Multivariate Real-Time Signal Extraction

Marc Wildi and Tucker McElroy

May 14, 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 **??**. But before discussing customization in Chapter 5, we discuss the applications of forecasting and nowcasting, as well as the impact of data vintage, in Chapter 7). Then the basic LPP treatment is extended to nonstationary processes in Chapter 4, followed by a discussion of filter constraints (Chapter 3). This treatment is extended to the case of co-integration in Chapter **??**. Applications to replicating and enhancing classical model-based approaches and HP/CF-filters are given in Chapter **??**, while more sophisticated gain/loss structures are discussed in Chapter **??**. Additional topics include inference (Chapter **??**), regularization (Chapter 6), data revisions (Chapter **??**), mixed-frequency data (Chapter 8), and adaptive filtering (Chapter **??**).

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",
            main = "Periodogram")
        axis(1, at = 1 + 0:6 * len/12, labels = c("0", "pi/6",
            "2pi/6", "3pi/6", "4pi/6", "5pi/6", "pi"))
        axis(2)
        box()
        plot(log(per), type = "l", axes = F, xlab = "Frequency",
            ylab = "Log-periodogram", main = "Log-periodogram")
        axis(1, at = 1 + 0:6 * len/12, labels = c("0", "pi/6",
            "2pi/6", "3pi/6", "4pi/6", "5pi/6", "pi"))
        axis(2)
        box()
    }
    return(list(DFT = DFT, per = per))
}
```

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)
```

```
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))
}
```

This function is nested in the multivariate MDFA, in the sense that the latter can replicate the former perfectly when suitably parametrized; see Section **??** 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)
```

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

The additional control parameters *lambda*, *eta* allow for customization of the filter, as discussed below in Chapter 5. The Boolean *i1* and *i2* can enforce useful filter constraints; see Chapter 3. 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 8. Another facet, is that we allow for the possibility of integrated processes; see Chapter 4. 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: 0x00000016c096a2a8>
<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 6); it can treat data revisions (Chapter **??**), mixed-frequency problems (Chapter 8), and non-stationarity (Chapters 4 and **??**. 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 **??**; 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

$$L X_t = X_{t-1}.$$

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 $1 \leq j, k \leq n$. 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 $\ell \geq 0$, 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

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

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 $1 \leq s, t \leq T$, 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 4 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 4. 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 5 Multi-step Ahead Forecasting. The LPP corresponds to optimal h-step forecasting, and the forecast error is $[L^{-h}\, e_1^{\prime} - \widehat{\Psi}(L)]\, X_t$. 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 6 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 that is stable, i.e., has 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), $\widehat{\Psi}(L) = A_{\Psi}(\Phi) = e_1' \, \Phi^h$. This formula demonstrates that it is essential that Φ be stable, and if fitting a VAR(1) we must parametrize Φ such that stability is guaranteed. The following parametrization (Roy, McElroy, and Linton (2019) provides a bijection from $\mathbb{R}^{n^2} \times \{\pm 1\}$ to the space of stable $n \times n$ matrices:

$$\begin{aligned} (\phi, \delta) &\rightarrow (V, Q) \\ V &= L \, D \, L' \\ Q &= E \, (1_n - S) \, (1_n + S)^{-1} \\ \Phi &= V^{1/2} \, Q \, (1_n + V)^{-1/2} \end{aligned}$$

provides the map from the parameters to Φ. Here V is a positive definite matrix with Cholesky decomposition in terms of a unit lower triangular matrix L and a positive diagonal matrix D; any such matrix can be described by $\binom{n}{2}$ arbitrary real entries in the lower portion of L, and by the exponential of n arbitrary real numbers for D. The matrix Q is orthogonal, given by the Cayley representation, where S is anti-symmetric – and hence can be described by $\binom{n}{2}$ arbitrary real numbers – and E is diagonal, with the first entry given by δ (which equals ± 1) and the remaining $n-1$ entries are equal to one. The inverse transformation is given by

$$\begin{aligned} \Phi &\rightarrow (V, Q) \\ V &= \sum_{j \geq 1} \Phi^j \, \Phi^{j\prime} \\ Q &= V^{-1/2} \, \Phi \, (1_n + V)^{1/2} \\ S &= (1_n + E \, Q)^{-1} \, (1_n - E \, Q) \\ \delta &= \det Q, \end{aligned}$$

where ϕ can be recovered from the entries of L and D in the Cholesky decomposition of V, along with the lower left entries of S.

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

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

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

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

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 $\widetilde{\Sigma}$. □

Remark 2 One of the conditions of Proposition 2 is that the PTV is unique. Non-uniqueness can arise in practice, but any PTV that minimizes the LPP criterion is suitable for generating an optimal solution. Hence, even though in this situation a particular PTV does not equal the true parameter, yet it generates the same minimal value of J_Ψ as the true parameter, and is "just as good" as the true parameter for purposes of the particular LPP. We do not care about correct model fitting *per se.*

Example 7 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

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

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 2 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, as described previously. 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

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

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

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

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

$$\Phi = \left[\begin{array}{cc} 1 & .5 \\ -.2 & .3 \end{array} \right]$$

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 (use the stable parametrization of Φ, using two choices of δ) 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. (As usual, the first of the two series is the target.) Do you obtain a unique minimizer? Do the parameter estimates appear to be consistent? How do the estimates compare to the Yule-Walker estimates? Repeat for $T = 200$ and $T = 500$.

```
> # Simulate a Gaussian VAR(1) of sample size 100:
> T <- 100
> phi.matrix <- rbind(c(1,.5),c(-.2,.3))
> innovar.matrix <- diag(2)
> true.psidelta <- var1.par2psi(phi.matrix,100)
> gamma.0 <- matrix(solve(diag(4) - phi.matrix %x% phi.matrix) %*%
+         matrix(innovar.matrix,ncol=1),nrow=2)
> x.init <- t(chol(gamma.0)) %*% rnorm(2)
> x.next <- x.init
> x.sim <- NULL
> for(t in 1:T)
+ {
+         x.next <- phi.matrix %*% x.next + rnorm(2)
+         x.sim <- cbind(x.sim,x.next)
+ }
> x.sim <- ts(t(x.sim))
> x.acf <- acf(x.sim,type="covariance",plot=FALSE,lag.max=T)[[1]]
> x.acf <- aperm(aperm(x.acf,c(3,2,1)),c(2,1,3))
> # Yule-Walker fit
> phi.yw <- x.acf[,,2] %*% solve(x.acf[,,1])        # phi coefficient from YW
> yw.psidelta <- var1.par2psi(phi.yw,100)
> # 1-step ahead forecasting
> psi.array <- array(0,c(1,2,1))
> psi.array[,,1] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))

[1] 1.430235 1.430235

> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))

         [,1]
[1,] 1.430235

> # phi coefficient from 1-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

           [,1]      [,2]
[1,]  0.9348059 0.5314934
[2,] -0.5589701 0.2793141

> # phi coefficient from 1-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

           [,1]       [,2]
[1,]  0.9347882  0.5313741
[2,] -1.5157224 -1.0416827

> # phi coefficient from Yule-Walker
> print(phi.yw)

           [,1]      [,2]
[1,]  0.9348056 0.5314925
[2,] -0.1783947 0.2670181

> # 2-step ahead forecasting
> psi.array <- array(0,c(1,2,2))
> psi.array[,,1] <- c(0,0)
> psi.array[,,2] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))

[1] 3.007243 3.007242

> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))
```

```
         [,1]
[1,] 3.017099

> # phi coefficient from 2-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

           [,1]      [,2]
[1,]  0.9183654 0.3693359
[2,] -0.2318667 0.5703253

> # phi coefficient from 2-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

           [,1]       [,2]
[1,] -0.3842571 -2.3329655
[2,] -0.2614505  0.1487464

> # phi coefficient from Yule-Walker
> print(phi.yw)

           [,1]      [,2]
[1,]  0.9348056 0.5314925
[2,] -0.1783947 0.2670181

> # low-pass LPP
> mu <- pi/24
> psi.array <- array(c(1,0) %x% sin(seq(1,T-1)*mu)/(pi*seq(1,T-1)),c(1,T-1,2))
> psi.array <- aperm(psi.array,c(1,3,2))
> acf.array <- x.acf
> theta <- rep(0,4)
> var1.fit.bk.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+          acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.bk.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+          acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.bk.Pos$value,var1.fit.bk.Neg$value))

[1] 0.7782000 0.8039344

> lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]])

          [,1]
[1,] 0.7784239

> # phi coefficient from low-pass LPP, delta = 1
> print(var1.psi2par(var1.fit.bk.Pos$par,1))
```

```
            [,1]      [,2]
[1,]  0.82390589 0.2256008
[2,] -0.02389912 0.7338540
```

```
> # phi coefficient from low-pass LPP, delta = -1
> print(var1.psi2par(var1.fit.bk.Neg$par,-1))
```

```
          [,1]          [,2]
[1,] 0.7708965 -1.432233e-05
[2,] 2.8936879 -9.986220e-01
```

```
> # phi coefficient from Yule-Walker
> print(phi.yw)
```

```
           [,1]      [,2]
[1,]  0.9348056 0.5314925
[2,] -0.1783947 0.2670181
```

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

$$\Theta = \left[\begin{array}{cc} 1 & 0 \\ .4 & 2 \end{array} \right]$$

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. (As usual, the first of the two series is the target.) Why do the parameter estimates not match those of Θ? How do the estimates compare to the Yule-Walker estimates? Repeat for $T = 200$ and $T = 500$, and explain your results.

```
> # Simulate a Gaussian VMA(1) of sample size 100:
> T <- 100
> theta.matrix <- rbind(c(1,0),c(.4,2))
> innovar.matrix <- diag(2)
> eps.old <- rnorm(2)
> x.sim <- NULL
> for(t in 1:T)
+ {
+         eps.next <- rnorm(2)
+         x.next <- eps.next + theta.matrix %*% eps.old
+         eps.old <- eps.next
+         x.sim <- cbind(x.sim,x.next)
+ }
```

```
> x.sim <- ts(t(x.sim))
> x.acf <- acf(x.sim,type="covariance",plot=FALSE,lag.max=T)[[1]]
> x.acf <- aperm(aperm(x.acf,c(3,2,1)),c(2,1,3))
> # Yule-Walker
> phi.yw <- x.acf[,,2] %*% solve(x.acf[,,1])        # phi coefficient from YW
> yw.psidelta <- var1.par2psi(phi.yw,100)
> # 1-step ahead forecasting
> psi.array <- array(0,c(1,2,1))
> psi.array[,,1] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))

[1] 1.865806 1.865806

> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))

         [,1]
[1,] 1.865806

> # phi coefficient from 1-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

          [,1]        [,2]
[1,] 0.4088079 -0.09822759
[2,] 0.1398298  0.71881066

> # phi coefficient from 1-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

           [,1]        [,2]
[1,]  0.4088452 -0.09821484
[2,] -0.9785868 -0.56418264

> # phi coefficient from Yule-Walker
> print(phi.yw)

            [,1]        [,2]
[1,]  0.40880769 -0.09822759
[2,] -0.05899914  0.33755351
```

```
> # 2-step ahead forecasting
> psi.array <- array(0,c(1,2,2))
> psi.array[,,1] <- c(0,0)
> psi.array[,,2] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))

[1] 2.230713 2.296949

> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))

         [,1]
[1,] 2.467927

> # phi coefficient from 2-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

          [,1]       [,2]
[1,] 0.0758301 -0.2361368
[2,] 0.6490909  0.1834502

> # phi coefficient from 2-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

            [,1]        [,2]
[1,] -0.05185672 -0.02473498
[2,] -0.06785990  0.45889013

> # phi coefficient from Yule-Walker
> print(phi.yw)

