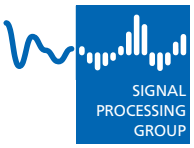
Bootstrapping Spectra

Bootstrappping-Verfahren angewandt auf die spektrale Leistungsdichte
Projektseminar von Momchil Nikolov
Tag der Einreichung: 11. Januar 2019

1. Gutachten: Dr.-Ing. Michael Fauss
2. Gutachten: Prof. Dr.-Ing. A.M. Zoubir

TECHNISCHE
UNIVERSITÄT
DARMSTADT



Bootstrapping Spectra
Bootstrappping-Verfahren angewandt auf die spektrale Leistungsdichte

Vorgelegtes Projektseminar von Momchil Nikolov

1. Gutachten: Dr.-Ing. Michael Fauss
2. Gutachten: Prof. Dr.-Ing. A.M. Zoubir

Tag der Einreichung: 11. Januar 2019

Declaration

To the best of my knowledge and belief this work was prepared without aid from any other sources except where indicated. Any reference to material previously published by any other person has been duly acknowledged. This work contains no material which has been submitted or accepted for the award of any other degree in any institution.

Hiermit versichere ich die vorliegende Arbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 11. Januar 2019

(Momchil Nikolov)

Abstract

In Chapter 1 we presents the seminar work. Chapter 2 introduces the reader to spectral estimation followed by the bootstrap method in Chapter 3. Chapter 4 combines the presiding chapters and shows how the bootstrap method can be applied in spectral estimation problems. Chapter 5 tests the methods and Chapter 6 summarizes the findings.

Abbreviations, Acronyms and Symbols

ACS	autocovariance sequence
b,B	bootstrap samples
DTFT	discrete-time Fourier transform
i.i.d.	independent and identically distributed
WSS	second-order stationary sequence
α	set level of a test
$\hat{\theta}$	estimate or estimator of a parameter θ
μ	mean
σ^2	variance
$f_X(x)$	probability density function of X
$F_X(x)$	cumulative distribution function of X

Contents

1 Introduction

The are many different approaches to spectrum estimation in Signal Processing. The most common technique used for solving this problem is the asymptotic one, but this project seminar explores one method, that can be useful in case the quantity of data is not abundant, or the data is non-Gaussian, namely the bootstrap approach. This method uses a re-sampling technique, that can generate numerous new samples from just one realization of a random process. The main goal is to preserve the original distribution of the data and build a confidence interval for its spectra. There are many different bootstrap methods to choose from. Here we will investigate some recently proposed ones, a time domain bootstrap and a frequency domain approach, the so called tapered block bootstrap and a frequency domain residual based bootstrap. After having implemented a simulation in Python, we will compare the results of both methods in order to see which one performs better.

2 Spectral Estimation

2.1 Spectral Estimation Problem

The main goal of the spectral estimation problem is captured by the following informal expression:

> *From a finite record of a stationary data sequence, estimate how the total power is distributed over frequency. [1]*

Spectral analysis is a helpful tool in diverse scientific fields, like vibration monitoring, where the wear and other characteristics of mechanical parts can be studied via the spectral content of measured signals. In economics, the spectral analysis may reveal ”hidden periodicities” in the studied data, which are to be derived from cyclic behavior or recurring processes. In medicine, spectral analysis of various signals gathered from a patient, such as electrocardiogram (ECG) or electroencephalogram (EEG) signals can provide useful information for diagnosis. Other interesting examples of spectrum usage can be found in meteorology, astronomy, sonar systems, seismology and many other fields. There are two main approaches to spectral analysis. The nonparametric (classical) one derives from the idea of calculating the desired signals periodogram, or a variation of it. The second approach to spectral estimation, called the parametric approach, is to hypothesize a model for the data, which provides a way of parameterizing the spectrum and thereby reduce the spectral estimation problem to that of estimating the parameters in the assumed model (e.g. find the lags for an AR-Modell). Parametric methods may give more accurate spectral estimates than the nonparametric ones in the cases where the data truly satisfies the model assumed by the former methods. However, in the more likely case that the data does not fulfill the assumed models, the nonparametric methods may outdo the parametric ones owing to the sensitivity of the latter to model oversimplification.
Many real–world signals can be characterized as being random (from the observer's viewpoint). Briefly speaking, this means that the variation of such a signal outside the observed interval cannot be determined precisely but only specified in terms of statistical properties. That is why we will not try to directly predict the spectral values of the signal, but put a confidence bound around our prediction with the help of bootstrap techniques, discussed in Chapter 3 and 4, in order to minimize inaccuracy.

2.2 Mathematical Representation of Spectral Estimation

Let $X(t)$ represent the real-valued, discrete-time signal of interest:

$$X(t) \in \mathbb{R},\ t = 0, 1, 2, \ldots, T. \tag{2.1}$$

Most commonly, $X(t)$ is obtained by sampling a continuous time signal. For notation ease, the time index t is expressed in units of sampling interval. Let us assume that $X(t)$ has finite energy, which means that:

$$\sum_{t=-\infty}^{\infty} |X(t)|^2 < \infty. \tag{2.2}$$

Then, under some additional regularity conditions, the sequence $X(t)$ possesses a *discrete-time Fourier transform* (DTFT) defined as:

$$X(\omega) = \sum_{t=-\infty}^{\infty} X(t)e^{-j\omega t} \tag{2.3}$$

and an energy spectral density:

$$C(\omega) = |X(\omega)|^2. \tag{2.4}$$

Most of the signals encountered in applications are such that their variation in the future cannot be known exactly. It is only possible to make probabilistic statements about that variation. The mathematical device to describe such a signal is that of a random sequence which contains an ensemble of possible realizations, each of which has some associated probability of occurrence. Obviously, from the whole ensemble of realizations, the experimenter can typically observe only one realization of the signal, and then it might be thought that the deterministic definitions of the previous section could be carried over unchanged to the current case. However, this is not possible because the realizations of a random signal, viewed as discrete-time sequences, do not have finite energy, and therefore do not possess DTFTs. A random signal usually has finite average power and therefore can be characterized by an average power spectral density. For simplicity reasons, in what comes next we will use the name spectral density for that quantity.
The discrete-time signal $X(t)$ will now be assumed to be a sequence of random variables with zero mean, $\mathbb{E}[X(t)] = 0$, and autocovariance sequence (ACS) or covariance function, defined as $c_{XX}(t_1, t_2) = \mathbb{E}[X_{t_1} X_{t_2}] = c_{XX}(\tau)$, where $\tau = t_1 - t_2$. The two assumptions imply that $X(t)$ is a second-order stationary sequence(WSS). Let $X(t)$ have spectral density:

