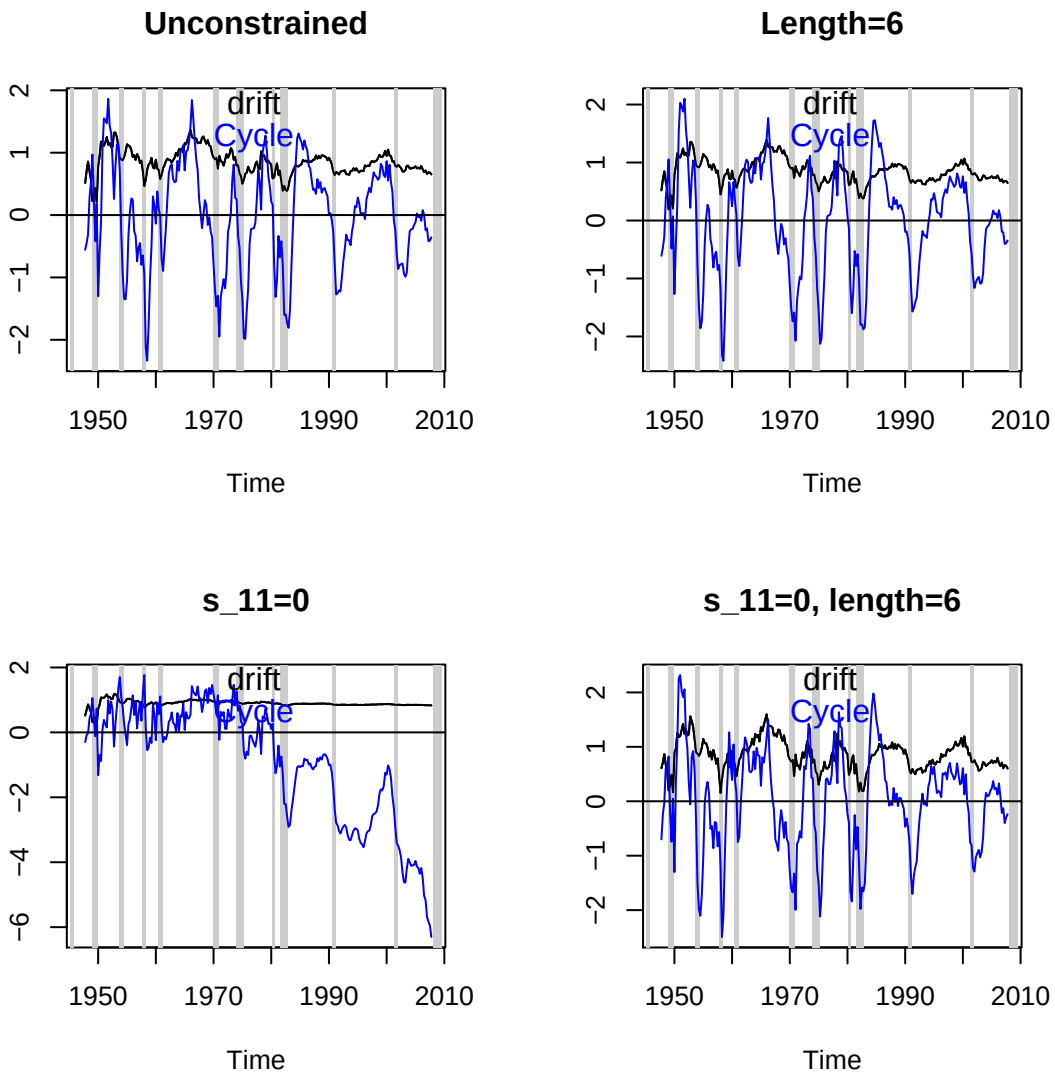


Figure 9.10: Cycles and drifts of I(2)-models

Analysis

- Optimization is tedious!
- Imposing both constraints – mean cycle-length and $\sigma^2_{w,11} = 0$ – affects the negative log-likelihood (the criterion value) noticeably: the resulting model would be rejected on ground of classical information criteria.
- The second model, with freely determined innovation variances and imposed cycle-length, fares best according to information criteria. It also (slightly) outperforms the previous I(1)-model, see table 9.3. Finally, the spectral density of the associated cycle does not peak in zero, see section 9.4.8.

- The stationary cycles in fig.9.10 (top and bottom-right panels) look similar. Counting-in the I(1)-model in the previous section, this observation may lead one to conclude that neither the integration-order nor the 'implied' cycle-length are clearly identified by the data. To some extent, the final decision rests up to the analyst's preference(s).

- Fig.9.10 suggests that drift (black line) and cycle (blue line) correlate, against the fact that the model assumes orthogonality of components.

We now return to the main topic and propose to replicate the estimated Trend-Cycle models by DFA.

9.4.7 Replicating the I(1)-Model by DFA

We rely on the I(1)-model estimated in section 9.4.3, as based on data up to Dec-2007: $\boldsymbol{\theta}$ = (1.52,-0.585,0.606,0.594).

Detrend the Data

Since the model assumes a constant trend with slope μ =0.85 we first detrend the data, see fig.9.11.

```
> drift<-mod2f_07$m[nrow(mod2f_07$m),2]
> trend<-ts(drift*(1:length(lgdp_07)),start=start_year,frequency=4)
> detrended<-lgdp_07-trend
> detrended<-detrended-mean(detrended)
```

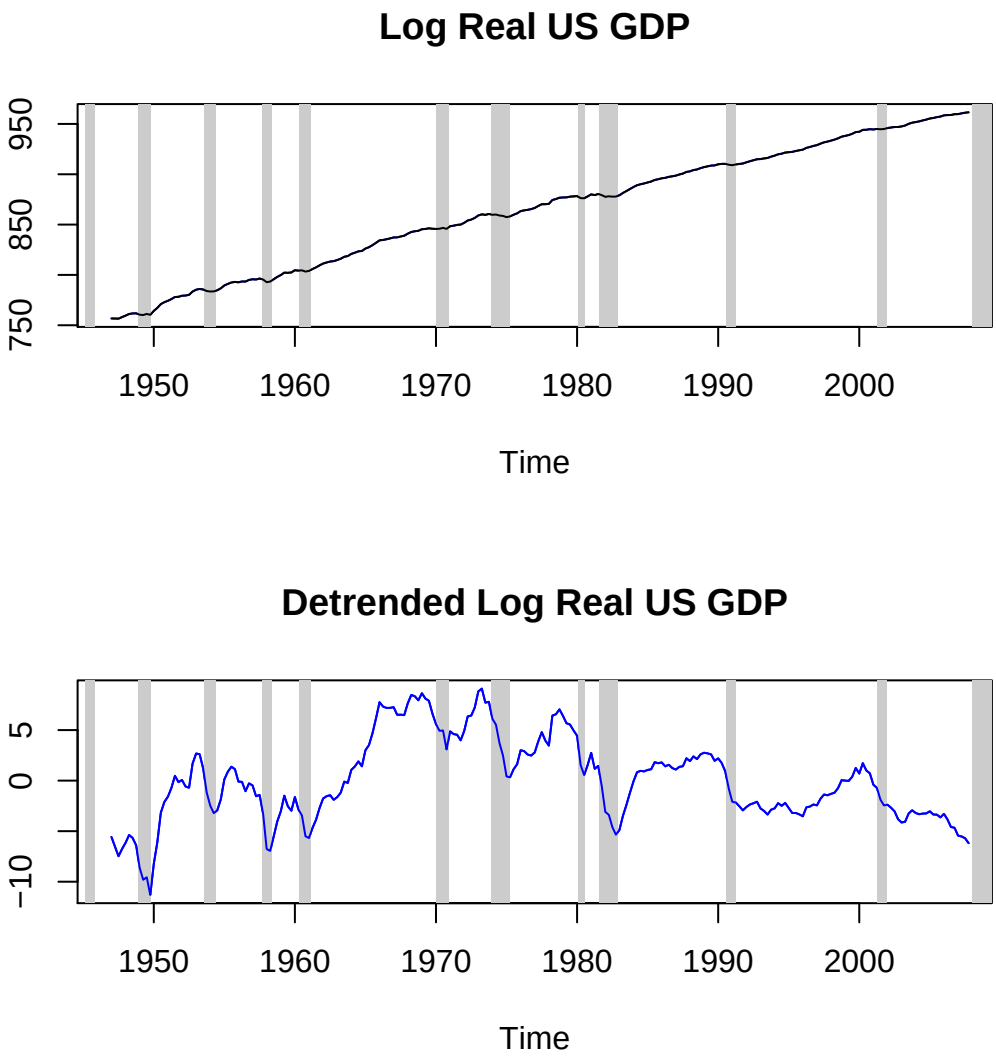


Figure 9.11: GDP (top) and trend-adjusted GDP (bottom)

Centering the detrended series about zero, as we did in the bottom panel, is facultative and does not affect results.

DGP and Replication

The DGP of the detrended series in fig.9.11 is (assumed to be) the sum of a random-walk (without drift) and of a stationary stochastic cycle. Since the latter two components are independent, by assumption, the (pseudo-) spectral density of the DGP is obtained by summing-up both compo-

nents

$$\begin{aligned} h_{cycle}(\omega) &= \frac{\sigma^2_{w,22}}{|1 - a_1 \exp(i\omega) - a_2 \exp(i2\omega)|^2} \\ h_{trend}(\omega) &= \frac{\sigma^2_{w,11}}{|1 - \exp(i\omega)|^2} \\ h_{Detrended\ GDP}(\omega) &= h_{cycle}(\omega) + h_{trend}(\omega) \end{aligned}$$

Specifically

$$h_{Detrended\ GDP}(\omega) := \frac{0.37}{|1 - \exp(-i\omega)|^2} + \frac{0.35}{|1 - 1.52\exp(-i\omega) + 0.59\exp(-i2\omega)|^2}$$

see fig.9.12 (the singularity in frequency zero has been skipped).

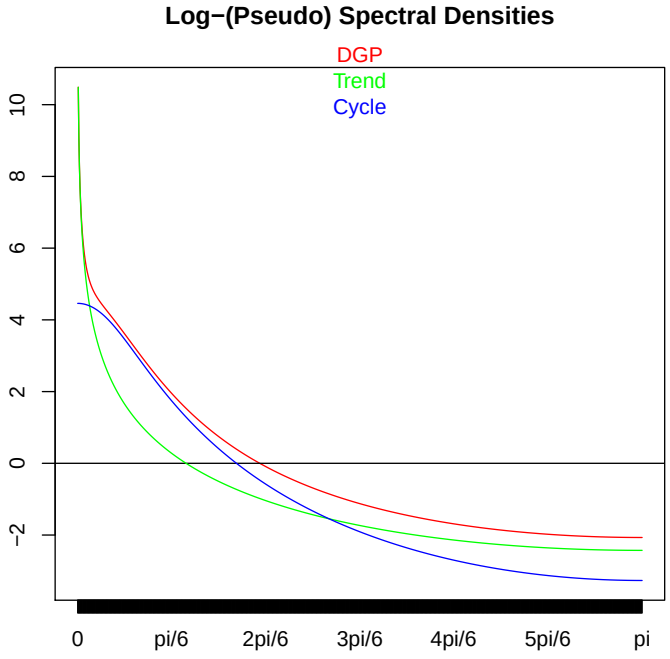


Figure 9.12: Log (pseudo) spectral densities of DGP (red), trend (green) and cycle (blue)

The pseudo-spectral density of the DGP (red line) can be plugged into 4.27 but the unit-root requires a first-order restriction, $i1 = T$, recall section 6.2[12]. In order to focus on the interesting business-cycle we address a bandpass target.

[12]See chapter 16 for a formal treatment.

Bandpass Target

Model 9.6 and its nested variant 9.5 generate model-based (symmetric) target filters. As an alternative, we here rely on a classic '2-10 years' business-cycle design

$$\Gamma(\omega) \quad = \quad \begin{cases} 1 & \frac{\pi}{20} \leq |\omega| \leq \frac{\pi}{4} \\ 0 & \text{otherwise} \end{cases} \tag{9.7}$$

where the cutoff-frequencies refer to quarters. This way, our research priorities are matched explicitly by the target specification.

Next, we compare outputs of the previous I(1)- and I(2)-models with the new (real-time) bandpass design, assuming $L = 100$. Since the DGP is assumed to be integrated of order one, we impose a simple level constraint $i1 = T$, see section 6.2.2.

1. Specify the filter design: cutoff, restriction ($i1 = T$), MSE-design ($\lambda = \eta = 0$).

```
> cutoff_len_upper<-4
> cutoff_len_lower<-20
> cutoff_upper<-pi/cutoff_len_upper
> L<-100
> # Spectrum: MDFA requires DFT i.e. square-root of density
> # The design is univariate i.e. input and output series are synchronized.
> # Therefore we can rely on absolute values (no phase information
> #   is required).
> weight_func<-cbind(sqrt(trffkt_GDP),sqrt(trffkt_GDP))
> # Ignore singularity in frequency zero
> weight_func[1,]<-0
> # Target
> Gamma<-(0:K)<=as.integer(cutoff_upper*K/pi)+1
> Gamma[1:(K/cutoff_len_lower+1)]<-0
> # Restrictions: i1 constraint
> i1<-T
> i2<-F
> weight_constraint<-Gamma[1]
> # MSE-design
> lambda<-eta<-0
```

2. Proceed to estimation:

```
> # Estimate MDFA MSE filter coefficients
> mdfa_obj<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,
+                   i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                   lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                   b0_H0,c_eta,weight_structure,
+                   white_noise,synchronicity,lag_mat,troikaner)
```

Note that we could replicate this object by the simpler context-specific call

```
> # Estimate MDFA MSE filter coefficients
> mdfa_obj<-MDFA_mse_constraint(L,weight_func,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
```

whereby *MDFA_mse_constraint* handles constrained MSE-designs, specifically.

3. Compute amplitude and time-shift functions, see fig.9.13: the time-shift is restricted to the passband only.

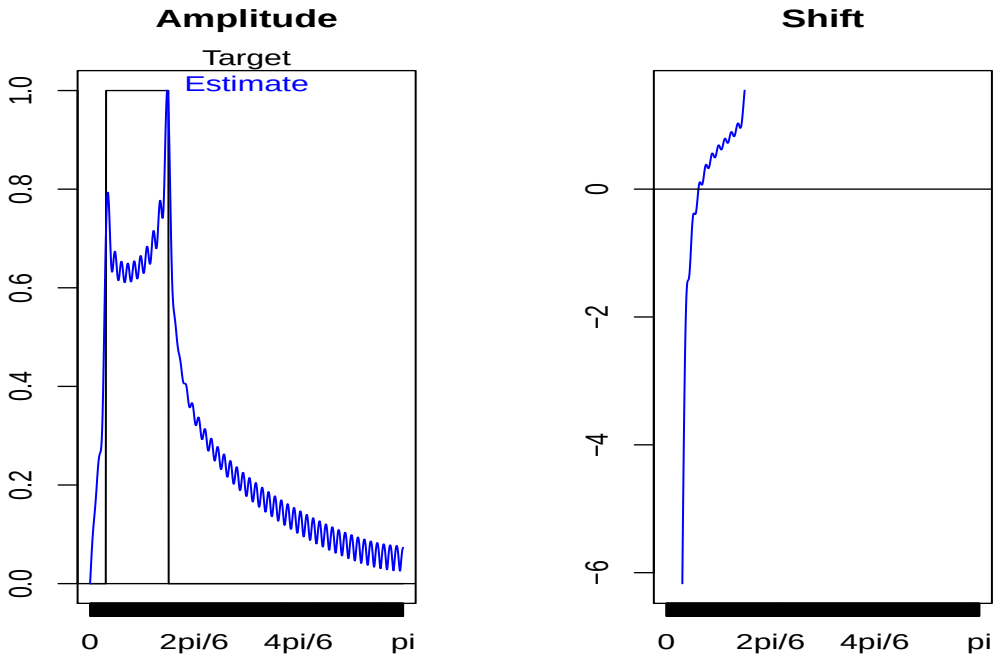


Figure 9.13: Amplitude (left) and time-shift functions of real-time bandpass MSE-design, i1=T

The observed ripples are innocuous: they are due to the discontinuity of the ideal bandpass which cannot be fitted perfectly by a finite-length filter ($L = 100$).

4. Compute the bandpass-MSE filter-output; compare cycles of I(1)-model (fig.9.8), best I(2)-model (fig.9.10, top-right panel) and bandpass-MSE, see fig.9.14.

```
> xf_i1<-rep(NA,length(detrended))
> for (i in L:length(detrended))
+   xf_i1[i]<-t(mdfa_obj$b)%*%(detrended[i:(i-L+1)]-mean(detrended))
```

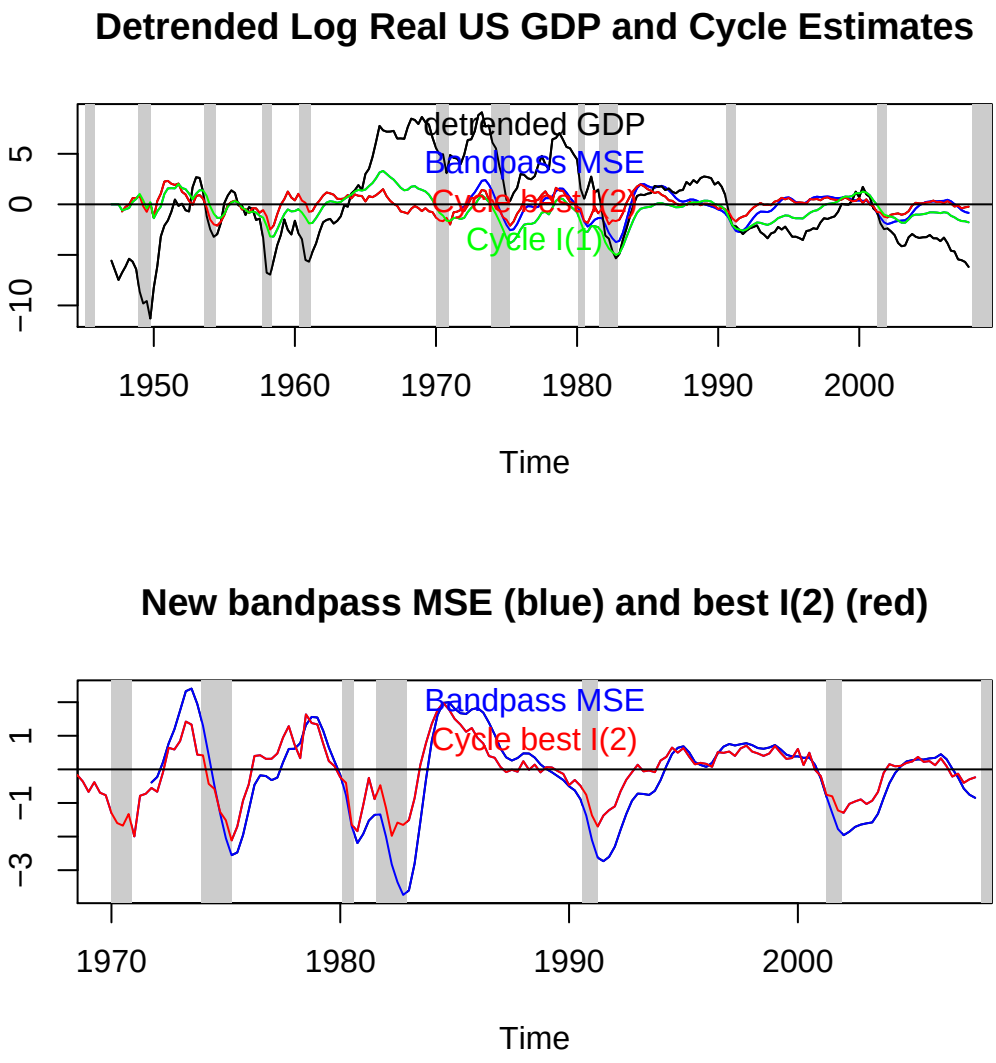


Figure 9.14: Detrended (log-real) US-GDP (black), new bandpass MSE (blue), best I(2)-model (red) and I(1)-model (green)

The new MSE-bandpass (blue line) fares well: it tracks expansions and recessions similarly to the best I(2)-model (red line). The former is a bit smoother and slightly delayed when compared to the latter: further customization could address both issues, at once.

9.4.8 Replicating the General I(2)-Model by DFA

DGP and Pseudo-Spectrum

If $\sigma^2_{w,22} > 0$ in the general model 9.6, then the DGP has a double unit-root in frequency zero. Its pseudo-spectrum is obtained by summing-up the spectra corresponding to the three orthogonal

innovation processes

$$\begin{aligned} h_{cycle}(\omega) &= \frac{\sigma^2_{w,33}}{|1 - a_1 \exp(i\omega) - a_2 \exp(i2\omega)|^2} \\ h_{level}(\omega) &= \frac{\sigma^2_{w,11}}{|1 - \exp(i\omega)|^2} \\ h_{drift}(\omega) &= \frac{\sigma^2_{w,22}}{|1 - \exp(i\omega)|^4} \\ h_{DGP}(\omega) &= h_{cycle}(\omega) + h_{level}(\omega) + h_{drift}(\omega) \end{aligned}$$

For the best I(2)-model (top right panel in fig.9.10) we obtain

$$h_{DGP}(\omega) = \frac{0.33}{|1 - 1.53 \exp(i\omega) + 0.63 \exp(i2\omega)|^2} + \frac{0.37}{|1 - \exp(i\omega)|^2} + \frac{0.000125}{|1 - \exp(i\omega)|^4}$$

see fig.9.15.

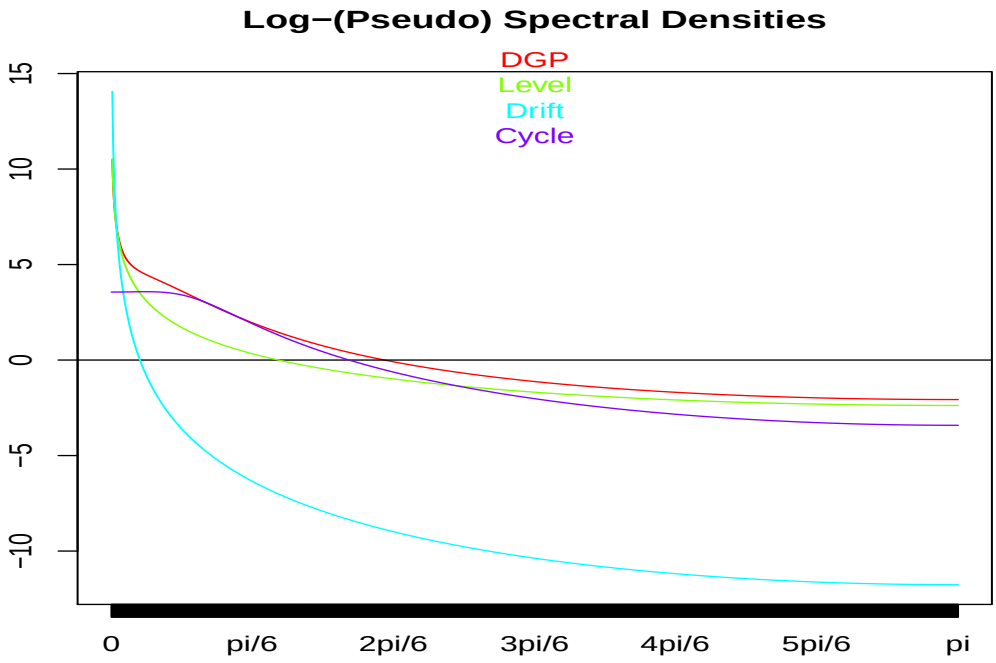


Figure 9.15: Log (pseudo) spectral densities of DGP (red), level (green), drift (cyan) and cycle (violet)

Note that the cycle-spectrum (violet) peaks in $\frac{\pi}{24.39}$ which corresponds to an effective (mean) cycle-length of 12.2 years (we imposed an 'implied' length of 6 years); the peak is very flat, though. The drift (cyan) is dominated by the sum of the cycle (violet) and of the level (green) on most of the frequency-band, except in a very narrow band centered about zero (the drift is slowly changing i.e. $\sigma_{w,22}$ is very small). As a result, the pseudo-spectral density of the I(1)-model in the

previous section (fig.9.12) and of the current I(2)-model look similar. Therefore, we expect that the resulting cycle-estimates should be similar, too.
The MBA can be replicated by the DFA by plugging $h_{DGP}(\omega)$ into 4.27. Since the process is integrated of order two, we have to impose first *and* second-order filter constraints, see section 6.2.

Ideal Bandpass

As in the previous section 9.4.7 we here target an ideal 2-10 years bandpass. In contrast to the previous section, however, the data does not need to be detrended: the drift is stochastic and the DGP is an I(2) process. Another distinguishing feature is that we propose an unconstrained design based on the following MSE-criterion

$$\frac{2\pi}{T}\sum_{k=-M}^{M}\left|\Gamma(\omega_k)-\hat{\Gamma}(\omega_k)(1-\exp(-i\omega_k))^2\right|^2 h_{GDP}(\omega_k)\rightarrow\min_{\mathbf{b}}$$

The composite filter $\hat{\Gamma}(\omega_k)(1-\exp(-i\omega_k))^2$ consists of a double difference filter, which transforms an I(2)-process into a stationary time series, and an unconstrained $\hat{\Gamma}(\omega_k)$ which 'undistords' the double difference such that the composite design matches the bandpass target $\Gamma(\cdot)$ for $\omega_k > 0$. The composite filter satisfies the required I(2)-constraint, by construction, and irrespective of the finite MA-filter $\hat{\Gamma}(\cdot)$, and therefore the above criterion is well defined (the singularity in frequency zero is cancelled[13]). For convenience we rewrite the criterion as follows

$$\frac{2\pi}{T}\sum_{k=-M}^{M}\left|\Gamma(\omega_k)\sqrt{h_{GDP}(\omega_k)}-\hat{\Gamma}(\omega_k)\left[\sqrt{h_{GDP}(\omega_k)}(1-\exp(-i\omega_k))^2\right]\right|^2\rightarrow\min_{\mathbf{b}}$$

The target variable relies on original data in levels; but the explanatory variable relies on second-order differences, as emphasized by the bracketed (square-root) spectral density[14].

1. Specify the filter design and feed the (pseudo-) spectral density to MDFA.

```
> cutoff_len_upper<-4
> cutoff_len_lower<-20
> cutoff_upper<-pi/cutoff_len_upper
> L<-100
> # Spectrum: the second-order difference filter is applied to the
> # explanatory variable (second column)
> #   Important: the relative phase information between target (data in level)
> #   and explanatory variable (differenced data) is required.
> #   Therefore one must supply the transferfunction (not the amplitude)
> #   of the differenced filter!
> weight_func<-cbind(sqrt(trffkt_i2_GDP),sqrt(trffkt_i2_GDP)*(1-exp(1.i*omega_k))^2)
> # Ignore singularity in frequency zero
```

[13]Note that the ideal bandpass target $\Gamma(\omega_k)$ has a zero of infinite order in frequency zero.
[14]Note that $\sqrt{h_{GDP}(\omega_k)}(1-\exp(-i\omega_k))^2$ is complex-valued: the second-order differences affect the phase relative to the target.

```
> weight_func[1,]<-0
> # Target
> Gamma<-(0:K)<=as.integer(cutoff_upper*K/pi)+1
> Gamma[1:(K/cutoff_len_lower+1)]<-0
> # Unconstrained design: the zero of order 2 is already obtained
> #   by the double difference filter
> i1<-F
> i2<-F
> # MSE-design
> lambda<-eta<-0
```

2. estimation:

```
> # Estimate MDFA MSE filter coefficients
> mdfa_obj<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,
+               eta,cutoff,i1,i2,weight_constraint,lambda_cross,
+               lambda_decay,lambda_smooth,lin_eta,shift_constraint,
+               grand_mean,b0_H0,c_eta,
+               weight_structure,white_noise,synchronicity,lag_mat,troikaner)
```

Note that this object could be replicated by the simpler context-specific call

```
> mdfa_obj<-MDFA_mse(L,weight_func,Lag,Gamma)$mdfa_obj
```

where $MDFA\_mse$ can handle the simplest (unconstrained) MSE case.

3. Compute amplitude and time-shift functions of the *composite* filter, see fig.9.16: the time-shift is restricted to the passband only.

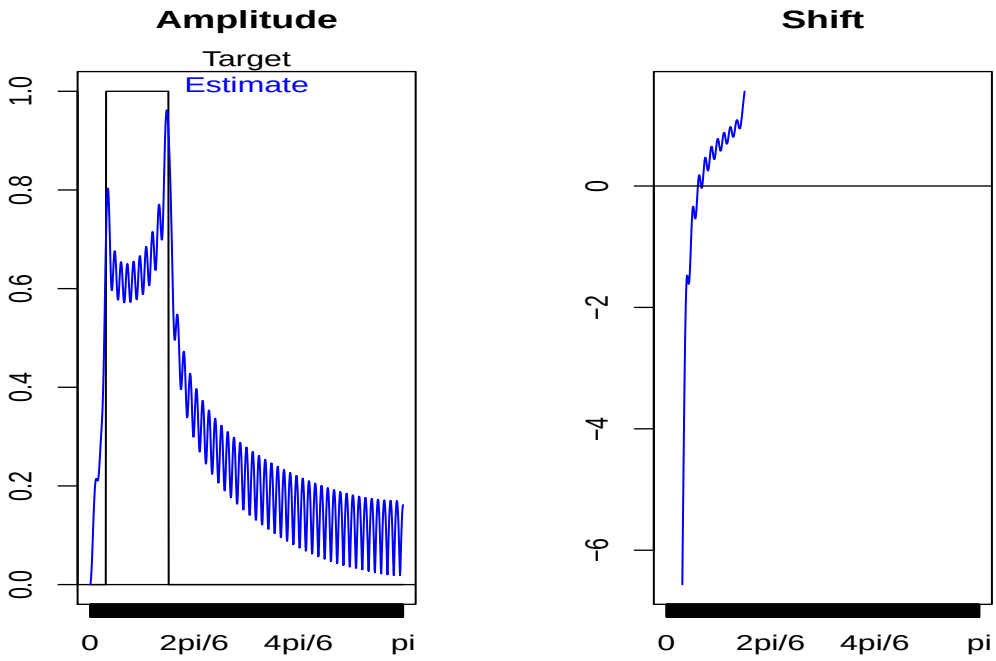


Figure 9.16: Amplitude (left) and time-shift functions (right) of real-time composite unconstrained bandpass MSE-design

As in the previous section, the observed ripples are innocuous (approximation of ideal bandpass target by finite-length filter).