            [,1]        [,2]
[1,]  0.40880769 -0.09822759
[2,] -0.05899914  0.33755351

> # low-pass LPP
> mu <- pi/24
> psi.array <- array(c(1,0) %x% sin(seq(1,T-1)*mu)/(pi*seq(1,T-1)),c(1,T-1,2))
> psi.array <- aperm(psi.array,c(1,3,2))
> acf.array <- x.acf
> theta <- rep(0,4)
> var1.fit.bk.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,method="BFGS")
> var1.fit.bk.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,method="BFGS")
> # fits using both delta values, compared
> print(c(var1.fit.bk.Pos$value,var1.fit.bk.Neg$value))

[1] 0.04783541 0.04782558

> lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]])

           [,1]
[1,] 0.05092779

> # phi coefficient from low-pass LPP, delta = 1
> print(var1.psi2par(var1.fit.bk.Pos$par,1))

            [,1]        [,2]
[1,] -0.06199006  0.07558949
[2,] -0.98694748 -0.38815893

> # phi coefficient from low-pass LPP, delta = -1
> print(var1.psi2par(var1.fit.bk.Neg$par,-1))

           [,1]       [,2]
[1,] -0.1448378 0.04558693
[2,]  0.7773912 0.50068117

> # phi coefficient from Yule-Walker
> print(phi.yw)

            [,1]        [,2]
[1,]  0.40880769 -0.09822759
[2,] -0.05899914  0.33755351
```

Exercise 3 Incorrect VAR(1) Forecasting. Simulate a sample of size $T = 1000$ from a bivariate Vector Moving Avergage process of order two, or VMA(2), given by

$$X_t = \epsilon_t + \Theta_1 \, \epsilon_{t-1} + \Theta_2 \, \epsilon_{t-2},$$

where

$$\Theta_1 = \left[\begin{array}{cc} .9 & 2 \\ 0 & .8 \end{array} \right]$$

$$\Theta_2 = \left[\begin{array}{cc} .8 & 3.5 \\ .2 & .6 \end{array} \right]$$

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 the 2-step ahead forecasting LPP. (As usual, the first of the two series is the target.) Then generate 2-step ahead forecasts for the entire sample. Compare the in-sample performance (for the first series) using the LPP result and the Yule-Walker result.

```
> # Simulate a Gaussian VMA(2) of sample size 1000:
> T <- 1000
> theta1.matrix <- rbind(c(.9,2),c(0,.8))
> theta2.matrix <- rbind(c(.8,3.5),c(0.2,.6))
> innovar.matrix <- diag(2)
> eps.old <- rnorm(2)
> eps.oldd <- rnorm(2)
> x.sim <- NULL
> for(t in 1:T)
+ {
+         eps.next <- rnorm(2)
+         x.next <- eps.next + theta1.matrix %*% eps.old +
+                 theta2.matrix %*% eps.oldd
+         eps.oldd <- eps.old
+         eps.old <- eps.next
+         x.sim <- cbind(x.sim,x.next)
+ }
> x.sim <- ts(t(x.sim))
> x.acf <- acf(x.sim,type="covariance",plot=FALSE,lag.max=T)[[1]]
> x.acf <- aperm(aperm(x.acf,c(3,2,1)),c(2,1,3))
> # Yule-Walker
> phi.yw <- x.acf[,,2] %*% solve(x.acf[,,1])        # phi coefficient from YW
> yw.psidelta <- var1.par2psi(phi.yw,100)
> # 1-step ahead forecasting
> psi.array <- array(0,c(1,2,1))
> psi.array[,,1] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,lower = c(-Inf,-30,-30,-Inf),
+         upper = c(Inf,20,20,Inf),method="L-BFGS-B")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,lower = c(-Inf,-30,-30,-Inf),
+         upper = c(Inf,20,20,Inf),method="L-BFGS-B")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))
```

```
[1] 8.036465 8.036465
```

```
> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))

         [,1]
[1,] 8.036465

> # phi coefficient from 1-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

           [,1]       [,2]
[1,] -0.0220511  2.3982423
[2,] -0.1852666 -0.5516525

> # phi coefficient from 1-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

            [,1]        [,2]
[1,] -0.02205007  2.39823878
[2,]  0.19520816 -0.09218997

> # phi coefficient from Yule-Walker
> print(phi.yw)

            [,1]      [,2]
[1,] -0.02205180 2.3982434
[2,] -0.08104211 0.7708112

> # 2-step ahead forecasting
> psi.array <- array(0,c(1,2,2))
> psi.array[,,1] <- c(0,0)
> psi.array[,,2] <- c(1,0)
> acf.array <- x.acf[,,1:3]
> theta <- rep(0,4)
> var1.fit.2step.Pos <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=1,lower = c(-Inf,-30,-30,-Inf),
+         upper = c(Inf,20,20,Inf),method="L-BFGS-B")
> var1.fit.2step.Neg <- optim(theta,lpp.var1,psi.array=psi.array,
+         acf.array=acf.array,delta=-1,lower = c(-Inf,-30,-30,-Inf),
+         upper = c(Inf,20,20,Inf),method="L-BFGS-B")
> # fits using both delta values, compared
> print(c(var1.fit.2step.Pos$value,var1.fit.2step.Neg$value))

[1] 10.44909 12.65618

> print(lpp.var1(yw.psidelta[[1]],psi.array,acf.array,yw.psidelta[[2]]))

         [,1]
[1,] 11.67957
```

```
> # phi coefficient from 1-step LPP, delta = 1
> print(var1.psi2par(var1.fit.2step.Pos$par,1))

            [,1]     [,2]
[1,]  0.08169268 1.852166
[2,] -0.26545460 1.355849

> # phi coefficient from 1-step LPP, delta = -1
> print(var1.psi2par(var1.fit.2step.Neg$par,-1))

           [,1]     [,2]
[1,] -0.4833857 1.578068
[2,] -0.3950306 1.420245

> # phi coefficient from Yule-Walker
> print(phi.yw)

            [,1]      [,2]
[1,] -0.02205180 2.3982434
[2,] -0.08104211 0.7708112

> # select choices of phi from above
> if(var1.fit.2step.Pos$value < var1.fit.2step.Neg$value) {
+         phi.lpp <- var1.psi2par(var1.fit.2step.Pos$par,1)
+ } else {
+         phi.lpp <- var1.psi2par(var1.fit.2step.Neg$par,-1)
+ }
> # LPP case
> fore.lpp <- phi.lpp %^% 2
> fore.lpp <- fore.lpp[1,]
> fore.casts <- fore.lpp %*% t(x.sim[1:(T-2),])
> plot(ts(as.vector(x.sim[3:T,1])))
> lines(ts(as.vector(fore.casts)),col=2)
> sum((x.sim[3:T,1] - fore.casts)^2)/(T-2)

[1] 10.38676

> # YW case
> fore.yw <- phi.yw %^% 2
> fore.yw <- fore.yw[1,]
> fore.casts <- fore.yw %*% t(x.sim[1:(T-2),])
> plot(ts(as.vector(x.sim[3:T,1])))
> lines(ts(as.vector(fore.casts)),col=2)
> sum((x.sim[3:T,1] - fore.casts)^2)/(T-2)