$$C_{XX}(\omega) = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} c_{XX}(\tau)e^{-j\omega\tau}, -\pi < \omega < \pi, \tag{2.5}$$

where $c_{XX}(\tau) = E[X_0 X_\tau]$ is the covariance function of $X(t)$. $C_{XX}(\omega)$ is a periodic function, with period equal to 2π and therefore is completely described by its variation on the interval $\omega = (-\pi, \pi)$ (radians per sampling interval). Also the spectral density is real-valued and non-negative. For real-valued signals, as in our case, $C_{XX}(\omega) = C_{XX}(-\omega)$.
Given observations $x_1, \ldots, x_T$ of the signal X, which are modeled by the random variables $X_1, \ldots, X_T$, we construct an estimator of the true spectrum $C_{XX}(\omega)$, as:

$$\hat{C}_{XX}(\omega; h) = \frac{1}{T \cdot h} \sum_{k=-N}^{N} K(\frac{\omega - \omega_k}{h}) I_{XX}(\omega_k) \tag{2.6}$$

where the kernel function $K(\cdot)$ is a known symmetric, non-negative, real-valued function, h is its bandwidth and N denotes the largest integer less than or equal to T/2. In equation (2.6), I_{XX} denotes the periodogram at frequencies $\omega_k = \frac{2\pi k}{T}$, $-N \leq \omega \leq N$, which is defined as:

$$I_{XX}(\omega) = \frac{1}{N} \left| \sum_{t=1}^{N} X(t)e^{-j\omega t} \right|^2 \tag{2.7}$$

and is a candidate estimator for the spectrum $C_{XX}(\omega)$ with the same symmetry, non-negativity and periodicity properties as $C_{XX}(\omega)$. When we apply only the periodogram it can be observed that the

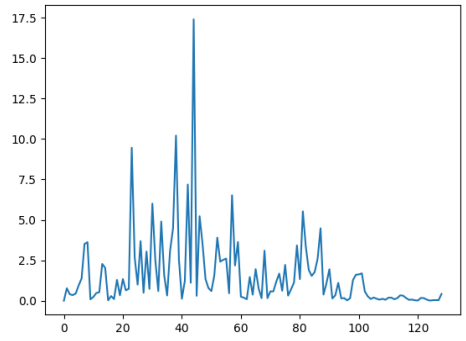


Figure 2.1: Periodogram $I_{XX}(\omega)$, spiky

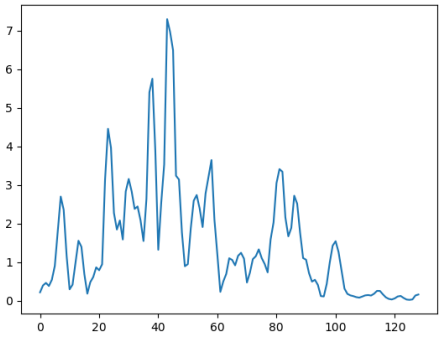


Figure 2.2: Kernel Spectral Density Estimate $\hat{C}_{XX}(\omega)$, smoothed version of $I_{XX}(\omega)$

variance, which corresponds to the fluctuations around the true spectrum, is quite large. Thus, the periodogram is not suitable for this estimation task alone and needs to be modified with a proper kernel function $K(\cdot)$, in order for the mean square error (MSE$[\hat{C}_{XX}]$ =Var$[\hat{C}_{XX}]$+bias$[\hat{C}_{XX}]$) to be minimized. The kernel function has a smoothing effect on the periodogram Fig.(2.2). It gives decreasing weights to a number of neighbors (which depends on the chosen bandwidth h) around a certain frequency ω, thus reducing the fluctuations around the true spectrum. In this way the variance is reduced as the number of neighbors increases, but this also increases the bias.
The spectral estimation problem can now be stated more formally as follows:

> *From a finite-length record $X_1, \ldots, X_T$ of a second-order stationary random process, determine an estimate $\hat{C}_{XX}(\omega)$ of its power spectral density $C_{XX}(\omega)$, for $\omega = [-\pi, \pi]$ [1]*

It would, of course, be desirable that $\hat{C}_{XX}(\omega)$ is as close to $C_{XX}(\omega)$ as possible. The main limitation on the quality of most power spectral density estimates is due to the small number of data samples T available for processing. In some cases, T is small since the price of obtaining large amounts of data is prohibitive. Most commonly, the value of T is limited by the fact that the observed signal can be considered second-order stationary only over short observation intervals.

3 Bootstrapping Method

3.1 Origin of "Bootstrap"

What is Bootstrapping? Before we explain the exact specifications of this method, it would be beneficiary to first have a quick look at the origin of the name “Bootstrapping”, because it gives us an idea of the imaginary nature of this “solution” to our small data sample problem. Almost every paper or article one can encounter, regarding this subject, associates the name “Bootstrapping” to the tale of Baron von Munchausen, who pulled himself up by the bootstraps from a sticky situation[2]. This analogy may suggest that the bootstrap is able to perform the impossible and has resulted sometimes in unrealistic expectations, especially when dealing with real data. Nevertheless, how can the tale of Munchausen be relevant to a statistical inference problem? The main problem with our estimation problem, encountered in Chapter 2, was that the number of realisations of the random variable at our disposal was limited. If we dare to use the ingenuity of Baron von Munchausen, we could try a trick, which at first seems impossible, but if applied with care a caution can lead to promising results. The principle is simple – just resample the original data. Samples can be constructed by drawing observations from a large data sample one at a time and returning them to the data sample after they have been chosen. This way one observation may be selected multiple times, while others may not be chosen at all.