4. Compute the output of the *composite* bandpass-MSE filter. Compare cycles of I(1)-model (fig.9.8), best I(2)-model (fig.9.10, top-right panel), and composite bandpass, see fig.9.17.

```
> xf_i2<-rep(NA,length(detrended))
> # Apply second-order differences to the data
> diff2_lgdp_07<-c(0,0,diff(lgdp_07,diff=2))
> for (i in L:length(detrended))
+   xf_i2[i]<-t(mdfa_obj$b)%*%diff2_lgdp_07[i:(i-L+1)]
```

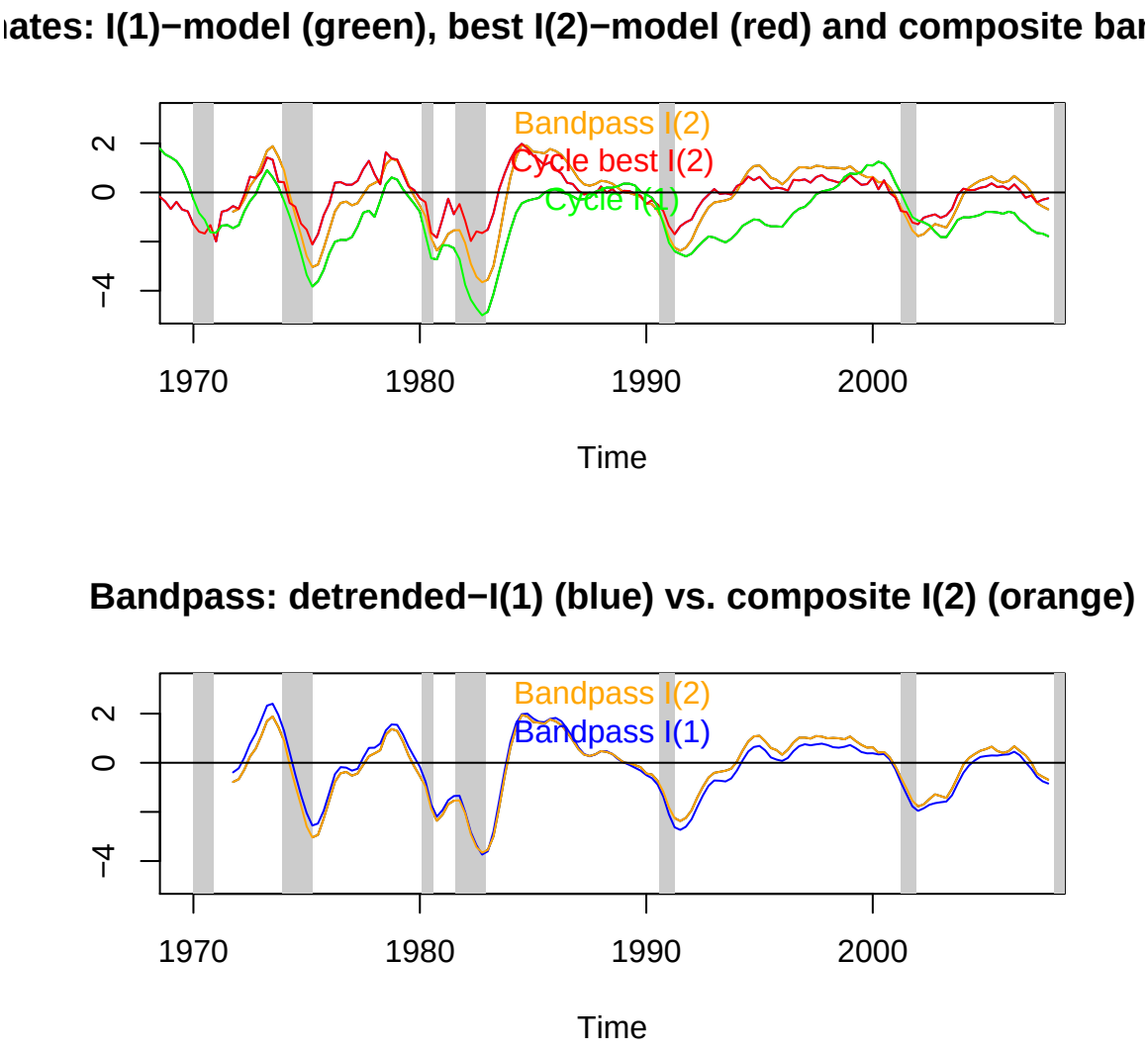


Figure 9.17: Composite bandpass (orange), bandpass I(1) (blue), best I(2)-model (red) and I(1)-model (green)

Analysis

- Both bandpass designs as well as the best I(2)-model generate zero-centric cycles which follow the alternating expansion and contraction episodes.
- The two bandpass (DFA) MSE-estimates are nearly identical despite different unit-root specifications or data transformation (detrending in the I(1)-case). This is because the pseudo-spectral densities of the DGP behave similarly outside a narrow band centered at the unit root frequency zero, recall section 9.4.8.
- The cycle of the best I(2)-model (red) appears to trade smoothness against speed: the series

is a bit noisier but turning points are detected earlier. Both issues could be tackled at once by suitable customization of the replicated MBA.

9.5 Customization of UC-Models

Once replicated by the DFA, the above trend-cycle models could be customized. However, since the model specification is sensitive to singular events, in particular to the protracted down-turn during the great recession, we here propose to analyze more robust designs, namely the classic Hodrick-Prescott and Christiano-Fitzgerald filters, which are replicated and customized by MDFA.

9.6 Replication and Customization of the Hodrick-Prescott HP-Filter)

9.6.1 Replication

The HP-filter is widely used in macroeconomics, for trend extraction or for business-cycle analysis. The original optimization principle

$$\frac{1}{T}\sum_{t=1}^{T}(x_t - y_t)^2 + \lambda_{HP}\sum_{t=3}^{T}[(1-B)^2 y_t]^2 \rightarrow \min_{y_t}$$

where x_t is the data, y_t is the HP-trend and $\lambda_{HP} > 0$ is a regularization term, was proposed by Whittaker (1923): the parameter λ_{HP} balances a trade off between data-fitting (left term) and smoothness (right term), whereby the squared second order differences $[(1-B)^2 y_t]^2$ of the trend correspond to the Curvature measure introduced in section 7.5 (omitting the normalization in the latter expression). For $\lambda_{HP} = 0$ the trend y_t (over)fits the data perfectly and for $\lambda_{HP} = \infty$ the trend is a linear function of time t (the curvature vanishes). For quarterly data, Hodrick and Prescott have proposed to select $\lambda_{HP} = 1600$.

The solution of the above minimization problem can be interpreted in terms of a formal signal extraction problem, see Maravall and Kaiser (2004) and McElroy (2008). In this framework the HP-trend y_t is the output of a symmetric filter[15]:

$$y_t = \frac{q}{q + (1-B)^2(1-F)^2}x_t$$

where $q = 1/\lambda_{HP}$ and where $F = B^{-1}$ is the forward operator. The implicit model underlying the approach assumes that

$$\begin{aligned} x_t &= T_t + \lambda_{HP} I_t \\ (1-B)^2 T_t &= \nu_t \end{aligned}$$

where I_t, ν_t are mutually independent Gaussian white noise sequences with identical variances. In this case, i.e. if the model applies, y_t is an optimal MSE estimate of the unobserved trend T_t.

[15] We here assume that y_t is estimated in the middle of a large sample, such that symmetric filters can be applied.

Note that λ_{HP} can be interpreted as an inverse SNR: for $\lambda_{HP} = 0$ the noise vanishes and therefore $x_t = T_t$ or, equivalently, $y_t = x_t$. For $\lambda_{HP} = 1600$ the DGP of x_t is found to be

$$(1 - B)^2 x_t = (1 - 1.7771B + 0.7994B^2)\epsilon_t \tag{9.8}$$

where ϵ_t is Gaussian white noise, see Maravall and Kaiser (2004) and McElroy (2008)[16]. The corresponding implicit pseudo-spectral density is

$$h(\omega) = \sigma^2 \left| \frac{1 - 1.7771 \exp(-i\omega) + 0.7994 \exp(-i2\omega)}{(1 - \exp(-i\omega))^2} \right|^2 \tag{9.9}$$

where σ^2 is the variance of ϵ_t. Note that the filter-coefficients are invariant to the scale of the data and therefore we may set $\sigma^2 = 1$, without affecting estimation results. The target, i.e. the symmetric HP-trend filter, could be obtained in R by relying on the so-called mFilter-library

```
> # Call HP-routines
> library(mFilter)
```

We conclude that optimality of the HP-trend, in terms of the MSE-metric, is reliant on an implicit model representation of the data. The corresponding DGP is integrated of order two, I(2), and its parameters are uniquely determined by the (inverse) SNR λ_{HP}.

9.6.2 Exercises: Replication

Since the HP-filter is less affected by the great recession than the previous state space models we do not skip recent data i.e. we consider the whole GDP series.

1. Load the full GDP-series, starting in 1960.

```
> # Load full data-set
> start_year<-1960
> end_date<-format(Sys.time(), "%Y-%m-%d")
> end_year<-as.double(substr(end_date,1,4))
> start_date=paste(start_year,"-01-01",sep="")
> # Select data between start_year and end_year
> data_sample<-mydata[paste("/",end_date,sep="")]
> data_sample<-data_sample[paste(start_date,"/",sep="")]
> lgdp <- ts(100*log(data_sample),start=start_year,frequency=4)
> nobs <- length(lgdp)
```

2. Load the R-function for deriving the MA-coefficients of the implicit time series model of the HP-filter, see McElroy (2008).

```
> source(file=paste(path.pgm,"hpFilt.r",sep=""))
> head(hpFilt)
```

[16]The latter author provides R-code for an exact derivation of the MA-parameters as a function of λ_{HP}, see section 9.6.2 below.

```
1 function (q, n)
2 {
3     absZ <- (sqrt(q) + sqrt(q + 16) + sqrt(2 * q + 2 * sqrt(q) *
4         sqrt(q + 16)))/4
5     c <- q/(absZ^2)
6     theta <- atan(sqrt(2 * q + 2 * sqrt(q) * sqrt(q + 16))/4)
```

The parameters q and n in the head of the function-call correspond to $1/\lambda_{HP}$ and L, the filter-length.

3. Verify that the function replicates the implicit model in the case $\lambda_{HP} = 1600$. Hint: for the filter length we select the length of the GDP-series.

```
> # Data: US-GDP
> x<-lgdp
> # Series length
> len<-L_hp<-length(x)
> # Select lambda
> lambda_hp<-1600
> q<-1/lambda_hp
> hp_filt_obj<-hpFilt(q,L_hp)
> tail(hpFilt,2)

21     return(list(filter_coef = filter_coef, ma_model = ma_model))
22 }

> hp_filt_obj$ma_model

[1]  0.0004996524 -1.7770908783  0.7994437833

> ma_coeff<-hp_filt_obj$ma_model[2:3]
```

The first model-coefficient is a normalizing constant, which is irrelevant for our application, and the remaining two parameters correspond to the MA(2)-part of the model. The function computes filter coefficients and implicit model-parameters for any $q = 1/\lambda_{HP}$.

4. Set $\lambda_{HP} = 1600$ (quarterly data) and apply the filter, as implemented in the *mFilter* package, see fig.9.18.

```
> # Resolution of frequency-grid: 2-times the sample length
> #   Selecting a higher resolution would tighten the approximation of HP by DFA
> K<-2*len
> # proceed to filtering
> x_hp <- hpfilter(x,type="lambda", freq=lambda_hp)
> # Extract the coefficients of the symmetric trend:
> #   hpfilter generates coefficients of the HP-gap (see below):
```

```
> #   we here transform back to trend filter
> parm<-diag(rep(1,len))-x_hp$fmatrix
> file = paste("z_HP_us_real_log_gdp.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6,
+     height = 6)
> # Plots: filter coefficients and series
> par(mfrow=c(2,1))
> title_more<-NA
> mplot<-cbind(parm[,len/2],parm[,1])
> plot_title<-"HP lambda=1600: symmetric (red) and real-time (blue)"
> axis_d<-1:len-1
> insamp<-1.e+99
> colo<-c("red","blue")
> mplot_func(mplot,axis_d,plot_title,title_more,insamp,colo)
> mplot<-cbind(rep(NA,len),rep(NA,len),rep(NA,len),x,x_hp$trend)
> plot_title<-"Log US-GDP (blue) vs HP-Trend (red)"
> plot(mplot[,4],col="blue",xlab="",ylab="",main=plot_title)
> nberShade()
> lines(mplot[,5],col="red")
> invisible(dev.off())
```

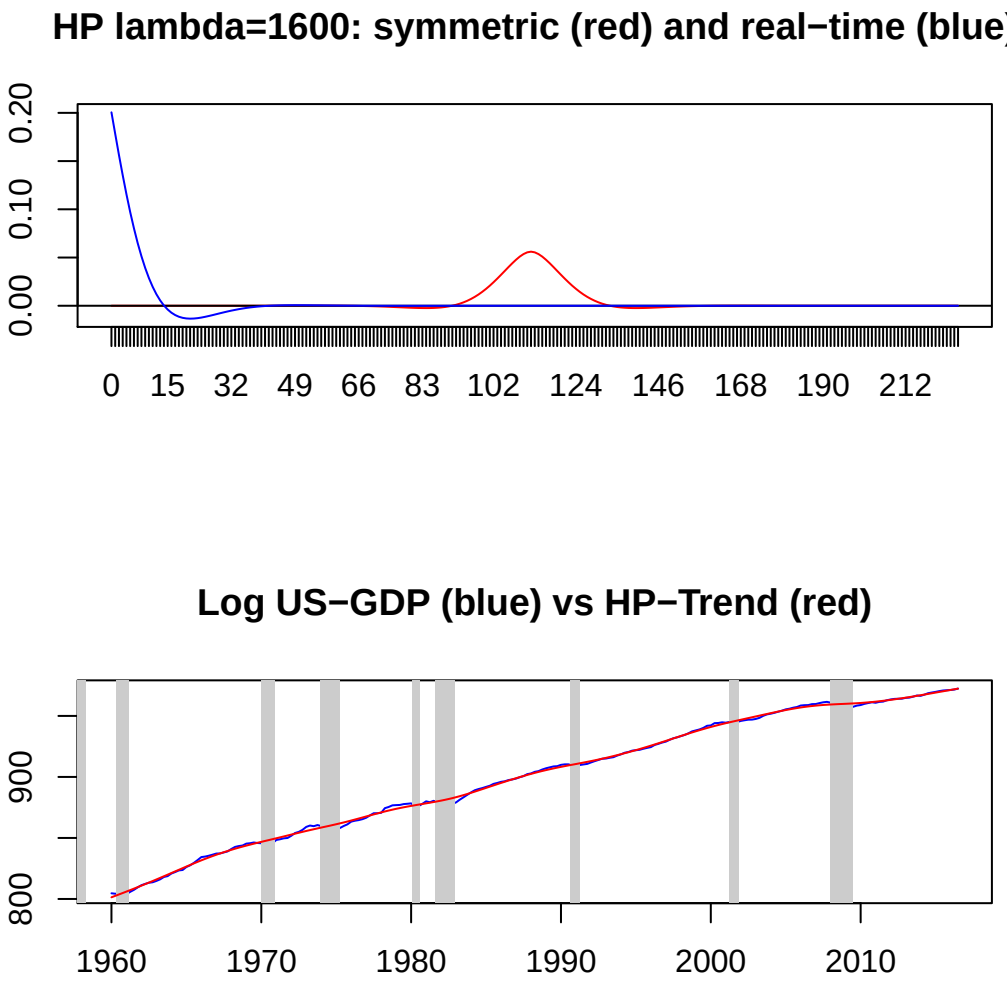


Figure 9.18: Filter coefficients of HP-trend, lambda=1600: symmetric (red) and real-time (blue) filters (top-graph). Original and filtered log US-GDP (bottom figure)

5. Compute the target signal, corresponding to the symmetric HP-filter, and the pseudo-spectral density 9.9, see fig.9.19.

```
> # Compute pseudo-spectral density underlying Wiener-Kolmogorov derivation of HP
> #   (see McElroy (2008) or Maravall-Kaiser p.179)
> # For lambda=1600 the MA coefficients are -1.77709 and 0.79944
> # Note that MDFA is fed with the square-root of the spectrum
> #   (this would correspond to the absolute value of the DFT)
> weight_func_h<-abs((1+ma_coeff[1]*exp(-1.i*(0:(K))*pi/(K))+
```

```
+                       ma_coeff[2]*exp(-1.i*2*(0:(K))*pi/(K)))/
+                       (1-exp(-1.i*(0:(K))*pi/(K)))^2)
> # Specify (square-root) spectra of target (first column)
> #   and of explanatory variable (second column): target and
> #   explanatory are the same here (univariate filter)
> weight_func<-cbind(weight_func_h,weight_func_h)
> # Compute target Gamma: HP-trend symmetric filter, see McElroy (2008)
> Gamma<-0:(K)
> for (k in 0:(K))
+ {
+   omegak<-k*pi/(K)
+   Gamma[k+1]<-(1/lambda_hp)/(1/lambda_hp+abs(1-exp(1.i*omegak))^4)
+ }
> file = paste("z_HP_filt_trffkt.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6,
+     height = 6)
> par(mfrow=c(2,1))
> colo<-c("blue","red")
> insamp<-1.e+99
> mplot<-as.matrix(Gamma)
> plot_title<-"Target Gamma, lambda=1600"
> freq_axe<-rep(NA,K+1)
> freq_axe[1]<-0
> freq_axe[1+(1:6)*K/6]<-c(paste(c("",2:5),"pi/6",sep=""),"pi")
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> # Plot log spectrum: weight_func must be squared
> mplot<-as.matrix(c(NA,log(weight_func[2:(K+1),1]^2)))
> plot_title<-"Log pseudo-spectrum, lambda=1600"
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```

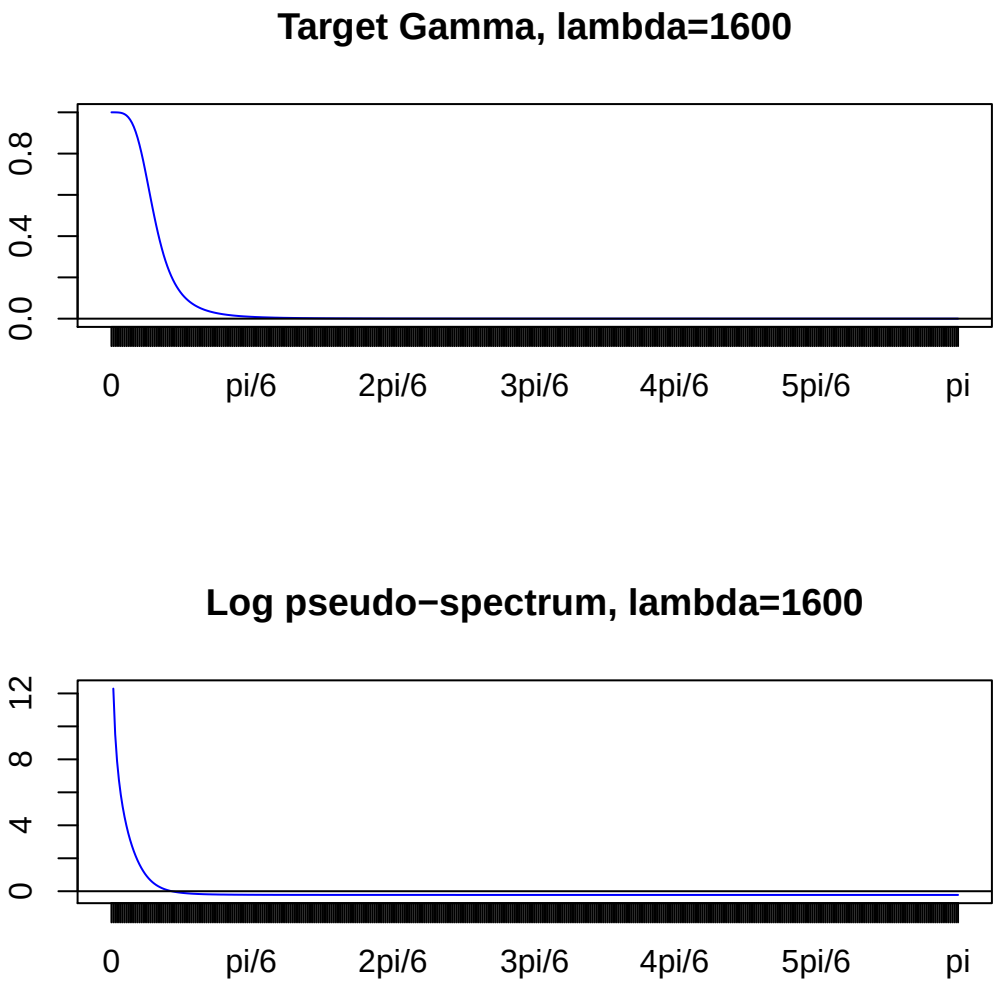


Figure 9.19: Target signal (top) and log-transformed pseudo-spectrum (bottom): the singularity in frequency zero is skipped

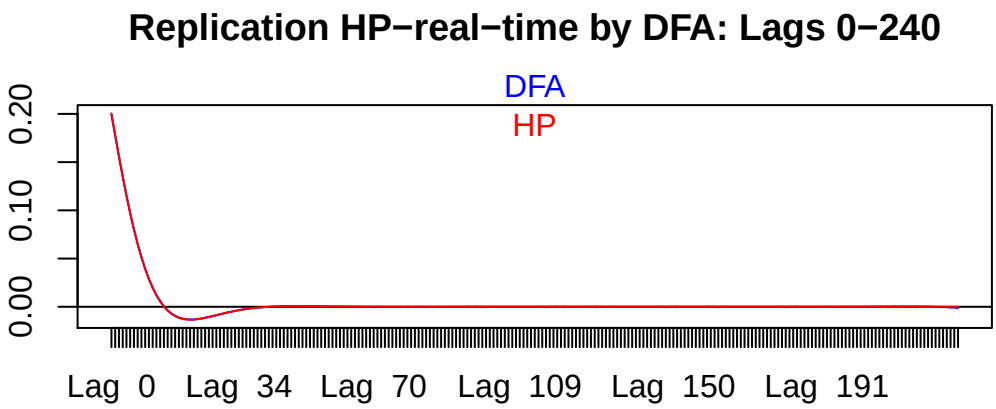
6. Replicate the HP real-time or concurrent filter (nowcast) by DFA: for that purpose insert the target and the pseudo-spectral density into 4.27 and impose first- and second-order constraints ($i1 = i2 = T$), as proposed in chapter 6[17]. Note that we propose (equivalent) generic as well as context specific estimation calls in our code below.

```
> # HP-spectrum: this will be squared in MDFA
> weight_func_hp<-weight_func
```

[17]Specifically we impose $\hat{\Gamma}(0) = \Gamma(0) = 1$ (level-constraint) and $\hat{\phi}(0) = 0$ (vanishing time-shift) which are required because the implicit DGP is assumed to be integrated of order two (double unit-root in frequency zero).

```
> K<-nrow(weight_func_hp)-1
> # Frequency zero is infinity (unit root)
> #   The singularity is removed by imposing first and second order
> #     restrictions
> #   For numerical computations we set the spectrum arbitrarily
> #     to zero in freq. zero
> weight_func_hp[1,]<-0
> # Filter length is identified with sample length
> L<-len
> # Set default settings for MDFA (MSE, no regularization)
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # First and second order constraints are imposed
> #   (the level constraint weight_constraint=1 is set in the default settings)
> i1<-T
> i2<-T
> # Cutoff: the frequency at which the target drops below 0.5
> cutoff<-pi*which(Gamma<0.5)[1]/length(Gamma)
> # Real-time (nowcast)
> Lag<-0
> # Estimation: generic
> imdfa_hp<-mdfa_analytic(L,lambda,weight_func_hp,Lag,Gamma,eta,cutoff,
+                         i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                         lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                         b0_H0,c_eta,weight_structure,
+                         white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_hp<-MDFA_mse_constraint(L,weight_func_hp,Lag,Gamma,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
> file = paste("z_HP_filt_coef.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6,
+     height = 6)
> par(mfrow=c(2,1))
> colo<-c("blue","red")
> insamp<-1.e+99
> mplot<-cbind(imdfa_hp$b,parm[1:L,max(0,Lag)+1])
> rownames(mplot)<-paste("Lag ",0:(nrow(mplot)-1))
> colnames(mplot)<-c("Replication by DFA","HP-real-time")
> plot_title<-"Replication HP-real-time by DFA: Lags 0-240"
> freq_axe<-rownames(mplot)
> title_more<-c("DFA","HP")
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
```

```
> mplot<-mplot[1:21,]
> rownames(mplot)<-paste("Lag ",0:(nrow(mplot)-1))
> colnames(mplot)<-c("Replication by DFA","HP-real-time")
> plot_title<-"Replication HP-real-time by DFA: Lags 0-20"
> freq_axe<-rownames(mplot)
> title_more<-c("DFA","HP")
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```



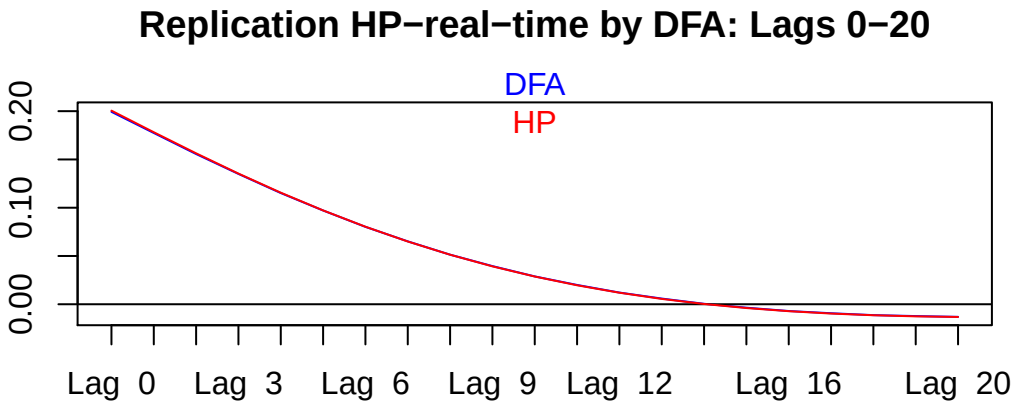


Figure 9.20: Concurrent filter coefficients: DFA (blue) vs. 'true' coefficients (red). Full lag-distribution (top) and first twenty lags (bottom).

Fig.9.20 confirms that the DFA replicates the true coefficients up to arbitrary precision: both coefficient series overlap almost perfectly. A numerical juxtaposition of filter coefficients below confirms that the approximation error is negligible by all practical means[18]:

```
> head(mplot)

       Replication by DFA HP-real-time
Lag  0         0.19932991   0.20055622
Lag  1         0.17741516   0.17820331
Lag  2         0.15560021   0.15635006
Lag  3         0.13501185   0.13538473
Lag  4         0.11520513   0.11559789
Lag  5         0.09712518   0.09719548
```

7. Check first- and second-order constraints of the DFA real-time filter:

```
> # Check first-order: should give 1
> print(paste("Transfer function in frequency zero: ",
+             round(sum(imdfa_hp$b),3),sep=""))

[1] "Transfer function in frequency zero: 1"

> # Check second-order: time-shift should vanish
> print(paste("Time-shift in frequency zero: ",
+             round((1:(L-1))%*%imdfa_hp$b[2:L],10),sep=""))

[1] "Time-shift in frequency zero: 0"
```

After replication of the real-time trend estimate we proceed to business-cycle analysis.

9.6.3 Business-Cycle Analysis: HP-Gap and HP-Cycle

We here propose two different designs, a highpass and a bandpass, and we compare both 'cycle' concepts.

HP-Gap

Consider the deviations of the GDP-series about the HP-trend

$$\text{gap}_t = \text{GDP}_t - \sum_{k=-\infty}^{\infty} \gamma_k^{HP-Trend,1600} \text{GDP}_{t-k}$$

where $\gamma_k^{HP-Trend,1600}$ are the coefficients of the symmetric HP-trend filter with $\lambda_{HP} = 1600$[19] and where, for the moment, it is assumed that the data-sample stretches infinitely into past and

[18] The magnitude of the error depends on the (finite) resolution of the discrete frequency-grid as specified by the parameter K in the R-code.