[1] 11.62949
```

Chapter 3

Filter Constraints

We propose and analyze a set of filter constraints that may be relevant for certain applications. Formally, the constraints ensure that the filter error has mean zero, and hence may be important to impose when the data process has a time-varying mean, or is non-stationary.

finiteness of the mean-square filter error in the case of integrated processes[1]. Section **??** sets-up the context; a link to integrated processes is proposed in section **??**; a general matrix notation is proposed in section **??**; section **??** extends the former (unconstrained) MDFA-criterion to the constrained case; finally, section **??** illustrates effects (of the constraints) on characteristics of real-time filters.

[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.

Chapter 4

Integrated Processes

The univariate DFA was introduced in Chapter 2 for the case of a stationary time series. However, most economic series are non-stationary, requiring suitable differencing to be rendered stationary. Such time series are called *integrated processes*. This chapter discusses the DFA methodology for multivariate integrated processes. We also give a thorough and novel treatment of co-integration, and how this is related to DFA filter constraints.

4.1 Stationary Multivariate DFA

Here we set forth our basic treatment of the stationary multivariate DFA, which builds on the univariate material of previous chapters. Moreover, we distinguish carefully between theoretical and empirical versions of the DFA criteria. Consider an m-variate time series $\{x_t\}$ with component series $\{x_t^{(j)}\}$ for $1 \leq j \leq m$. Let our target signal be defined as $y_t = \Gamma(B) x_t$, where $\Gamma(z)$ is a $1 \times m$ dimensional multivariate filter. Let the component filters be denoted $\Gamma^{(j)}(B)$, so that

$$y_t = \sum_{j=1}^{m} \Gamma^{(j)}(B) x_t^{(j)},$$

where each $\Gamma^{(j)}(B)$ is a univariate filter that operates on the jth component input series. The real-time estimation problem is to find m univariate concurrent filters $\widehat{\Gamma}^{(j)}(B)$ to causally estimate the signal via

$$\widehat{y}_t = \sum_{j=1}^{m} \widehat{\Gamma}^{(j)}(B) x_t^{(j)}.$$

Together the concurrent filters form a single multivariate filter $\widehat{\Gamma}(B)$ that is $1 \times m$ dimensional. We seek concurrent filters such that the univariate filter error process $y_t - \widehat{y}_t$ is stationary with minimal variance.

The coefficients of the filters are denoted $\{\gamma_k\}$ and $\{\widehat{\gamma}_k\}$ respectively for the target and real-time cases. These are each $1 \times m$ dimensional matrices, with a superscript denoting the particular component. Thus, $\gamma_k^{(j)}$ weights the jth series, at k time units in the past of the present time of consideration, and $\widehat{\gamma}_k^{(j)}$ is similarly defined for the real-time estimator.

Recall that in the univariate DFA (Chapter 4), the concurrent filters may satisfy certain filter constraints that are imposed by the user. As we discuss below, some of these constraints may be necessary when the input process is integrated, in order that the filter error be stationary and mean zero. Because $\{x_t\}$ is stationary, there exists a spectral representation in terms of a vector orthogonal increments process $\mathbb{Z}(\omega)$, as described in Brockwell and Davis (1991):

$$x_t = \int_{-\pi}^{\pi} e^{i\omega t} \, d\mathbb{Z}(\omega).$$

The orthogonal increments process is a random measure with the uncorrelated increments property, which says that for any two disjoint subsets $A_1, A_2 \subset [-\pi, \pi]$, the random vectors $\mathbb{Z}(A_1)$ and $\mathbb{Z}(A_2)$ are mean zero and uncorrelated with one another. Moreover, for any such set A we have

$$\mathrm{Var}\mathbb{Z}(A) = \mathbb{E}\mathbb{Z}(A)\overline{\mathbb{Z}(A)}' = \frac{1}{2\pi} \int_A dH(\omega), \tag{4.1}$$

where H is the spectral distribution function of the process $\{x_t\}$. When the process is purely nondeterministic (which we henceforth assume), the spectral distribution is absolutely continuous in each of its m^2 component functions, and its derivative h is called the spectral density. It is a matrix-valued function of frequency $\omega \in [-\pi, \pi]$.

It is convenient to introduce notation for the integration in (4.1):

$$\langle g \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} g(\omega) \, d\omega$$

for any function g; hence $\mathrm{Var}\mathbb{Z}(A)$ can be rewritten $\langle 1_A \, h \rangle$. The autocovariances $R(j)$ of the process have the relationship

$$R(j) = \mathbb{E}[x_t x_{t-j}'] = \langle e^{ij\cdot} H \rangle.$$

Applying the filter $\Gamma(B)$ and using the linearity of the spectral representation, we obtain

$$y_t = \int_{-\pi}^{\pi} e^{i\omega t} \, \Gamma(e^{-i\omega}) \, d\mathbb{Z}(\omega).$$

Recall that $\Gamma(e^{-i\omega})$ is the frequency response function (frf) of the filter. For the real-time filter, we also have

$$\widehat{y}_t = \int_{-\pi}^{\pi} e^{i\omega t} \, \widehat{\Gamma}(e^{-i\omega}) \, d\mathbb{Z}(\omega).$$

In this case, the real-time filter $\widehat{\Gamma}(B)$ is causal, so its frf will typically be complex-valued. This differs from the target frf, which in many cases is a symmetric filter, and hence real-valued.

Next, the real-time estimation error is

$$y_t - \widehat{y}_t = \int_{-\pi}^{\pi} e^{i\omega t} \, \left(\Gamma(e^{-i\omega}) - \widehat{\Gamma}(e^{-i\omega}) \right) \, d\mathbb{Z}(\omega).$$

Note that the integrand only depends on ω through $e^{-i\omega}$. It is therefore convenient to use the abbreviation $z = e^{-i\omega}$. Although the phase properties of this error row vector may be of interest, in DFA we focus on the mean squared error, which works out to be

$$\mathbb{E}(y_t - \widehat{y}_t)(y_t - \widehat{y}_t)' = \langle \left(\Gamma(z) - \widehat{\Gamma}(z) \right) \, h \, \left(\Gamma(\overline{z}) - \widehat{\Gamma}(\overline{z}) \right)' \rangle. \tag{4.2}$$

Pretending that the spectral density h is known, we could use (4.2) as a criterion to obtain an optimal real-time approximation to the given Γ. The optimization would be computed over a particular class of concurrent filters.

For an illustration (that generalizes the earlier univariate treatments), suppose we consider $\widehat{\Gamma}$ from the space of causal length L filters, with no constraint on the numerical values of the filter coefficients other than they are real numbers. Write these coefficients as a single vector ϕ as follows:

$$\phi' = [\widehat{\Gamma}_0, \widehat{\Gamma}_1, \cdots, \widehat{\Gamma}_{L-1}].$$

So ϕ is a column vector of length Lm. Then the criterion (4.2) can be rewritten as

$$\mathcal{L}(\phi) = \phi' \, A \, \phi - \phi' \, b - \overline{b'} \, \phi + \langle \Gamma(z) \, h \, \Gamma'(\overline{z}) \rangle,$$

where

$$\overline{b'} = [\langle \Gamma(z) \, h \rangle, \langle \Gamma(z) \, h \overline{z} \rangle, \cdots, \langle \Gamma(z) \, h \overline{z}^{L-1} \rangle]$$

and A is a block matrix consisting of the autocovariances $R(j)$ of the data process. Because this objective function is a quadratic in ϕ, we obtain the minimizer by an analytic formula, namely

$$\phi = A^{-1} \, \Re(b).$$

This follows from $\nabla \mathcal{L}(\phi) = -b - \overline{b} + 2A\phi$.

However, the discussion so far has involved the true spectral density, which in practice is unknown. In that case, one just substitutes some estimate of h, typically either the periodogram or a tapered version of such, or a model-based estimator of h. For example, one could fit a high order Vector Autoregression (VAR) to the data to obtain a plug-in estimator of h. In this chapter we focus upon the periodogram.

In keeping with previous notation, the DFT is $T^{-1/2} \sum_{t=1}^T x_t \exp\{-it\omega\}$, denoted $\Xi_{TX}(\omega)$. Taking the conjugate outer product of this quantity yields the periodogram, a rank one matrix:

$$I_{TX}(\omega) = \Xi_{TX}(\omega) \Xi_{TX}(-\omega)'.$$

It is a simple exercise to show that

$$\widehat{R}_j = T^{-1} \sum_{t=1}^{T-j} x_{t+j} x_t' = \langle I_{TX} \overline{z}^j \rangle$$

for $j \geq 0$. With these estimators, it is possible to plug directly into the same multivariate DFA equations, obtaining the empirical version of (4.2):

$$\langle \left(\Gamma(z) - \widehat{\Gamma}(z) \right) \, I_{TX} \, \left(\Gamma(\overline{z}) - \widehat{\Gamma}(\overline{z}) \right)' \rangle.$$

In the previous example, this has the effect of replacing A with a matrix of sample autocovariances, and updating the entries of the b vector with quantities based on the periodogram. But the final solution has the same form.

The basic treatment of length L concurrent filters can be extended to handle filter constraints. The general problem still involves optimization of $\mathcal{L}(\phi)$, but now ϕ is subject to linear constraints of the form

$$\phi = R \, \varphi + c,$$

where R and c are known quantities. Essentially, we end up optimizing over φ instead. Plugging in, we obtain

$$\begin{aligned}\mathcal{L}(\varphi) &= \varphi'\, R'\, A\, R\, \varphi - \phi'\, R'\, [b - A\, c] - [\overline{b'} - c'\, A]\, R\, \varphi + \langle \Gamma(z)\, h\, \Gamma'(\overline{z}) \rangle \\ &\quad + c'\, A\, c - c'\, b - \overline{b}^{\prime}\, c.\end{aligned}$$

The minimum of this function has the formula

$$\varphi = [R'\, A\, R]^{-1}\, R'\, \left(\Re(b) - A\, c\right).$$

In Chapter 4, the level constraint $i1$ was studied, which in the multivariate context states that

$$\Gamma(1) = \widehat{\Gamma}(1).$$

By writing $\widehat{\Gamma}_0$ in terms of the other free coefficients, we can express the level constraint in terms of the general linear constraint framework. We have $\widehat{\gamma}_0 = \Gamma(1) - \sum_{\ell=1}^{L-1} \widehat{\gamma}_\ell$, so that

$$\begin{aligned}\varphi' &= [\widehat{\gamma}_1, \widehat{\gamma}_2, \cdots, \widehat{\gamma}_{L-1}] \\ c' &= [\Gamma(1)', 0, \cdots]\end{aligned}$$

and the constraint matrix is

$$R = \begin{bmatrix} -1_m & \cdots & -1_m \\ 1_m & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 1_m \end{bmatrix}.$$

Alternatively, the time-shift constraint $i2$ is expressed as $\dot{\Gamma}(1) = \dot{\widehat{\Gamma}}(1)$, which holds if and only if

$$\sum_\ell \ell \gamma_\ell = \sum_{\ell=0}^{L-1} \ell \widehat{\gamma}_\ell.$$

Solving for $\widehat{\gamma}_1$ in terms of the other coefficients gives $\widehat{\gamma}_1 = \dot{\Gamma}(1) - \sum_{\ell=2}^{L-1} \ell \widehat{\gamma}_\ell$, so that

$$\begin{aligned}\varphi' &= [\widehat{\gamma}_0, 0, \widehat{\gamma}_2, \cdots, \widehat{\gamma}_{L-1}] \\ c' &= [0, \dot{\Gamma}(1)^{\prime}, 0, \cdots]\end{aligned}$$

with constraint matrix

$$R = \begin{bmatrix} 1_m & 0 & 0 & \cdots \\ 0 & -21_m & -31_m & \cdots \\ 0 & 1_m & 0 & \cdots \\ 0 & 0 & 1_m & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}.$$

Finally, we might consider imposing both $i1$ and $i2$ constraints, which is equivalent to

$$\begin{aligned}\widehat{\gamma}_1 &= \sum_\ell \ell \gamma_\ell - \sum_{\ell=2}^{L-1} \ell \widehat{\gamma}_\ell \\ \widehat{\gamma}_0 &= \sum_\ell (\ell - 1) \gamma_\ell + \sum_{\ell=2}^{L-1} (\ell - 1) \widehat{\gamma}_\ell.\end{aligned}$$

Then the linear constraint quantities are expressed

$$\begin{aligned} \varphi' &= [\widehat{\gamma}_2, \cdots, \widehat{\gamma}_{L-1}] \\ c' &= [\sum_{\ell} (\ell - 1) \gamma_\ell', \sum_{\ell} \ell \gamma_\ell', 0, \cdots] \end{aligned}$$

with constraint matrix

$$R = \left[\begin{array}{llll} 1_m & 21_m & 31_m & \cdots \\ -21_m & -31_m & -41_m & \cdots \\ 1_m & 0 & \cdots & \cdots \\ 0 & 1_m & 0 & \cdots \\ \vdots & \vdots & \vdots & \\ 0 & 0 & 1_m & \end{array} \right].$$

These are the main cases of interest for stationary multivariate series. The resulting concurrent filters are denoted as moving average MDFA filters.

4.2 Non-stationary Multivariate Processes

We next consider processes that when differenced are stationary, which are the most common type occuring in econometrics and finance. The general treatment of co-integration is complicated when multiple unit roots are present. For example, if there are trend and seasonal roots present, application of a co-integrating vector to the data process may only reduce the order of non-stationarity somewhat, rather than making the series stationary.

We first illustrate this point through dynamic factor component models. Let the differencing polynomial be $\delta(B) = \prod_{\ell=1}^p (1 - e^{i\omega_\ell} B)^{q_\ell}$, where q_ℓ is the multiplicity of each unit root at frequency ω_ℓ. When ω_ℓ is not 0 or π, we know a conjugate factor must appear in $\delta(B)$. By a convenient abuse of notation, we denote the pairing of such conjugate factors by $(1 - e^{i\omega_\ell} B)^{q_\ell}$, with the understanding that q_ℓ is even and denotes the produce of conjugate factors.

Suppose that the data process can be written as the sum of non-stationary latent processes, each of which has differencing polynomial $(1 - e^{i\omega_\ell} B)^{q_\ell}$, plus a residual stationary process. We write this as

$$x_t = \sum_{\ell=1}^p s_t^{(\ell)} + s_t^{(0)}, \tag{4.3}$$

where $(1 - e^{i\omega_\ell} B)^{q_\ell} s_t^{(\ell)}$ is stationary for each $1 \leq \ell \leq p$, and $s_t^{(0)}$ is stationary as well. Let the reduced polynomials $\delta^{(\ell)}(B) = \delta(B) \left(1 - e^{i\omega_\ell} B\right)^{-q_\ell}$ be defined. Then applying $\delta(B)$ to the structural equation (4.3) yields

$$\partial x_t := \delta^{(\ell)}(B) x_t = \sum_{\ell=1}^p \delta^{(\ell)}(B) \partial s_t^{(\ell)} + \delta(B) s_t^{(0)}.$$

Here the ∂ notation before a process refers to the suitably differenced version of that process, which is stationary. Each stationary latent process $\partial s_t^{(\ell)}$ may have singularities in its spectral density matrix, such that it can be represented as $\Lambda^{(\ell)}$ times some $c_t^{(\ell)}$, a stationary process of reduced dimension with spectral density matrix invertible at all frequencies. Such a latent process is governed by a dynamic factor model (DFM), with $\Lambda^{(\ell)} = I_m$ recovering the general case. We actually require $\Lambda^{(0)} = I_m$ in order to guarantee that the spectrum of ∂x_t is non-singular except at a finite number of frequencies.

Suppose that β is a vector such that $\beta' \Lambda^{(k)} = 0$ for some $1 \leq k \leq p$. Then

$$\beta' \, \partial x_t = \sum_{\ell \neq k} \beta' \Lambda^{(\ell)} \, \delta^{(\ell)}(B) c_t^{(\ell)} + \beta' \delta(B) s_t^{(0)},$$

and note that $(1 - e^{i \omega_k} B)^{q_k}$ can be factored out of all terms on the right hand side. Hence $\beta' x_t$ only requires $\delta^{(k)}(B)$ differencing to become stationary; the frequency ω_k co-integrating vector β reduces the order of non-stationarity by the factor $(1 - e^{i \omega_k} B)^{q_k}$. Moreover, if β is in the left null space of several factor loadings $\Lambda^{(\ell)}$, the order of non-stationarity can be reduced further. In an extreme case, $\beta' \Lambda^{(\ell)} = 0$ for $1 \leq \ell \leq p$, so that $\beta' x_t$ is stationary; however, whether or not the factor loadings have a non-trivial intersection of left null space depends on each process.

We now proceed with a general treatment of vector non-stationary processes to explore what types of filter constraints are necessary when co-integrating vectors are present. Crucially supposing that the differencing operator $\delta(B)$ is the same for each component series, we can write

$$x_t = \sum_{j=1}^{d} A_{j,t+d} \, x_{j-d} + \int_{-\pi}^{\pi} \frac{e^{i \omega t} - \sum_{j=1}^{d} A_{j,t+d} \, e^{-i \omega (d-j)}}{\delta(e^{-i \omega})} \, d\mathbb{Z}(\omega),$$

where $d = \sum_{\ell=1}^{m} q_\ell$ and each $A_{j,t+d}$ is a time varying function for each j, and the x_{j-d} are initial values. This representation is chiefly useful when $t > 1$, though it is still valid when $t \leq 0$. The $\mathbb{Z}(\omega)$ is the orthogonal increments process in the spectral representation of ∂x_t.

Each of the time-varying functions is in the null space of $\delta(B)$, i.e., $\delta(B) A_{j,t+d} = 0$ for $1 \leq j \leq d$, where the backshift operator works on the $t + d$ index. As a consequence, we can rewrite each $A_{j,t+d}$ as a linear combination of the basis functions of the null space of $\delta(B)$, which yields a more convenient representation. Let the basis functions be $\phi_j(t)$ for $1 \leq j \leq d$; the existence and form of such functions are a basic staple of difference equation theory, treated briefly in Brockwell and Davis (1991). Then we can write $A_{j,t+d} = \sum_{k=1}^{d} G_{jk} \phi_k(t)$ for each $1 \leq j \leq d$, for some coefficients G_{jk}. It follows that

$$\sum_{j=1}^{d} A_{j,t+d} B^{d-j} = \sum_{k=1}^{d} \phi_k(t) \left(\sum_{j=1}^{d} G_{jk} B^{d-j} \right).$$

Each expression in parentheses on the right hand side is a degree $d - 1$ polynomial in B, and will henceforth be denoted as $p^{(k)}(B)$. Substituting the new formulation, we obtain

$$x_t = \sum_{j=1}^{d} \phi_j(t) p^{(j)}(B) \, x_0 + \int_{-\pi}^{\pi} \frac{e^{i \omega t} - \sum_{j=1}^{d} \phi_j(t) \, p^{(j)}(e^{-i \omega})}{\delta(e^{-i \omega})} \, d\mathbb{Z}(\omega),$$