3.2 Introduction to the bootstrap method

Suppose that we have a set of measured realizations $x = \{x_1, x_2, ..., x_n\}$ of the random sample $X = \{X_1, X_2, ..., X_n\}$, drawn from some unspecified distribution F_X. Let $\hat{\theta} = \hat{\theta}(X)$ be an estimator of some parameter θ of F_X, which could be, for example, the mean $\theta = \mu_X$ of F_X estimated by the sample mean $\hat{\theta} = \hat{\mu}_X = \frac{1}{N}\sum_{i=1}^{N} X_i$. The aim is to find characteristics of $\hat{\theta}$ such as the distribution of $\hat{\theta}$. If the distribution function, F_X is known or is assumed to be known and given that the function $\hat{\theta}(X)$ is relatively simple and then it is possible to exactly evaluate the distribution of the parameter estimator $\hat{\theta}$. In many practical applications, either the distribution F_X is unknown or the parameter estimator $\hat{\theta}_X$ is too complicated for its distribution to be derived in a closed form[2]. The question is how reliable is the parameter estimator $\hat{\theta}$? The easiest way to answer this is to take advantage of the central limit theorem: asymptotically approximate the distribution of the parameter and assume that the distribution of $\hat{\mu}_X$ is Gaussian.. However, what other options one has, if there is no way of repeating the process and the fact that the theorem does not apply for small sample sizes is also taken into account? The bootstrap is one way out. Its spirited logic suggest the substitution of the unknown distribution F_X by the empirical distribution of the data $\hat{F}_X$. The reuse of the original data through re-sampling creates what we call a bootstrap sample. It has the size of the original sample, i.e., ${x_b}^* = {x_1}^*, {x_2}^*, \ldots, {x_n}^*$ for $b = 1, \ldots, B$, where ${x_i}^* = x_m,\ i, m \in \{1, 2, \ldots, n\}$, are obtained, commonly, by drawing at random with replacement from x.

A straightforward example of the bootstrap procedure can be seen in (3.1), while the figure on the right (3.2) showcases an example of how the bootstrap samples are generated (Note, that in these two

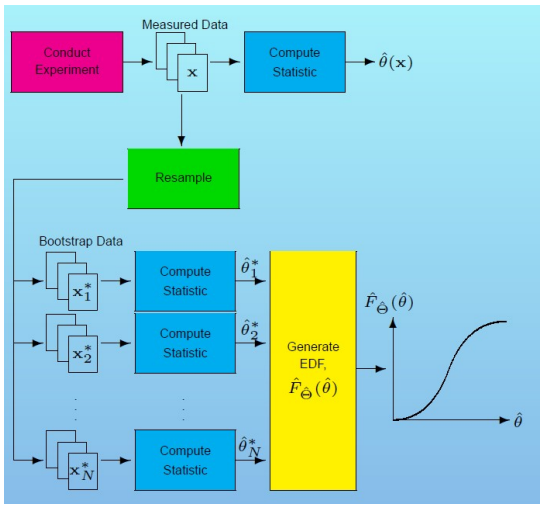


Figure 3.1: The Bootstrap Procedure[3]

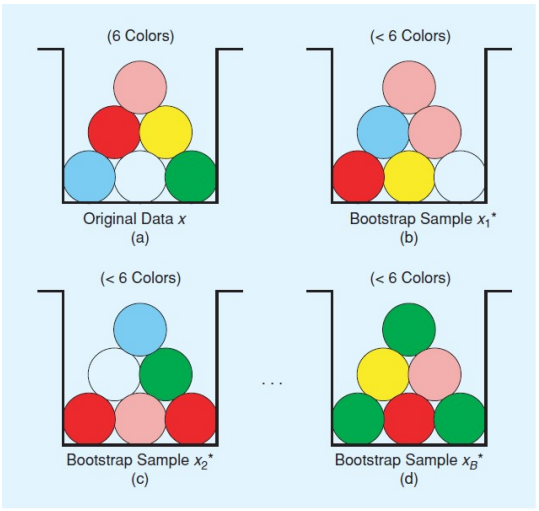


Figure 3.2: The independent data bootstrap re-sampling principle[2]

figures $N = B$). Each of the bootstrap samples in the figure is considered as new data. Based on the bootstrap sample ${x_b}^*$, bootstrap parameter estimates $\hat{\theta}_b^* = \hat{\theta}({x_b}^*)$ for $b = 1, 2, \ldots, B$ are calculated. Given a large number B of bootstrap parameter estimates, we can then approximate the distribution of $\hat{\theta}$ by the distribution of $\hat{\theta}^*$, which is derived from the bootstrap sample x^*, i.e., we approximate the distribution $F_{\hat{\theta}}$ of $\hat{\theta}$ by $\hat{F}_{\hat{\theta}^*}$, the distribution of $\hat{\theta}^*$.

There are several technical points that need to be mentioned, considering the performance of the bootstrap. One key factor for the success of the bootstrap method is the number of observations at disposal. Bootstrapping is generally applied, when the realizations of a measurement are scares to begin with. However, as every statistical estimation problem in practice, the amount of data influences the accuracy of the estimator greatly.

Andrews and Buchinsky have thoroughly examined [4], considering the quantity of bootstrap samples B needed, so that the distribution of a parameter estimator is approximated. One unwritten rule is for the number of bootstrap samples B to take a value between 25 and 50 for variance estimation and to be set to about 1,000 where a 95% confidence interval is sought. However, with the fast increasing computational power, there are no limitations to exceeding these numbers. In theory, the "ideal" bootstrap estimate of the parameter we are seeking consists of an infinite amount of bootstrap re-samples.

Note that the bootstrap simulation error, which quantifies the difference between the true distribution and the estimated distribution, comes from two independent errors of different sources, i.e., a bootstrap (statistical) error and a simulation (Monte Carlo) error. The first error can not be avoided and does not depend on the number of bootstrap samples B but on the size N of the original sample. The second one, on the other hand, can be minimized by increasing B. The aim is therefore to choose B so that the the bootstrap error is no greater than the simulation error. If the sample size N is large to begin with, we might be forced to reduce the amount of bootstrap samples B, because of the fact, that the time it takes to bootstrap a sample is positively correlated with the number of samples N we need to draw. But as the amount of samples N increases, so does the accuracy of our estimator, while the bootstrap error decreases. We found that the rule of thumb of choosing $B = 40N$, proposed by Davison and Hinkley [5], is appropriate in many applications. In practice, the value of B is situation dependent and is left to the experimenter to choose.