[19] Note that the coefficients as computed by the function $hpfilter$ ($mFilter$-package) correspond to the gap. The trend coefficients in the previous section were obtained by the transformation $\gamma_k^{trend} = 1 - \gamma_k^{gap}$.

into future. The resulting time series gap_t is likely to be positive during expansions and negative during contractions. Although these dynamic characteristcs typically alude to a 'cycle', the target filter

$$\Gamma^{gap}(\omega) := 1 - \Gamma^{trend}(\omega)$$

is not a bandpass but a highpass instead, see fig.9.21, red line. The highpass has a double zero in the unit-root frequency zero: it transforms a non-stationary I(2)-process in a stationary component. The real-time (one-sided) filter $\hat{\Gamma}^{gap}(\omega)$ inherits this property, too. Note that $\hat{\Gamma}^{gap}(\omega) := 1 - \hat{\Gamma}^{trend}(\omega)$ where $\hat{\Gamma}^{trend}(\omega)$ is the real-time trend filter replicated in the previous exercise. The assertion follows directly from

$$\begin{aligned} & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| \Gamma^{gap}(\omega_k) - \hat{\Gamma}^{gap}(\omega_k) \right|^2 h(\omega_k) \\ = & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| (1 - \Gamma^{trend}(\omega_k)) - (1 - \hat{\Gamma}^{trend}(\omega_k)) \right|^2 h(\omega_k) \\ = & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| \Gamma^{trend}(\omega_k) - \hat{\Gamma}^{trend}(\omega_k) \right|^2 h(\omega_k) \end{aligned}$$

where $h(\omega_k)$ is the pseudo spectral density defined in 9.9: $\hat{\Gamma}^{trend}$ minimizes the lower term if and only if $\hat{\Gamma}^{gap}(\omega)$ minimzes the upper term.

HP-Cycle

An effective bandpass design is obtained when applying the HP-trend filter to the *differenced* GDP series

$$\begin{aligned} cycle_t &= \sum_{k=-\infty}^{\infty} \gamma_k^{HP-Trend,1600} \Delta GDP_{t-k} \\ &= \sum_{k=-\infty}^{\infty} (\gamma_k^{HP-Trend,1600} - \gamma_{k-1}^{HP-Trend,1600}) GDP_{t-k} \end{aligned}$$

where $\Delta GDP_{t-k} = GDP_{t-k} - GDP_{t-k-1}$. The filter with coefficients

$$\gamma_k^{Cycle} := \gamma_k^{HP-Trend,1600} - \gamma_{k-1}^{HP-Trend,1600} \tag{9.10}$$

is called HP Cycle. Note that the filter coefficients γ_k^{Cycle} are applied to the original data in *levels*: the transfer function is effectively a bandpass, see fig.9.21, blue line. The filter 9.10 transforms an I(1)-process (single unit-root with a possible linear drift) into a stationary cycle centered about the drift-constant[20]. The MSE-criterion 4.27 for deriving the real-time filter is

$$\begin{aligned} & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| \Gamma^{trend}(\omega_k)(1 - \exp(-i\omega_k)) - \hat{\Gamma}(\omega_k)(1 - \exp(-i\omega_k)) \right|^2 h(\omega_k) \\ = & \frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| \Gamma^{trend}(\omega_k) - \hat{\Gamma}(\omega_k) \right|^2 \tilde{h}(\omega_k) \rightarrow \min_{\mathbf{b}} \end{aligned} \tag{9.11}$$

[20] We here ignore the fact that the implicit model is I(2). First differences of GDP are (nearly) stationary which is all we need at this stage.

where

$$\begin{aligned}\tilde{h}(\omega) &= |1-\exp(-i\omega)|^2 \left|\frac{1-1.7771\exp(-i\omega)+0.7994\exp(-i2\omega)}{(1-\exp(-i\omega))^2}\right|^2 \\ &= \left|\frac{1-1.7771\exp(-i\omega)+0.7994\exp(-i2\omega)}{1-\exp(-i\omega)}\right|^2 \end{aligned} \tag{9.12}$$

Note that we may apply either $\hat{\Gamma}(\omega_k)(1-\exp(-i\omega_k))$ to GDP or $\hat{\Gamma}(\omega_k)$ to first differences of GDP: the latter is done in our R-code below. Since the differenced process is assumed to be I(1), instead of I(2), we can select $i1 = T$ and $i2 < -F$: a first-order level constraint is imposed only[21].

As we shall see, real-time gap and cycle estimates are closely linked, since the former corresponds to first differences of the latter.

Exercises: Implement HP-Gap and HP-Cycle

1. Compute amplitude functions of HP-Gap and of HP-Cycle and show that the first is a highpass and that the second is a bandpass.

```
> Gamma_cycle<-abs(Gamma*(1-exp(1.i*pi*(0:K)/K)))
> Gamma_gap<-1-Gamma
> Gamma_gap[1]<-0
> file = paste("z_HP_us_real_log_gdp_hp_diff__gap_amp.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> insamp<-1.e+99
> par(mfrow=c(2,2))
> # Cycle
> mplot<-as.matrix(Gamma_cycle)
> plot_title<-"Target HP-Cycle"
> freq_axe<-rep(NA,K+1)
> freq_axe[1]<-0
> freq_axe[1+(1:6)*K/6]<-c(paste(c("",2:5),"pi/6",sep=""),"pi")
> title_more<-NA
> colo<-"blue"
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> # Gap
> mplot<-as.matrix(Gamma_gap)
> plot_title<-"Target HP-Gap"
> freq_axe<-rep(NA,K+1)
> freq_axe[1]<-0
> freq_axe[1+(1:6)*K/6]<-c(paste(c("",2:5),"pi/6",sep=""),"pi")
> title_more<-NA
```

[21] The user is free to relax this restriction since first differences of GDP are not integrated.

```
> colo<-"red"
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> # At frequency zero
> len2<-30
> mplot<-as.matrix(c(Gamma_cycle[(len2+1):1],
+                    as.matrix(Gamma_cycle)[2:(len2+1)]))
> plot_title<-"Target HP-Cycle at frequency zero"
> freq_axe<-rep(NA,2*len2+1)
> freq_axe[1]<-0
> freq_axe[c(1,len2+1,2*len2+1)]<-c(paste("-pi/",as.integer(K/len2),sep=""),
+                                   "0",paste("pi/",as.integer(K/len2),sep=""))
> title_more<-NA
> colo<-"blue"
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> # Gap
> mplot<-as.matrix(c(Gamma_gap[(len2+1):1],as.matrix(Gamma_gap)[2:(len2+1)]))
> plot_title<-"Target HP-Gap at frequency zero"
> freq_axe<-rep(NA,2*len2+1)
> freq_axe[1]<-0
> freq_axe[c(1,len2+1,2*len2+1)]<-c(paste("-pi/",as.integer(K/len2),sep=""),"0",
+                                   paste("pi/",as.integer(K/len2),sep=""))
> title_more<-NA
> colo<-"red"
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```

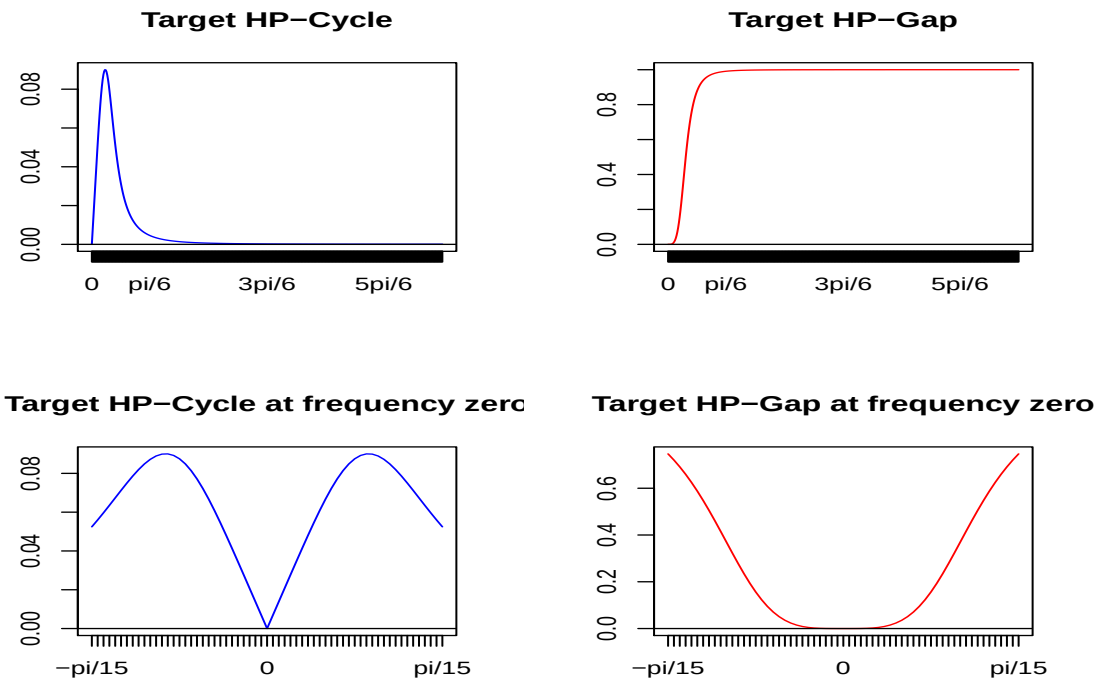


Figure 9.21: Amplitude functions: HP-Cycle (blue) and HP-Gap (red). Whole positive frequency-band (top) vs. close-up at frequency zero (bottom)

Remarks

- The bandpass (left-side plots) 'shrinks' the input signal: the peak-value of the amplitude is smaller than 1/10.
- The HP-Gap (right-side plots) is a highpass: noise will pass the filter without hindrance.
- The amplitude function of the highpass flattens towards frequency zero, see bottom-right plot: its derivative vanishes or, stated otherwise, HP-Gap has a zero of order two in frequency zero.
- In contrast, the amplitude of the bandpass does not flatten towards frequency zero, see bottom-left plot: its derivative does not exist and the zero is of order one only.

2. Compute the cycle-periodicity, measured in years:

```
> peak_frequency<-(which(Gamma_cycle==max(Gamma_cycle))-1)*pi/K
> periodicity_in_quarters<-round(2*pi/peak_frequency,3)
> print(paste("Cycle-periodicity in years: ",periodicity_in_quarters/4,sep=""))

[1] "Cycle-periodicity in years: 13.353"
```

The effective duration of the cycle, as measured by the peak of the amplitude function, exceeds the mean duration of business-cycles. A correspndingly shorter length could be obtained by selecting a smaller λ_{HP}. As an example, we here compute the effective cycle-length for $\lambda_{HP} = 200$:

```
> # Select lambda
> lambda<-200
> # proceed to filtering
> x_l <- hpfilter(x,type="lambda", freq=lambda)
> # Extract the coefficients of the symmetric trend:
> parm_l<-diag(rep(1,len))-x_l$fmatrix
> Gamma_l<-0:(K)
> for (k in 0:(K))
+ {
+   omegak<-k*pi/(K)
+   Gamma_l[k+1]<-parm_l[len/2,len/2]+2*parm_l[(len/2+1):len,len/2]%*%
+               cos((1:(len/2))*omegak)
+ }
> # Specify cycle-target
> Gamma_cycle_l<-abs(Gamma_l*(1-exp(1.i*pi*(0:K)/K)))
> # Compute cycle-length
> peak_frequency_l<-(which(Gamma_cycle_l==max(Gamma_cycle_l))-1)*pi/K
> periodicity_in_quarters_l<-round(2*pi/peak_frequency_l,3)
> print(paste("Cycle-periodicity in years for lambda=",lambda,": ",
+             periodicity_in_quarters_l/4,sep=""))

[1] "Cycle-periodicity in years for lambda=200: 7.8275"
```

The mean duration of 7.827 (years) would match business-cycles of US-GDP better. Note, however, that the AR(2)-coefficients in the implicit model 9.8, and therefore the pseudo-spectral densities 9.9 and 9.12, depend on λ_{HP} and should be recomputed accordingly, see exercise 3, section 9.6.2, for usage of the corresponding R-function.

3. Compute an optimal (MSE) real-time filter of the target $cycle_t$: apply 9.11 and set $i1 = T, i2 = F$ and $L = 50$[22]. Once again we propose generic and context-specific estimation calls in our code.

```
> L<-50
> # Specify the pseudo-spectral density of the differenced implicit model equation
> #   The following line of code is tricky since a two-column matrix multiplies
> #   a vector whose length corresponds to the number of rows of the matrix.
> # In such a case, R automatically applies the vector to each column separately
> #   and multiplication is performed elementwise, as desired.
```

[22] Fig.9.20 suggests that real-time filter coefficients decay rapidly to zero.

```
> weight_func_hp_diff<-weight_func_hp*
+   abs(1-exp(-1.i*(0:(nrow(weight_func_hp)-1)*pi/(nrow(weight_func_hp)-1))))
> K<-nrow(weight_func_hp_diff)-1
> # Set default settings for MDFA (MSE, no regularization)
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Set the filter constraints: level but no time-shift constraint
> i1<-T
> i2<-F
> # Estimate the corresponding MSE real-time filter
>
> imdfa_hp_cycle<-mdfa_analytic(L,lambda,weight_func_hp_diff,Lag,Gamma,
+                         eta,cutoff,i1,i2,weight_constraint,lambda_cross,
+                         lambda_decay,lambda_smooth,lin_eta,shift_constraint,
+                         grand_mean,b0_H0,c_eta,weight_structure,
+                         white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_hp_cycle<-MDFA_mse_constraint(L,weight_func_hp_diff,Lag,Gamma,i1,i2,
+                                     weight_constraint,shift_constraint)$mdfa_obj
>
```

4. Check the filter-constraints:

```
> # Check first-order: should give 1
> print(paste("Transfer function in frequency zero: ",
+             sum(imdfa_hp_cycle$b),sep=""))

[1] "Transfer function in frequency zero: 1"

> # Check second-order: this is not imposed anymore
> print(paste("Time-shift in frequency zero: ",
+             round((1:(L-1))%*%imdfa_hp_cycle$b[2:L],3),sep=""))

[1] "Time-shift in frequency zero: 8.257"
```

As expected, the first-order level constraint is met but the time-shift in frequency zero is now 8.257.

5. Compare real-time coefficients for HP-trend (previous section) with those obtained for $\hat{\Gamma}(\omega_k)$ (the latter filter is applied directly to differenced data), see fig.9.22.

```
> # Specify the pseudo-spectral density of the differenced implicit
> #   model equation
> file = paste("z_HP_us_real_log_gdp_hp_diff_mse.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
```

```
> insamp<-1.e+99
> colo<-c("red","blue")
> mplot<-cbind(imdfa_hp$b[1:L],imdfa_hp_cycle$b)
> plot_title<-"Filter coefficients: Level (red) vs. Difference (blue)"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,1],col="red",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> lines(mplot[,2],col="blue")
> invisible(dev.off())
```

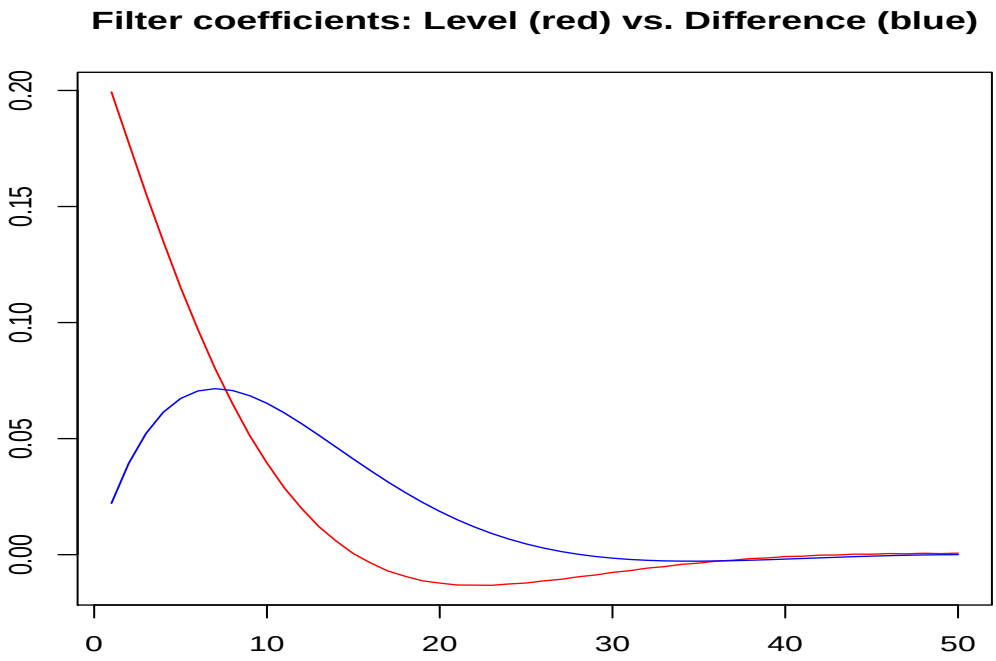


Figure 9.22: Real-time MSE filters: HP-trend in level (red) vs. HP Trend in differences (blue)

The new coefficients (blue line) decay less rapidly to zero because the difference operator magnifies high-frequency noise i.e. a stronger smoothing is required.

6. Filter the data according to the above real-time Gap and Cycle specifications:

   - HP-Gap

     ```
     > x_gap<-x
     > # Specify the real-time HP-Gap coefficients
     > #   We rely on the estimates as computed by hpfilter
     ```

```
> #   We provide an arbitrary series of length L
> #   The real-time gap-coefficients are in the first column of fmatrix
> gamma_gap<-hpfilter(1:L,type="lambda", freq=lambda_hp)$fmatrix[,1]
> # The sum must be zero: bandpass filter applied to
> #   original GDP in level
> sum(gamma_gap)

[1] 7.264155e-17

> for (j in L:length(x_gap))
+   x_gap[j] <- gamma_gap[1:L]%*%x[j:(j-(L-1))]
> x_gap[1:(L-1)]<-NA
```

- HP-Cycle:

```
> x_cycle<-xdiff<-diff(x)
> gamma_cycle<-as.vector(imdfa_hp_cycle$b)
> # Coefficients should add to one: trend filter applied to
> #   differenced data
> sum(gamma_cycle)

[1] 1

> for (j in L:length(x_cycle))
+   x_cycle[j] <- gamma_cycle%*%xdiff[j:(j-(L-1))]
> x_cycle[1:(L-1)]<-NA
```

7. Compare the proposed real-time cycle estimates and verify that the differenced cycle corresponds to the gap, see fig.9.23. Hint: standardize all series for ease of visual inspection.

```
> file = paste("z_HP_us_real_log_gdp_hp_bp.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> par(mfrow=c(1,2))
> # Plots: filter coefficients and series
> insamp<-1.e+99
> # We bind data and filter outputs
> #   The shorter x_cycle (relies on differences) is automatically
> #     shifted/adjusted
> mploth<-cbind(c(NA,diff(x)),x_gap,x_cycle,diff(x_cycle))
> # Standardization
> mplot<-scale(na.omit(mploth))
> plot_title<-"Gap (red) vs. cycle (blue)"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,3],col="blue",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
```

```
> nberShade()
> lines(mplot[,3],col="blue")
> lines(mplot[,2],col="red")
> plot_title<-"Gap (red) vs. diff-cycle (green)"
> plot(mplot[,2],col="red",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> nberShade()
> lines(mplot[,2],col="red")
> lines(mplot[,4]+1,col="green")
> invisible(dev.off())
```

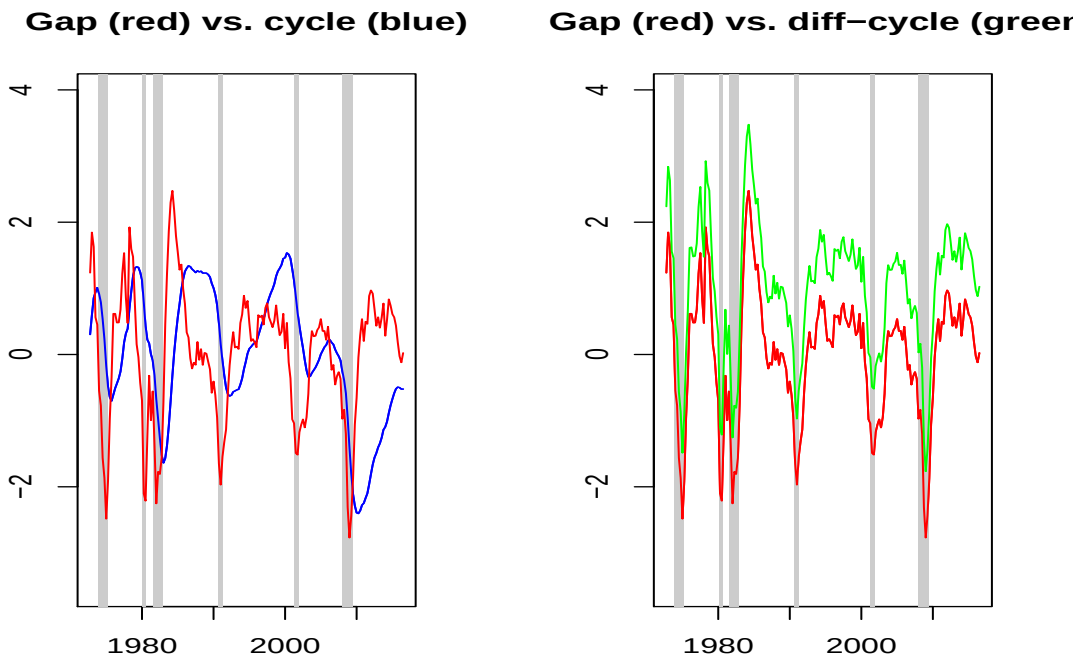


Figure 9.23: HP-gap (red) and HP-cycle (blue) applied to log-transformed US-GDP in left panel. HP-gap (red) and differenced HP-cycle (green) in right panel: the green line is shifted upwards by +1 in order to differentiate both series. All filter outputs are standardized for ease of visual inspection

The cycle (blue line) relying on the pseudo-spectral density 9.12 (differenced data) is smoother, as expected. We note, also, that HP-gap (red) remains centered at zero towards the sample end, whereas the cycle appears 'negatively biased'. This effect illustrates the double zero of the HP-gap design in frequency-zero: the filter is less sensitive to the changing drift (smaller

growth-rate) of US-GDP after the great recession[23]. It is debatable which concept is more pertinent from a business-cycle perspective: a cycle centered about the slowly changing drift (blue) or a highpass centered at zero (red). As confirmed by the right panel, first differences of the cycle (green) are identical to the gap (red): both series would perfectly merge without the artificial shift introduced in the differenced cycle. The difference operator, linking gap and cycle series, explains the dissimilitude of the amplitude functions in frequency zero, recall fig.9.21.

We now customize the cycle, trying to maintain smoothness while improving speed.

9.6.4 Customization

A look at HP-Cycle in fig.9.23, blue line, suggests that the customization should emphasize Timeliness strongly in the ATS-trilemma, since the series appears 'shifted to the right' (delayed).

1. Set $\lambda = 100$, $\eta = 0.5$ and compute a customized filter based on criterion 9.12. The constraints are left unchanged. Note that our R-code proposes generic as well as context-specific (equivalent) function-calls.

```
> # Customization
> lambda<-100
> eta<-0.5
> # Delimit passband and stopband for customization:
> #   Pi/12 corresponds to a mean duration of 24 quarters
> #   which is approximately the mean duration of historical
> #   business-cycles
> cutoff<-pi/12
> imdfa_hp_cust<-mdfa_analytic(L,lambda,weight_func_hp_diff,Lag,Gamma,eta,cutoff,
+                         i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                         lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                         b0_H0,c_eta,weight_structure,
+                         white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_hp_cust<-MDFA_cust_constraint(L,weight_func_hp_diff,Lag,Gamma,cutoff,
+               lambda,eta,i1,i2,weight_constraint,shift_constraint)$mdfa_obj
```

2. Compare filter coefficients of MSE- and customized filters, see fig.9.24.

```
> file = paste("z_HP_cust_coef.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 4, height = 4)
> # Plots: filter coefficients MSE and customized
> title_more<-NA
> mplot<-cbind(imdfa_hp_cust$b,imdfa_hp_cycle$b)
```

[23]The declining drift can be observed in fig.9.18, too.

```
> parma<-parm
> plot_title<-"HP cycle: MSE (blue) vs Customized (black)"
> axis_d<-0:(L-1)
> insamp<-1.e+99
> colo<-c("black","blue")
> mplot_func(mplot,axis_d,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```

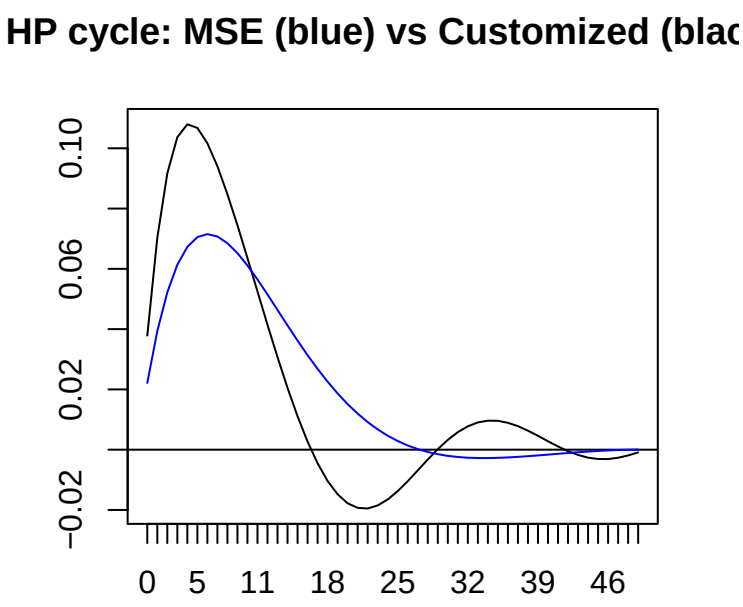


Figure 9.24: Filter coefficients HP Cycle: MSE (blue) vs. customized (black), based on spectrum of differenced data.

3. Filter the data and compare the corresponding cycle nowcasts, see fig.9.25.

```
> x_cycle_cust<-diff(x)
> gamma_cust<-imdfa_hp_cust$b[1:L]
> for (j in L:length(x_cycle_cust))
+   x_cycle_cust[j] <- gamma_cust%*%diff(x)[j:(j-(L-1))]
> x_cycle_cust[1:(L-1)]<-NA
> file = paste("z_HP_us_real_log_gdp_hp_bp_cust.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> # Plot
```

```
> insamp<-1.e+99
> colo<-c("red","blue")
> mplot<-scale(cbind(x_cycle,x_cycle_cust))
> plot_title<-"HP Cycle: MSE (blue) vs Customized (black): both standardized"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,2],col="black",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> nberShade()
> lines(mplot[,2],col="black")
> lines(mplot[,1],col="blue")
> invisible(dev.off())
```

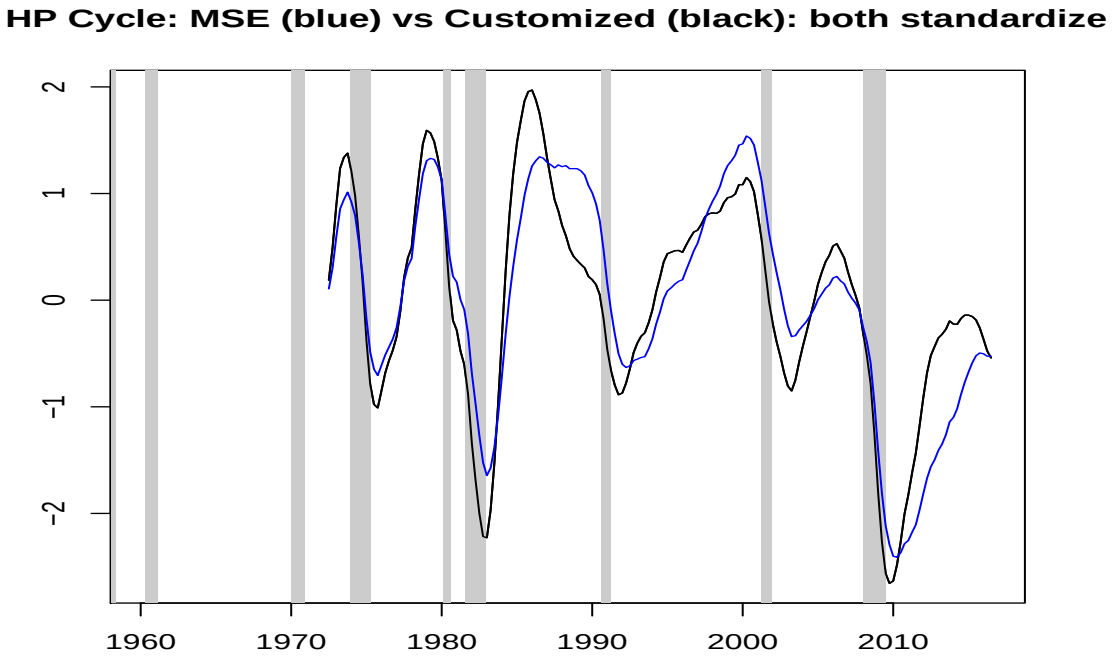


Figure 9.25: HP real-time Cycle designs: MSE (blue) vs. customized (black). Both filter outputs are standardized for ease of visual inspection

The customized filter output is smooth and it tends to lie 'to the left' of the MSE-design, as desired.