where $p^{(j)}(B)$ acts on x_0 by shifting the time index $t = 0$ back in time for each power of B. This representation is now extremely convenient, because application of any factor of $\delta(B)$ will annihilate a corresponding basis function (when roots are repeated, some basis functions will also be transformed into others that are instead annihilated).

Suppose that we left multiply by β', which is a co-integrating vector at frequency ω_k:

$$\beta' x_t = \sum_{j=1}^{d} \phi_j(t) p^{(j)}(B)\, \beta'\, x_0 + \int_{-\pi}^{\pi} \frac{e^{i\omega t} - \sum_{j=1}^{d} \phi_j(t)\, p^{(j)}(e^{-i\omega})}{\delta(e^{-i\omega})}\, \beta'\, d\mathbb{Z}(\omega). \tag{4.4}$$

From our previous discussion, we know that the result is a non-stationary process with differencing operator $\delta^{(k)}(B)$; this implies that there should be a cancelation of $\beta'\, d\mathbb{Z}(\omega)$ with the $(1 - e^{i\omega_k} e^{-i\omega})^{q_k}$ term in $\delta(e^{-i\omega})$. As a result, we have the following spectral formalization of the co-integrating relation:

$$\beta'\, d\mathbb{Z}(\omega) = (1 - e^{i\omega_k} e^{-i\omega})^{q_k}\, d\mathbb{Z}^{(k)}(\omega), \tag{4.5}$$

where $d\mathbb{Z}^{(k)}(\omega)$ is the orthogonal increments measure of another stationary invertible process. This condition (4.5) is readily satisfied by the latent dynamic factor process discussed earlier, which is exemplary of the general situation of interest. The extreme case, where the co-integrating vector lies in all the left null spaces of the component processes, allows us to factor $\delta(e^{-i\omega})$ completely from $\beta' d\mathbb{Z}(\omega)$, though such a property need not hold in practice.

In order to see the full effect of condition (4.5) on $\beta' x_t$, we re-organize terms in equation (4.4). Let us suppose, without loss of generality, that frequency ω_k has corresponding basis functions $\phi_1, \cdots, \phi_{q_k}$, so that the first q_k basis functions are annihilated by $(1 - e^{i\omega_k} B)^{q_k}$. Then we can write

$$\beta' x_t = \sum_{j=q_k+1}^{d} \phi_j(t) p^{(j)}(B)\, \beta'\, x_0 + \int_{-\pi}^{\pi} \frac{e^{i\omega t} - \sum_{j=q_k+1}^{d} \phi_j(t)\, p^{(j)}(e^{-i\omega})}{\delta^{(k)}(e^{-i\omega})}\, d\mathbb{Z}^{(k)}(\omega)$$
$$+ \sum_{j=1}^{q_k} \phi_j(t) \left(p^{(j)}(B) \beta'\, x_0 - \int_{-\pi}^{\pi} \frac{p^{(j)}(e^{-i\omega})}{\delta^{(k)}(e^{-i\omega})}\, d\mathbb{Z}^{(k)}(\omega) \right).$$

The first two terms are immediately recognized as the deterministic and stochastic portions respectively of a non-stationary process that has $\delta^{(k)}(B)$ for differencing operator. The third term is left over, and consists of deterministic time series that are in the null space of $(1 - e^{i\omega_k} B)^{q_k}$. To see this, observe that for the third term the expression in parentheses is stochastic, but does not depend on time t, so that the resulting series is predictable.

It is true that $\delta^{(k)}(B)$ always divides $p^{(j)}(B)$, and hence the stochastic portion of the third term is well-defined. We cannot prove that the coefficients of the $\phi_j(t)$ for $1 \leq j \leq q_k$ must be zero, as counter-examples are easy to construct; consider two series that have a common stochastic trend with null vector $\beta' = [1,\, 1]$, but whose underlying linear deterministic trends have different slopes. In our analysis henceforth, we will assume that this third term is identically zero.

4.3 DFA for Co-Integrated Processes

This is the general treatment of co-integration. Now consider the filter error $\epsilon_t = y_t - \widehat{y}_t$. Let $\Delta(z) = \Gamma(z) - \widehat{\Gamma}(z)$, so that

$$\epsilon_t = \sum_{j=1}^{d} \Delta(B) \phi_j(t) p^{(j)}(B)\, x_0 + \int_{-\pi}^{\pi} \frac{e^{i\omega t}\, \Delta(e^{-i\omega}) - \sum_{j=1}^{d} \Delta(B) \phi_j(t)\, p^{(j)}(e^{-i\omega})}{\delta(e^{-i\omega})}\, d\mathbb{Z}(\omega),$$

where $\Delta(B)$ acts only upon the basis functions $\phi_j(t)$. In order to write this expression, we really need the common differencing operators assumption. Note that $\Delta(B)$ is a row vector of filters, and it gets multiplied by the initial value vectors and the orthogonal increments process $d\mathbb{Z}(\omega)$. Clearly the error process is stationary if all the basis functions are annihilated by $\Delta(B)$, because in that case we must be able to factor $\Delta(B) = \tau(B)\delta(B)$ (where $\tau(B)$ is a $1 \times m$ multivariate filter) and $\epsilon_t = \int_{-\pi}^{\pi} e^{i\lambda t}\, \tau(e^{-i\lambda})\, d\mathbb{Z}(\lambda)$. This is the case of full filter constraints, analogous to the stationary case considered above.

We next consider some natural properties of target and concurrent filters. Let us first factor $\delta(B) = \delta^S(B)\delta^N(B)$ according to signal and noise unit roots. We will henceforth suppose that $\delta^S(B) = (1 - e^{i\omega k} B)^{q_k}$ for some unit root $\zeta_k = e^{-i\omega k}$ of multiplicity q_k. Thus $\delta^N(B) = \delta^{(k)}(B)$. Both $\Gamma(B)$ and $\widehat{\Gamma}(B)$ should preserve signal basis functions, which are those $\phi_j(t)$ corresponding to the unit root ζ_k. In order to preserve all these functions (i.e., act as the identity filter on them all) when multiplicity q_k is present, we must have that

$$\frac{\Gamma(z) - \Gamma(\zeta_k)}{(z - \zeta_k)^{q_k}} \qquad \frac{\widehat{\Gamma}(z) - \widehat{\Gamma}(\zeta_k)}{(z - \zeta_k)^{q_k}}$$

are both bounded in z. Equivalently, the differences $\Gamma(z) - \Gamma(\zeta_k)$ and $\widehat{\Gamma}(z) - \widehat{\Gamma}(\zeta_k)$ are each divisible by $\delta^S(z)$. We call this the *signal preservation* property of the filters. For example, the signal extraction filters described in McElroy and Trimbur (2012) always satisfy this sort of condition. In addition, because signal extraction filters must eradicate all basis functions associated with noise frequencies, it follows that $\delta^N(z)$ must be a factor of $\Gamma(z)$ and $\widehat{\Gamma}(z)$. We call this the *noise annihilation* property of the filters. Introduce the notation

$$\Gamma^{N,k}(z) = \Gamma(\zeta_k)\delta^N(z)/\delta^N(\zeta_k) \qquad \widehat{\Gamma}^{N,k} = \widehat{\Gamma}(\zeta_k)\delta^N(z)/\delta^N(\zeta_k).$$

Then we can write

$$\Delta(z) = \left(\frac{\Gamma(z) - \Gamma^{N,k}(z)}{\delta(z)} \right) \delta(z) - \left(\frac{\widehat{\Gamma}(z) - \widehat{\Gamma}^{N,k}(z)}{\delta(z)} \right) \delta(z) + \left(\frac{\Gamma(\zeta_k) - \widehat{\Gamma}(\zeta_k)}{\delta^N(\zeta_k)} \right) \delta^N(z). \quad (4.6)$$

We claim that the first two expressions involve a bounded rational function times $\delta(z)$. To see this true, observe that we only have to check boundedness of $[\Gamma(z) - \Gamma^{N,k}(z)]/\delta(z)$ at z values that are either roots of $\delta^N(B)$ or $\delta^S(B)$ – the same argument applies to the second term involving $\widehat{\Gamma}$. For a signal unit root, we have $z = \zeta_k$, and boundedness follows from the signal preservation property. For a noise unit root, observe that we may always factor $\delta^N(B)$ from $\Gamma(B)$ by the noise annihilation property. As for the third term of (4.6), it is also always well-defined by the noise annihilation property.

It is paramount that $\Delta(B)$ reduce the non-stationary process to stationarity, and this can only happen in two ways: first, a $\delta(B)$ can be factored out, which accomplishes the requirement by differencing. Second, the filter may have a linear combination that is a co-integrating vector (associated with the single signal frequency, which is important!), which together with noise differencing accomplishes the requirement as well. Note that application of a co-integrating vector alone only removes signal non-stationarity, and noise non-stationarity will remain. The above decomposition for $\Delta(B)$ accomplishes this (under the signal preservation and noise annihilation properties) if

$$\frac{\Gamma(\zeta_k) - \widehat{\Gamma}(\zeta_k)}{\delta^N(\zeta_k)} = \beta' \tag{4.7}$$

for some vector β to be described. If we impose that β is the zero vector, then we obtain the first case above, where $\Delta(B)$ maintains stationarity by full differencing. If instead we relax this to only imposing that $\beta = \beta_k$ be a co-integrating vector for the signal, then we obtain the second case above. Because there may be many choices for β_k, depending on the co-integrating rank (the dimension of the left null space of the factor loading matrix in the latent dynamic factors formulation), this is a milder condition that may allow for more flexibility in filter estimation. Note that since $\Gamma(\zeta_k)/\delta^N(\zeta_k)$ is a given quantity, imposing (4.7) amounts to setting $\widehat{\Gamma}(\zeta_k)/\delta^N(\zeta_k) = \beta' + \Gamma(\zeta_k)/\delta^N(\zeta_k)$ for a known co-integrating β.

We next develop the consequences of (4.7), where β is either zero or a signal co-integrating vector (the second case will reduce to the first when we set $\beta = 0$ in the following formulas). Let $c_t = \delta^N(B)\beta' x_t$, which by our prior expression for $\beta' x_t$ is shown to be equal to $\int_{-\pi}^{\pi} e^{i\omega t}\, d\mathbb{Z}^{(k)}(\omega)$. The signal extraction error is

$$\epsilon_t = \int_{-\pi}^{\pi} e^{i\omega t} \left(\frac{\Gamma(z) - \Gamma^{N,k}(z)}{\delta(z)} \right) d\mathbb{Z}(\omega) - \int_{-\pi}^{\pi} e^{i\omega t} \left(\frac{\widehat{\Gamma}(z) - \widehat{\Gamma}^{N,k}(z)}{\delta(z)} \right) d\mathbb{Z}(\omega) + \int_{-\pi}^{\pi} e^{i\omega t}\, d\mathbb{Z}^{(k)}(\omega).$$

Its variance involves a spectral density matrix that combines information from the differenced series ∂s_t as well as the noise-differenced co-integrated series c_t. We now suppose that the joint spectral density matrix of these series is available to us, which is certainly possible in the case of latent dynamic factor models. Letting h_c, $h_{c\partial x}$, and $h_{\partial x}$ denote the spectra and cross-spectra, we have the joint spectra for $[c_t, \partial x_t']'$ is

$$h(\omega) = \left[\begin{array}{cc} h_c(\omega) & h_{\partial x c}(\omega) \\ h_{c\partial x}(\omega) & h_{\partial x}(\omega) \end{array} \right].$$

Then the signal extraction variance is $(2\pi)^{-1}$ times the integral of

$$\left[1, - \left(\frac{\Gamma(z) - \Gamma^{N,k}(z)}{\delta(z)} \right) + \left(\frac{\widehat{\Gamma}(z) - \widehat{\Gamma}^{N,k}(z)}{\delta(z)} \right) \right] h(\omega) \left[1, - \left(\frac{\Gamma(\overline{z}) - \overline{\Gamma^{N,k}}(\overline{z})}{\delta(\overline{z})} \right) + \left(\frac{\widehat{\Gamma}(\overline{z}) - \overline{\widehat{\Gamma}^{N,k}}(\overline{z})}{\delta(\overline{z})} \right) \right]'.$$

Substituting the periodogram for h at this point will allow empirical estimation, where we impose the co-integrating relations on $\widehat{\Gamma}$ and then optimize.

In the case that β is the zero vector, we are merely imposing full filter constraints by (4.7), and the multivariate DFA (M-DFA) condition can be simplified. Introduce the notation

$$\Gamma^{\delta,k}(z) = \frac{\Gamma(z) - \Gamma^{N,k}(z)}{\delta(z)} \qquad \widehat{\Gamma}^{\delta,k}(z) = \frac{\widehat{\Gamma}(z) - \widehat{\Gamma}^{N,k}(z)}{\delta(z)}.$$

Although the former quantity can be computed from a knowledge of the target filter and the goals of analysis, the latter quantity is obliquely related to the parameters of the proposed concurrent filter. In terms of these quantities, the M-DFA MSE is

$$\langle \left[\Gamma^{\delta,k}(z) - \widehat{\Gamma}^{\delta,k}(z) \right] \; h_{\partial x} \; \left[\Gamma^{\delta,k}(\overline{z}) - \widehat{\Gamma}^{\delta,k}(\overline{z}) \right]' \rangle.$$

The filter conditions can be rephrased in terms of coefficient constraints, as in the stationary case.

4.4 Examples of Multivariate DFA

4.5 Examples of Co-Integrated Multivariate DFA

Chapter 5

Filter Customization

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 **??**; section **??** highlights the classic dichotomic forecast paradigm; quadratic optimization criteria and closed-form solutions are presented in section **??**; an application of customization is proposed in section **??**; performance measures are presented in section **??**; finally, sections **??** and **??** assess performances of customized designs when benchmarked against classic MSE-approaches.

[1] Backcasts were discussed in chapter 7.

Chapter 6

Overfitting and Regularization

6.1 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 6.2 we contrast overfitting from a classic (one-step ahead forecasting) and from a general signal extraction perspective; section 6.3 briefly reviews methods for tackling overfitting and introduces the so-called Regularization Troika; sections 6.4 to 6.6 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 6.7; section 6.8 derives closed form solutions of the regularized MDFA-estimate and section 6.9 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.

6.2 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.

6.2.1 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

$$y_t = \sum_{k=-\infty}^{\infty} \gamma_k x_{t-k} \tag{6.1}$$

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 6.1 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.

6.2.2 Empirical Examples

To be filled...

6.3 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.

6.3.1 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 3 ($i1 = i2 = T$) 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.

6.3.2 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 **??**, 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.

6.3.3 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

$$MDFA + f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}^m_{decay}\mathbf{b} + f(\lambda_{smooth})\mathbf{b}'\mathbf{\Lambda}^m_{smooth}\mathbf{b} + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}^m_{cross}\mathbf{b} \to \min_{\mathbf{b}}$$

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.

6.4 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.

6.4.1 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

$$DFA + \lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}^0_{decay}\mathbf{b} \to \min_{\mathbf{b}}$$

where *DFA* stands for any of the previous DFA-criteria (MSE or customized, with or without constraints) and where $\mathbf{\Lambda}_{decay}^0$ is a diagonal matrix of dimension $L*L$ with diagonal-elements $(1+\lambda_{decay,1})^k$, $k=0,...,L-1$

$$\mathbf{\Lambda}_{decay}^0=\left(\begin{array}{ccccc}1 & 0 & 0 & ... & 0\\ 0 & 1+\lambda_{decay,1} & 0 & ... & 0\\ 0 & 0 & (1+\lambda_{decay,1})^2 & ... & 0\\ 0 & 0 & 0 & ... & (1+\lambda_{decay,1})^{L-1}\end{array}\right)$$

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

$$\mathbf{b}'\mathbf{\Lambda}_{decay}^0\mathbf{b}=\sum_{k=0}^{L-1}(1+\lambda_{decay,1})^k b_k^2$$

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

$$DFA+f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b}\to\min_{\mathbf{b}} \tag{6.2}$$

such that $f(0)=0$ and $f(1)=\infty$. If $\lambda_{decay,2}=0$ then 6.2 reduces to the original DFA-criterion; if $\lambda_{decay,2}\to 1$ 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 $\lambda_{decay,2}=1$.

6.4.2 Backcasting and Forecasting

For a backcast $\hat{y}_{T+h}^h:=\sum_{k=h}^{L-1+h}b_{kh}x_{T+h-k}$ 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 $\mathbf{\Lambda}_{decay}^0$ by

$$\left(\begin{array}{ccccc}(1+\lambda_{decay,1})^{-h} & 0 & 0 & ... & 0\\ 0 & (1+\lambda_{decay,1})^{-h-1} & 0 & ... & 0\\ 0 & 0 & (1+\lambda_{decay,1})^{-h-2} & ... & 0\\ ::: & & & & \\ 0 & 0 & 0 & ... & (1+\lambda_{decay,1})^{L-1+h}\end{array}\right)$$

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

$$\mathbf{\Lambda}_{decay}^{0,h}=\left(\begin{array}{cccc}(1+\lambda_{decay,1})^{\max(0,-h)} & 0 & 0\ ... & 0\\ 0 & (1+\lambda_{decay,1})^{\max(0,-h)-1} & 0\ ... & 0\\ ::: & & & \\ 0 & 0 & 0\ ... & (1+\lambda_{decay,1})^{L-1-\max(0,-h)}\end{array}\right)\qquad(6.3)$$

The maximum $\max(0,-h)$ in the exponent ensures that forecasting and nowcasting are handled equally.

6.4.3 Multivariate Framework

The extension to a multivariate framework, with explaining series $x_t,\ w_{1t},...,w_{mt},\ m>0$, is straightforward:

$$MDFA+\lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}^{m}\mathbf{b}\to\min_{\mathbf{b}}$$

where $\mathbf{\Lambda}_{decay}^{m}$ is an $L(m+1)*L(m+1)$-dimensional diagonal matrix

$$\mathbf{\Lambda}_{decay}^{m}=\left(\begin{array}{cccc}\mathbf{\Lambda}_{decay}^{0} & 0 & ... & 0\\ 0 & \mathbf{\Lambda}_{decay}^{0} & ... & 0\\ ::: & & & \\ 0 & 0 & ... & \mathbf{\Lambda}_{decay}^{0}\end{array}\right)$$

where $\mathbf{\Lambda}_{decay}^{0}$ from the univariate case is replicated $m+1$ times along the diagonal of $\mathbf{\Lambda}_{decay}^{m}$ 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 $\mathbf{\Lambda}_{decay}^{0}$ is used throughout. Otherwise, the more general expression 6.3 could be substituted.

6.4.4 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.6.1.