STEP 1) FIT A MODEL TO THE DATA.
STEP 2) SUBTRACT THE FITTED MODEL FROM THE ORIGINAL DATA TO OBTAIN RESIDUALS.
STEP 3) CENTER (OR RESCALE) THE RESIDUALS.
STEP 4) RESAMPLE THE RESIDUALS.
STEP 5) CREATE NEW BOOTSTRAP DATA BY ADDING THE RESAMPLED RESIDUALS TO THE FITTED MODEL FROM STEP 1.
STEP 6) FIT THE MODEL TO THE NEW BOOTSTRAP DATA.
STEP 7) REPEAT STEPS 4–6 MANY TIMES TO OBTAIN DISTRIBUTIONS FOR THE MODEL PARAMETER ESTIMATORS.

Figure 3.3: Residual-based Bootstrap Procedure For Dependent Data[2]

The assumption that the data is i.i.d. is not always true. Here we provide some guidance as to how to re-sample dependent data. Note that if the data was i.i.d., standard bootstrap re-sampling with replacement gives an accurate representation of the underlying distribution. However, if the data shows heteroskedasticity (the random variables in the sequence or vector may have different variances) or serial correlation, randomly re-sampled data would lead to errors.
One way to extend the basic bootstrap principle to dependent data is the concept of data modeling and the subsequent assumption of i.i.d. residuals that approximate the modeling and measurement errors. In analogy to the linearization of a nonlinear problem, the idea here is to reformulate the problem so that the i.i.d. component of the data may be used for re-sampling. In most cases, the procedure follows the structure described in 3.3. We can use the above procedure in many signal processing problems, in our case the one discussed in Chapter 2.2, the residuals in Step 2 of the above procedure can be found as a ratio of the two parameter estimators, for example the ratios between the periodogram (2.7) and the kernel spectrum density estimator (2.6) at distinct frequency bins. The residuals are than assumed to be i.i.d.

4 Bootstrap Methods Applied in Signal Processing

4.1 Introduction to bootstrap methods for spectral estimation

As mentioned in the previous chapter, there are bootstrap procedures, which can be applied to signal processing problems in order to bypass difficult situations. Such a problem could be, for example, a limited amount of available samples due to the stationarity requirement holding true for a limited amount of time. The Bootstrap method promises a simple and straightforward solution for this, namely re-sampling by drawing at random with replacement. By sampling with replacement, the sample values are assumed to be independent. Practically, this means that the covariance between the two equal zero. We also force the unknown distribution of the parameter we seek to assume the form of a normal distribution (see 3.1). These two assumptions lead to the conclusion that our data is i.i.d. However, this conclusion is an idealization of what could be expected, given a real signal.
There are two main approaches to bootstrapping spectra. One could use time domain methods, where the re-sampling is performed in the time domain, followed by a calcualtion of the resamples based bootstrap periodogram, $I_{XX}{}^{*}(\omega_k)$. Afterwards, the bootstrap spectral estimate $C_{XX}{}^{*}(\omega)$ is obtained via kernel spectral density estimation as in Eq.(2.6), but with $I_{XX}{}^{*}(\omega_k)$ taking the place of the periodogram $I_{XX}(\omega_k)$. Alternatively, one could use frequency domain approaches, in which the periodogram itself is re-sampled. These methods rely on simpler independent data re-sampling schemes, which are justified by the fact that periodogram ordinates at frequencies $\omega_k = 2\pi k/T, k = 0, 1, 2, \ldots, N$, are nearly independent. In what follows, we will introduce and test these two approaches as suggested in [6].

4.2 Frequency domain residual based bootstrap

Franke and Härdle [7] proposed to express the relationship between the periodogram (2.7) and the spectral density (2.5) as a multiplicative regression and to bootstrap the so obtained frequency domain residuals. The method which has been shown to be asymptotically valid is summarized in Figure 4.1(Eq.(4)= Eq.2.7, Eq.(3)= Eq.2.6). Note that in the first step we have to center the random variable realizations $X_1, X_2, \ldots, X_T$ by substracting the sample mean μ_X. Afterwards, we must compute a initial spectral density estimate via the periodogram and an initial bandwidth $h_i > 0$. In Step 3 we express the frequency domain residuals $\hat{\varepsilon}_k$ as the relation between the periodogram and the estimate of the spectral density. Here the residuals are marked with the index $k = 1, \ldots, N$ and are half the length of our original sample X_t, because of the symmetry property of the periodogram and the spectral estimate. After that, the residuals $\hat{\varepsilon}_k$ are normalized, by division with the residual mean $\varepsilon = \frac{1}{N}\sum_{t=1}^{N} \hat{\varepsilon}_k$. These re-scaled real valued residuals $\tilde{\varepsilon_k}$ are approximately i.i.d. for N sufficiently large. The re-scaling of residuals performed in Step 3 serves essentially the same purpose as re-centering the data in an additive regression model in order to avoid an additional bias at the re-sampling stage. Now that we have prepared our data, we can start the bootstrap procedure. First (Step 4) we have to draw with replacement from the re-scaled residuals $\tilde{\varepsilon_1}, \ldots, \tilde{\varepsilon_N}$ into our bootstrap residual sample $\tilde{\varepsilon_1}^*, \ldots, \tilde{\varepsilon_N}^*$. Next (Step 5), we take the bootstrap residuals $\tilde{\varepsilon}^*$ and multiply them with our initial spectral estimate $\hat{C}_{XX}(\omega_k)$ in order to get the bootstrap periodogram $I^*{}_{XX}(\omega_k)$. Here we use the bandwidth g. We set the $I^*{}_{XX}(0) = 0$, because the periodogram at $\omega = 0$ represents the average of the random variable $X_t, t = 0, 1, \ldots, T$, which is in our case $\mu_X = 0$, because we centered X at Step 1. Now we once more use equation (2.6) to estimate the spectrum, but this time we do it with the bootstrapped periodogram. Here we use the bandwith h. Note, that this bootstrap algorithm uses three distinct kernel bandwidths, namely, an initial bandwidth h_i, a re-sampling bandwidth g and finally h, which may or may not be one and the same. The choice of bandwidth in kernel spectral density estimation is of utmost importance. For more information check [8]. Step 6 represents the repetition of the bootstrap procedure. The optimal number of bootstrap samples was discussed in Chapter 3.2. Finally, after all the bootstrap spectrum estimates $\hat{C}^*{}_{XX}$ have been calculated, a confidence interval with an arbitrary α can be set around the initial estimate $\hat{C}_{XX}$.