4. Compare differences of the customized cycle with the MSE gap estimate, see fig.9.26. Hint: recall that MSE gap and first differences of MSE cycle are identical, see fig.9.23, right panel.

```
> file = paste("z_HP_us_real_log_gdp_hp_bp_cust_1.pdf", sep = "")
```

```
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> insamp<-1.e+99
> colo<-c("red","blue")
> mplot<-scale(cbind(x_gap,diff(x_cycle_cust)))
> plot_title<-"Diff-customized cycle (green) vs. HP-Gap (red)"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,2],col="green",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> nberShade()
> lines(mplot[,2],col="green")
> lines(mplot[,1],col="red")
> invisible(dev.off())
```

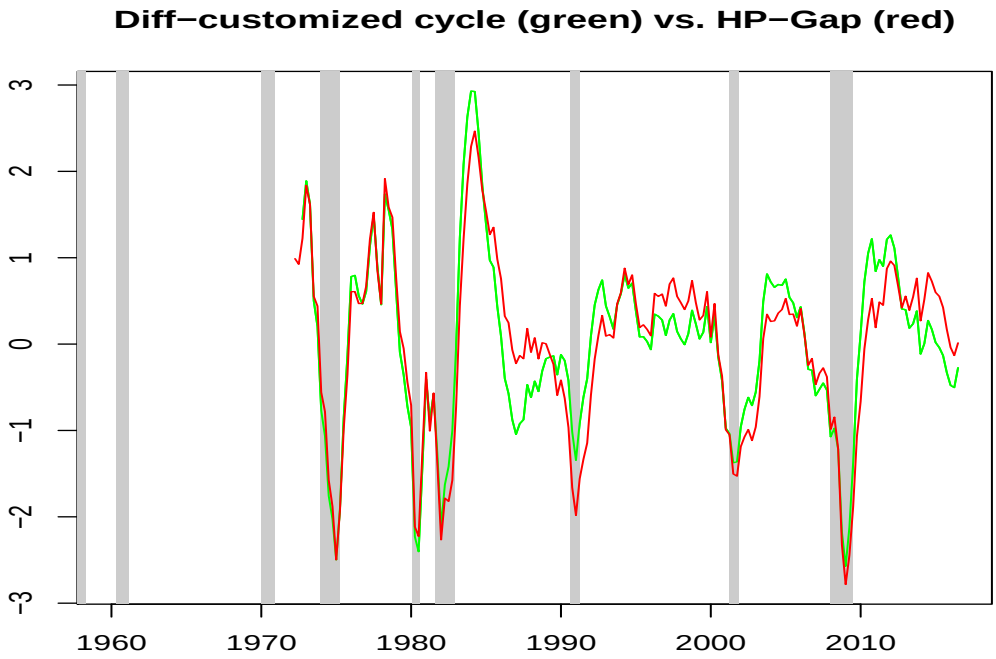


Figure 9.26: Differences of customized cycle (green) vs. MSE HP-gap (red). Both filter outputs are standardized for ease of visual inspection

As expected, the differenced customized cycle tends to lie to the left of the MSE gap.

9.7 Christiano Fitzgerald CF-Filter

9.7.1 Definition

The bandpass CF-filter relies on the ideal bandpass target

$$\Gamma^{CF}(\omega) := \begin{cases} 1 & , \text{ cutoff}^l < |\omega| < \text{cutoff}^u \\ 0 & , \text{ otherwise} \end{cases}$$

where $0 < \text{cutoff}^l < \text{cutoff}^u$ designate lower and upper cutoff frequencies. Typically, real-time estimates are obtained by assuming that the DGP is a random-walk. In this case, the DFA-MSE criterion 4.27 becomes

$$\frac{2\pi}{T} \sum_{k=-[T/2]}^{[T/2]} \left| \Gamma^{CF}(\omega_k) - \hat{\Gamma}(\omega_k) \right|^2 \frac{1}{|1-\exp(-i\omega_k)|^2} \rightarrow \min_{\mathbf{b}} \tag{9.13}$$

where $h(\omega) = \dfrac{1}{|1-\exp(-i\omega_k)|^2}$ is the pseudo-spectral density of the random-walk[24]. A first-order restriction $\hat{\Gamma}(0) = \Gamma(0) = 0$ is necessary in order to cancel the singularity in frequency zero, see section 6.1 (the case $i1 = T$). In contrast to HP-gap, which is a highpass, the CF-filter is an effective bandpass design. In contrast to HP-cycle, which is also a bandpass, upper and lower cutoff-frequencies are specified explicitly. Finally, CF assumes an I(1)-model in contrast to HP whose implicit DGP is I(2).

9.7.2 Replication

We replicate the bandpass CF-filter with own R-code and establish a link to the functions *cffilter* and *mFilter* in the R-package *mFilter*. As for previous sections we rely on (log-transformed) US-GDP.

1. Specify a $[2, 10]$-years bandpass (typical business-cycle specification) and implement a real-time CF-filter according to the implicit model-specification in the previous section. Impose a first order level-restriction (i1=T), set $L = T$ and select $K = 2 * T$ for the resolution of the discret frequency-grid. Note that we propose generic as well as context-specific function-calls in our R-code below.

```
> x<-lgdp
> # Series length
> len<-length(x)
> # Resolution of frequency-grid: 2-times the sample length
> #   Selecting a higher resolution would tighten the approximation of
> #   CF by DFA
> K<-2*len
```

[24] An additional scaling-term σ^2 is unnecessary since filter coefficients are scale invariant in this application. The numerator of the pseudo-spectral density could be enriched by an arbitrary MA-specification if the differenced data is autocorrelated.

```
> # Upper and lower cutoffs of bandpass: lengths in quarters
> len1<-8
> len2<-40
> cutoff1<-2*pi/len1
> cutoff2<-2*pi/len2
> # Specify target bandpass
> Gamma_cf<-((0:K)>K*cutoff2/pi)&((0:K)<K*cutoff1/pi)
> # Specify (square-root of) implicit pseudo-spectral density
> weight_func_cf<-matrix(rep(1/abs(1-exp(1.i*(0:K)*pi/K)),2),ncol=2)
> K<-nrow(weight_func_cf)-1
> # Remove singularity in frequency zero (one can assign an arbitrary value)
> weight_func_cf[1,]<-0
> # Filter length
> L<-len
> # Set default settings for MDFA (MSE, no regularization)
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Filter constraints
> i1<-T
> i2<-F
> # We have to specify the level-constraint
> #   Real-time filter must equal target-Gamma in frequency zero
> weight_constraint<-Gamma_cf[1]
> # Proceed to estimation
>
> imdfa_cf<-mdfa_analytic(L,lambda,weight_func_cf,Lag,Gamma_cf,eta,cutoff,
+                         i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                         lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                         b0_H0,c_eta,weight_structure,
+                         white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_cf<-MDFA_mse_constraint(L,weight_func_cf,Lag,Gamma_cf,i1,i2,weight_constraint,
+                               shift_constraint)$mdfa_obj
>
```

2. Plot the amplitude function of the real-time filter, see fig.9.27.

```
> file = paste("z_amp_shift_dfa_mse_cf_0.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> omega_k<-pi*(0:K)/K
> amp_mse<-abs(imdfa_cf$trffkt)
> mplot<-as.matrix(amp_mse)
```

```
> mplot[1,]<-NA
> colnames(mplot)<-NA
> ax<-rep(NA,nrow(mplot))
> ax[1+(0:6)*((nrow(mplot)-1)/6)]<-c(0,"pi/6","2pi/6","3pi/6",
+                                    "4pi/6","5pi/6","pi")
> plot_title<-paste("Amplitude of Christiano Fitzgerald with
+                   cutoffs pi/",len2/2,", pi/",len1/2,sep="")
> insamp<-1.e+90
> title_more<-dimnames(mplot)[[2]]
> colo<-c("blue","red")
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
```

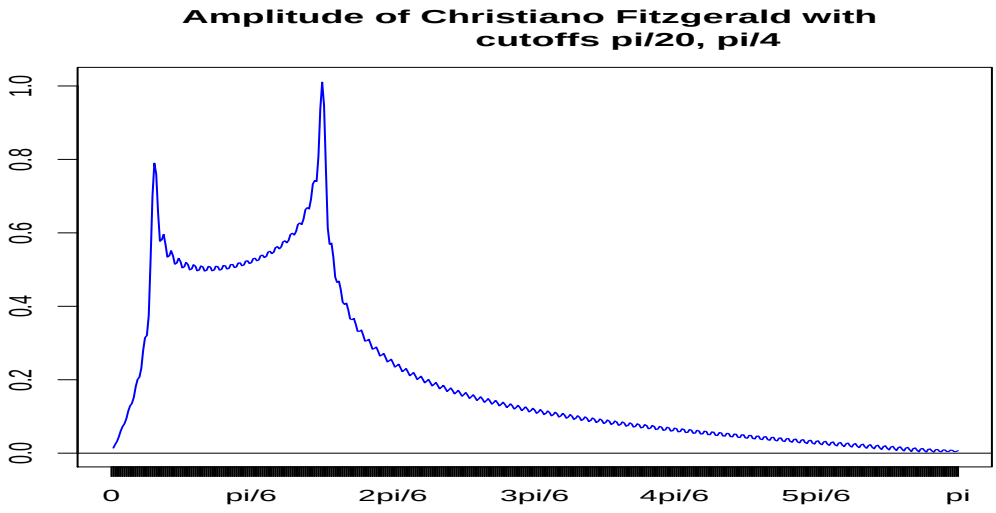


Figure 9.27: Amplitude function of the real-time Christiano Fitzgerald filter as replicated by DFA and assuming that the DGP is a random-walk

3. Check the filter constraints: the transfer function should vanish in the unit-root frequency zero.

```
> # Check first-order: should give 0
> print(paste("Transfer function in frequency zero: ",
+             round(sum(imdfa_cf$b),3),sep=""))

[1] "Transfer function in frequency zero: 0"

> # Check second-order: time-shift is not constrained
> print(paste("Time-shift in frequency zero: ",
+             round((1:(L-1))%*%imdfa_cf$b[2:L],3),sep=""))
```

```
[1] "Time-shift in frequency zero: -2.154"
```

The first-order restriction $\hat{\Gamma}^{CF}(0) = 0$ is fulfilled, as expected.

4. Compute real-time CF-filter coefficients according to the function $cffilter$ (package $mFilter$[25]) and check the (first-order) filter constraint.

```
> # Setting theta=1 replicates the random-walk model
> #   We first set root=F (no unit-root)
> x_cf<-cffilter(x,pu=len2,pl=len1,root=F,drift=F, nfix=NULL,theta=1)
> parm_cf<-x_cf$fmatrix
> # Check first-order: should give 0
> print(paste("Transfer function in frequency zero: ",
+             round(sum(parm_cf[,1]),3),sep=""))

[1] "Transfer function in frequency zero: 0.096"
```

The filter constraint is not fulfilled. Note, however, that the function $cffilter$ allows to specify a unit-root by setting $root = T$.

5. Select $root = T$, reestimate filter coefficients and check the filter constraint:

```
> x_cf_T<-cffilter(x,pu=len2,pl=len1,root=T,drift=F, nfix=NULL,theta=1)
> parm_cf_T<-x_cf_T$fmatrix
> # Check first-order: should give 0
> print(paste("Transfer function in frequency zero: ",
+             round(sum(parm_cf_T[,1]),3),sep=""))

[1] "Transfer function in frequency zero: -1.491"
```

Obviously, the code does not seem to work properly.

6. Another option would be to use the function $mFilter$ and to specify *filter="CF"*:

```
> x_mF_T<-mFilter(x,filter="CF",pu=len2,pl=len1,root=T,drift=F,
+                 nfix=NULL,theta=1)
> parm_mF_T<-x_mF_T$fmatrix
> # Check first-order: should give 0
> print(paste("Transfer function in frequency zero: ",
+             round(sum(parm_mF_T[,1]),3),sep=""))

[1] "Transfer function in frequency zero: -1.491"
```

Neither $mFilter$ nor $cffilter$ comply with the first-order constraint in frequency zero. Selecting $root = T$ results in a severly misspecified filter.

7. Compare real-time filter coefficients of DFA and $mFilter$ (or $cffilter$), see fig.9.28.

[25] Version 0.1-3.

```
> file = paste("z_CF_us_real_log_gdp.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> colo<-c("blue","red","green")
> mplot<-cbind(imdfa_cf$b,parm_cf[,1],parm_mF_T[,1])
> colnames(mplot)<-c("DFA","mFilter: plot=F","mFilter: plot=T")
> plot_title<-"Real-time CF-filters: DFA (blue) vs. mFilter (red and green)"
> freq_axe<-paste("Lag ",0:(len-1),sep="")
> title_more<-colnames(mplot)
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```

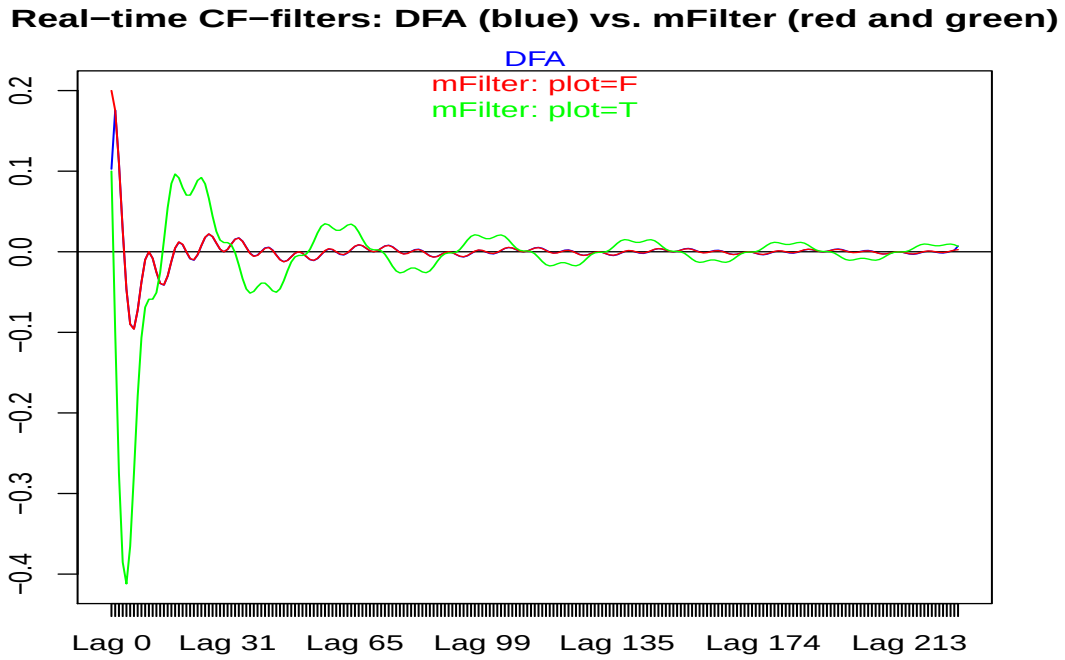


Figure 9.28: Real-time filter coefficients: DFA (blue) vs. mFilter with root=F (red) and root=T (green)

As can be seen, DFA (blue) and $mFilter$ or $cffilter$ based on $root = F$ (red) match closely up to the boundaries, lags 0 and 227, where non-negligible discrepancies can be observed. In contrast, the coefficients of $mFilter$ for $root = T$ (green line) are 'off the mark'.

8. Cross-check real-time DFA coefficients with the (time-domain) model-based solution proposed in section 9.2.2, see fig.9.29. Hint: our R-code relies on 9.3, whereby $a_1 = 1$ (random-walk).

```
> # Coefficients of symmetric filter (truncated at length 100000)
> ord<-100000
> b<-0:ord
> b[1+1:ord]<-(sin((1:ord)*2*pi/len1)-sin((1:ord)*2*pi/len2))/(pi*(1:ord))
> b[1]<-2/len1-2/len2
> # Real-time filter based on for- and backcasts
> b_finite<-b[1:len]
> # The lag-0 coefficient is augmented by forecasts
> b_finite[1]<-b_finite[1]+sum(b[2:ord])
> # The lag-len coefficient is augmenetd by backcasts
> b_finite[len]<-b_finite[len]+sum(b[(len+1):ord])
> # Compare DFA and model-based coefficients
> file = paste("z_CF_us_real_log_gdp_fb.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> colo<-c("red","blue")
> mplot<-cbind(b_finite,imdfa_cf$b)
> plot_title<-"Real-time CF-filters: Forecast/backcast (red) vs. DFA (blue)"
> freq_axe<-rep(NA,len)
> freq_axe[1]<-0
> freq_axe[(1:6)*len/6]<-paste("Lag ",as.integer(1+(1:6)*len/6),sep="")
> mplot_func(mplot,freq_axe,plot_title,title_more,insamp,colo)
> invisible(dev.off())
```

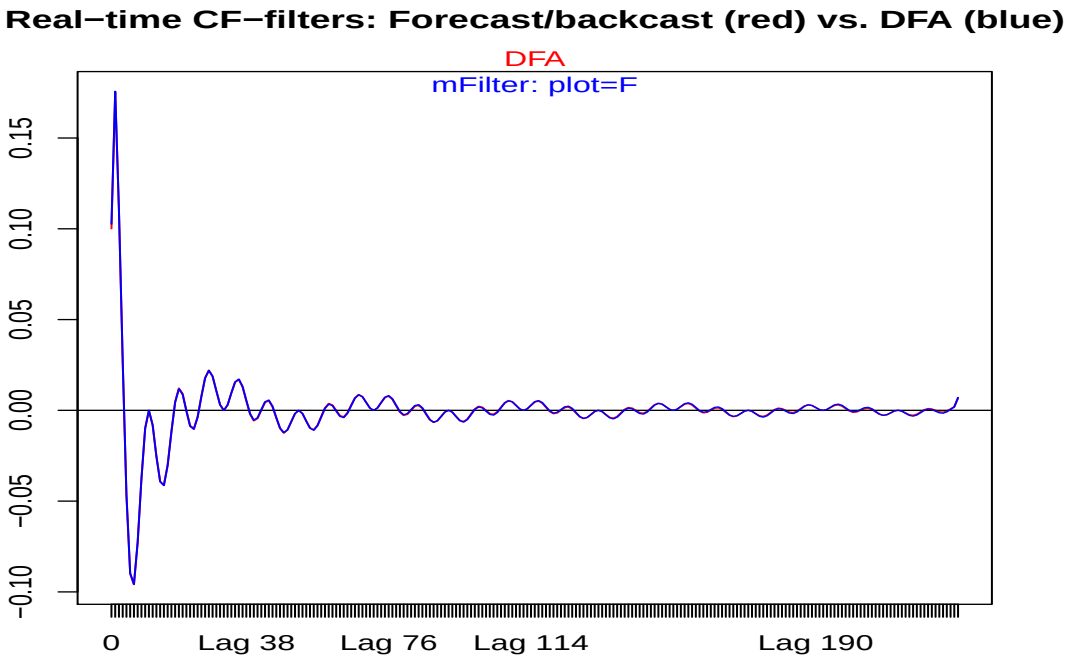


Figure 9.29: Real-time filter coefficients: model-based approach (red) vs. DFA (blue)

Real-time model-based and DFA filters now match at all lags[26] which confirms replication of CF by DFA.

9. Compute real-time cycle estimates based on DFA and on $mFilter$: select $root = F$ as well as $root = T$. Hint: set $L = 50$[27] and re-estimate filter coefficients accordingly.

```
> # Filter length
> L<-50
> weight_func<-weight_func_cf
> K<-nrow(weight_func)-1
> # Default-settings for DFA
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Filter constraints
> i1<-T
> i2<-F
> # We have to specify the level-constraint
> #   Real-time filter must equal target-Gamma in frequency zero
> weight_constraint<-Gamma_cf[1]
```

[26] Minor (negligible) deviations are due to the finite resolution of the frequency-grid.
[27] The filter length L must be smaller than T in order to obtain a time series of length $T - L + 1$ of filtered data.

```
> # Proceed to estimation
>
> imdfa_cf_L<-mdfa_analytic(L,lambda,weight_func_cf,Lag,Gamma_cf,eta,cutoff,
+                           i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                           lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                           b0_H0,c_eta,weight_structure,
+                           white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_cf_L<-MDFA_mse_constraint(L,weight_func_cf,Lag,Gamma_cf,i1,i2,weight_constraint,
+                                 shift_constraint)$mdfa_obj
> gamma_dfa<-as.vector(imdfa_cf_L$b)
> #--------------
> # Compute filter according to cffilter
> # We provide an arbitrary input series of length L
> # We set root=T
> x_cf_L_T<-mFilter(as.ts(1:L),filter="CF",pu=len2,pl=len1,root=T,drift=F, nfix=NULL,theta=1
> gamma_cf_L_T<-x_cf_L_T$fmatrix[,1]
> # Here we set root=F
> x_cf_L_F<-mFilter(as.ts(1:L),filter="CF",pu=len2,pl=len1,root=F,drift=F, nfix=NULL,theta=1
> gamma_cf_L_F<-x_cf_L_F$fmatrix[,1]
```

10. Apply these filters to the GDP-data .

```
> # Filter the data
> dfa_cycle<-cff_cycle_T<-cff_cycle_F<-x
> for (j in L:len)
+ {
+ # DFA
+   dfa_cycle[j]<-gamma_dfa%*%x[j:(j-L+1)]
+ # mFilter, root=T
+   cff_cycle_T[j]<-gamma_cf_L_T%*%x[j:(j-L+1)]
+ # mFilter, root=F
+   cff_cycle_F[j]<-gamma_cf_L_F%*%x[j:(j-L+1)]
+ }
> # Skip initial values
> dfa_cycle[1:L]<-cff_cycle_T[1:L]<-cff_cycle_F[1:L]<-NA
```

11. Compare the resulting filter outputs. Hint: standardize the series for ease of visual inspection, see fig.9.30.

```
> #-------------
> # Plot the filtered outputs
> file = paste("z_HP_us_real_log_gdp_cf_out.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
```

```
+     width = 6, height = 6)
> # Plots: filter coefficients and series
> insamp<-1.e+99
> mplot<-scale(cbind(dfa_cycle,cff_cycle_T,cff_cycle_F,
+                    x_cf$cycle))
> plot_title<-"CF real-time cycle: DFA (blue) and mFilter root=T (red) and
+ root=F (green): all standardized"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,2],col="red",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> nberShade()
> lines(mplot[,2],col="red")
> lines(mplot[,1],col="blue")
> lines(mplot[,3],col="green")
> #lines(mplot[,4],col="black")
> invisible(dev.off())
```

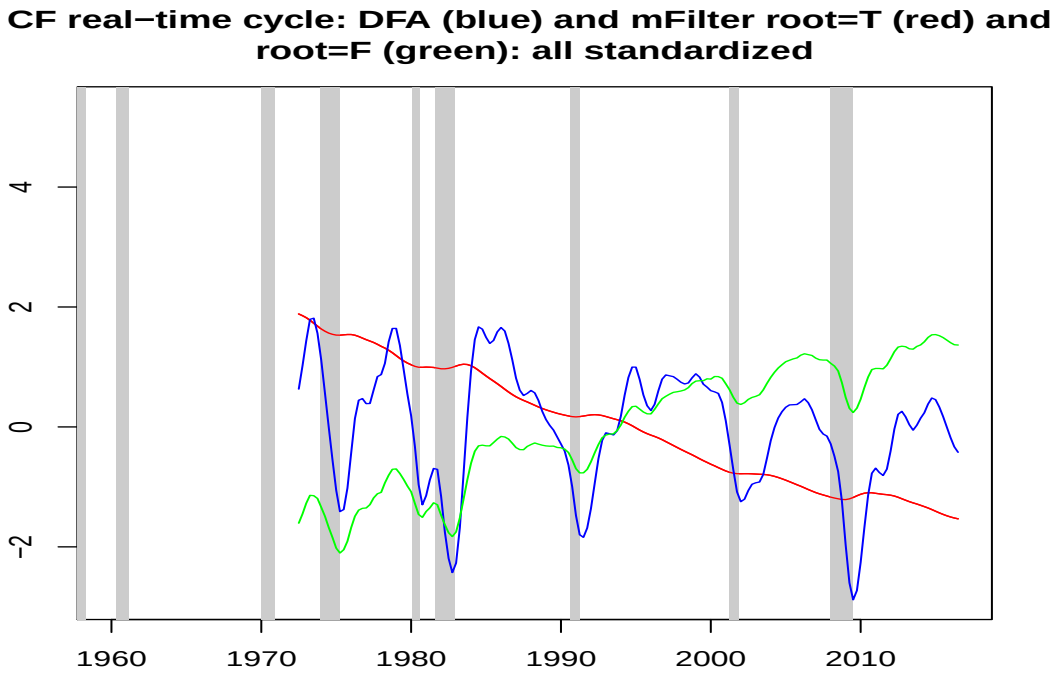


Figure 9.30: CF real-time cycle estimates: DFA (blue), mFilter with unit-root (red) and mFilter without unit root (green). All filter outputs are standardized for ease of visual inspection

As expected, the alleged 'cycles' computed by $mFilter$ (or $cffilter$) are non-stationary because the filters do not comply with the first-order constraint in frequency zero.

12. Plot the cycle as computed (internally) by the function $mFilter$ (select $root = F$), see fig.9.31.

```
> # Plot the filtered outputs
> file = paste("z_HP_us_real_log_gdp_cf_cycle.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special",
+     width = 6, height = 6)
> # Plots: filter coefficients and series
> insamp<-1.e+99
> mplot<-as.matrix(x_cf$cycle)
> plot_title<-"Cycle as computed by cffilter"
> ymin<-min(mplot,na.rm=T)
> ymax<-max(mplot,na.rm=T)
> plot(mplot[,1],col="black",xlab="",ylab="",main=plot_title,type="l",
+      ylim=c(ymin,ymax))
> nberShade()
> lines(mplot[,1],col="black")
> invisible(dev.off())
```

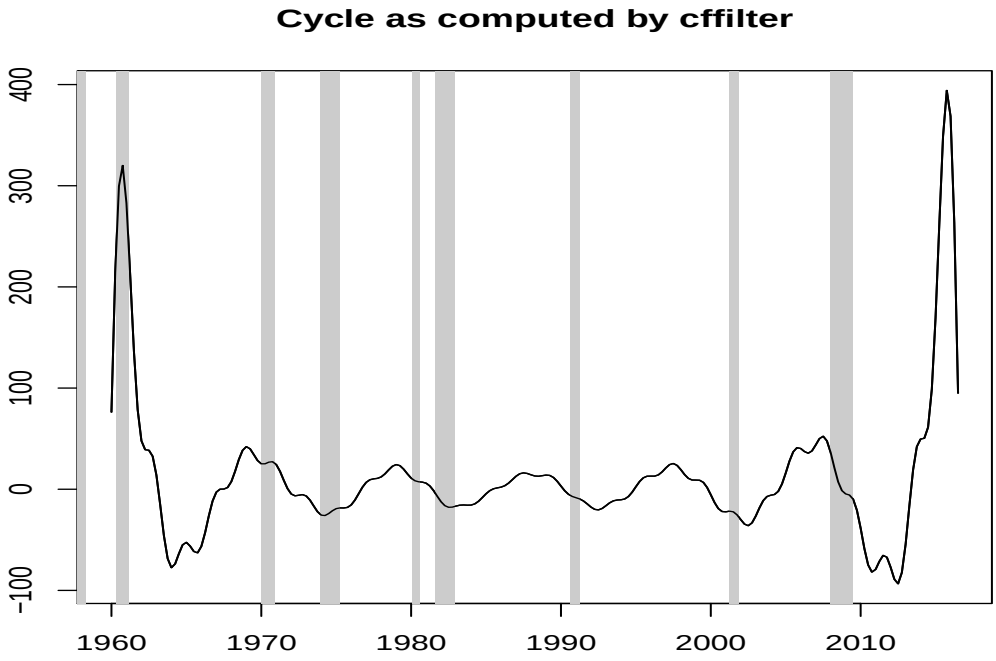


Figure 9.31: Christiano Fitzgerald cycle as computed by function mFilter

The plot confirms that the CF-filter is not implemented properly in the package *mFilter*.

The DFA replicates perfectly the CF-filter, as confirmed by our comparison with the classic model-based (time-domain) approach, and it allows for straightforward extensions to arbitrary DGP and/or target specifications.

9.7.3 Customization

Customization of a Bandpass Design: the Case of the CF-Filter

Once replicated by DFA, the real-time CF-filter can be customized. In contrast to all previous examples, which were based on lowpass trend targets, we here illustrate the proceeding in the case of a *bandpass* cycle target. The main distinguishing feature of the former is the two-sided stopband: to the left and to the right of the passband. This distinction is important in so far that Smoothness refers to the stopband. By default, the MDFA-routine *mdfa_analytic* emphasizes the right hand portion, corresponding to the higher frequencies, only. The weighting function $W(\omega_k, \eta, \text{cutoff})$ in 7.7 becomes

$$W(\omega_k, \eta, \text{cutoff}^u) = \begin{cases} 1 \ , \ \text{if } |\omega_k| < \text{cutoff}^u \\ (1 + |\omega_k| - \text{cutoff}^u)^\eta \ , \ \text{otherwise} \end{cases}$$

where cutoff^u is the upper cutoff-frequency. In general the left hand portion of the stopband, corresponding to the trend, is damped anyway by imposing constraints at frequency zero and therefore a corresponding extension of the above weight function is unnecessary.

Lowpass and bandpass designs generally differ in terms of Timeliness, too. As an example, amplitude and time-shift functions of a real-time CF (MSE) filter of length L =50 are plotted in fig.9.32[28] (for ease of visual inspection time-shift values smaller than -1 are discarded):

```
> file = paste("z_amp_shift_dfa_mse_cf.pdf", sep = "")
> pdf(file = paste(path.out,file,sep=""), paper = "special", width = 6, height = 6)
> omega_k<-pi*(0:K)/K
> amp_mse<-abs(imdfa_cf_L$trffkt)
> shift_mse<-shift_forecast<-Arg(imdfa_cf_L$trffkt)/omega_k
> shift_mse[which(shift_mse<(-1))]<-NA
> mplot<-cbind(amp_mse,shift_mse)
> mplot[1,1:2]<-NA
> dimnames(mplot)[[2]]<-c("Amplitude MSE","Time-shift MSE")
> ax<-rep(NA,nrow(mplot))
> ax[1+(0:6)*((nrow(mplot)-1)/6)]<-c(0,"pi/6","2pi/6","3pi/6","4pi/6","5pi/6","pi")
> plot_title<-"Amplitude and time-shift CF (MSE) real-time filter"
> insamp<-1.e+90
> title_more<-dimnames(mplot)[[2]]
```

[28] Note that the amplitude functions in figs.9.27 and 9.32 differ sligthly because the filter-lengths are different: L =227 for the former vs. L =50 for the latter.