```
RStudioGD
        2
```

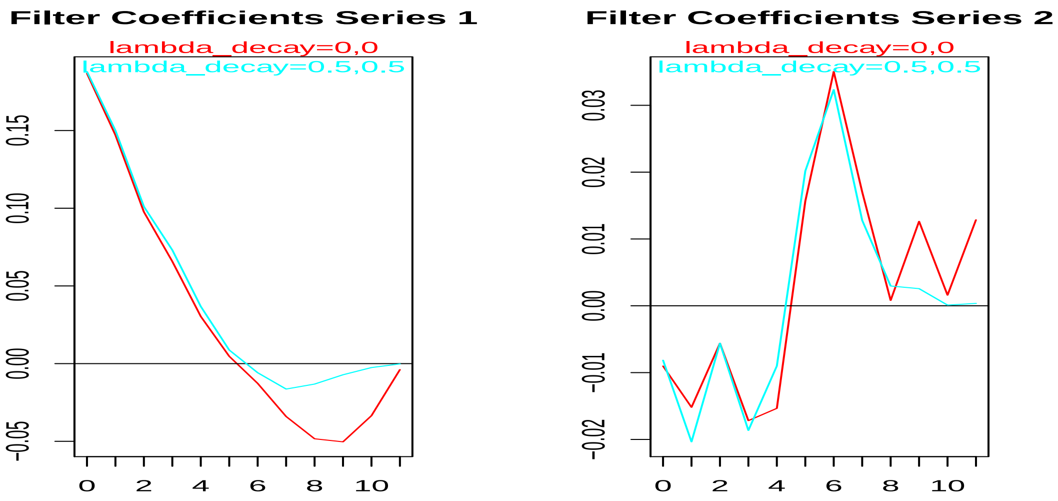

Figure 6.1: Filter Coefficients: original unregularized (red) vs. decay-regularization (cyan) for series 1 (left) and 2 (right) of the bivariate design, lambda-decay=(0.5,0.5)

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).

6.4.5 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.6.2: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
RStudioGD
        2
```