Step 1. *Centering.* Center $X_1, \ldots, X_T$ by subtracting the sample mean.

Step 2. *Initial Estimate.* Compute the periodogram according to Eq. (4). With an initial bandwidth $h_i > 0$, calculate $\hat{C}_{XX}(\omega; h_i)$ according to Eq. (3).

Step 3. *Compute and Rescale Residuals.* Calculate the residuals

$$\hat{\varepsilon}_k = \frac{I_{XX}(\omega_k)}{\hat{C}_{XX}(\omega_k; h_i)}, \quad k = 1, \ldots, N$$

and rescale them to obtain

$$\tilde{\varepsilon}_k = \frac{\hat{\varepsilon}_k}{\varepsilon.}, \quad k = 1, \ldots, N, \quad \varepsilon. = \frac{1}{N}\sum_{k=1}^{N} \hat{\varepsilon}_k.$$

Step 4. *Bootstrap Residuals.* Draw independent bootstrap residuals $\tilde{\varepsilon}_1^*, \ldots, \tilde{\varepsilon}_N^*$ from $\tilde{\varepsilon}_1, \ldots, \tilde{\varepsilon}_N$.

Step 5. *Bootstrap Estimate.* Choose a resampling kernel bandwidth g and define the bootstrap periodogram as $I^*_{XX}(\omega_k) = I^*_{XX}(-\omega_k) = \hat{C}_{XX}(\omega_k; g)\tilde{\varepsilon}_k^*$,

setting $I^*_{XX}(0) = 0$. Then find the bootstrap kernel spectral density estimate

$$\hat{C}^*_{XX}(\omega; h; g) = \frac{1}{T \cdot h}\sum_{k=-N}^{N} K\left(\frac{\omega - \omega_k}{h}\right) I^*_{XX}(\omega_k).$$

Step6. *Repetition.* Repeat Steps 4 and 5 a large number of times.

Step 7. *Confidence Interval Estimation.* With an arbitrary α find c_U^* such that

$$\text{Prob}^*\left(\sqrt{T \cdot h}\frac{\hat{C}^*_{XX}(\omega; h; g) - \hat{C}_{XX}(\omega; g)}{\hat{C}_{XX}(\omega; g)} \le c_U^*\right) = \frac{\alpha}{2}$$

and set the upper bound of the $100(1-\alpha)\%$ confidence interval for $C_{XX}(\omega)$ as $\hat{C}_{XX}(\omega; h)/(1 + c_U^*(T \cdot h)^{-1/2})$. Herein, $\text{Prob}^*(\cdot)$ denotes probability conditioned on the measured data. Analogously using a c_L^*, compute the lower bound of the confidence interval.

Figure 4.1: The residual based bootstrap procedure[6]

4.3 Time domain tapered block bootstrap

How can we bootstrap non-i.i.d. data without imposing a parametric model? Can we re-sample the data non-parametrically? Künsch [9] was the first one to try and answer these questions with his concept of re-sampling sequences (chunks) of data. The method is referred to as the moving block bootstrap. In essence, rather than re-sampling with replacement single data points, sets of consecutive points are re-sampled to maintain, in a non-parametric fashion, the structure between neighboring data points. The segments chosen for bootstrapping can be either non-overlapping or overlapping. The idea of a recently proposed method by Paparoditis and Politis [10], known as the tapered block bootstrap, is to assign different weights to the data points according to their position within their block,specifically,giving reduced weight to data near the endpoints of the window.

Step 1. *Centering.* Center $X_1,\dots,X_T$ by removing the sample mean.

Step 2. *Spectrum Estimation.* Calculate the periodogram $I_{XX}(\omega_k)$ from $X_1,\dots,X_T$ according to Eq. (4) and the corresponding kernel spectral density estimate $\hat{C}_{XX}(\omega_k)$, $k=1,\dots,N$, as in Eq. (3).

Step 3. *Set the Bootstrap Parameters.* Choose a block size b, where b is a positive integer less than T and define $k=\lfloor T/b \rfloor$ as the number of blocks to draw from, where $\lfloor\cdot\rfloor$ denotes the integer part.

Step 4. *Draw Bootstrap Resamples.* Let the integers $i_0,i_1,\dots,i_{k-1}$ be drawn independently and identically distributed with distribution uniform on the set $\{1,2,\dots,T-b+1\}$. Then for $m=0,1,\dots,k-1$, compute

$$X^*_{mb+j} = w_b(j)\frac{\sqrt{b}}{\|w_b\|_2}X_{i_m+j-1}, \quad j=1,2,\dots,b,$$

which yields the pseudo-series $X^*_1,X^*_2,\dots,X^*_l$ of size $l=k\cdot b$. Herein w_b denotes the tapering window defined in Eq. (9) and $\|w_b\|_2=(\sum_{t=1}^{b} w_b(t)^2)^{1/2}$.

Step 5. *Construct Bootstrap Estimates.* Center the pseudo-series $X^*_1,X^*_2,\dots,X^*_l$ and find the bootstrap periodogram $I^*_{XX}(\omega_k)$ for $-N^*<k<N^*$, computed as in Eq. (4), replacing T by l and $X_1,\dots,X_T$ by $X^*_1,X^*_2,\dots,X^*_l$, where $N^*=\lfloor l/2 \rfloor$ and $I^*_{XX}(0)=0$. Then find the corresponding bootstrap kernel spectral density estimate $\hat{C}^*_{XX}(\omega)$ as in Eq. (3), with l replacing T.

Step 6. *Repetition.* Repeat Steps 4 and 5 a large number of times.

Step 7. *Confidence Interval Estimation.* Find confidence bounds as in Step 7 of Table 1, replacing T by l.