```
> colo<-c("blue","red")
> mplot_func(mplot, ax, plot_title, title_more, insamp, colo)
> invisible(dev.off())
```

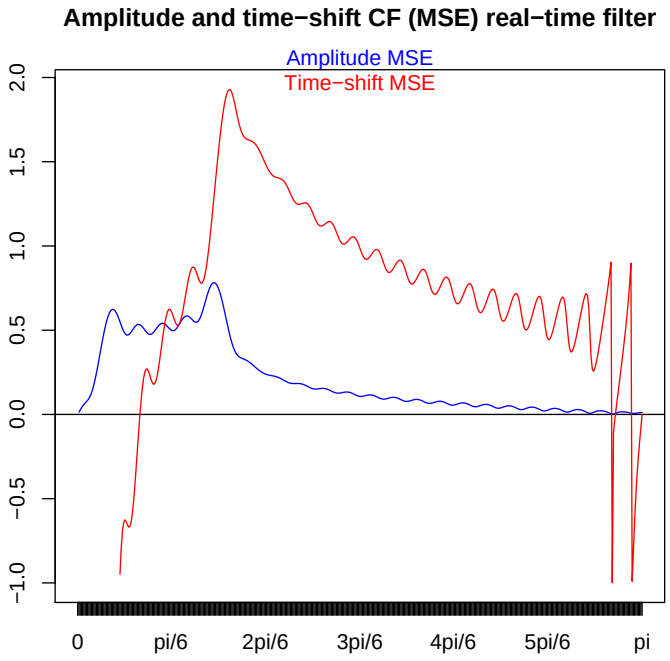


Figure 9.32: Amplitude (blue) and time-shift (red) functions of CF MSE real-time filter: time-shifts smaller than -1 are skipped

We observe that the time-shift is small, even negative, in the passband of the filter[29] and therefore timeliness is less relevant, at least in a nowcast perspective ($Lag = 0$). However, for some particular applications, like for example the construction of a *leading* business-cycle indicator, an effective anticipation is desirable[30]. We can address this problem by emphasizing a signal-*forecast*, $Lag < 0$, instead of a nowcast, $Lag = 0$. As proposed in section 7.8 we here extend the original customization pair (λ, η) and consider the customization triplet (Lag, λ, η), instead.

[29]This property is typical for bandpass designs and, more generally, for filters whose amplitude functions dip at frequency zero. The negative time-shift of the bandpass is inherited from the difference filter: typically, the transfer function $\hat{\Gamma}^{bp}(\omega)$ of a (finite length) real-time bandpass can be decomposed into $\hat{\Gamma}^{bp}(\omega) = (1 - \exp(-i\omega))^d \hat{\Gamma}(\omega)$ where d is the order of the zero at frequency zero and where $\hat{\Gamma}(\omega)$ is a lowpass which damps high-frequency noise. The negative time-shift of the bandpass towards lower frequencies is inherited from the difference operator(s).

[30]The leading indicator LEI of the OECD relies on a bandpass concept. It targets a lead of 6 months.

Exercises: Customization of CF-Bandpass

We emphasize Timeliness by targetting a forecast of the signal and we emphasize Smoothness by emphasizing the right portion of the stopband.

1. Set $Lag = -4$ (target is shifted by one year), $\lambda = 10$ (emphasize Timeliness) and $\eta = 1$ (emphasize Smoothness) and derive the cofficients of a corresponding filter of length 50. We propose generic as well as context-specific function-calls in our R-code below.

```
> # Filter length
> L<-50
> weight_func<-weight_func_cf
> K<-nrow(weight_func)-1
> source(file=paste(path.pgm,"control_default.r",sep=""))
> # Filter constraints
> i1<-T
> i2<-F
> Lag<--4
> lambda<-10
> eta<-1
> cutoff<-2*pi/len1
> # We have to specify the level-constraint
> #   Real-time filter must equal target-Gamma in frequency zero
> weight_constraint<-Gamma_cf[1]
> # Proceed to estimation
>
> imdfa_cf_L_cust<-mdfa_analytic(L,lambda,weight_func_cf,Lag,Gamma_cf,eta,cutoff,
+                         i1,i2,weight_constraint,lambda_cross,lambda_decay,
+                         lambda_smooth,lin_eta,shift_constraint,grand_mean,
+                         b0_H0,c_eta,weight_structure,
+                         white_noise,synchronicity,lag_mat,troikaner)
> # Alternative (identical) context-specific estimation:
> imdfa_cf_L_cust<-MDFA_cust_constraint(L,weight_func_cf,Lag,Gamma_cf,cutoff,
+               lambda,eta,i1,i2,weight_constraint,shift_constraint)$mdfa_obj
>
```

2. Filter the data and compare outputs of MSE-nowcast and customized-forecast filters, see fig.9.33.

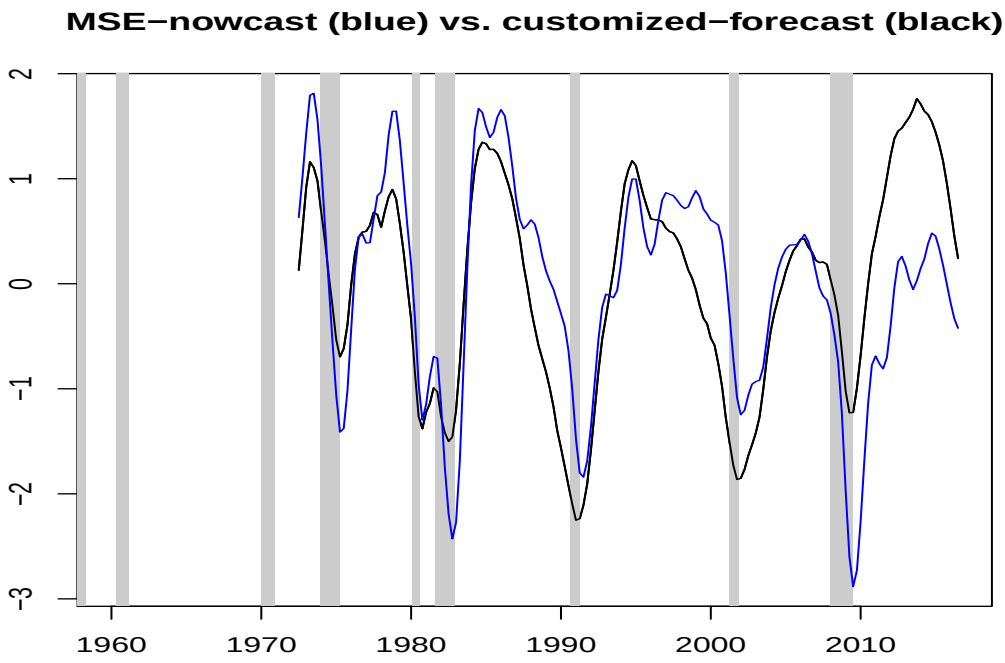


Figure 9.33: CF real-time cycle estimates: DFA MSE-nowcast (blue) vs. DFA customized-forecast (black): both standardized for ease of visual inspection

The output of the customized forecast filter is smooth and tends to lie to the left of the MSE-nowcast, as desired. Note that the emphasis of Smoothness, as entailed by our selection $\eta = 1$, precludes the forecast filter to effectively anticipate the nowcast MSE-filter by $-Lag$ =4 quarters. In any case, alternative settings of the customization triplet (Lag, λ, η) are open for further experimentation.

To conclude, let us re-emphasize that neither of the above results, derived in the context of classic HP- and CF-filter designs, rely on data: implicit models, and thus implicit spectra, are specified exogeneously. Let us remind, also, that the quarterly US-GDP time series is subject to publication-lags and to revisions. A proper proceeding in the context of data revisions is proposed in chapter 14.

9.8 Summary

- The generic spectrum-interface of MDFA allows straightforward replication of alternative ARIMA or state-space model-based approaches.
- Implicit model-based perspectives can be assigned to classic Hodrick-Prescott and Christiano-Fitzgerald filter designs. Therefore, replication by the MDFA is straightforward too, as

illustrated by our empirical examples.

- Once replicated, any of these approaches can be customized by relying on the generic ATS-trilemma. Applications to US-GDP illustrate the proceeding.

- A customized explicit or implicit model-based approach is a *hybrid*: the spectrum relies on (pseudo-) maximum likelihood but the filter is obtained by emphasizing the filter error and by leaving a restricted mean-square perspective by ways of the ATS-trilemma.

- Model-based spectra are generally smooth (except for unit-root singularities), because time series models typically rely on few parameters. If the model is neither overfitted nor misspecified, then the derived DFA-filter, be it MSE or customized, be it a forecast or a nowcast, is free of overfitting and misspecification, too, irrespective of the size of the filter-length L.

- A selection of a particular spectrum (model-based, DFT or other) remains a matter of preference and of experience of the user.

- The customization triplet (Lag, λ, η) usefully extends the action of the ATS-trilemma to arbitrary real-valued leads, $Lag < 0$, or lags, $Lag > 0$. A corresponding application to the bandpass Christiano-Fitzgerald filter illustrates the proceeding when designing a business-cycle indicator based on US-GDP.

Chapter 10

Reverse Engineering

Consider the generic MSE-expression

$$\sum_k |\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)|^2 h(\omega_k)$$

In contrast to the previous chapters, where $\hat{\Gamma}(\omega_k)$ was obtained as the outcome of the MSE-criterion

$$\sum_k |\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)|^2 h(\omega_k) \rightarrow \min_{\hat{\Gamma}}$$

for given Γ and h, we here consider cases where $\hat{\Gamma}(\omega_k)$ is **already known/given** (for example by a customer) and we try to determine either $\Gamma(\omega_k)$, for given $h(\omega_k)$, or the positive weighting function $h(\omega_k) > 0$[1], for given $\Gamma(\omega_k)$, from either

$$\sum_k |\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)|^2 h(\omega_k) \rightarrow \min_{\Gamma}$$

or from

$$\sum_k |\Gamma(\omega_k) - \hat{\Gamma}(\omega_k)|^2 h(\omega_k) \rightarrow \min_{h(\omega_k)}$$

This is called reverse-engineering since we look for the problem formulation which lead to the pre-specified 'solution' $\hat{\Gamma}$.

- In the first criterion, we assume the weighting function $h(\omega_k)$ to be given (for a model-based spectrum or DFT) and we look for a target $\Gamma(\omega_k)$ which minimzes the MSE i.e. we look for determining the original 'problem', as specified by the unknown target $\Gamma(\omega_k)$, given $h(\omega_k)$ (a model or the DFT). Note that this is essentially the same as prior (real-time/concurrent) estimation problem except that the roles of Γ and $\hat{\Gamma}$ are now interchanged: the new target $\hat{\Gamma}$ is an asymmetric filter and the filter subject to optimization, Γ, must by symmetric which is easily obtained by setting $Lag = T/2$ in MDFA.
- In the second problem formulation we seek an 'optimal' model description which would generate the given $\hat{\Gamma}(\omega_k)$ as solution of the MSE-problem given the target Γ, but recall above footnote.

[1] The function h is determined up to a scaling constant i.e. it is assumed that $\sum_k h(\omega_k)$ is fixed.

Examples:

- Assume $h = constant > 0$ (white noise) and $\hat{\Gamma}$ is equally weighted MA(200). Which target (problem) is associated to $\hat{\Gamma}$ in the first place?
- Assume $\Gamma(\omega_k) = \exp(i\omega_k)$ is the anticipative (one-step ahead) allpass and $\hat{\Gamma}$ is the equally weighted MA(T) of length T equal to the sample length. Which weighting function h (model) is associated to $\hat{\Gamma}$ in the first place?

Background: filter zoo in financial trading.

Reverse-engineering is crucial when replicating designs (for example classic model-based designs or, more generally, customer solutions) in MDFA for later customization. It is also useful when assessing the pertinence of an alleged 'proven' solution $\hat{\Gamma}(\omega_k)$:

- For given $\hat{\Gamma}(\omega_k)$ and target $\Gamma(\omega_k)$ obtain $S(\omega_k)$ Translation: under which data-conditions (data-generating process or DGP) is the solution optimal in view of the specified target (this proceeding is sometimes found in the literature when discussing the pertinence of a particular design)
- For given $\hat{\Gamma}(\omega_k)$ and $S(\omega_k)$ obtain $\Gamma(\omega_k)$ Translation: for what kind of problems (target) is the proposed solution optimal under the given data conditions (I haven't seen this kind of evaluation in the literature because the target is mostly one-step ahead forecasting i.e. the target is fixed (not assumed to be the outcome of the optimization)).

Chapter 11

Exotic Optimization Criteria

11.1 Introduction

Classic MSE and Customization: From AT-dilemma to ATS-trilemma. Alternative performances (not directly signal extraction): exotic criteria.

11.2 Hybrid Criterion

11.2.1 SE-Performance

11.2.2 Emphasizing Trading Performance

Maximizing Profits: Time vs. Frequency-Domain Perspectives

A Scaling Problem

A Projection Approach

11.3 Double-Decker

multi-stage

11.3.1 Instilling Performance

ATS+one-step ahead

11.3.2 Extracting Performance

11.4 Radical departure MSE

Emphasize turning-points (zero-crossings): holding time (width) and delay (relative position) EaSY GENERALIZATION TO MULTIVARIate: like multivariate customization (for holding-

time we are only interested in the acf(1) corresponding to univariate output of multivariate filter; and similarly for delay)

Use crossover bandpass as start-value for optimization

Chapter 12

Inference

Refer to DFA-article with Tucker. Refer to 2005 and 2008 books + elements paper. I won't assign too much time to this topic because it is more or less irrelevant for applications; it is less relevant than for model-based approaches because we emphasize the filter (not the DGP). Generalization(s) of well-known model-based statistics. Maybe additional contributions by Tucker about the multivariate case. My own results include asymptotic distribution of filter coefficients (2005-book) and a generalized unit-root test (2008-book).

Chapter 13

Overfitting and Regularization

13.1 Introduction

Overfitting refers to an undesirable phenomenon concomitant to fitting a filter (or a model) to data according to a particular optimization criterion: the fitted device matches 'perfectly', in the sense of the criterion, a singular realization of the DGP at the expense of generally poorer fit for yet unseen data. We here propose to tackle overfitting by shrinking the ambient space $\mathbb{R}^{L(m+1)}$ of the *filter* coefficients in such a way that desirable – universal – properties of the latter are obtained[1]. The commonly observed trade off between misspecification and overfitting can then be alleviated, to some extent, by realizing that the required universal features, as emphasized by the regularized designs, are shared by the 'truly best' (unobserved) coefficients, too. Succinctly, we here address the hyper-parameters $\lambda_{decay}, \lambda_{cross}$ and λ_{smooth} in the call of our generic MDFA-function

```
> head(mdfa_analytic)
```

```
1 function (L, lambda, weight_func, Lag, Gamma, eta, cutoff, i1,
2     i2, weight_constraint, lambda_cross, lambda_decay, lambda_smooth,
3     lin_eta, shift_constraint, grand_mean, b0_H0, c_eta, weight_structure,
4     white_noise, synchronicity, lag_mat, troikaner)
5 {
6     lambda <- abs(lambda)
```

In section 13.2 we contrast overfitting from a classic (one-step ahead forecasting) and from a general signal extraction perspective; section 13.3 briefly reviews methods for tackling overfitting and introduces the so-called Regularization Troika; sections 13.4 to 13.6 introduce, describe, analyze and illustrate the quadratic bilinear forms of the Regularization Troika; interaction of the regularization terms and implicit data-requirements are discussed in section 13.7; section 13.8 derives closed form solutions of the regularized MDFA-estimate and section 13.9 generalizes the concept of 'degrees of freedom' to the Regularization Troika.

[1] In contrast to classic model-based approaches, the DFA emphasizes the relevant filter coefficients.

13.2 Overfitting: a Signal-Extraction Perspective

Overfitting of a particular device – filter, model – in association with a particular optimization criterion, refers to the fact that in-sample realizations of the criterion systematically better out-of-sample realizations thereof. This undesirable discrepancy arises as a direct consequence of the perfected in-sample fit, which feigns an unrealistically optimistic picture of the underlying estimation problem. Ultimately, tackling overfitting is concerned about improving out-of-sample performances of the fitted device.

13.2.1 In- and Out-of-Sample Spans

In a classic one-step ahead forecast perspective, the target $y_t = x_{t+1}$ is either observed (for $t = 1, ..., T-1$) or unobserved: there is no mid-way. Accordingly, the in-sample span coincides unequivocally with the time period $t = 1, ..., T-1$ for which the target is observed. In contrast, the more general signal extraction target

$$y_t \quad = \quad \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k} \tag{13.1}$$

which can involve arbitrarily many future and past realizations, does not allow for a clear-cut distinction of in-sample and out-of-sample time spans anymore. Near the middle of the sample, $t \approx T/2$, the target y_t can be determined accurately, assuming γ_k converge to zero sufficiently rapidly, and therefore observations in the middle of the sample may be claimed to be 'in sample'; towards the sample-end $t = T$, however, the weights assigned to the out-of-sample portion $x_{T+1}, x_{T+2}, ...$ of the target in 13.1 generally increase; eventually, for h sufficiently large, y_{T+h} may be claimed to be 'entirely' out-of-sample. In contrast to classic one-step ahead forecasting, the transition from in-sample performances (fully observed target) to out-of-sample performances (unobserved target) is therefore generally gradual and smooth instead of abrupt and discontinuous; the transition depends essentially on the rate of decay of the filter coefficients γ_k; for one-step ahead forecasting, the decay is instantaneous and therefore an unequivocal clear-cut distinction of in-sample and out-of-sample spans is obtained. If L is large, then an overly optimistic fit of y_t by the real-time estimate

$$\hat{y}_t^0 := \sum_{k=0}^{L-1} b_{k0} x_{t-k}$$

in the middle of the sample could result in a poor real-time estimate $\hat{y}_T^0$ of y_T towards the practically relevant end-point $t = T$, in which case the real-time filter b_{k0} would have been overfitted. It is important to realize that $\hat{y}_T^0$ is, to a large extent, an out-of-sample estimate although the time point to which it refers, namely T, is literally in (the) sample. To conclude, note that the DFA targets $\Gamma(\cdot)\Xi_{TX}(\cdot)$ which, contrary to y_t, is observable. Unfortunately, the transformation of the data into the frequency-domain cannot prevent against the inescapable fact that y_t is less well determined towards the sample-end[2] and therefore the (mean-square) filter-error in the middle of the sample will generally be smaller than towards the boundaries.

[2] In the frequency-domain, the DFT 'does the trick' by assuming circularity of the data. In the absence of circularity (periodicity) overfitting applies in a similar way.

13.2.2 Empirical Examples

To be filled...

13.3 Tackling Overfitting: a Short Review

Restricting the ability of a device – filter or model – to 'fit' data generally improves congruency of in-sample and out-of-sample performances. We here review some well-known methods, ranging from simple brute-force to more refined non-parametric and parametric approaches. To conclude, we prolong the list by introducing the Regularization Troika.

13.3.1 Hard (Filter-) Constraints

Brute Force

The ability of a filter to fit data depends on the number $L(m+1)$ of its freely determined coefficients. Overfitting can be tamed by reducing either L (shorter filters) or m (less explaining variables), or both. Eventually, such decisions might be rationalized by relying on pertinent expert-knowledge, in some specific cases. However, the overall proceeding can be qualified as a brute-force approach for tackling a problem whose essence would deserve a more refined – statistical – treatment.

Loaded Constraints

The number of freely determined coefficients can be reduced by imposing filter-constraints which match a particular problem structure and/or particular user-priorities: the level- and time-shift constraints proposed in chapter 6 ($i1 = i2 = T$) are typical examples. Note that neither L nor m are affected by these constraints; instead, a particular meaningful – loaded – structure is imposed upon the longitudinal interplay of the coefficients.

13.3.2 Smoothing the Spectrum

In the (M)DFA-framework, the data is expressed in terms of its frequency-domain representation, the DFT. Instead of addressing degrees of freedom of the filter coefficients, we could emphasize degrees of freedom of the DFT. This way, we may maintain the (full-blown) ambient space $\mathbb{R}^{L(m+1)}$ of the (M)DFA intact, conditional on the DFT being of 'reduced rank'.

Non-Parametric Approaches

Classical distribution theory suggests that the periodogram is an unbiased but noisy estimate of the true spectral density of the DGP. A better estimate of the spectrum, in terms of mean-square performances, could be obtained by trading unbiasedness against variance-reduction or, more explicitly, by applying a suitable Kernel-smoother to adjacent periodogram ordinates in order to reduce randomness[3]. If $L = T$ (full-blown ambient space), then the (M)DFA-filter based on the smoothed spectrum inherits degrees of freedom directly from the latter. Increased smoothness

[3]For various reasons we abstain from smoothing the periodogram or the DFT.

shrinks degrees of freedom and tames overfitting (eventually ending-up in more or less heavy misspecification and inefficiency).

Parametric (Model-Based) Approach

According to chapter 9, the (M)DFA can replicate classic model-based approaches by exchanging the periodogram (or the DFT) for a corresponding model-based estimate in the optimization criterion. If $L = T$, then the MBA can be replicated up to arbitrary precision by the (M)DFA and therefore degrees of freedom of (M)DFA and MBA must coincide. In the case of replication and/or customization, the (M)DFA-filter thus inherits parsimony from the model.

13.3.3 Regularization Troika

Far from being exhaustive, the above discussion delineates a path from simple brute-force to more refined parametric approaches for taming overfitting. The Regularization Troika, to be detailed below, prolongs this path: instead of constraining the spectrum (the DFT), it emphasizes the filter coefficients; instead of emphasizing one-step ahead forecasting performances, it addresses the filter error, directly; instead of imposing hard constraints, it incentivizes the (M)DFA-criterion. The last point is achieved by parametrizing a triplet of universal 'features': the longitudinal rate of decay, the longitudinal smoothness and the cross-sectional similarity of the filter coefficients. Specifically, the original MDFA-criterion is augmented by three positive definite quadratic bilinear forms

$$MDFA + f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}^m_{decay}\mathbf{b} + f(\lambda_{smooth})\mathbf{b}'\mathbf{\Lambda}^m_{smooth}\mathbf{b} + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}^m_{cross}\mathbf{b} \rightarrow \min_{\mathbf{b}}$$

The newly introduced quadratic terms penalize departures of the original MDFA-estimate from the proposed requirements. We now define and analyze each penalty-term of the Regularization Troika and provide empirical illustrations of its specific capacities.

13.4 Longitudinal (Rate of) Decay

In general, observations in the remote past of a time series are less relevant for determining current and future states – nowcasts or forecasts – of a time series than fresher data. As a consequence one expects that filter coefficients should converge to zero with increasing lag. This universal pattern can be facilitated by introducing a suitable penalty term amending additively the original optimization criterion.

13.4.1 Quadratic Bilinear Form: Univariate Framework

We here emphasize univariate designs in a nowcast-perspective. Extensions to multivariate designs as well as to backcasting and/or forecasting are discussed below. Let $y_T = \sum_{k=-\infty}^{\infty} \gamma_k x_{T-k}$ and $\hat{y}_T = \sum_{k=0}^{L-1} b_k x_{T-k}$. The regularized criterion is obtained as

$$DFA + \lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}^0_{decay}\mathbf{b} \rightarrow \min_{\mathbf{b}}$$

where *DFA* stands for any of the previous DFA-criteria (MSE or customized, with or without constraints) and where $\mathbf{\Lambda}^0_{decay}$ is a diagonal matrix of dimension $L * L$ with diagonal-elements $(1+\lambda_{decay,1})^k$, $k = 0, ..., L-1$

$$\mathbf{\Lambda}^0_{decay} = \begin{pmatrix} 1 & 0 & 0 & ... & 0 \\ 0 & 1+\lambda_{decay,1} & 0 & ... & 0 \\ 0 & 0 & (1+\lambda_{decay,1})^2 & ... & 0 \\ 0 & 0 & 0 & ... & (1+\lambda_{decay,1})^{L-1} \end{pmatrix}$$

and where $\lambda_{decay,1} \geq 0, \lambda_{decay,2} \geq 0$ account for the *shape* as well as for the *strength* of the decay-regularization. We note that

$$\mathbf{b}'\mathbf{\Lambda}^0_{decay}\mathbf{b} = \sum_{k=0}^{L-1}(1+\lambda_{decay,1})^k b_k^2$$

is a strictly positive definite bilinear form of the filter coefficients. For convenience, a non-linear monotonic transformation $f(\lambda_{decay,2})$ of $\lambda_{decay,2}$ can be used

$$DFA + f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b} \rightarrow \min_{\mathbf{b}} \tag{13.2}$$

such that $f(0) = 0$ and $f(1) = \infty$. If $\lambda_{decay,2} = 0$ then 13.2 reduces to the original DFA-criterion; if $\lambda_{decay,2} \rightarrow 1$ then full (asymptotically infinite) weight is assigned to the penalty term i.e. the data is ignored and the solution is projected onto zero by the (strictly positive definite) bilinear-form ($\hat{\mathbf{b}} = \mathbf{0}$, asymptotically); for $0 < \lambda_{decay,2} < 1$ the criterion balances data-requirements, on one side, and regularization-preferences, on the other side. For $\lambda_{decay,1} = 0$ all coefficients are shrunken equally, irrespective of their lag; for $\lambda_{decay,1} > 0$ high-lag coefficients are shrunken more heavily such that data in the remote past will be discounted accordingly. The degrees of freedom shrink continuously from the maximum value L, when $\lambda_{decay,2} = 0$ (no regularization), to zero when $\lambda_{decay,2} = 1$.

13.4.2 Backcasting and Forecasting

For a backcast $\hat{y}^h_{T+h} := \sum_{k=h}^{L-1+h} b_{kh}x_{T+h-k}$ of y_{T+h}, $h < 0$, the most important observation is no more x_T, in general, but $x_{T+h}, h < 0$, instead. As a consequence, the decay-regularization should be amended in order to emphasize shrinkage on both sides – to the left and to the right – of $x_{T+h}, h < 0$. Specifically, we may want to replace the above bilinear form $\mathbf{\Lambda}^0_{decay}$ by

$$\begin{pmatrix} (1+\lambda_{decay,1})^{-h} & 0 & 0 & ... & 0 \\ 0 & (1+\lambda_{decay,1})^{-h-1} & 0 & ... & 0 \\ 0 & 0 & (1+\lambda_{decay,1})^{-h-2} & ... & 0 \\ ::: & & & & \\ 0 & 0 & 0 & ... & (1+\lambda_{decay,1})^{L-1+h} \end{pmatrix}$$

If $h = 0$ (nowcast) then this expression reduces to the earlier one. In the case of backcasting, $h < 0$, a symmetric shrinkage is exerted on both sides of x_{T+h}, as desired. In the case of forecasting, i.e. if $h > 0$, then the most important observation is generally (though not always) x_T and therefore

we may want to apply exactly the same scheme as for nowcasting. The following slightly modified bilinear-form fulfills all requirements

$$\mathbf{\Lambda}^{0,h}_{decay} = \begin{pmatrix} (1+\lambda_{decay,1})^{\max(0,-h)} & 0 & 0 & ... & 0 \\ 0 & (1+\lambda_{decay,1})^{\max(0,-h)-1} & 0 & ... & 0 \\ ::: & & & & \\ 0 & 0 & 0 & ... & (1+\lambda_{decay,1})^{L-1-\max(0,-h)} \end{pmatrix} \quad (13.3)$$

The maximum $\max(0,-h)$ in the exponent ensures that forecasting and nowcasting are handled equally.

13.4.3 Multivariate Framework

The extension to a multivariate framework, with explaining series $x_t,\ w_{1t},...,w_{mt},\ m > 0$, is straightforward:

$$MDFA + \lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}^{m}_{decay}\mathbf{b} \rightarrow \min_{\mathbf{b}}$$

where $\mathbf{\Lambda}^{m}_{decay}$ is an $L(m+1)*L(m+1)$-dimensional diagonal matrix

$$\mathbf{\Lambda}^{m}_{decay} = \begin{pmatrix} \mathbf{\Lambda}^{0}_{decay} & 0 & ... & 0 \\ 0 & \mathbf{\Lambda}^{0}_{decay} & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & \mathbf{\Lambda}^{0}_{decay} \end{pmatrix}$$

where $\mathbf{\Lambda}^{0}_{decay}$ from the univariate case is replicated $m+1$ times along the diagonal of $\mathbf{\Lambda}^{m}_{decay}$ and where filter coefficients are stacked in a long column vector $\mathbf{b} = Vec(\mathbf{B})$, of dimension $L(m+1)$. Note that it is implicitly assumed that all time series are coincident (synchronized) since the same block $\mathbf{\Lambda}^{0}_{decay}$ is used throughout. Otherwise, the more general expression 13.3 could be substituted.

13.4.4 Simple Example

In order to illustrate the new regularization feature we here rely on a simple bivariate design based on the second process (AR(1) with coefficient $a_1 = 0.1$) and realizations of length $T = 120$.

```
> set.seed(1)
> len<-120
> eps1<-arima.sim(list(ar=0.1),n=len)
> eps2<-arima.sim(list(ar=0.1),n=len)
> # Define the data-matrix:
> # The first column must be the target series.
> data_matrix<-cbind(eps1,eps1,eps2)
```

We then compute the DFTs of the data and specify the target signal, an ideal trend with cutoff $\pi/6$.