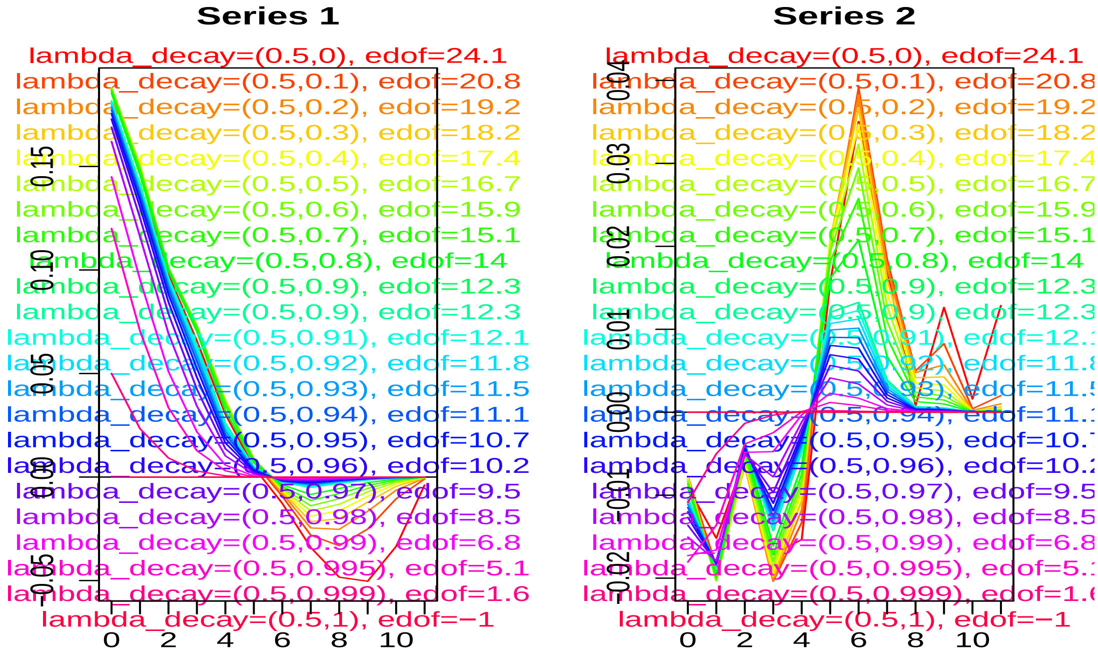

Figure 6.2: Filter Coefficients: effect of shape-parameter (grid of points -0.1, 0, 0.1, 0.2, ..., 1.1) for fixed strength parameter (0.5), series 1 (left) and 2 (right)

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.6.3: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
RStudioGD
        2
```

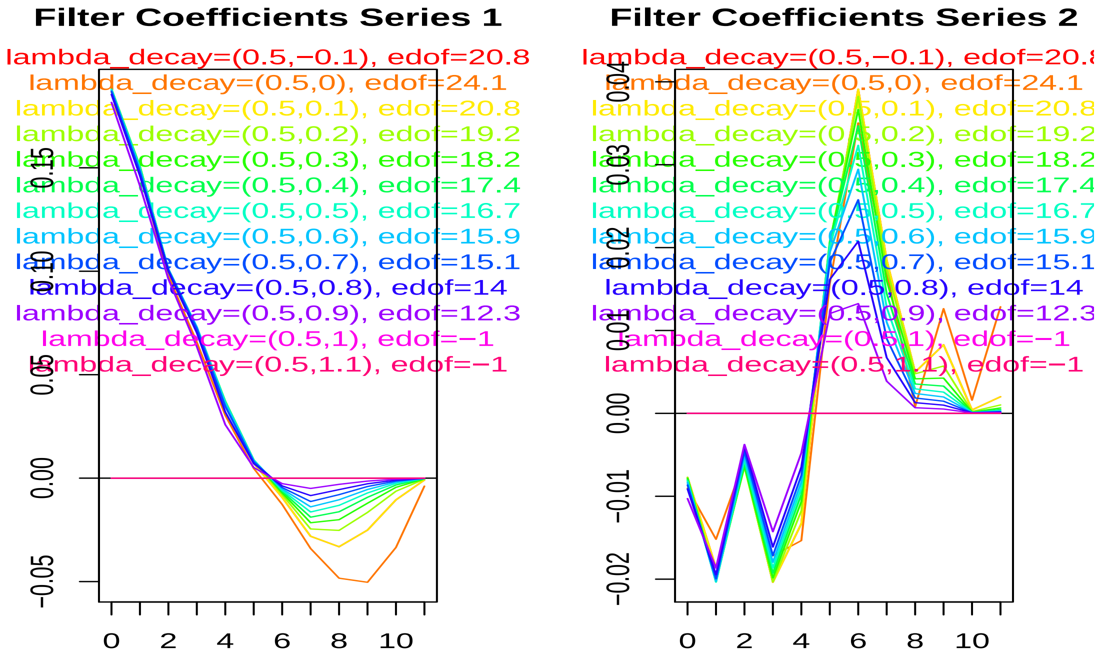

Figure 6.3: Filter Coefficients: effect of strength-parameter (grid of points -0.1,0, 0.1, 0.2, ..., 1.1) for fixed shape parameter (0.5), series 1 (left) and 2 (right)

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).

6.4.6 Constraints vs. Regularization

Hard-constraints, such as imposed by $i1$ or $i2$, see chapter 3, 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.6.4: the effective degrees of freedom, $edof$, as well as the regularization settings are reported too.

```
RStudioGD
        2
```

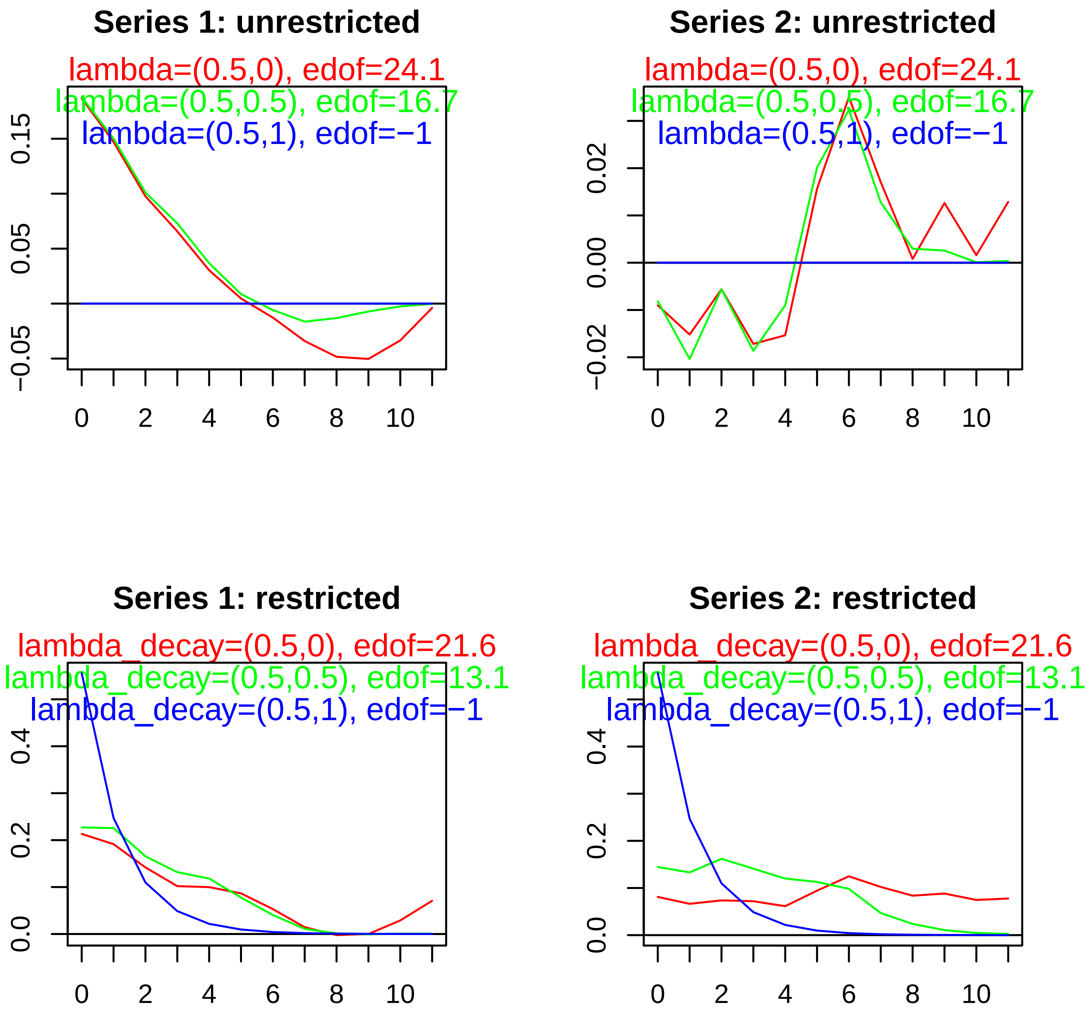

Figure 6.4: Filter Coefficients: effect of constraints and strength-parameter (0, 0.5, 1) for fixed shape parameter (0.5), series 1 (left) and 2 (right); unconstrained design (top) vs. constrained design (bottom)

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 ??.

6.5 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)

$$\sum_{k=0}^{L-3}\left((1-B)^2b_{ik}\right)^2=\sum_{k=0}^{L-3}\left(b_{ik}-2b_{i,k+1}+b_{i,k+2}\right)^2 \tag{6.4}$$

are large.

6.5.1 Quadratic Bilinear Form

We may thus introduce a penalty-term whose bilinear form replicates 6.4. Specifically, the regularized criterion becomes

$$MDFA+f(\lambda_{smooth})\mathbf{b}'\mathbf{\Lambda}_{smooth}^m\mathbf{b}\to\min_{\mathbf{b}} \tag{6.5}$$

where $\mathbf{\Lambda}_{smooth}^m$ is a block-diagonal matrix of dimension $L(m+1)*L(m+1)$

$$\mathbf{\Lambda}_{smooth}^m=\left(\begin{array}{cccc}\mathbf{\Lambda}_{smooth}^0 & 0 & ... & 0\\ 0 & \mathbf{\Lambda}_{smooth}^0 & ... & 0\\ ::: & & & \\ 0 & 0 & ... & \mathbf{\Lambda}_{smooth}^0\end{array}\right)$$

with $L*L$-dimensional blocks $\mathbf{\Lambda}_{smooth}^0$

$$\mathbf{\Lambda}_{smooth}^0=\left(\begin{array}{cccccccccccccc}1&-2&1&0&0&0&0&....&&&&&&0\\ -2&5&-4&1&0&0&0&....&&&&&&0\\ 1&-4&6&-4&1&0&0&....&&&&&&0\\ 0&1&-4&6&-4&1&0&....&&&&&&0\\ ::&&&&&&&&&&&&&\\ 0&0&0&0&0&0&...&1&-4&6&-4&1&0&\\ 0&0&0&0&0&0&0&...&1&-4&6&-4&1&\\ 0&0&0&0&0&0&0&...&0&1&-4&5&-2&\\ 0&0&0&0&0&0&0&...&0&0&1&-2&1&\end{array}\right)$$

[6] The case $i=0$ refers to the explaining data of x_t.

The latter replicate the curvature measure 6.4

$$\sum_{k=0}^{L-3}\left((1-B)^2 b_{ik}\right)^2=\mathbf{b}_i'\mathbf{\Lambda}_{smooth}^0\mathbf{b}_i$$

for $i=0,...,m$[7]. In the univariate case we have $m=0$ and therefore $\mathbf{\Lambda}_{smooth}^m=\mathbf{\Lambda}_{smooth}^0$. 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.

6.5.2 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.6.5: the effective degrees of freedom, $edof$, as well as the accompanying regularization settings are reported too.