Figure 4.2: Tapered Block Bootstrap[6]

In Figure 4.2 we can see an example of a tapered block bootstrap algorithm (Eq.(4)= Eq.2.7, Eq.(3)= Eq.2.6). Step 1 and 2 are the same as the frequency domain residual based bootstrap introduced in the previous chapter. In Step 3, we choose the block size b and draw $k = \lfloor T/b \rfloor$ number of blocks. Selection of the block length b is crucial as it heavily affects the performance of the tapered block bootstrap in practice. In Step 4 we begin to draw blocks of length b from our recorded sample X. But how do we decide which blocks to draw, so that we can assume the data is i.i.d.? Let index integers $i_0, i_1, \dots, i_{k-1}$ represent the start of the block inside the sample, which are drawn i.i.d. from $[1, 2, \dots, T-b+1]$. In order for the block to have the full length b, the biggest index we can draw should be limited to $i_{max} = T-b+1$. Additionally, this algorithm must be transformed from a moving block bootstrap to a tapered block bootstrap. When we draw a block we must assign weights to it via a tapering window. In this case the tapering window is a trapezoid. For more information check [6], Page 1426. After we are done drawing and reshaping the k blocks, we arrange them to get the newly obtained sample X^*, which has the length $l = kb$. Note that in most cases $l \neq T$. It is recommended in this scenario to draw an additional block and truncate it, so that our bootstrap sample X^* has the same length as the original sample X. Step 5 represents the calculation of the periodogram (Eq.2.7) and the spectral density estiamte (Eq.2.6) of the newly obtained tapered block bootstrap sample X^*. Steps 4 and 5 need to be repeated a large number of times in order for the estimator to asymptotically converge to the real value (Step 6). The optimal number of bootstrap samples was discussed in Chapter 3.2. In Step 7, we calculate the confidence interval from the bootstrap spectrum estimates $\hat{C}^*{}_{XX}$, as in Step 7 in Fig. 4.1.

5 Test of Frequency and Time Domain Bootstrap Methods

5.1 Test Model

In the previous chapter we introduced two bootstrap methods, which promise to deliver inferential statistics about an estimator of some parameter $\hat{\theta} = \hat{\theta}(X)$ of some unspecified distribution F_X given realizations of a random variable X. One of the bootstrap methods re-samples the residuals of the fitted model in the frequency domain, while the other draws blocks of the random variable in the time domain in order to reconstruct a new version of the random variable. But which one performs better? We will use a AR(5) model from [3]:

$$X(t) = 0.5X_{t-1} - 0.6X_{t-2} + 0.3X_{t-3} - 0.4X_{t-4} + 0.2X_{t-5} + N_t, \tag{5.1}$$

N_t are independently and uniformly distributed variables on the interval $[0, 1]$, with a sample size of $T = 256, 128$, in order to test these both methods and see how they performe with a reduced amount of observations. We used the Bartlett-Priestley kernel function ([11],p.444) for the spectral kernel estimator (2.6) with a bandwidth of $h_i = g = h = 0.05$ for both methods.
A reasonable block size for the tapered block bootstrap was obtained by manually inspecting $\hat{C}^*{}_{XX}(\tau)$ to find the smallest integer τ after which the covariance appears negligible. Specifically, we used $b = 38$.
The programming language Python was the chosen programming environment in order to code the two bootstrap methods and test them. The code can be found in the public Github page:
`https://github.com/momchi93/Bootstrapp`.

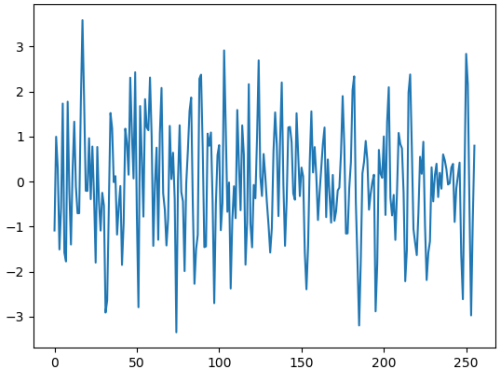


Figure 5.1: AR(5) Model in time domain

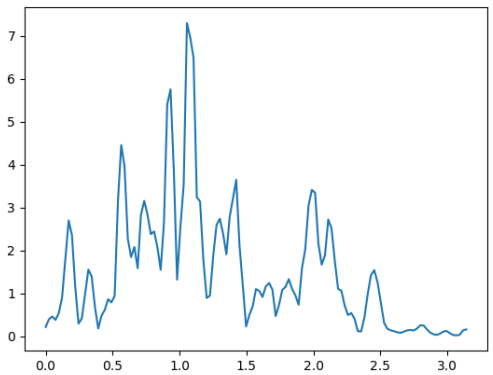


Figure 5.2: AR(5) Model in freq. domain, represented by the kernel estimate $\hat{C}^*{}_{XX}$

T = 256						
bootstrap samples	1	3	6	10	100	1000
coverage probability in %	50	83.24	95.33	98.81	99.51	99.61
T = 128						
bootstrap samples	6	50	55	60	100	1000
coverage probability in %	69.38	94.0	95.84	96.76	98.61	99.23

Table 5.1: Frequency Domain Residual based Bootstrap: Coverage probability for different number of bootstrap samples and length of X_t

We have implemented the frequency domain residual based bootstrap method as shown in Chapter 4.2. The results are shown in Table 5.1. 100 Monte Carlo runs were used to generate these results. We used an $\alpha = 0.05$, which suggests that the interval we set around the kernel spectral density estimate $\hat{C^*}_{XX}$ should cover at lest 95% of all frequency bins $\omega_k = 2\pi k/T, k = 0, 1, 2, \ldots, N$. The way we measured the coverage probability in the Python simulation is the following: After we calculate the interval around the kernel estimate, we count how many times the upper bound estimate is bigger than the kernel estimate, we added that to the count of how many times the lower bound was actually lower than the kernel estimate and than we divide that number by the number of frequency bins times 2. As we can see in the table above, we start with a coverage probability of 50% for 1 bootstrap, which is equal to a coin flip. That gives us no information. But as soon as we reach 6 bootstraps for $T = 256$ we get our desired 95% coverage probability. Further increasing the bootstrap quantity gives us even better results, and by reaching 1000 bootstraps as suggested in Chapter 3.2 we get coverage probability close to 100%. But what happens if for example our available samples reduce to $T = 128$, due to to loss of stationarity? As we can see in Table 5.1, we reach our 95% coverage around 55 bootstrap samples, which is almost 10 times bigger than what we get if we have 2 times more samples to work with. Interestingly, if we can afford to draw 1000 bootstrap samples, we get almost identical results in both cases. This leaves us to the assumption, that with an increasing amount of bootstrap samples, we can compensate for the lack of initial samples to begin with. Note that in Chapter 3.2 we made the assumption, that the bootstrap samples B can be reduced, if we have a bigger amount of samples N, which is confirmed by these results.
But what does increasing the bootstrap samples really do in this model? If we take in consideration the fact, that if we inspect the possible residuals that can be drawn, there is a maximum value we can get (considering the upper bound, the same can be applied to the lower bound, but with a minimum value). Every time we draw residuals, there is a chance, that we draw this maximum value for a certain frequency. If we start with fewer bootstraps, the chance is low that we will draw this value, but if we can draw in infinite amount of times, eventually we will get this maximum value for all frequencies, making the interval the biggest it can get. But that is no good, because firstly, we don't have the computational power required for that, secondly we want our interval to be as tight as possible to the kernel estimate. Theoretically, we can just say that the power spectrum can take values between $(-\infty, \infty)$ at all frequencies and we will always be right. That is the definition of a random variable, but in reality it doesn't help us estimate anything. In order to demonstrate the growth of the interval for increasing bootstrap samples, see the following figures.