```
> # Determine the in-sample period (full in sample)
> insample<-nrow(data_matrix)
> # Compute the DFT: d=0 for stationary data (default settings)
> weight_func<-spec_comp(insample, data_matrix, d)$weight_func
> # Target
> Gamma<-(1:nrow(weight_func))<=(nrow(weight_func)-1)/6+1
```

Next, we estimate filter coefficients of a 'plain vanilla' real-time MSE-design (default hyperparameters: $\boldsymbol{\lambda}_{decay} = (0, 0)$) with filter-length $L = 12$ per series i.e. with $2 \cdot 12 = 24$ degrees of freedom.

```
> L<-12
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> troikaner<-T
> # Estimate filter coefficients: MSE
> mdfa_obj<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,i1,i2,weight_constraint,lambd
```

Next, we impose a non-vanishing decay-regularization $\boldsymbol{\lambda}_{decay} = (0.5, 0.5)$ and re-estimate filter coefficients: $\lambda_{decay,1} = 0.5$ enforces stronger shrinkage for high-lag coefficients and $\lambda_{decay,2} = 0.5$ assigns some kind of 'mid-term' weight: neither vanishing nor very strong.

```
> # Estimate filter coefficients: Decay-regularization
> lambda_decay[2]<-0.5
> lambda_decay[1]<-0.5
> mdfa_obj_decay<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,i1,i2,weight_constraint
```

The resulting filter-coefficients are plotted in fig.13.1.

```
null device
          1
```

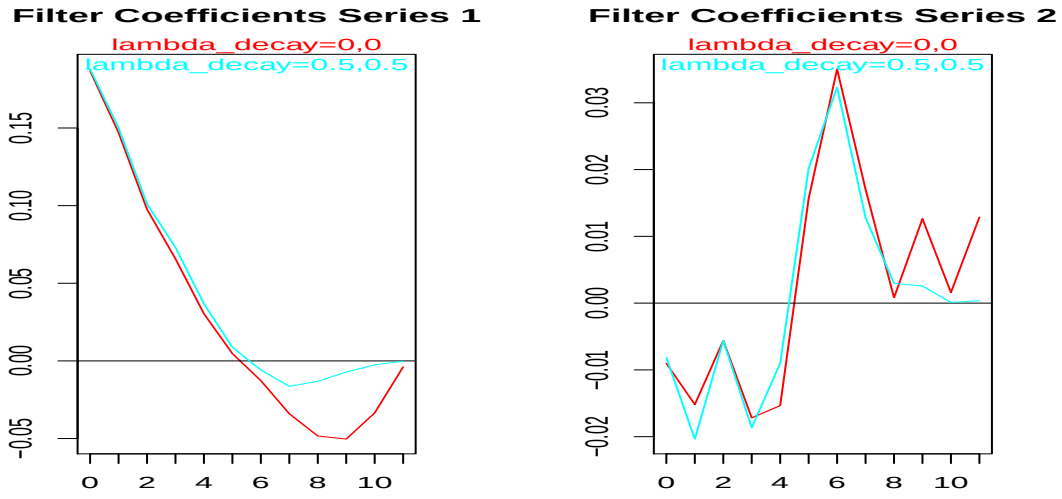


Figure 13.1: Filter Coefficients: original unregularized (red) vs. decay-regularization (cyan) for series 1 (left) and 2 (right) of the bivariate design, lambda-decay=(0.5,0.5)

As expected, the rate of decay of the regularized coefficients (cyan) is more pronounced. In order to measure the 'effective' strength of the regularization we can compute the so-called *effective degrees of freedom* of the estimates (see below for details): the regularization has shrunken the original unconstrained space, of dimension 24.09 ($2 * L$), to a subspace of (generally non-integer) dimension 16.66: approximately ten degrees of freedom were lost by imposing the decay regularization and therefore in-sample and out-of-sample performances of the regularized design will be less incongruent (overfitting is tamed).

13.4.5 Grid-Screening of Effects

We here attempt to provide more insights into the decay-regularization by screening the specific effects controlled by $\lambda_{decay,1}$ and $\lambda_{decay,2}$.

Shape-Parameter $\lambda_{decay,1}$

We first fix the strength-parameter $\lambda_{decay,2} := 0.5$ and compute estimates of the filter coefficients for a discrete grid of the shape-parameter $\lambda_{decay,1} = k \cdot 0.1$ with $k = -1, 0, ..., 11$:

```
> # Estimate filter coefficients: Decay-regularization
> lambda_decay_1<-0.5
> lambda_decay_2<-0.1*0:9
> lambda_decay_2<-c(lambda_decay_2,lambda_decay_2[length(lambda_decay_2)]+0.01*0:9,0.995,0.999,1)
```

The resulting filter-coefficients are plotted in fig.13.2: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
null device
          1
```

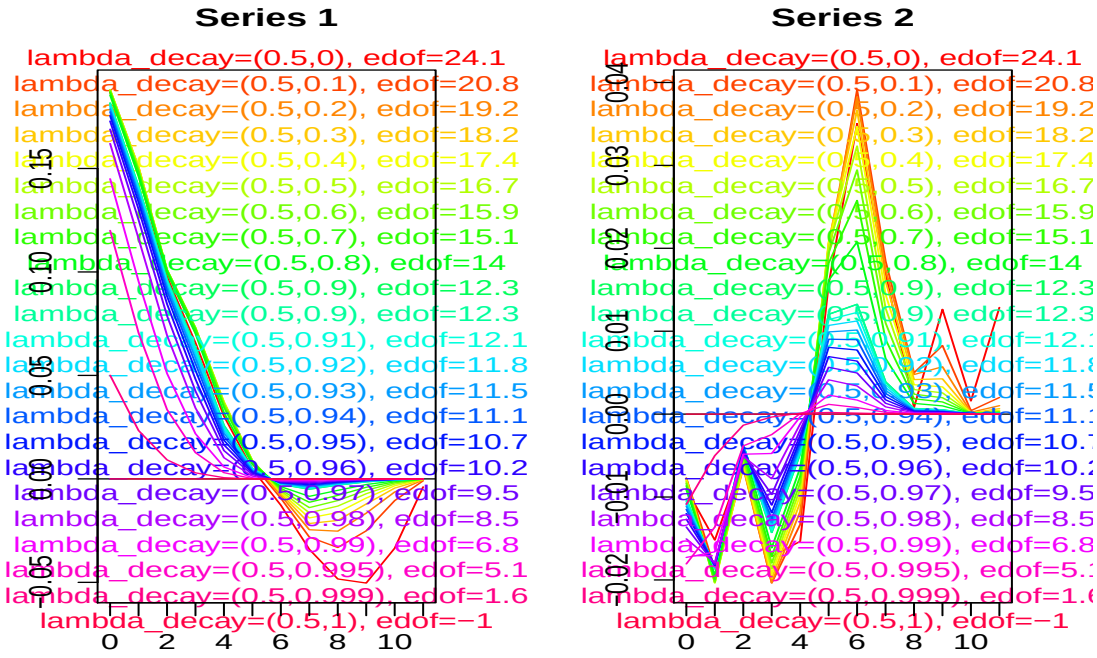


Figure 13.2: Filter Coefficients: effect of shape-parameter (grid of points -0.1, 0, 0.1, 0.2, ..., 1.1) for fixed strength parameter (0.5), series 1 (left) and 2 (right)

A small value of the shape-parameter $\lambda_{decay,1}$ implies that nearly equal weight is assigned to all coefficients, irrespective of the lag. In this case, the number of effective degrees of freedom, $edof$, is smallest and the overall shrinkage-effect is most pronounced. For increasing shape parameter, higher-lag coefficients are shrunken more heavily whereas small-lag coefficients get relaxed, in relative terms. Note that a larger shape-parameter $\lambda_{decay,1}$ tends, ceteris paribus, to enforce the strength of the regularization[4]. This effect explains the non-monotonicity of $edof$ as a function of $\lambda_{decay,1}$. Note, also, that the effect of $\lambda_{decay,1}$ is bounded to the interval $[0, 1]$: our R-code relies on $\min(|\lambda_{decay,1}|, 1)$, which explains why the estimates for $\lambda_{decay,1} = -0.1$ and $\lambda_{decay,1} = 0.1$ (or for $\lambda_{decay,1} = 1$ and $\lambda_{decay,1} = 1.1$) are indistinguishable.

[4]This entanglement could be avoided by normalizing $\lambda_{decay,2}$ by $\frac{1}{L}\sum_{k=0}^{L-1}|1 + \lambda_{decay,1}|^k$, for example (but we didn't).

Strength-Parameter $\lambda_{decay,2}$

We now fix the shape-parameter $\lambda_{decay,1} = 0.5$ and analyze effects by the strength-parameter $\lambda_{decay,2} = k \cdot 0.1$ where $k = -1, 0, 1, ..., 11$:

```
> # Estimate filter coefficients: Decay-regularization
> lambda_decay_1<-0.5
> lambda_decay_2<-0.1*-1:11
```

The resulting filter-coefficients are plotted in fig.13.3: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
null device
          1
```

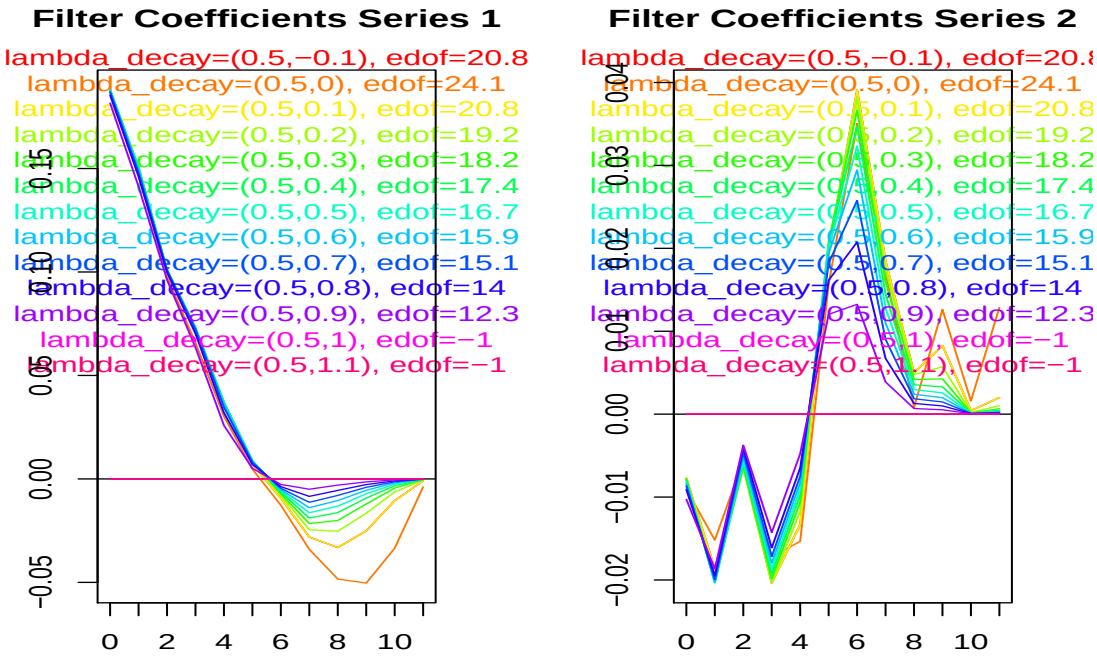


Figure 13.3: Filter Coefficients: effect of strength-parameter (grid of points -0.1,0, 0.1, 0.2, ..., 1.1) for fixed shape parameter (0.5), series 1 (left) and 2 (right)

The strength-parameter $\lambda_{decay,2}$ is bounded to the interval $[0, 1]$, too (the code relies on $f(\min(|\lambda_{decay,2}|, 1))$ where $f(\cdot)$ is a monotonic function with $f(0) = 0$ and $f(1) = \infty$). If $\lambda_{decay,2} = 0$ then no regularization is imposed and thus $edof = 24.0946826274948$ $(= 2L)$. The number of effective degrees of freedom shrinks monotonically as a function of $\lambda_{decay,2}$: for 'large' $\lambda_{decay,2} \to 1$, an asymptotically infinite weight[5] is obtained and therefore the coefficients are shrunken to zero i.e. $edof = 0$ (the

[5]For numerical reasons we use a finite upper-bound in our R-code.

corresponding coefficients are confounded with the zero line).

13.4.6 Constraints vs. Regularization

Hard-constraints, such as imposed by $i1$ or $i2$, see chapter 6, are maintained irrespective of regularization settings: hard-constraints are prioritized and dominate the design. In the following example we compare unrestricted ($i1 = i2 = F$) and restricted filter estimates, whereby a simple level-constraint $i1 = T$ is imposed in the latter case: $\hat{\Gamma}_1(0) = \hat{\Gamma}_2(0) = 1$ (the transferfunctions of the two real-time filters of the bivariate design must equate one in frequency zero).

```
> # Estimate filter coefficients: Decay-regularization
> lambda_decay_1<-0.5
> lambda_decay_2<-c(0,0.5,1)
```

We now estimate coefficients for all four combinations of regularized/unregularized and constrained/unconstrained designs:

```
> # Source the default (MSE-) parameter settings
> source(file=paste(path.pgm,"control_default.r",sep=""))
> troikaner<-T
> # Unconstrained designs: with and without regularization
> b_mat_unrestricted<-matrix(nrow=L,ncol=(ncol(weight_func)-1)*length(lambda_decay_2))
> edof_vec_unrestricted<-rep(NA,length(lambda_decay_2))
> for (i in 1:length(lambda_decay_2))
+ {
+   lambda_decay<-c(lambda_decay_1,lambda_decay_2[i])
+   mdfa_obj_decay<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,i1,i2,weight_constrai
+   b_mat_unrestricted[,(i-1)*(ncol(weight_func)-1)+1:(ncol(weight_func)-1)]<-mdfa_obj_decay$b
+   edof_vec_unrestricted[i]<-mdfa_obj_decay$degrees_freedom
+ }
> # Impose level-constraint
> i1<-T
> # Both transfer functions must equal one in frequency zero
> weight_constraint<-c(1,1)
> b_mat_restricted<-matrix(nrow=L,ncol=(ncol(weight_func)-1)*length(lambda_decay_2))
> edof_vec_restricted<-rep(NA,length(lambda_decay_2))
> for (i in 1:length(lambda_decay_2))
+ {
+   lambda_decay<-c(lambda_decay_1,lambda_decay_2[i])
+   mdfa_obj_decay<-mdfa_analytic(L,lambda,weight_func,Lag,Gamma,eta,cutoff,i1,i2,weight_constrai
+   b_mat_restricted[,(i-1)*(ncol(weight_func)-1)+1:(ncol(weight_func)-1)]<-mdfa_obj_decay$b
+   edof_vec_restricted[i]<-mdfa_obj_decay$degrees_freedom
+ }
```

The resulting filter-coefficients are plotted in fig.13.4: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
null device
          1
```

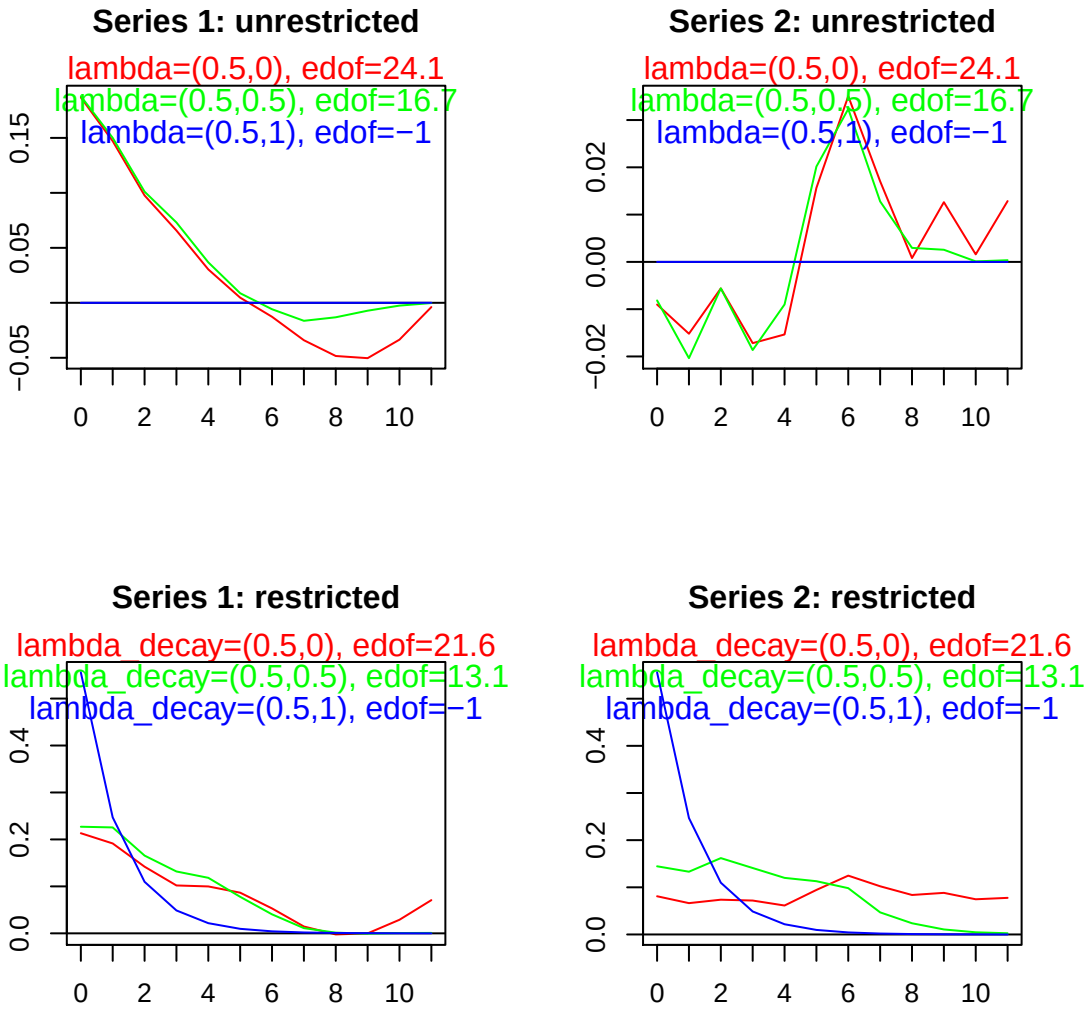


Figure 13.4: Filter Coefficients: effect of constraints and strength-parameter (0, 0.5, 1) for fixed shape parameter (0.5), series 1 (left) and 2 (right); unconstrained design (top) vs. constrained design (bottom)

Without regularization imposed (red lines), the constrained design (bottom) looses two degrees of freedom, dropping from $edof = 24.1$ to $edof = 21.6$. With full regularization imposed (blue lines) the unconstrained design (top) is shrunken to zero; in contrast the constrained design (bottom)

must satisfy $\hat{\Gamma}_i(0) = \sum_{k=0}^{L-1} b_{ki} = 1$ for $i = 1, 2$. Indeed, the latter constraints are satisfied by all filters, irrespective of regularization settings. To conclude, we note that level and time-shift constraints were parametrized in such a way that potential conflicts with the proposed regularization-term are avoided, irrespective of the lead (forecast) or lag (backcast) of the estimate, see section 6.3.

13.5 Longitudinal Smoothness

For each explaining time series x_t, w_{it}, $i = 1, ..., m$, in a real-time filter design, the corresponding filter coefficients b_{ik}[6], for i fixed, may be considered as functions of the lag $k = 0, ..., L-1$. A typical indication of overfitting is given when these lag-dependent functions are ragged i.e. when the series b_{ik}, for fixed i, are 'noisy' or, more formally, when the curvatures (squared second order differences)

$$\sum_{k=0}^{L-3} \left((1-B)^2 b_{ik}\right)^2 = \sum_{k=0}^{L-3} \left(b_{ik} - 2b_{i,k+1} + b_{i,k+2}\right)^2 \tag{13.4}$$

are large.

13.5.1 Quadratic Bilinear Form

We may thus introduce a penalty-term whose bilinear form replicates 13.4. Specifically, the regularized criterion becomes

$$MDFA + f(\lambda_{smooth}) \mathbf{b}' \mathbf{\Lambda}^m_{smooth} \mathbf{b} \rightarrow \min_{\mathbf{b}} \tag{13.5}$$

where $\mathbf{\Lambda}^m_{smooth}$ is a block-diagonal matrix of dimension $L(m+1) * L(m+1)$

$$\mathbf{\Lambda}^m_{smooth} = \begin{pmatrix} \mathbf{\Lambda}^0_{smooth} & 0 & ... & 0 \\ 0 & \mathbf{\Lambda}^0_{smooth} & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & \mathbf{\Lambda}^0_{smooth} \end{pmatrix}$$

with $L * L$-dimensional blocks $\mathbf{\Lambda}^0_{smooth}$

$$\mathbf{\Lambda}^0_{smooth} = \begin{pmatrix} 1 & -2 & 1 & 0 & 0 & 0 & 0 & .... & & & & & 0 \\ -2 & 5 & -4 & 1 & 0 & 0 & 0 & .... & & & & & 0 \\ 1 & -4 & 6 & -4 & 1 & 0 & 0 & .... & & & & & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & .... & & & & & 0 \\ :: & & & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & ... & 1 & -4 & 6 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 1 & -4 & 6 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 1 & -4 & 5 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 1 & -2 & 1 \end{pmatrix}$$

[6]The case $i = 0$ refers to the explaining data of x_t.

The latter replicate the curvature measure 13.4

$$\sum_{k=0}^{L-3}\left((1-B)^2 b_{ik}\right)^2 = \mathbf{b}_i'\mathbf{\Lambda}^0_{smooth}\mathbf{b}_i$$

for $i = 0, ..., m$[7]. In the univariate case we have $m = 0$ and therefore $\mathbf{\Lambda}^m_{smooth} = \mathbf{\Lambda}^0_{smooth}$. As for the previous decay-regularization, $f(\lambda_{smooth})$ is a non-linear function of λ_{smooth}, with $f(0) = 0, f(1) = \infty$, which measures the strength of the smoothness-regularization. If $\lambda_{smooth} = 1$, then full regularization is imposed but, in contrast to the previous decay-term, the data is not completely discarded in this case, because shrinkage is applied to second-order differences. The proposed quadratic bilinear form is positive but not *strictly* positive definite: its kernel consists of all functions which are linear in the lag k. Therefore, asymptotically, filter coefficients b_{ik} must be linear, which still allows for multiple dependency from the data (such as scale and sign, for example). In particular the degrees of freedom shrink continuously from $L * (m + 1)$ when $\lambda_{smooth} = 0$ to $2 * (m + 1)$[8] when $\lambda_{smooth} = 1$. Note that the smoothness regularization is not affected by the lag parameter (Lag or $-h$) and therefore the above mentioned applies indifferently to backcasting, nowcasting or forecasting, as well.

13.5.2 Empirical Examples

We rely on the above example and compute coefficients for a discrete grid $\lambda_{smooth} = k \cdot 0.1$, $k = -1, 0, ..., 11$, of parameter values:

```
> # Estimate filter coefficients: Decay-regularization
> #lambda_smooth_vec<-0.1*-1:11
> lambda_smooth_vec<-0.1*0:9
> lambda_smooth_vec<-c(lambda_smooth_vec,lambda_smooth_vec[length(lambda_smooth_vec)]+0.01*0:9,0.
```

The resulting filter-coefficients are plotted in fig.13.5: the effective degrees of freedom, $edof$, as well as the accompanying regularization settings are reported too.

```
null device
          1
```

[7] Checking this identity is graciously left as an exercise.
[8] Two parameters, namely intercept and slope, determine a linear function.

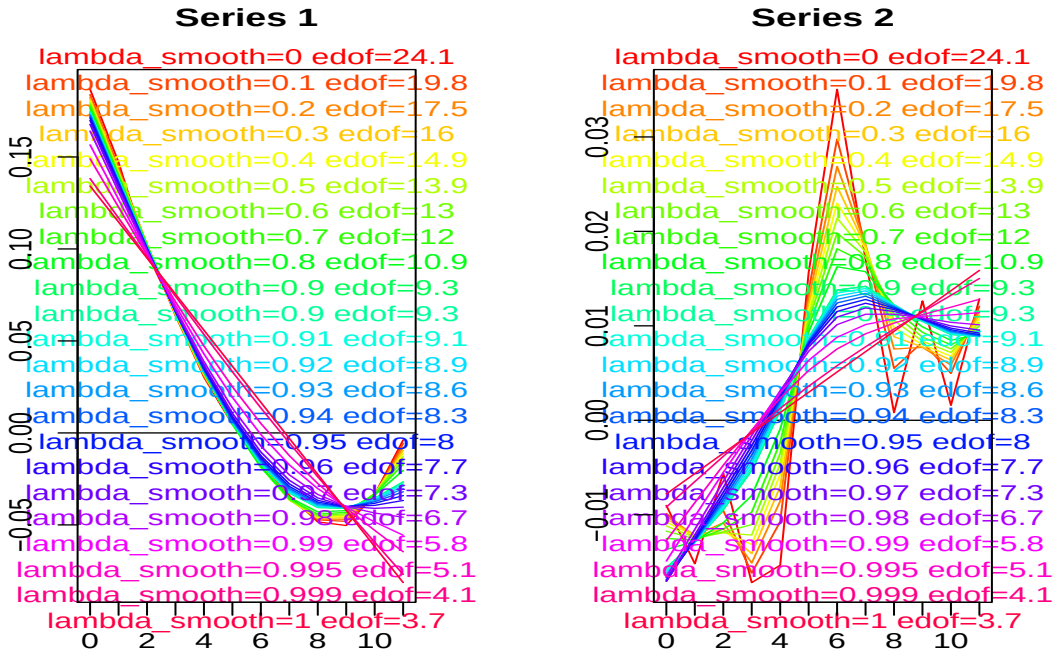


Figure 13.5: Filter Coefficients: effect of smoothness regularization for grid-points -0.1, 0, 0.1, 0.2, ..., 1.1: series 1 (left) and 2 (right)

The longitudinal smoothness of the filter coefficients is improved as λ_{smooth} increases and the coefficients end-up as linear functions of the lag when full regularization is imposed. Unlike the previous decay-regularization, the coefficients are not shrunken towards zero for $\lambda_{smooth} = 1$; in this case $edof = 2 * (m + 1) = 4$, as expected. In conformity with the previous decay-term, the effect of λ_{smooth} is bounded to the interval $[0, 1]$: our R-code relies on $f(\min(|\lambda_{decay,1}|, 1))$ where $f(\cdot)$ is a monotonic function with $f(0) = 0$ and $f(1) = \infty$. Therefore, the estimates for $\lambda_{smooth} = -0.1$ and $\lambda_{smooth} = 0.1$ (or for $\lambda_{smooth} = 1$ and $\lambda_{smooth} = 1.1$) are identical.

13.6 Cross-Sectional Similarity

Often, in practice, the time series x_t, w_{it}, $i = 1, ..., m$ of a multivariate design are redundant, at least to some extent[9]. In such a case it is not illegitimate to assume that filter coefficients should share common features, too: for example the rate of decay and/or the sign and/or the scale.

[9]Business-cycles, for example, are defined as pervasive co-movements across a range of interesting time series. In this context, it is not uncommon to assume that time series can be decomposed additively (eventually after suitable data-transformation) into a common generally non-stationary term (common factor) and a stationary idiosyncratic component (cointegration rank one). In such a case the common factor stands for the redundant information across the original data.

13.6.1 Quadratic Bilinear Form

In order to account for this possibility we here fix the lag-index k and consider b_{ik} as a function of the cross-sectional index $i = 0, ..., m$: for each (fixed) k, b_{ik} should be similar or, stated otherwise, b_{ik} should be close to the cross sectional mean $\frac{1}{m+1}\sum_{i=0}^{m} b_{ik}$[10]. Accordingly, the penalty term amending additively the (M)DFA-criterion can be specified as

$$\sum_{k=0}^{L-1}\sum_{i=0}^{m}\left(b_{ik} - \frac{1}{m+1}\sum_{i'=0}^{m} b_{i'k}\right)^2 \tag{13.6}$$

Smaller values of this statistic signify enforced cross-sectional similarity of the coefficients, as desired. A suitable positive definite bilinear form replicating this penalty term is given by

$$\mathbf{\Lambda}_{cross}^{m} = \mathbf{I} - \boldsymbol{M} \tag{13.7}$$

where $\mathbf{I}$ is an $L(m+1) * L(m+1)$-dimensional identity and

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{\mu} & ... & \boldsymbol{\mu} \\ ::: & & \\ \boldsymbol{\mu} & ... & \boldsymbol{\mu} \end{pmatrix}$$

is made of $(m+1)^2$ replications of the $L * L$-dimensional block $\boldsymbol{\mu}$

$$\boldsymbol{\mu} = \begin{pmatrix} -\frac{1}{m+1} & 0 & ... & 0 \\ 0 & -\frac{1}{m+1} & ... & 0 \\ ::: & & & \\ 0 & 0 & ... & -\frac{1}{m+1} \end{pmatrix}$$

A juxtaposition of these blocks along the rows of $\boldsymbol{M}$ builds-up the cross-sectional means from the stacked (long) coefficient vector $\mathbf{b} = Vec(\mathbf{B})$. The corresponding regularized criterion becomes

$$MDFA + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}_{cross}^{m}\mathbf{b} \to \min_{\mathbf{b}} \tag{13.8}$$

where, once again, $f(\lambda_{cross})$ is a non-linear monotonic function with $f(0) = 0, f(1) = \infty$. In the latter case, when full regularization is imposed, the cross-sectional differences are projected onto zero. As for the smoothness-term, the cross-sectional bilinear form is not strictly positive since its kernel is made of the common 'grand-mean'. Asymptotically, the data still interacts with the regularized estimate: the remaining degrees of freedom shrink continuously from $(m+1)L$, when $\lambda_{cross} = 0$, to L when $\lambda_{cross} = 1$. Note that the cross-sectional regularization is not affected by the lag parameter (Lag or $-h$) and therefore the above mentioned applies indifferently to backcasting, nowcasting or forecasting, as well.