```
RStudioGD 
        2 
```

[7] Checking this identity is graciously left as an exercise.

[8] Two parameters, namely intercept and slope, determine a linear function.

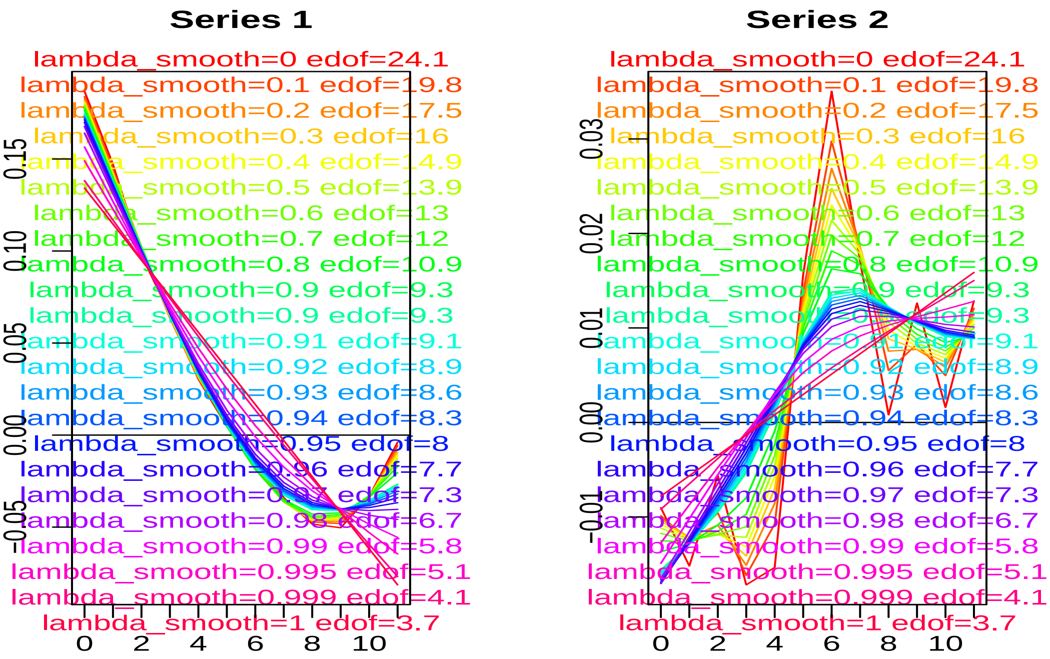

Figure 6.5: Filter Coefficients: effect of smoothness regularization for grid-points -0.1, 0, 0.1, 0.2, ..., 1.1: series 1 (left) and 2 (right)

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.

6.6 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.

6.6.1 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

$$\sum_{k=0}^{L-1}\sum_{i=0}^{m}\left(b_{ik}-\frac{1}{m+1}\sum_{i'=0}^{m} b_{i'k}\right)^2 \tag{6.6}$$

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

$$\mathbf{\Lambda}_{cross}^{m} = \mathbf{I} - \boldsymbol{M} \tag{6.7}$$

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

$$MDFA + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}_{cross}^{m}\mathbf{b} \to \min_{\mathbf{b}} \tag{6.8}$$

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.

6.6.2 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 **??**. 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.6.6: the effective degrees of freedom, *edof*, as well as the accompanying regularization settings are reported too.