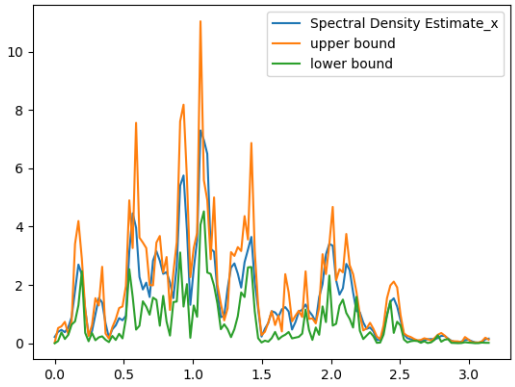


Figure 5.3: Residual-Based Bootstrap, B = 3

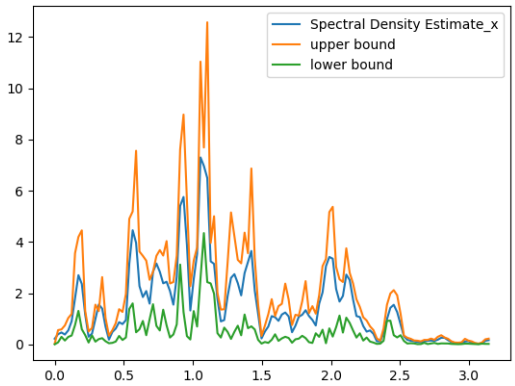


Figure 5.4: Residual-Based Bootstrap, B = 6

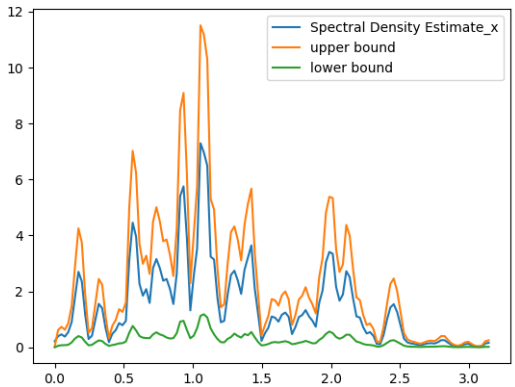


Figure 5.5: Residual-Based Bootstrap, B = 1000

5.3 Time domain tapered block bootstrap

T = 256						
bootstrap samples	1	3	6	10	100	1000
coverage probability in %	50	67.32	74.88	80.71	96.67	99.32
T = 128						
bootstrap samples	6	11	16	21	46	1000
coverage probability in %	89.61	93.40	94.98	95.90	97.57	99.15

Table 5.2: Time domain tapered block bootstrap: Coverage probability for different number of bootstrap samples and length of X_t

We have implemented the time domain tapered block bootstrap method as shown in Chapter 4.3. The results are shown in Table 5.2. 100 Monte Carlo runs were used to generate these results. We used an $\alpha = 0.05$, which suggests that the interval we set around the kernel spectral density estimate $\hat{C^*}_{XX}$ should cover at lest 95% of all frequency bins $\omega_k = 2\pi k/T, k = 0, 1, 2, \ldots, N$. The way we measured the coverage probability in the Python simulation is the same as in the previous section. As we can see in the table above (for $T = 256$), we start with a coverage probability of 50% for 1 bootstrap, which is equal to the previous method and is not useful. As we increase the bootstrap samples B, we start to get better results, but the increase in the coverage probability is substantially slower compared to the frequency domain residual based bootstrap method. We reach the 95% mark around 100 bootstrap samples, which is around 16 times more than what we need with the frequency algorithm. If we decrease the amount of observations to $T = 128$, counter intuitively we get better results for the same length T. With just 20 bootstrap samples we reach the 95% interval and get even better results, than the frequency domain method. If we look at Figure 5.6, we can get a better understanding of this phenomena. As we can see, the tapered samples X^* look nothing like the original sample X. If we calculate the spectrum of these newly obtained samples, the frequency components will be totally altered and will not look anything like the original spectrum C_{XX}. But as we decrease the length of the signal, while keeping the block length b constant, we will have to draw less blocks and thus the signal will be closer to its original shape. This leaves us to the conclusion, that drawing block in the time domain is not a good strategy to begin with, making the frequency domain approach a better method.
It should also be noted, that if one desires to use the tapered method, keeping the size l of the bootstrap sample X^* the same as the size T of the original sample X is critical. We have not compared the coverage probability of these two cases, because it is clear from Figure 5.6, that when $l \neq T$ the interval does not cover the kernel estimate.
Finally, if we compare both methods for both time lengths $T = (128, 256)$ at $b = 1000$ bootstraps, we get roughly the same coverage probability. That suggest, that the "Bootstrap Method" can be generally useful for inferential signal processing tasks, given that the computational power is sufficient to generate a large amount of re-samples, which in our day and time is already not a problem.