13.6.2 Empirical Examples

We rely on the previous example and compute coefficients for a discrete grid $\lambda_{cross} = k \cdot 0.1$, $k = -1, 0, ..., 11$, of parameter values:

[10] The cross-sectional means are the grand-means, see section 4.5. However, our proceeding here avoids the undesirable asymmetry of the explicit grand-mean parametrization.

```
> # Estimate filter coefficients: Decay-regularization
> #lambda_cross_vec<-0.1*0:9
> lambda_cross_vec<-0.1*0:9
> lambda_cross_vec<-c(lambda_cross_vec,lambda_cross_vec[length(lambda_cross_vec)]+0.01*0:9,0.995,
```

The resulting filter-coefficients are plotted in fig.13.6: the effective degrees of freedom, $edof$, as well as the accompanying regularization settings are reported too.

```
null device
          1
```

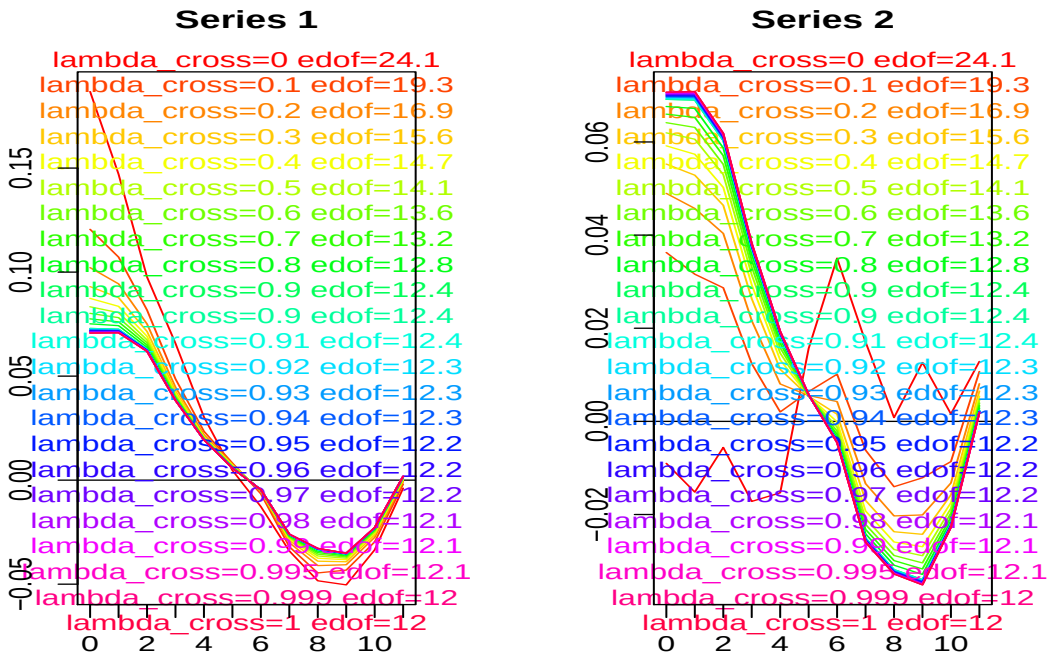


Figure 13.6: Filter Coefficients: effect of cross-sectional regularization for grid-points -0.1, 0, 0.1, 0.2, ..., 1.1: series 1 (left) and 2 (right)

For increasing λ_{cross}, filter coefficients for series 1 and 2 are merged more effectively, ending-up absorbed in the common grand-mean, when full regularization is imposed (the coefficients in left and right panels are identical if $\lambda_{cross} \geq 1$). As for the smoothness-regularization, the coefficients are not shrunken towards zero for $\lambda_{cross} = 1$; instead, the gap to the common grand-mean is closed and $edof = L(= 12)$. In conformity with the previous regularization terms, the effect of λ_{cross} is bounded to the interval $[0, 1]$: our R-code relies on $f(\min(|\lambda_{cross,1}|, 1))$ where $f(\cdot)$ is a monotonic function with $f(0) = 0$ and $f(1) = \infty$. Therefore, the estimates for $\lambda_{cross} = -0.1$ and $\lambda_{cross} = 0.1$ (or for $\lambda_{cross} = 1$ and $\lambda_{cross} = 1.1$) are identical. let us conclude here by emphasizing that the cross-sectional regularization is inadequate in the specific case of our toy-example because

the second series of our bivariate design is independent from the target i.e. its coefficients should vanish[11].

13.7 Regularization Troika: a Triplet of Universal Filter Requirements

Instead of imposing possibly misspecified 'hard' constraints, the proposed triplet of regularization-terms assigns 'soft' preferences to particular sub-spaces of the ambient space $\mathbb{R}^{L(m+1)}$, which are felt to be of universal relevance.

13.7.1 Quadratic (Bilinear) Form

An arbitrary combination and weighting of all three requirements can be obtained formally by the so-called Regularization Troika

$$MDFA + f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}^m_{decay}\mathbf{b} + f(\lambda_{smooth})\mathbf{b}'\mathbf{\Lambda}^m_{smooth}\mathbf{b} + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}^m_{cross}\mathbf{b} \rightarrow \min_{\mathbf{b}} \quad (13.9)$$

If $\lambda_{decay,2} = \lambda_{smooth} = \lambda_{cross} = 0$ then 13.9 replicates the original MDFA-criterion. Otherwise, degrees of freedom are shrunken by projecting filter coefficients onto sub-spaces which are *un*likely to conflict with data (universality postulate) if the regularization is suitably balanced. Our credo – backed-up by experience – is that the MDFA, endowed with the proposed regularization features, is able to tame overfitting without necessarily inflicting 'misspecification' or statistical inefficiency: shrinkage-gains outweigh misspecification-losses such that, overall, out-of-sample performances improve.

13.7.2 Empirical Examples

To be filled...

13.7.3 Entanglement and Conflicting Requirements

The proposed regularization features are not mutually orthogonal: as an example a strong decay-regularization may favor simultaneously smoothness as well as cross-sectional similarity by attracting coefficients towards the 'common' and 'smooth' zero-line. Per contra, the decay-regularization might also conflict with the smoothness requirement, since a rapid decay could be disruptive or discontinuous. It should be clear, therefore, that the regularization settings, as controlled by the hyperparameters $\boldsymbol{\lambda}_{decay}, \lambda_{smooth}$ and λ_{cross} interact, either positively or negatively, to some extent. Interestingly, part of the conflicts can be resolved by specifying alternative origins of the shrinkage-spaces, see section 13.11. Finally, let us remind here that regularization sub-spaces can be affected by imposing hard-constraints, see the example in section 13.4.6.

[11] The common coefficients obtained for $\lambda_{cross} \geq 1$ in fig.13.6 are a shrunken version of the unregularized coefficients for series 1: whereas the first series is informative, the second-one just adds undesirable noise, which must be tamed by zero-shrinkage.

13.7.4 Limitations: Data Transformations

Some of the proposed regularization features assume, at least implicitly, that the data is subject to specific homogeneity constraints. As a trivial example, the cross-sectional term assumes similarity of time series in the data set (against the premicesses of our toy example). Filter-performances are no more invariant to (arbitrary) sign-changes or transformation of scales of time series if $\lambda_{cross} > 0$. In a similar vein, the zero-shrinkage peculiar to the strictly positive definite decay-term assumes that the scales of the time series are identical or at least similar. Otherwise, the series with the smallest scale would be penalized more heavily because its coefficients are generally (though not always) larger, in absolute value. Performances are no more invariant to scale transformations if $\lambda_{decay,2} > 0$. We therefore recommend to transform the data prior to regularization in such a way that the implicit homogeneity assumptions are met. In applications so far we found useful to match scales and signs of series[12].

13.7.5 Empirical Examples

To be filled...

13.8 Optimization Criterion (Zero-Shrinkage)

We note that 13.9 is quadratic in the filter coefficients and therefore a unique closed-form solution exists. Using bilinear forms simplifies the derivation (as well as the notation) of the closed-form solution. We distinguish the cases of regularization with and without 'hard' constraints (for example i1- and i2-constraints).

13.8.1 Unconstrained Design

Consider

$$\begin{aligned} & (\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b})'(\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}) \\ + \quad & \lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}\mathbf{b} + \lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}\mathbf{b} + \lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b} \to \min \end{aligned} \tag{13.10}$$

where $\mathbf{Y}^{\text{Cust}}(\eta) \in \mathbb{R}^{[T/2]+1}$ is a $[T/2]+1$-dimensional real-valued vector of dependent variables and where $\mathbf{X}^{\text{Cust}}(\lambda,\eta) \in \mathbb{C}^{([T/2]+1)(m+1)L}$ is a $([T/2]+1) * ((m+1)L)$-dimensional complex-valued matrix of explaining data, see section 8.1.4: for $\lambda = \eta = 0$ no customization is imposed and therefore a classic MSE-estimate is obtained. Derivation of the criterion with respect to the

[12] In order to measure performances in a consistent way, data transformations should always be causal i.e. they should not rely on future observations.

stacked parameter vector $\mathbf{b}$ leads to

$$\begin{aligned}\frac{d}{d\mathbf{b}}\ \text{Criterion} &= \frac{d}{d\mathbf{b}}\ (\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b})'(\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}) \\ &+ \frac{d}{d\mathbf{b}}\ \left(\lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}\mathbf{b}+\lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}\mathbf{b}+\lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b}\right) \\ &= -2\mathbf{Y}^{\text{Cust}}(\eta)'\Re\left(\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right)+2\mathbf{b}'\Re\left\{\mathbf{X}^{\text{Cust}}(\lambda,\eta)'\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right\} \\ &\quad +2\lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}+2\lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}+2\lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\end{aligned}$$

where we used the fact that the proposed bilinear forms are symmetric. Equating this expression to zero, the generalized regularized solution is obtained as

$$\begin{aligned}&\hat{\mathbf{b}}^{\text{Cust-Reg}}(\lambda,\eta,\lambda_{smooth},\lambda_{cross},\lambda_{decay,1},\lambda_{decay,2})= \\ &\left(\Re\left\{\mathbf{X}^{\text{Cust}}(\lambda,\eta)'\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right\}+\lambda_{smooth}\mathbf{\Lambda}_{smooth}+\lambda_{cross}\mathbf{\Lambda}_{cross}+\lambda_{decay,2}\mathbf{\Lambda}_{decay}\right)^{-1} \\ &\Re(\mathbf{X}^{\text{Cust}}(\lambda,\eta))'\mathbf{Y}^{\text{Cust}}(\eta)\end{aligned} \tag{13.11}$$

As can be seen, each term of the Regularization Troika contributes in 'regularizing' the matrix subject to inversion: ill-posed problems with L large (large filter order) and/or m large (high-dimensional design) can be solved effectively by imposing suitable shrinkage towards idealized filter patterns. In particular, the number $(m+1)L$ of unknown filter coefficients may exceed the sample size T[13].

13.8.2 Constrained Design

We assume that the filter constraints can be written in the form

$$\mathbf{b} = \mathbf{R}(i1,i2)\mathbf{b_f}+\mathbf{c}(i1,i2) \tag{13.12}$$

where $\mathbf{b_f}$ is the vector of freely determined coefficients and where $\mathbf{R}(i1,i2)$ and $\mathbf{c}(i1,i2)$ depend on the booleans $i1,i2$, see section 6.3.4 (alternative constraints could be considered, of course). Plugging this expression into 13.10 and taking derivatives with respect to the stacked vector of freely determined coefficients $\mathbf{b_f}$, we obtain (note that the functional dependencies of $\mathbf{Y}^{\text{Cust}}(\eta),\mathbf{X}^{\text{Cust}}(\lambda,\eta)$

[13] This would allow for a formal treatment of high-dimensional designs (m large), for example.

on the customization parameters λ, η are omitted for notational simplicity):

$$\begin{aligned}
\frac{d}{d\mathbf{b_f}}\text{ Criterion} &= \frac{d}{d\mathbf{b_f}}\left(\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\left(\mathbf{Rb_f} + \mathbf{c}\right)\right)'\left(\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\left(\mathbf{Rb_f} + \mathbf{c}\right)\right) \\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{smooth}\left(\mathbf{Rb_f} + \mathbf{c}\right)'\mathbf{\Lambda}_{smooth}\left(\mathbf{Rb_f} + \mathbf{c}\right) \\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{cross}\left(\mathbf{Rb_f} + \mathbf{c}\right)'\mathbf{\Lambda}_{cross}\left(\mathbf{Rb_f} + \mathbf{c}\right) \\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{decay,2}\left(\mathbf{Rb_f} + \mathbf{c}\right)'\mathbf{\Lambda}_{decay}\left(\mathbf{Rb_f} + \mathbf{c}\right) \\
&= -2(\mathbf{Y}^{\text{Cust}})'\Re\left(\mathbf{X}^{\text{Cust}}\right)\mathbf{R} + 2\Re\Big\{(\mathbf{X}^{\text{Cust}}\mathbf{c})'(\mathbf{X}^{\text{Cust}}\mathbf{R})\Big\} + 2\mathbf{b}_f'\Re\Big\{(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{R}\Big\} \\
&\quad +2\lambda_{smooth}\mathbf{b_f'R'\Lambda}_{smooth}\mathbf{R} + 2\lambda_{smooth}(\mathbf{c'\Lambda}_{smooth}\mathbf{R}) \\
&\quad +2\lambda_{cross}\mathbf{b_f'R'\Lambda}_{cross}\mathbf{R} + 2\lambda_{cross}(\mathbf{c'\Lambda}_{cross}\mathbf{R}) \\
&\quad +2\lambda_{decay,2}\mathbf{b_f'R'\Lambda}_{decay}\mathbf{R} + 2\lambda_{decay,2}(\mathbf{c'\Lambda}_{\text{decay}}\mathbf{R})
\end{aligned}$$

where, again, we used the fact that the proposed bilinear forms are symmetric. Equating this expression to zero, the *customized, regularized and constrained* solution is obtained as

$$\begin{aligned}
&\hat{\mathbf{b}}_f^{\text{Cust-Reg-Const}}(\lambda, \eta, \lambda_{smooth}, \lambda_{cross}, \lambda_{decay,1}, \lambda_{decay,2}, i1, i2) \\
= &\Big\{\Re\Big[(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{R}\Big] + \lambda_{smooth}\mathbf{R'\Lambda}_{smooth}\mathbf{R} + \lambda_{cross}\mathbf{R'\Lambda}_{cross}\mathbf{R} + \lambda_{decay,2}\mathbf{R'\Lambda}_{decay}\mathbf{R}\Big\}^{-1} \\
&\Big((\Re(\mathbf{X}^{\text{Cust}})\mathbf{R})'\mathbf{Y}^{\text{Cust}} - \Re\Big\{(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{c}\Big\} - \mathbf{R}'(\lambda_{smooth}\mathbf{\Lambda}_{\text{smooth}} + \lambda_{cross}\mathbf{\Lambda}_{cross} + \lambda_{decay,2}\mathbf{\Lambda}_{cross})\mathbf{c}\Big) \\
= &\Big\{\Re\Big[\mathbf{R}'(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\mathbf{R}\Big] + \lambda_{smooth}\mathbf{R'\Lambda}_{smooth}\mathbf{R} + \lambda_{cross}\mathbf{R'\Lambda}_{cross}\mathbf{R} + \lambda_{decay,2}\mathbf{R'\Lambda}_{decay}\mathbf{R}\Big\}^{-1} \\
&\Big((\Re(\mathbf{X}^{\text{Cust}})\mathbf{R})'\mathbf{Y}^{\text{Cust}} + \mathbf{Const}\Big) \qquad (13.13)
\end{aligned}$$

where

$$\mathbf{Const} = -\mathbf{R}'\Big\{\Re\Big[(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\Big] + \lambda_{smooth}\mathbf{\Lambda}_{smooth} + \lambda_{cross}\mathbf{\Lambda}_{cross} + \lambda_{decay,2}\mathbf{\Lambda}_{decay}\Big\}\mathbf{c}$$

This last term collects all level constraints[14]. A comparison with 13.11 illustrates that both expressions - with or without constraints - are formally quite similar, up to the additional transformation by $\mathbf{R}$ and the emergence of a new generalized level-shift **Const**: setting $\mathbf{R} = \mathbf{Id}$ and $\mathbf{c} = \mathbf{0}$ (no constraints involved) in the above solution indeed replicates 13.11. The result of the optimization is $\mathbf{b_f}$, the vector of freely determined coefficients. The sought-after 'full-coefficient' vector $\mathbf{b}$, which is indispensable for the filtering-task, is then obtained from 13.12.

13.8.3 Empirical Example: Constrained Regularized Customized Design

To be filled...

[14] Indeed, $\mathbf{c} = \mathbf{0}$ if $i1 = F$.

13.9 Effective Degrees of Freedom*

In the absence of regularization, the number of effective degrees of freedom *edof* coincides with the number of freely determined filter coefficients (the length of the stacked vector $\mathbf{b_f}$); otherwise, *edof* is reduced. We here derive a statistic which extends *edof* to the generic Regularization Troika. In contrast to the classic (real-valued) linear regression framework, the MDFA-criterion is subject to technical peculiarities because the filter coefficients, as determined in a complex-valued space (frequency-domain), are required to be real-valued. As a result, the hat-matrix is neither symmetric, nor a projection anymore. As we shall see, this problem can be overcome by introducing a so-called adjoined imaginary estimate (the purely imaginary least-squares solution) and by augmenting the original hat-matrix by the resulting adjoined hat-matrix. For technically-averse readers, the results in this section can be skipped without impairing comprehension of later topics.

13.9.1 Hat-Matrix and Residual-Matrix

For simplicity of exposition we here treat the unconstrained case as obtained by criterion 13.10. The so-called *hat-matrix* $\mathbf{H}$ is defined by

$$\mathbf{H} = \mathbf{X}^{\mathrm{Cust}} \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} + \lambda_{smooth} \mathbf{\Lambda}_{smooth} + \lambda_{cross} \mathbf{\Lambda}_{cross} + \lambda_{decay,2} \mathbf{\Lambda}_{decay} \right)^{-1} \Re(\mathbf{X}^{\mathrm{Cust}})' \tag{13.14}$$

Thus

$$\hat{\mathbf{Y}}^{\mathrm{Cust}} := \mathbf{X}^{\mathrm{Cust}} \hat{\mathbf{b}} = \mathbf{H} \mathbf{Y}^{\mathrm{Cust}}$$

i.e. $\mathbf{H}$ maps the dependent variable $\mathbf{Y}$ onto $\hat{\mathbf{Y}}^{\mathrm{Cust}}$. In the absence of regularization, the expression for $\mathbf{H}$ simplifies to

$$\mathbf{H} = \mathbf{X}^{\mathrm{Cust}} \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} \Re(\mathbf{X}^{\mathrm{Cust}})'$$

Note that $\hat{\mathbf{Y}}^{\mathrm{Cust}}$ lies in a particular subspace of the complex plane spanned by the data $\mathbf{X}^{\mathrm{Cust}}$, determined by the requirement that the least-squares solution must be real-valued[15]. Therefore, the complex-valued hat-matrix $\mathbf{H}$ is generally asymmetric ($\mathbf{H} \neq \mathbf{H}'$) and it is no more a projection since

$$\begin{aligned} \mathbf{H}'\mathbf{H} &= \overline{\mathbf{H}}^{\mathbf{T}} \mathbf{H} \\ &= \Re(\mathbf{X}^{\mathrm{Cust}}) \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} \Re(\mathbf{X}^{\mathrm{Cust}}) \\ &\neq \mathbf{H} \end{aligned}$$

Interestingly, though, an idempotency is obtained when focusing on the real-parts only:

$$\begin{aligned} \Re\left(\mathbf{H}'\mathbf{H}\right) &= \Re(\mathbf{X}^{\mathrm{Cust}}) \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} \Re\Big((\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \Big) \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} \Re(\mathbf{X}^{\mathrm{Cust}}) \\ &= \Re(\mathbf{X}^{\mathrm{Cust}}) \left(\Re\left\{ (\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right\} \right)^{-1} \Re(\mathbf{X}^{\mathrm{Cust}}) \\ &= \Re(\mathbf{H}) \end{aligned}$$

[15] If we relaxed this restriction, i.e. if we allowed for an unrestricted complex-valued least-squares solution, then the resulting hat-matrix would be

$$\mathbf{H} = \mathbf{X}^{\mathrm{Cust}} \left((\mathbf{X}^{\mathrm{Cust}})' \mathbf{X}^{\mathrm{Cust}} \right)^{-1} (\mathbf{X}^{\mathrm{Cust}})'$$

This matrix would be a symmetric (hermitian) projection with eigenvalues either one or zero.

Also, in contrast to the classic linear regression framework, the eigenvalues of $\mathbf{H}$ are not restricted to unity or zero: they are generally complex-valued and can be larger or smaller than one, in absolute value. Finally, the trace of the hat-matrix does not correspond to the effective degrees of freedom, anymore

$$tr(\mathbf{H}) \neq (m+1)L$$

In fact, $tr(\mathbf{H})$ is a (data-dependent) random-variable.

The *residual-matrix* **Res** is defined by

$$\mathbf{Res} := \mathbf{I} - \mathbf{H}$$

It maps the dependent variable $\mathbf{Y}$ onto the filter error $(\mathbf{I}-\mathbf{H})\mathbf{Y} = \mathbf{Y} - \hat{\mathbf{Y}}$. In analogy to the former hat-matrix, the latter residual-matrix is no more symmetric, it is no more a projection and its trace is a (data-dependent) random-variable. Interestingly, though,

$$\begin{aligned} \Re\left((\mathbf{I}-\mathbf{H})'\hat{\mathbf{Y}}\right) &= \Re\left((\mathbf{I}-\mathbf{H})'\mathbf{H}\mathbf{Y}\right) \\ &= \left(\Re(\mathbf{H}) - \Re(\mathbf{H}'\mathbf{H})\right)\mathbf{Y} \\ &= \mathbf{0} \end{aligned}$$

i.e. the residual-matrix retains orthogonality, at least for real parts. We now propose an extension of these (classic) concepts which allow for a formal and general definition of the number of effective degrees of freedom *edof* in the generic MDFA-framework.

13.9.2 A Generalization of *edof*: Adjoined Complex Estimate and Symmetric Augmentation of the Hat-Matrix

The origin of the above difficulties resides in the asymmetry of the hat-matrix $\mathbf{H}$ which is imputable to its imaginary part. Fundamentally, this asymmetry is caused by requiring the least-squares coefficient-vector $\hat{\mathbf{b}}$ to be real-valued. Consider now the following (asymmetric) optimization problem

$$\begin{aligned} & (\mathbf{Y}^{\mathrm{Cust}}(\eta) - \mathbf{X}^{\mathrm{Cust}}(\lambda,\eta)i\mathbf{b})'(\mathbf{Y}^{\mathrm{Cust}}(\eta) - \mathbf{X}^{\mathrm{Cust}}(\lambda,\eta)i\mathbf{b}) \\ = & \left|\mathbf{Y}^{\mathrm{Cust}}(\eta) - \mathbf{X}^{\mathrm{Cust}}(\lambda,\eta)i\mathbf{b}\right|^2 \\ = & \left|-i\mathbf{Y}^{\mathrm{Cust}}(\eta) - \mathbf{X}^{\mathrm{Cust}}(\lambda,\eta)\mathbf{b}\right|^2 \end{aligned}$$

where $i\mathbf{b}$ is supposed to be purely imaginary. The last equation suggests that we may be looking for a purely real solution $\mathbf{b}$ after rotating the (real-valued) target by $\pi/2$, from $\mathbf{Y}^{\mathrm{Cust}}(\eta)$ to $-i\mathbf{Y}^{\mathrm{Cust}}(\eta)$. We name this problem *adjoined optimization criterion* and the resulting purely imaginary (least-squares) solution is called *adjoined estimate*. Its closed-form is obtained from

$$\begin{aligned} d/d\mathbf{b}\ \text{Adjoined Criterion} &= d/d\mathbf{b}\ (-i\mathbf{Y}^{\mathrm{Cust}} - \mathbf{X}^{\mathrm{Cust}}\mathbf{b})'(-i\mathbf{Y}^{\mathrm{Cust}} - \mathbf{X}^{\mathrm{Cust}}\mathbf{b}) \\ &= -(-i\mathbf{Y}^{\mathrm{Cust}} - \mathbf{X}^{\mathrm{Cust}}\mathbf{b})'\mathbf{X}^{\mathrm{Cust}} - (-i\mathbf{Y}^{\mathrm{Cust}} - \mathbf{X}^{\mathrm{Cust}}\mathbf{b})^{\mathbf{T}}\overline{\mathbf{X}^{\mathrm{Cust}}} \\ &= 2\mathbf{Y}^{\mathrm{Cust}\prime}\Im\left(\mathbf{X}^{\mathrm{Cust}}\right) + 2\mathbf{b}'\Re((\mathbf{X}^{\mathrm{Cust}})'\mathbf{X}^{\mathrm{Cust}}) \end{aligned}$$

where we used the fact that $(i\mathbf{Y}^{\text{Cust}})' = -i(\mathbf{Y}^{\text{Cust}})'$, since $\mathbf{Y}^{\text{Cust}}$ is real-valued. We deduce that the adjoined (purely imaginary) least-squares estimate $\hat{\mathbf{b}}^{ad}$ is obtained as

$$\begin{aligned}\hat{\mathbf{b}}^{ad} &= -i\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{-\mathbf{1}}\Im\left(\mathbf{X}^{\text{Cust}}\right)'\mathbf{Y}^{\text{Cust}} \\ &= i\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{-\mathbf{1}}\Im\left((\mathbf{X}^{\text{Cust}})'\right)\mathbf{Y}^{\text{Cust}}\end{aligned}$$

where the last equality follows from $\Im\left(\mathbf{X}^{\text{Cust}}\right)' = -\Im\left((\mathbf{X}^{\text{Cust}})'\right)$. The corresponding adjoined hat-matrix $\mathbf{H}^{ad}$ is

$$\mathbf{H}^{ad} = i\mathbf{X}^{\text{Cust}}\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{-\mathbf{1}}\Im\left((\mathbf{X}^{\text{Cust}})'\right)$$

Let us now define the *augmented hat-matrix* $\mathbf{H}^{aug}$ as the sum of original and adjoined hat-matrices

$$\begin{aligned}\mathbf{H}^{aug} &= \mathbf{H} + \mathbf{H}^{ad} \\ &= \mathbf{X}^{\text{Cust}}\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{-\mathbf{1}}\left(\mathbf{X}^{\text{Cust}}\right)'\end{aligned}$$

From the last equality we infer that the augmented hat-matrix is symmetric[16] $\mathbf{H}^{aug} = (\mathbf{H}^{aug})'$ and therefore its diagonal must be real-valued[17]. We then obtain

$$\begin{aligned}tr(\mathbf{H}^{aug}) &= tr\left(\mathbf{X}^{\text{Cust}}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}(\mathbf{X}^{\text{Cust}})'\right) \\ &= tr\left((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}\right) \\ &= tr\left(\Re\Big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\Big\}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}\right) \\ &= tr(\mathbf{I}_{(m+1)L}) \\ &= (m+1)L\end{aligned}$$

where the third equality is a consequence of the trace being real and linear and where $\mathbf{I}_{(m+1)L}$ is an $(m+1)L * (m+1)L$-dimensional identity. Since the trace of the augmented hat-matrix $\mathbf{H}^{aug}$ is a fixed number (not a random variable depending on the data) coinciding with $edof = (m+1)L$[18] in the absence of regularization, we are now in a position to extend the concept of the effective degrees of freedom to the case of arbitrary regularization by defining

$$edof(\boldsymbol{\lambda}_{decay}, \lambda_{smooth}, \lambda_{cross}) := tr(\mathbf{H}^{aug}(\boldsymbol{\lambda}_{decay}, \lambda_{smooth}, \lambda_{cross}))$$

where the augmented hat-matrix $\mathbf{H}^{aug}(\cdot)$ is now considered as a function of the general regularization settings. This number has been reported in the previous figures: in the presence of regularization it is generally non-integer valued. To conclude, note that the constrained and regularized optimization criterion 13.13 can be tackled analogously and does not deserve a separate treatment here.

13.9.3 Empirical Examples

To be filled

[16] Note however that it is not a projection.

[17] Recall that $(\mathbf{H}^{aug})' = \overline{\mathbf{H}^{aug}}^T$ is the hermitian conjugate.

[18] It is assumed that the data is of full rank.

13.10 The Troikaner

To be filled...

13.10.1 Generalized Information Criterion

13.11 General H0-Shrinkage

To be filled...

13.11.1 Zero-Shrinkage vs. Regularization: Potentially Conflicting Requirements

13.11.2 Inclusion of A Priori Knowledge

13.11.3 Replicating (and Enhancing) Clients' Performances

13.12 Optimization Criterion under General H0-Shrinkage

To be filled...

13.13 MDFA-Stages

To be filled...