```
RStudioGD
        2
```

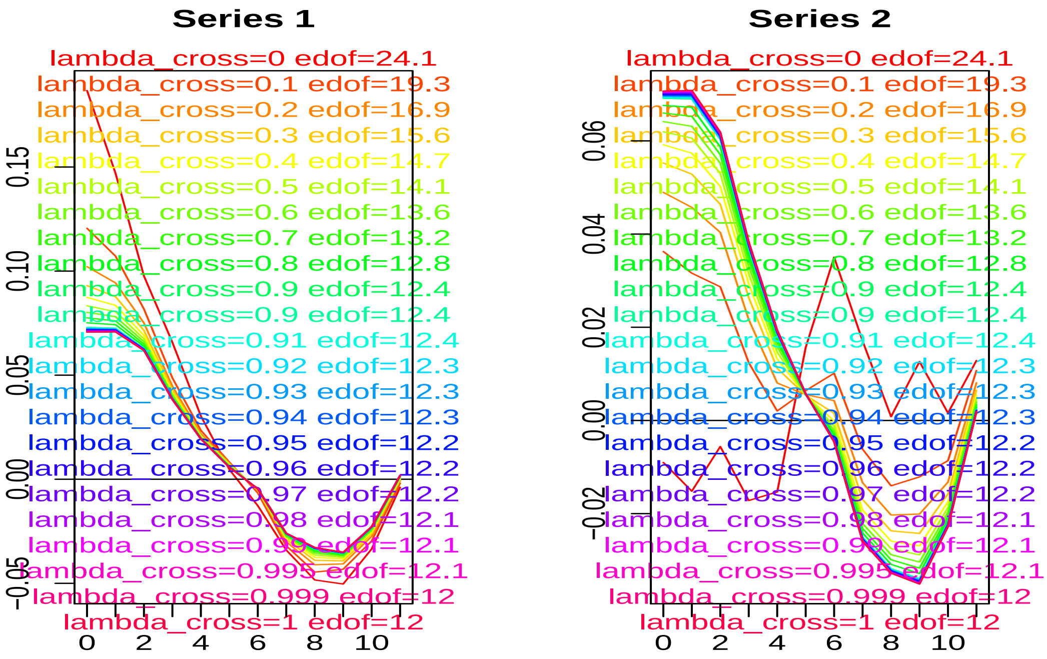

Figure 6.6: Filter Coefficients: effect of cross-sectional regularization for grid-points -0.1, 0, 0.1, 0.2, ..., 1.1: series 1 (left) and 2 (right)

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].

6.7 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.

6.7.1 Quadratic (Bilinear) Form

An arbitrary combination and weighting of all three requirements can be obtained formally by the so-called Regularization Troika

$$MDFA + f(\lambda_{decay,2})\mathbf{b}'\mathbf{\Lambda}^m_{decay}\mathbf{b} + f(\lambda_{smooth})\mathbf{b}'\mathbf{\Lambda}^m_{smooth}\mathbf{b} + f(\lambda_{cross})\mathbf{b}'\mathbf{\Lambda}^m_{cross}\mathbf{b} \to \min_{\mathbf{b}} \tag{6.9}$$

If $\lambda_{decay,2} = \lambda_{smooth} = \lambda_{cross} = 0$ then 6.9 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.

6.7.2 Empirical Examples

To be filled...

6.7.3 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 6.11. Finally, let us remind here that regularization sub-spaces can be affected by imposing hard-constraints, see the example in section 6.4.6.

[11] The common coefficients obtained for $\lambda_{cross} \geq 1$ in fig.6.6 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.

6.7.4 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].

6.7.5 Empirical Examples

To be filled...

6.8 Optimization Criterion (Zero-Shrinkage)

We note that 6.9 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).

6.8.1 Unconstrained Design

Consider

$$\begin{aligned} & (\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b})'(\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}) \\ + \quad & \lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}\mathbf{b} + \lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}\mathbf{b} + \lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b} \to \min \end{aligned} \tag{6.10}$$

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 **??**: 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

$$\begin{aligned}\frac{d}{d\mathbf{b}}\text{ Criterion} &= \frac{d}{d\mathbf{b}}\left(\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}\right)'\left(\mathbf{Y}^{\text{Cust}}(\eta)-\mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}\right)\\ &+ \frac{d}{d\mathbf{b}}\left(\lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}\mathbf{b}+\lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}\mathbf{b}+\lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\mathbf{b}\right)\\ &= -2\mathbf{Y}^{\text{Cust}}(\eta)'\Re\left(\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right)+2\mathbf{b}'\Re\left\{\mathbf{X}^{\text{Cust}}(\lambda,\eta)'\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right\}\\ &\quad +2\lambda_{smooth}\mathbf{b}'\mathbf{\Lambda}_{smooth}+2\lambda_{cross}\mathbf{b}'\mathbf{\Lambda}_{cross}+2\lambda_{decay,2}\mathbf{b}'\mathbf{\Lambda}_{decay}\end{aligned}$$

where we used the fact that the proposed bilinear forms are symmetric. Equating this expression to zero, the generalized regularized solution is obtained as

$$\begin{aligned}&\hat{\mathbf{b}}^{\text{Cust-Reg}}(\lambda,\eta,\lambda_{smooth},\lambda_{cross},\lambda_{decay,1},\lambda_{decay,2}) = \\ &\left(\Re\left\{\mathbf{X}^{\text{Cust}}(\lambda,\eta)'\mathbf{X}^{\text{Cust}}(\lambda,\eta)\right\}+\lambda_{smooth}\mathbf{\Lambda}_{smooth}+\lambda_{cross}\mathbf{\Lambda}_{cross}+\lambda_{decay,2}\mathbf{\Lambda}_{decay}\right)^{-1}\\ &\Re(\mathbf{X}^{\text{Cust}}(\lambda,\eta))'\mathbf{Y}^{\text{Cust}}(\eta)\end{aligned} \qquad (6.11)$$

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}.

6.8.2 Constrained Design

We assume that the filter constraints can be written in the form

$$\mathbf{b} \;=\; \mathbf{R}(i1,i2)\mathbf{b_f}+\mathbf{c}(i1,i2) \qquad (6.12)$$

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 **??** (alternative constraints could be considered, of course). Plugging this expression into 6.10 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):

$$\begin{aligned}
\frac{d}{d\mathbf{b_f}}\text{ Criterion} &= \frac{d}{d\mathbf{b_f}}\left(\mathbf{Y}^{\text{Cust}}-\mathbf{X}^{\text{Cust}}\left(\mathbf{Rb_f}+\mathbf{c}\right)\right)'\left(\mathbf{Y}^{\text{Cust}}-\mathbf{X}^{\text{Cust}}\left(\mathbf{Rb_f}+\mathbf{c}\right)\right)\\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{smooth}\left(\mathbf{Rb_f}+\mathbf{c}\right)'\mathbf{\Lambda}_{smooth}\left(\mathbf{Rb_f}+\mathbf{c}\right)\\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{cross}\left(\mathbf{Rb_f}+\mathbf{c}\right)'\mathbf{\Lambda}_{cross}\left(\mathbf{Rb_f}+\mathbf{c}\right)\\
&+ \frac{d}{d\mathbf{b_f}}\lambda_{decay,2}\left(\mathbf{Rb_f}+\mathbf{c}\right)'\mathbf{\Lambda}_{decay}\left(\mathbf{Rb_f}+\mathbf{c}\right)\\
&= -2(\mathbf{Y}^{\text{Cust}})'\Re\left(\mathbf{X}^{\text{Cust}}\right)\mathbf{R}+2\Re\left\{(\mathbf{X}^{\text{Cust}}\mathbf{c})'(\mathbf{X}^{\text{Cust}}\mathbf{R})\right\}+2\mathbf{b}_f'\Re\left\{(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{R}\right\}\\
&\quad +2\lambda_{smooth}\mathbf{b_f}'\mathbf{R}'\mathbf{\Lambda}_{smooth}\mathbf{R}+2\lambda_{smooth}(\mathbf{c}'\mathbf{\Lambda}_{smooth}\mathbf{R})\\
&\quad +2\lambda_{cross}\mathbf{b_f}'\mathbf{R}'\mathbf{\Lambda}_{cross}\mathbf{R}+2\lambda_{cross}(\mathbf{c}'\mathbf{\Lambda}_{cross}\mathbf{R})\\
&\quad +2\lambda_{decay,2}\mathbf{b_f}'\mathbf{R}'\mathbf{\Lambda}_{decay}\mathbf{R}+2\lambda_{decay,2}(\mathbf{c}'\mathbf{\Lambda}_{\text{decay}}\mathbf{R})
\end{aligned}$$

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

$$\begin{aligned}
&\hat{\mathbf{b}}_f^{\text{Cust-Reg-Const}}(\lambda,\eta,\lambda_{smooth},\lambda_{cross},\lambda_{decay,1},\lambda_{decay,2},i1,i2)\\
=\ &\left\{\Re\left[(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{R}\right]+\lambda_{smooth}\mathbf{R}'\mathbf{\Lambda}_{smooth}\mathbf{R}+\lambda_{cross}\mathbf{R}'\mathbf{\Lambda}_{cross}\mathbf{R}+\lambda_{decay,2}\mathbf{R}'\mathbf{\Lambda}_{decay}\mathbf{R}\right\}^{-1}\\
&\left((\Re(\mathbf{X}^{\text{Cust}})\mathbf{R})'\mathbf{Y}^{\text{Cust}}-\Re\left\{(\mathbf{X}^{\text{Cust}}\mathbf{R})'\mathbf{X}^{\text{Cust}}\mathbf{c}\right\}-\mathbf{R}'(\lambda_{smooth}\mathbf{\Lambda}_{\text{smooth}}+\lambda_{cross}\mathbf{\Lambda}_{cross}+\lambda_{decay,2}\mathbf{\Lambda}_{cross})\mathbf{c}\right)\\
=\ &\left\{\Re\left[\mathbf{R}'(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\mathbf{R}\right]+\lambda_{smooth}\mathbf{R}'\mathbf{\Lambda}_{smooth}\mathbf{R}+\lambda_{cross}\mathbf{R}'\mathbf{\Lambda}_{cross}\mathbf{R}+\lambda_{decay,2}\mathbf{R}'\mathbf{\Lambda}_{decay}\mathbf{R}\right\}^{-1}\\
&\left((\Re(\mathbf{X}^{\text{Cust}})\mathbf{R})'\mathbf{Y}^{\text{Cust}}+\mathbf{Const}\right) \qquad (6.13)
\end{aligned}$$

where

$$\mathbf{Const}=-\mathbf{R}'\left\{\Re\left[(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\right]+\lambda_{smooth}\mathbf{\Lambda}_{smooth}+\lambda_{cross}\mathbf{\Lambda}_{cross}+\lambda_{decay,2}\mathbf{\Lambda}_{decay}\right\}\mathbf{c}$$

This last term collects all level constraints[14]. A comparison with 6.11 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 6.11. 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 6.12.

6.8.3 Empirical Example: Constrained Regularized Customized Design

To be filled...

[14] Indeed, $\mathbf{c}=\mathbf{0}$ if $i1=F$.

6.9 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.

6.9.1 Hat-Matrix and Residual-Matrix

For simplicity of exposition we here treat the unconstrained case as obtained by criterion 6.10. The so-called *hat-matrix* $\mathbf{H}$ is defined by

$$\mathbf{H} \quad = \quad \mathbf{X}^{\text{Cust}}\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}+\lambda_{smooth}\mathbf{\Lambda}_{smooth}+\lambda_{cross}\mathbf{\Lambda}_{cross}+\lambda_{decay,2}\mathbf{\Lambda}_{decay}\right)^{-1}\Re(\mathbf{X}^{\text{Cust}})' \tag{6.14}$$

Thus

$$\hat{\mathbf{Y}}^{\text{Cust}} := \mathbf{X}^{\text{Cust}}\hat{\mathbf{b}} = \mathbf{H}\mathbf{Y}^{\text{Cust}}$$

i.e. $\mathbf{H}$ maps the dependent variable $\mathbf{Y}$ onto $\hat{\mathbf{Y}}^{\text{Cust}}$. In the absence of regularization, the expression for $\mathbf{H}$ simplifies to

$$\mathbf{H} \quad = \quad \mathbf{X}^{\text{Cust}}\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}\Re(\mathbf{X}^{\text{Cust}})'$$

Note that $\hat{\mathbf{Y}}^{\text{Cust}}$ lies in a particular subspace of the complex plane spanned by the data $\mathbf{X}^{\text{Cust}}$, 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

$$\begin{aligned}
\mathbf{H}'\mathbf{H} &= \overline{\mathbf{H}}^{\mathbf{T}}\mathbf{H}\\
&= \Re(\mathbf{X}^{\text{Cust}})\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}\Re(\mathbf{X}^{\text{Cust}})\\
&\neq \mathbf{H}
\end{aligned}$$

Interestingly, though, an idempotency is obtained when focusing on the real-parts only:

$$\begin{aligned}
\Re\left(\mathbf{H}'\mathbf{H}\right) &= \Re(\mathbf{X}^{\text{Cust}})\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}\Re\Big((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\Big)\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}\Re(\mathbf{X}^{\text{Cust}})\\
&= \Re(\mathbf{X}^{\text{Cust}})\left(\Re\big\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\big\}\right)^{-1}\Re(\mathbf{X}^{\text{Cust}})\\
&= \Re(\mathbf{H})
\end{aligned}$$

[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

$$\mathbf{H} = \mathbf{X}^{\text{Cust}}\Big((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\Big)^{-1}(\mathbf{X}^{\text{Cust}})'$$

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.

6.9.2 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

$$\begin{aligned}&(\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)i\mathbf{b})'(\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)i\mathbf{b})\\ =\ &\left|\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)i\mathbf{b}\right|^2\\ =\ &\left|-i\mathbf{Y}^{\text{Cust}}(\eta) - \mathbf{X}^{\text{Cust}}(\lambda,\eta)\mathbf{b}\right|^2\end{aligned}$$

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 $\mathbf{Y}^{\text{Cust}}(\eta)$ to $-i\mathbf{Y}^{\text{Cust}}(\eta)$. 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

$$\begin{aligned}d/d\mathbf{b}\text{ Adjoined Criterion} &= d/d\mathbf{b}\ (-i\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\mathbf{b})'(-i\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\mathbf{b})\\ &= -(-i\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\mathbf{b})'\mathbf{X}^{\text{Cust}} - (-i\mathbf{Y}^{\text{Cust}} - \mathbf{X}^{\text{Cust}}\mathbf{b})^{\mathbf{T}}\overline{\mathbf{X}^{\text{Cust}}}\\ &= 2\mathbf{Y}^{\text{Cust}'}\Im\left(\mathbf{X}^{\text{Cust}}\right) + 2\mathbf{b}'\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\end{aligned}$$

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

$$\begin{array}{rcl}\hat{\mathbf{b}}^{ad} & = & -i\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{\mathbf{-1}}\Im\left(\mathbf{X}^{\text{Cust}}\right)'\mathbf{Y}^{\text{Cust}}\\ & = & i\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{\mathbf{-1}}\Im\left((\mathbf{X}^{\text{Cust}})'\right)\mathbf{Y}^{\text{Cust}}\end{array}$$

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

$$\begin{array}{rcl}\mathbf{H}^{aug} & = & \mathbf{H}+\mathbf{H}^{ad}\\ & = & \mathbf{X}^{\text{Cust}}\big(\Re((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}})\big)^{\mathbf{-1}}\left(\mathbf{X}^{\text{Cust}}\right)'\end{array}$$

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

$$\begin{array}{rcl}tr(\mathbf{H}^{aug}) & = & tr\left(\mathbf{X}^{\text{Cust}}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}(\mathbf{X}^{\text{Cust}})'\right)\\ & = & tr\left((\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}\right)\\ & = & tr\left(\Re\left\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\right\}\left(\Re\{(\mathbf{X}^{\text{Cust}})'\mathbf{X}^{\text{Cust}}\}\right)^{-1}\right)\\ & = & tr(\mathbf{I}_{(m+1)L})\\ & = & (m+1)L\end{array}$$

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 6.13 can be tackled analogously and does not deserve a separate treatment here.

6.9.3 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.

6.10 The Troikaner

To be filled...

6.10.1 Generalized Information Criterion

6.11 General H0-Shrinkage

To be filled...

6.11.1 Zero-Shrinkage vs. Regularization: Potentially Conflicting Requirements

6.11.2 Inclusion of A Priori Knowledge

6.11.3 Replicating (and Enhancing) Clients' Performances

6.12 Optimization Criterion under General H0-Shrinkage

To be filled...

6.13 MDFA-Stages

To be filled...

- Numerically (very) fast
- Statistically efficient
- Immunized against Overfitting
- Highly Adaptive

6.14 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

6.15 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 7

Vintage Data: Working with Data-Revisions

refer to 2011-paper

7.1 Introduction

7.2 Data Organization

7.2.1 Vintage and Release Triangles

7.2.2 Vintage Triangle in the Frequency-Domain

7.3 Reconcile Real-Time Signalextraction and Data-Revisions

7.3.1 Vintage-Filtering/Smoothing

7.3.2 Setting-Up MDFA-Designs: the (Pseudo-) Stationary Case

7.3.3 Extension to Non-Stationary Time Series

Refer and link to section **??**

Chapter 8

Mixed-Frequency Data: Combining and Working with Data Sampled at Different Time Scales

8.1 Introduction

- Refer to section ?? for a quantitative analysis of up-dating effects.

- Refer to recent work with Chris

8.2 Target is Low-Frequency Data

8.2.1 Folding the Frequency Interval

8.2.2 Generalized Optimization Criterion

8.3 Explaining Series is/are Low-Frequency Data

8.3.1 Folding and Expanding the Frequency Interval

8.3.2 Generalized Optimization Criterion

8.4 General Case: Arbitrary Mix

8.5 Unequally Distributed Release Dates

8.6 Missing Data

Lomb-Scargle, DFT with leads/lags: analyze orthogonality.

Chapter 9

Summary and Links

9.1 Survey of MDFA Optimization Criteria

9.2 Consistency and Efficiency: a Tale of Two Philosophies

9.2.1 Knowing the Truth: Omniscience

9.2.2 Believing in Truth: Faith and Fatalism

9.2.3 From Truth to Effectiveness: Emphasizing Performances

Bibliography

[1] Lutkepohl 2007

[2] Brockwell and Davis 1991

[3] Baxter and King 1999

[4] Self and Liang 1987

[5] Hosoya and Taniguchi 1982

[6] Taniguchi and Kakizawa 2000

[7] McElroy and Trimbur 2015

[8] McElroy 2008

[9] Roy McElroy Linton 2019

[10] McElroy and Findley 2015

[11] Wildi 2008