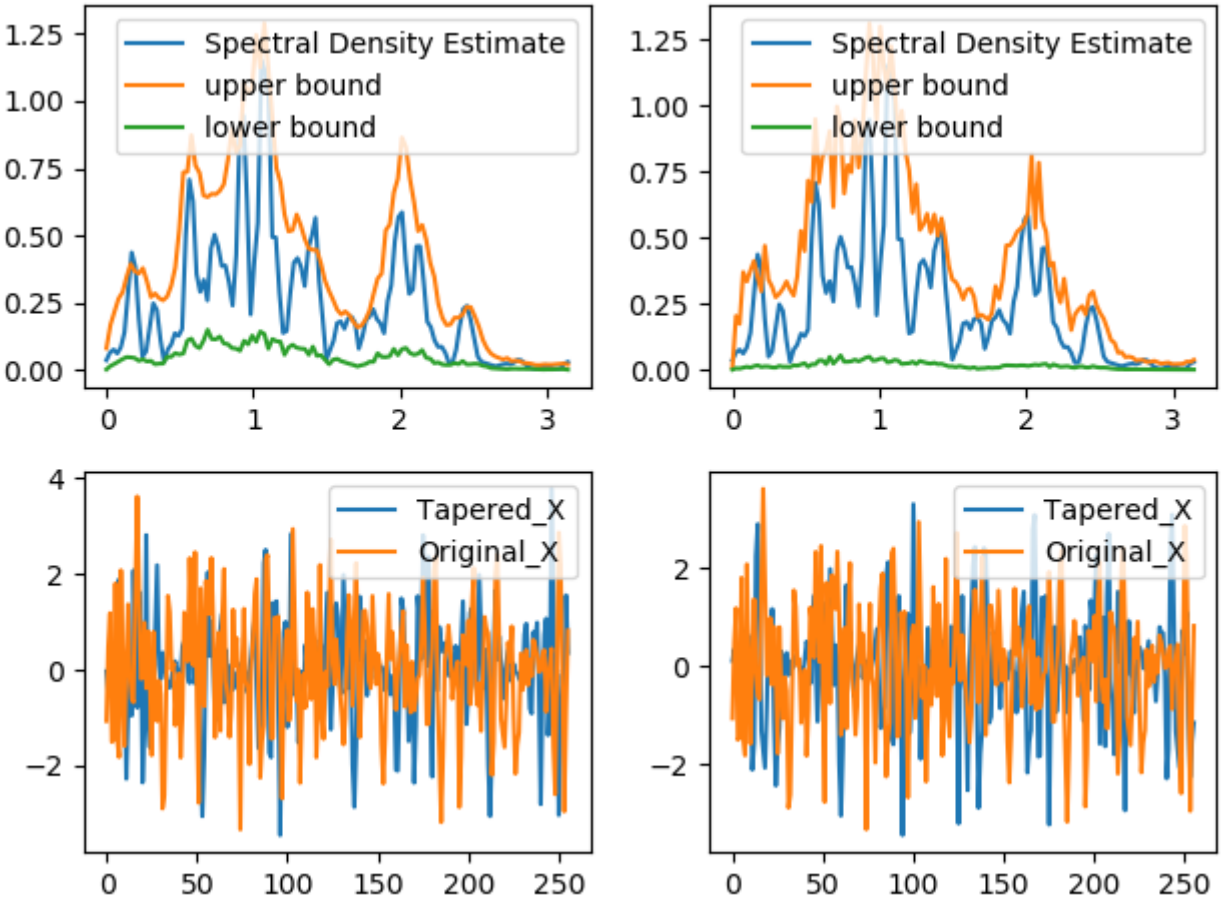


Figure 5.6: Time domain tapered block bootstrap: In the upper left figure the length of the tapered sample is not equal to original sample $l \neq T$, upper right figure $l = T$. It can be seen by the naked eye, that the second method covers the estimate $\hat{C^*}_{XX}$ better. The bottom figures show 2 cases of original X versus tapered X^* sample, $l = T$ in both cases.

6 Conclusions

In this seminar work, we have discussed the estimation of confidence intervals for spectra with the bootstrap method. Specifically, we have compared a frequency domain method to a time domain method. The result was clear, the frequency domain residual based bootstrap method is to be chosen, based on the fact, that it needs less bootstrap samples b in order to perform the inferential statistics task it is given. Also it has been shown, that the tapered block bootstrap distorts the original signal, which hinders further work with the data. It should be noted, that the underlying real-life signals do not follow the assumed model of linear processes, the frequency domain residual based bootstrap has shown in this seminar work its good performance in that it maintains a coverage level close to the preset one and therefore can be considered robust.
For someone looking to investigate the subject of "Bootstrapping" further, the paper of Prof. Zoubir [6], which this seminar work is based on, suggests a third method, which is a combination of the time domain and frequency domain method. It has been found, that this method is superior.

List of References

[1] P. Stoica and R. L. Moses, *Spectral analysis of signals*. Upper Saddle River, N.J.: Pearson/Prentice Hall, 2005.

[2] A. Zoubir and D. Iskandler, "Bootstrap methods and applications," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 10–19, 2007.

[3] A. M. Zoubir and D. R. Iskander, *Bootstrap Techniques for Signal Processing*. Cambridge: Cambridge University Press, 2004.

[4] Donald W. K. Andrews, and Moshe Buchinsky, ". "a three-step method for choosing the number of bootstrap repetitions."," *Econometrica*, no. 1, pp. 23–51, 2000.

[5] D. Kushary, A. C. Davison, and D. V. Hinkley, "Bootstrap methods and their application," *Technometrics*, vol. 42, no. 2, p. 216, 2000.

[6] A. M. Zoubir, "Bootstrapping spectra: Methods, comparisons and application to knock data," *Signal Processing*, vol. 90, no. 5, pp. 1424–1435, 2010.

[7] J. Franke and W. Hardle, "On bootstrapping kernel spectral estimates," *The Annals of Statistics*, vol. 20, no. 1, pp. 121–145, 1992.

[8] P. M. Robinson, "Automatic frequency domain inference on semiparametric and nonparametric models," *Econometrica*, vol. 59, no. 5, p. 1329, 1991.

[9] H. R. Kunsch, "The jackknife and the bootstrap for general stationary observations," *The Annals of Statistics*, vol. 17, no. 3, pp. 1217–1241, 1989.

[10] E. Paparoditis and D. N. Politis, "Tapered block bootstrap," pp. 1105–1119, 2001.

[11] M. B. Priestley, *Spectral analysis and time series*. Probability and mathematical statistics, London: Academic Press, 1981.