- Numerically (very) fast
- Statistically efficient
- Immunized against Overfitting
- Highly Adaptive

13.14 Unsmoothing

Difficult task: heavy smoothing. It requires long filters which are prone to overfitting (if not regularized). Idea:

- Apply a very simple (univariate) 0-stage filter which oversmooths the data (for example equally weighted or white noise DFA with very small cutoff)
- At the first MDFA-stage apply a filter of length 2 or 3 with 0-shrinkage, based on the multivariate DFT, whose purposes are
  - Account for sign (unemployment vs- production)
  - Account for scale (if series are not standardized)

  - Account for time-shift by unsmoothing the stage 0 filter (L>=2 allows for difference-like designs which unsmooth the filtered data and provide an anticipation)

- Statistically efficient
- Immunized against Overfitting
- Highly Adaptive

13.15 Summary

- Overfitting is an unavoidable and direct consequence of determining filter coefficients according to an optimization principle: the perfected in-sample fit systematically understates the difficulty of the estimation problem.
- The transition from in-sample to out-of-sample performances is generally smooth and gradual in signal-extraction problems (in contrast to classic one-step ahead forecasting) and the transition depends on the rate of decay of the coefficients γ_k of the target.
- Improving out-of-sample performances by taming overfitting is an imbalancing act, whereby shrinkage-gains must outweigh misspecification-losses.
- The Regularization Troika complies with this view by assigning soft preferences to subspaces which are felt to be of universal relevance in terms of longitudinal decay, longitudinal smoothness and cross-sectional similarity of filter coefficients. If these characteristics are shared by the truly best (but unobserved) coefficients, then shrinkage does not conflict with efficiency.
- Finding optimal regularization weights $\boldsymbol{\lambda}_{decay}, \lambda_{smooth}, \lambda_{cross}$ is a balancing act whose outcome strongly depends on user-preferences (utility function[19]).
- The solution of the regularized quadratic criterion can be obtained in closed form.
- An expression for the effective degrees of freedom of a regularized filter can be derived by augmenting the original hat-matrix, corresponding to the purely real-valued least-squares solution, by the adjoined hat-matrix, corresponding to the purely imaginary least-squares solution.

[19] As a factual example, mean-square performances and trading performances are incongruent to a large extent, due to scale-invariance, anisotropy and non-linearity effects.

Chapter 14

Vintage Data: Working with Data-Revisions

refer to 2011-paper

14.1 Introduction

14.2 Data Organization

14.2.1 Vintage and Release Triangles

14.2.2 Vintage Triangle in the Frequency-Domain

14.3 Reconcile Real-Time Signalextraction and Data-Revisions

14.3.1 Vintage-Filtering/Smoothing

14.3.2 Setting-Up MDFA-Designs: the (Pseudo-) Stationary Case

14.3.3 Extension to Non-Stationary Time Series

Refer and link to section 17

Chapter 15

Mixed-Frequency Data: Combining and Working with Data Sampled at Different Time Scales

15.1 Introduction

- Refer to section 4.7.2 for a quantitative analysis of up-dating effects.

- Refer to recent work with Chris

15.2 Target is Low-Frequency Data

15.2.1 Folding the Frequency Interval

15.2.2 Generalized Optimization Criterion

15.3 Explaining Series is/are Low-Frequency Data

15.3.1 Folding and Expanding the Frequency Interval

15.3.2 Generalized Optimization Criterion

15.4 General Case: Arbitrary Mix

15.5 Unequally Distributed Release Dates

15.6 Missing Data

Lomb-Scargle, DFT with leads/lags: analyze orthogonality.

Chapter 16

Non-Stationarity: Integrated Processes

16.1 Introduction

Briefly review classical model-based concepts

16.2 Integration

16.2.1 Unit-Roots vs. Filter Constraints

Refer to DFA-paper with Tucker and to 2005 and 2008-books (generalized unit-root test)

16.2.2 I(1)- (Level) Constraint

16.2.3 I(2)- (Time-Shift) Constraint

16.3 Optimization Criterion

Chapter 17

Non-Stationarity: Cointegrated Processes

Contribution by Tucker? Refer to elements paper with regards to empirical criteria.

17.1 Cointegration Relations vs. Filter Constraints

17.2 The Rank-One Case

17.3 Arbitrary Rank

17.4 Universal Time-Domain Decomposition of the Filter Error

17.5 Frequency-Domain Decomposition of the Filter Error

17.6 I(1)-MSE Criterion

17.7 Unveiling the Unit-Root Singularity

17.8 Matrix Notation (Frequency Domain)

17.9 Customization

17.10 Regularization

17.11 An Application of Cointegration to Data Revisions

Link to section 14.3.3

Chapter 18

Non-Stationarity: Adaptive Filtering

18.1 Introduction

Contrast with difference-stationary processes

18.2 Filter Up-Dating

18.2.1 Univariate Up-Dating

18.2.2 Multivariate Up-Dating

18.3 Adaptive State Space and MDFA

18.4 MDFA-Stages

Chapter 19

Seasonal Adjustment

19.1 Introduction

Chapter 20

Summary and Links

20.1 Survey of MDFA Optimization Criteria

20.2 Consistency and Efficiency: a Tale of Two Philosophies

20.2.1 Knowing the Truth: Omniscience

20.2.2 Believing in Truth: Faith and Fatalism

20.2.3 From Truth to Effectiveness: Emphasizing Performances

Appendix

Appendix Discrete Sums of Trigonometric Terms

Proposition 5 *Let* $\omega_k = \dfrac{2k\pi}{T}$ *and* $j = 0, ..., T-1$. *For odd sample sizes* T *we have*

$$\begin{aligned}
\sum_{k=-(T-1)/2}^{(T-1)/2} \exp(ij\omega_k) &= \begin{cases} T & j = 0 \\ 0 & otherwise \end{cases} \\
\sum_{k=-(T-1)/2}^{(T-1)/2} \cos(j\omega_k) &= \begin{cases} T & j = 0 \\ 0 & otherwise \end{cases} \\
\sum_{k=-(T-1)/2}^{(T-1)/2} \sin(j\omega_k) &= 0
\end{aligned}$$

For even sample sizes T *we obtain similarly*

$$\begin{aligned}
\sum_{k=-T/2}^{T/2} w_k \exp(ij\omega_k) &= \begin{cases} T & j = 0 \\ 0 & otherwise \end{cases} \\
\sum_{k=-T/2}^{T/2} w_k \cos(j\omega_k) &= \begin{cases} T & j = 0 \\ 0 & otherwise \end{cases} \\
\sum_{k=-T/2}^{T/2} w_k \sin(j\omega_k) &= 0
\end{aligned}$$

where $w_k = \begin{cases} 1 & |k| < T/2 \\ 1/2 & |k| = T/2 \end{cases}$

Proof

Sums of sines vanish by (a)symmetry. Therefore sums of cosines and sums of complex exponentials must be identical. For odd T and $j \neq 0$ we have

$$\begin{aligned}
\sum_{k=-(T-1)/2}^{(T-1)/2} \exp(ij\omega_k) &= \exp\left(ij\omega_{-(T-1)/2}\right) \sum_{k=0}^{T-1} \exp\left(ij\omega_k\right) \\
&= \exp\left(ij\omega_{-(T-1)/2}\right) \frac{1 - \exp\left(ij\frac{2T\pi}{T}\right)}{1 - \exp\left(ij\frac{2\pi}{T}\right)} \\
&= 0
\end{aligned}$$

For $j = 0$ the proof follows from $\exp(ij\omega_k) = 1$. For even T we obtain

$$\begin{aligned}
\sum_{k=-T/2}^{T/2} w_k \exp(ij\omega_k) &= \exp\left(ij\omega_{-T/2}\right) \sum_{k=0}^{T} w_{k-T/2} \exp\left(ij\omega_k\right) \\
&= \exp\left(ij\omega_{-T/2}\right) \sum_{k=0}^{T-1} \exp\left(ij\omega_k\right) \\
&= \exp\left(ij\omega_{-T/2}\right) \frac{1 - \exp\left(ij\frac{2T\pi}{T}\right)}{1 - \exp\left(ij\frac{2\pi}{T}\right)} \\
&= 0
\end{aligned}$$

The following corollary is a direct consequence of the above results

Corollary 1

$$\frac{1}{T} \sum_{j=0}^{T-1} \exp(-ij\omega_k) = \begin{cases} 1 & \text{if } \omega_k = 0 \\ 0 & \text{otherwise} \end{cases}$$

Appendix Discrete Convolution

Proposition 6 *Let $Y_{it} = \sum_{k=-\infty}^{\infty} \gamma_{ik} X_{i,t-k}, i = 1, 2$ be the outputs of arbitrary (not necessarily symmetric) filters with transfer functions $\Gamma_i(\cdot), i = 1, 2$ and input series X_{it}, $i = 1, 2$. Define*

$$\begin{aligned}
r_T &:= \frac{2\pi}{T} \sum_{j=-(T-1)/2}^{(T-1)/2} \Xi_{TY_1}(\omega_j)\overline{\Xi_{TY_2}(\omega_j)} \\
&- \frac{2\pi}{T} \sum_{j=-(T-1)/2}^{(T-1)/2} \Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)\overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)} \qquad \text{(A.1)}
\end{aligned}$$

If $X_{it} \in C_f^0, i = 1, 2$ then

$$E[|r_T|] = \begin{cases} o(1/\sqrt{T}) & \text{if } \Gamma_i(\cdot) \in C_f^{1/2}, i = 1, 2 \\ O(1/T) & \text{if } \Gamma_i(\cdot) \in C_f^1, i = 1, 2 \end{cases} \qquad \text{(A.2)}$$

Proof:

Recall that we assume the sample size T to be an odd integer for notational simplicity (similar though slightly modified developments apply in the case of even T). Let

$$R_j(\omega) := \Xi_{TY_j}(\omega) - \Gamma(\omega)\Xi_{TX_j}(\omega)$$

For stationary $X_t \in C_f^0$ theorem 4.8 in Wildi (2005) shows that

$$E[|R_j(\omega)|^2] = \mathrm{O}(1/T) \qquad \text{(A.3)}$$

uniformly in j. Consider now the relevant error term

$$\begin{aligned}
r_T &= \frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}\Xi_{TY_1}(\omega_j)\overline{\Xi_{TY_2}(\omega_j)} \\
&\quad -\frac{2\pi}{T}\sum_{j=(T-1)/2}^{(T-1)/2}\Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)\overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)} \\
&= \frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}(R_1(\omega_j)+\Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)) \\
&\quad \overline{(R_2(\omega_j)+\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j))} \\
&\quad -\frac{2\pi}{T}\sum_{j=(T-1)/2}^{(T-1)/2}\Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)\overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)} \\
&= \frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}R_1(\omega_j)\overline{R_2(\omega_j)} \\
&\quad +\frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}R_1(\omega_j)\overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)} \\
&\quad +\frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}\Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)\overline{R_2(\omega)}
\end{aligned} \tag{A.4}$$

Using A.3 and the Cauchy-Schwartz-inequality, the first summand is of order $\mathrm{O}(1/T)$ and can be neglected. We now show that the second summand is of order $\mathrm{o}(1/\sqrt{T})$ (in absolute mean). A similar reasoning would apply to the third one by symmetry.
From

$$\begin{aligned}
\Xi_{TY_j}(\omega) &= \frac{\sqrt{2\pi}}{\sqrt{T}}\sum_{t=1}^{T}Y_{jt}\exp(-it\omega) \\
&= \frac{\sqrt{2\pi}}{\sqrt{T}}\sum_{t=1}^{T}\left(\sum_{k=-\infty}^{\infty}\gamma_{jk}X_{j,t-k}\right)\exp(-it\omega) \\
&= \sum_{k=-\infty}^{\infty}\gamma_{jk}\exp(-ik\omega)\frac{\sqrt{2\pi}}{\sqrt{T}}\sum_{t=1}^{T}X_{j,t-k}\exp(-i(t-k)\omega) \\
&= \sum_{k=-\infty}^{\infty}\gamma_{jk}\exp(-ik\omega)\frac{\sqrt{2\pi}}{\sqrt{T}}\sum_{t=1-k}^{T-k}X_{jt}\exp(-it\omega)
\end{aligned}$$

one deduces

$$\begin{aligned}
R_j(\omega_n) &:= \Xi_{TY_j}(\omega_n) - \Gamma(\omega_n)\Xi_{TX_j}(\omega_n) \\
&= \sum_{k=-\infty}^{\infty} \gamma_{jk} \exp(-ik\omega_n) \\
&\quad \frac{\sqrt{2\pi}}{\sqrt{T}} \left(\sum_{t=1-k}^{T-k} X_{jt} \exp(-it\omega_n) - \sum_{t=1}^{T} X_{jt} \exp(-it\omega_n) \right) \\
&= \sum_{k=-\infty}^{\infty} \gamma_{jk} \exp(-ik\omega_n) \frac{\sqrt{2\pi}}{\sqrt{T}} \\
&\quad \left[\sum_{t=1}^{\min(k,T)} \Big(X_{j,t-k} \exp(-i(t-k)\omega_n) \right. \\
&\quad \left. -X_{j,T+t-k} \exp(-i(T+t-k)\omega_n) \Big) \right. \\
&\quad \left. + \sum_{t=1}^{\min(-k,T)} \Big(X_{j,T+1-t-k} \exp(-i(T+1-t-k)\omega_n) \right. \right. \\
&\quad \left. \left. -X_{j,1-t-k} \exp(-i(1-t-k)\omega_n) \Big) \right] \\
&= \sum_{k=-\infty}^{\infty} \gamma_{jk} \exp(-ik\omega_n) \frac{\sqrt{2\pi}}{\sqrt{T}} \\
&\quad \left[\sum_{t=1}^{\min(k,T)} \Big(X_{j,t-k} - X_{j,T+t-k} \Big) \exp(-i(t-k)\omega_n) \right. \\
&\quad \left. + \sum_{t=1}^{\min(-k,T)} \Big(X_{j,T+1-t-k} - X_{j,1-t-k} \Big) \exp(-i(1-t-k)\omega_n) \right]
\end{aligned}$$

where the last equality follows from $T\omega_n$ being a multiple of 2π. We now analyze the error component A.4.

$$\begin{aligned}
&\frac{2\pi}{T} \sum_{j=-(T-1)/2}^{(T-1)/2} R_1(\omega_j) \overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)} \\
= &\left(\frac{2\pi}{T} \right)^2 \sum_{j=-(T-1)/2}^{(T-1)/2} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} I_{\{l>0\}} \\
&\times \sum_{r=1}^{\min(l,T)} \sum_{t=1}^{T} \gamma_{2k}\gamma_{1l} X_{2t} \Big(X_{1,r-l} - X_{1,T+r-l} \Big) \exp(-i(r-l-k-t)\omega_j) \\
&+ \left(\frac{2\pi}{T} \right)^2 \sum_{j=-(T-1)/2}^{(T-1)/2} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} I_{\{l<0\}} \\
&\times \sum_{r=1}^{\min(-l,T)} \sum_{t=1}^{T} \gamma_{2k}\gamma_{1l} X_{2t} \Big(X_{1,T+1-r-l} - X_{1,1-r-l} \Big) \\
&\exp(-i(1-r-l-k-t)\omega_j)
\end{aligned}$$

where $I_{\{l>0\}}$ and $I_{\{l<0\}}$ are indicator functions. For k, l, r fixed let

$s = r - l - k \bmod(T)$ so that

$$\frac{1}{T}\sum_{j=-(T-1)/2}^{(T-1)/2} \exp(-i(r-l-k-t)\omega_j) = \begin{cases} 0 & t \neq s \\ 1 & \text{else} \end{cases}$$

Thus

$$\begin{aligned} & E\left[\left|\frac{1}{T}\sum_{t=1}^{T}\gamma_{2k}\gamma_{1l}X_{2t}\Big(X_{1,r-l} - X_{1,T+r-l}\Big)\right.\right. \\ & \left.\left.\sum_{j=-(T-1)/2}^{(T-1)/2}\exp(-i(r-l-k-t)\omega_j)\right|\right] \\ \leq\ & 2|\gamma_{2k}\gamma_{1l}|\sqrt{E[X_{1t}^2]E[X_{2t}^2]} \end{aligned}$$

and analogously

$$\begin{aligned} & E\left[\left|\frac{1}{T}\sum_{t=1}^{T}\gamma_{2k}\gamma_{1l}X_{2t}\Big(X_{1,T+1-r-l} - X_{1,1-r-l}\Big)\right.\right. \\ & \left.\left.\sum_{j=-(T-1)/2}^{(T-1)/2}\exp(-i(1-r-l-k-t)\omega_j)\right|\right] \\ \leq\ & 2|\gamma_{2k}\gamma_{1l}|\sqrt{E[X_{1t}^2]E[X_{2t}^2]} \end{aligned}$$

Therefore

$$\begin{aligned} E[|r_T|] \leq\ & E\left[\left|\frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2} R_1(\omega_j)\overline{\Gamma_2(\omega_j)\Xi_{TX_2}(\omega_j)}\right|\right] \\ & + E\left[\left|\frac{2\pi}{T}\sum_{j=-(T-1)/2}^{(T-1)/2}\Gamma_1(\omega_j)\Xi_{TX_1}(\omega_j)\overline{R_2(\omega)}\right|\right] + \mathrm{O}(1/T) \\ \leq\ & \frac{4\pi^2}{T}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} 2|\gamma_{2k}\gamma_{1l}||\min(|l|,T)|\sqrt{E[X_{1t}^2]E[X_{2t}^2]} \\ & + \frac{4\pi^2}{T}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} 2|\gamma_{2k}\gamma_{1l}||\min(|k|,T)|\sqrt{E[X_{1t}^2]E[X_{2t}^2]} \\ & + \mathrm{O}(1/T) \\ =\ & \begin{cases} \mathrm{o}(1/\sqrt{T}) & \text{if } \Gamma_i(\cdot) \in C_f^{1/2}, i = 1,2 \\ \mathrm{O}(1/T) & \text{if } \Gamma_i(\cdot) \in C_f^{1}, i = 1,2 \end{cases} \end{aligned}$$

where the last equality - for $\Gamma_1(\cdot) \in C_f^{1/2}$ - follows from

$$\begin{aligned} & \frac{1}{T}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}|\gamma_{2k}\gamma_{1l}||\min(|l|,T)| \\ \leq\ & \frac{1}{\sqrt{T}}\sum_{k=-\infty}^{\infty}|\gamma_{2k}|\sum_{l=-\infty}^{\infty}|\gamma_{1l}||\sqrt{|l|}|\frac{\sqrt{\min(|l|,T)}}{\sqrt{T}} \\ =\ & \mathrm{o}(1/\sqrt{T}) \end{aligned}$$

since $\Gamma_1(\cdot) \in C_f^{1/2}$ and $\lim_{T\to\infty} \frac{\sqrt{\min(|l|,T)}}{\sqrt{T}} = 0$ for each l (and analogously for $\Gamma_2(\cdot) \in C_f^{1/2}, C_f^1$). This completes the proof of the proposition.

Appendix Performances Customized Designs

A.3 Peak Correlations, Curvatures and MSEs: In- and Out-of-Sample Distributions

The following graphs correspond to the in- and out-of-sample distributions of peak-correlations, curvatures and sample MSEs for the processes $a_1 = 0.9$ and $a_1 = -0.9$ in section 7.6.

$a_1 = 0.9$

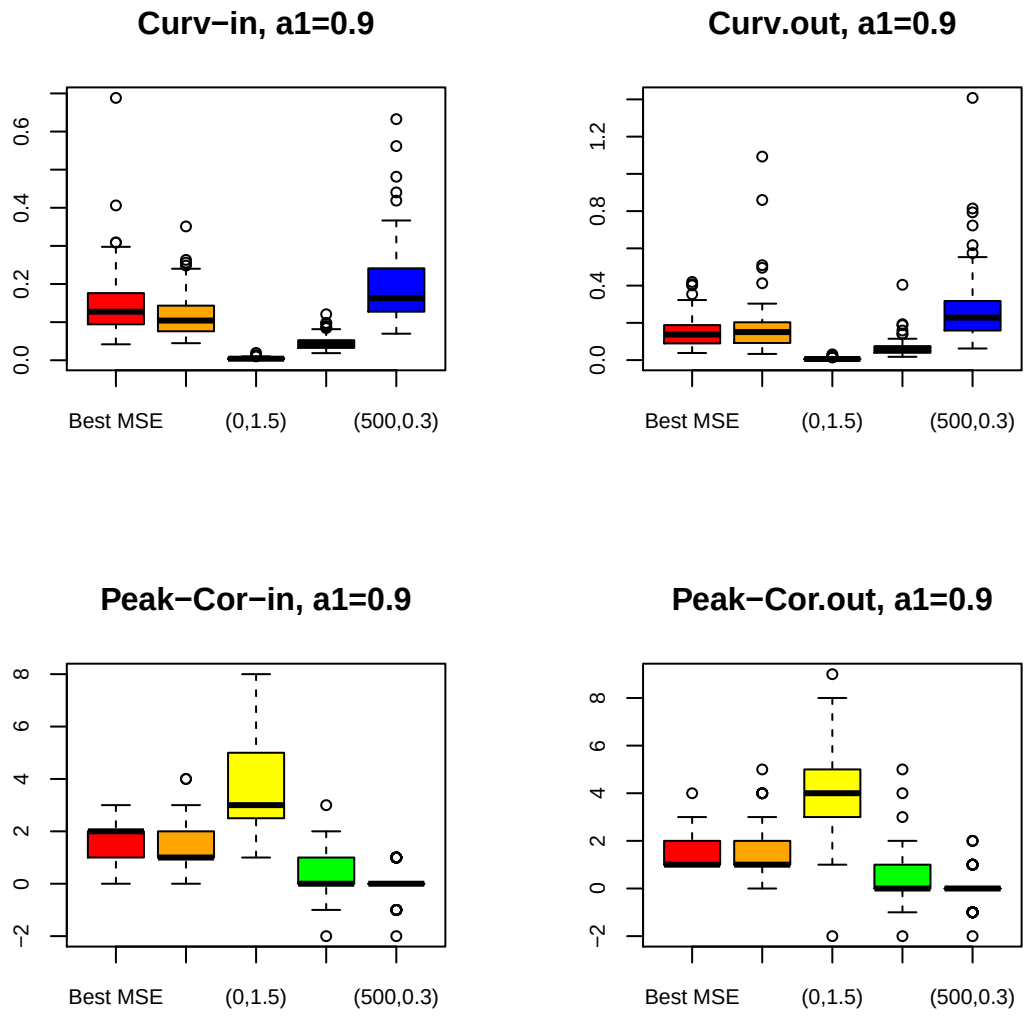


Figure 1: Empirical distributions of Curvature and Peak-Correlation of best theoretical MSE (red), empirical MSE (yellow), balanced customized (green), unbalanced smoothness (cyan) and unbalanced timeliness (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=-0.9

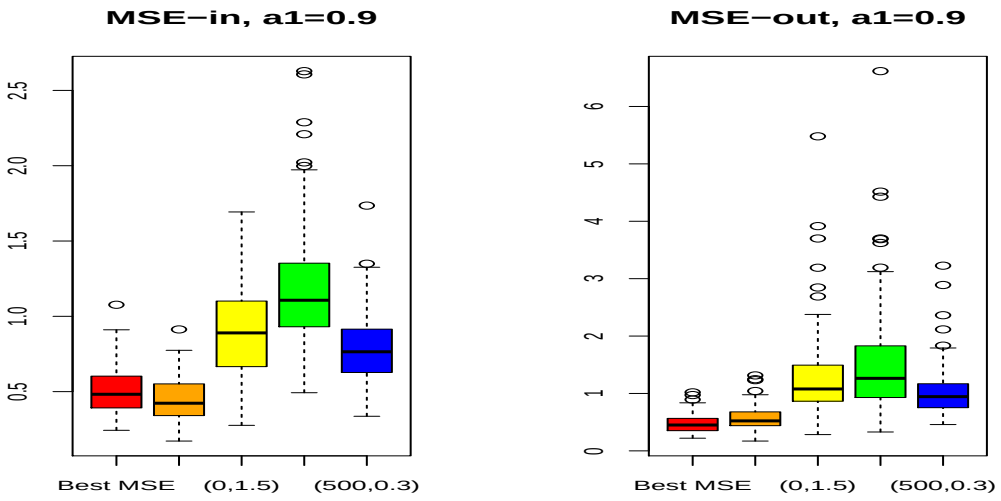


Figure 2: Empirical distributions of Sample MSEs of best theoretical MSE (red), empirical MSE (yellow), balanced customized (green), unbalanced smoothness (cyan) and unbalanced timeliness (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=-0.9

$a_1 = -0.9$

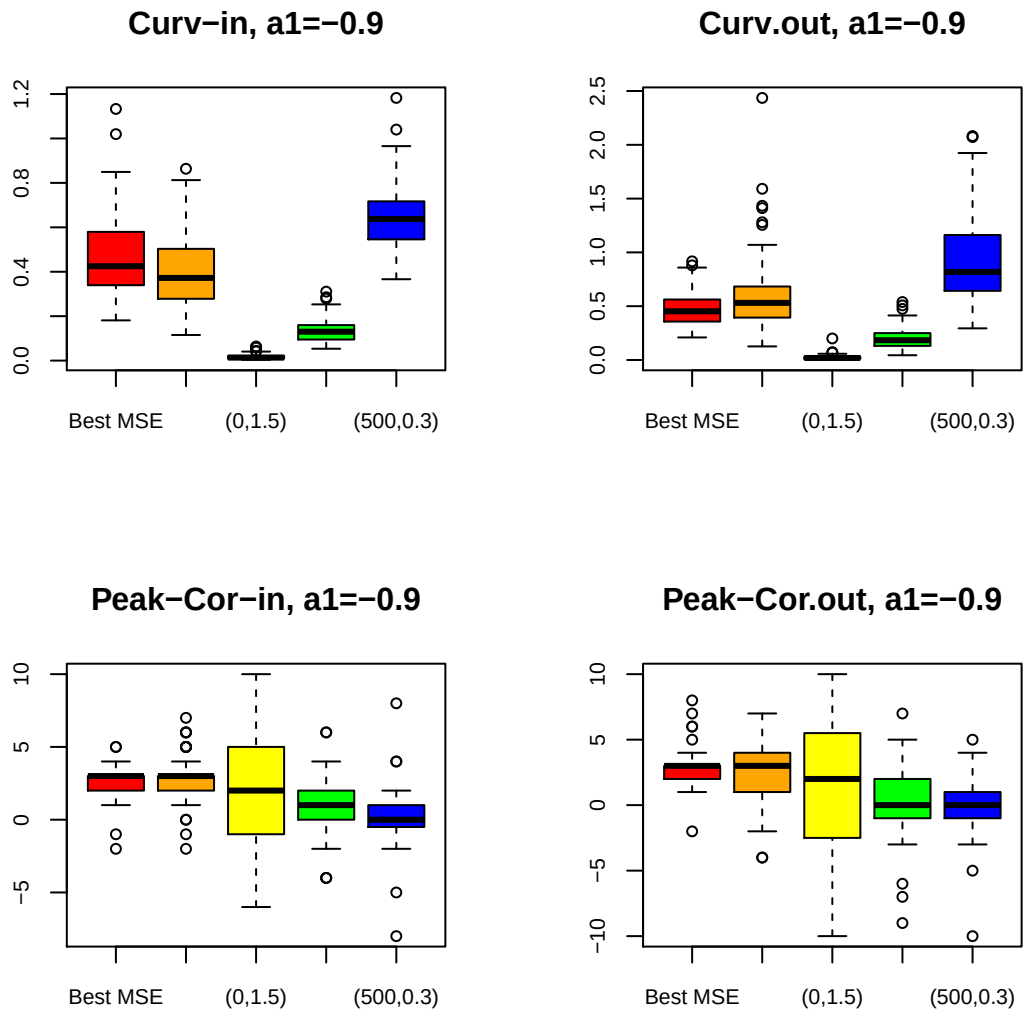


Figure 3: Empirical distributions of Curvature and Peak-Correlation of best theoretical MSE (red), empirical MSE (yellow), balanced customized (green), unbalanced smoothness (cyan) and unbalanced timeliness (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=0.9

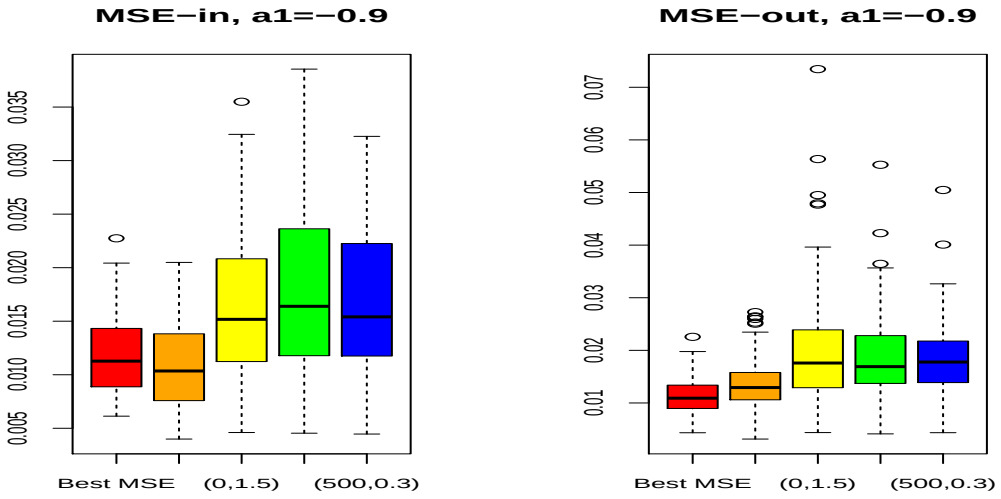


Figure 4: Empirical distributions of Sample MSEs of best theoretical MSE (red), empirical MSE (yellow), balanced customized (green), unbalanced smoothness (cyan) and unbalanced timeliness (blue) filters. All empirical filters are based on the periodogram: in-sample (left plots) and out-of-sample (right plots) for a1=0.9

Appendix MDFA: R-Code