Quantile-Based Spectral Analysis in an Object-Oriented Framework and a Reference Implementation in **R**: The quantspec Package

Tobias Kley

Ruhr-Universität Bochum

Abstract

Quantile-based approaches to the spectral analysis of time series have recently attracted a lot of attention. Several methods for estimation have been proposed in the literature and their statistical properties were analyzed. Yet, so far, neither a systematic method for computation nor a comprehensive software implementation are available to date. This paper contains two main contributions. First, an extensible framework for quantile-based spectral analysis of time series is developed and documented using object-oriented models. A comprehensive, open source reference implementation of this framework is provided in the **R** package **quantspec**, which is available from the Comprehensive **R** Archive Network. The second contribution of the present paper is to provide a detailed tutorial, with worked examples, for this **R** package. A reader who is already familiar with quantile-based spectral analysis and whose primary interest is not the design of the **quantspec** package, but how to use it, can read the tutorial and worked examples (Sections 3 and 4) independently.

Keywords: time series, spectral analysis, periodogram, quantile regression, copulas, ranks, **R**, **quantspec**, framework, object-oriented design.

1. A short introduction to quantile-based spectral analysis

1.1. Laplace and copula cumulants and their spectral representation

Quantification of serial dependence in a second-order stationary process $(X_t)_{t \in \mathbb{Z}}$ is traditionally based on its autocovariance and autocorrelation functions, which measure linear dependencies among observations at different times. Periodicities of a time series are then most commonly analyzed by decomposing the autocovariance function, into a sum of sines and

cosines. This approach is referred to as (ordinary) spectral analysis of time series and has been known for decades. As a statistical method, it has been investigated many times and is well understood. In the analysis of centered Gaussian time series this approach is particularly attractive, because the autocovariance function completely characterizes the distribution of the underlying process. If that process is not Gaussian, ordinary spectral analysis suffers from typical weaknesses of L_2-methods: It is lacking robustness against outliers and heavy tails, and it is unable to capture important dynamic features such as changes in the conditional shape (skewness, kurtosis), time-irreversibility, or dependence in the extremes. In addition, only time series with an existing second moment can be analyzed at all. All of this was previously realized by many researchers, and various extensions and modifications of the L^2-periodogram have been proposed to remedy those drawbacks.

Approaches to robustifying the traditional spectral methods against outliers and deviations from the distributional assumptions were taken, among others, by Kleiner, Martin, and Thomson (1979); Klüppelberg and Mikosch (1994); Mikosch (1998); Katkovnik (1998); Hill and McCloskey (2013). To account for more general dynamic features alternative spectral concepts and tools were recently proposed. A first step in that direction was taken by Hong (1999). In order to obtain a complete description of the two-dimensional distributions at lag k, he introduced a *generalized spectrum* where the covariances $\mathsf{COV}(X_t, X_{t-k})$ are replaced by the covariances $\mathsf{COV}(e^{ix_1 X_t}, e^{ix_2 X_{t-k}})$ yielding a spectrum closely related to the joint characteristic functions of the pairs (X_t, X_{t-k}). In the quantile-based approach to spectral analysis the objects of interest are the *Laplace cross-covariance kernel*

$$\gamma_k(q_1, q_2) := \mathsf{COV}\Big(I\{X_t \leq q_1\}, I\{X_{t-k} \leq q_2\}\Big), \quad q_1, q_2 \in \bar{\mathbb{R}}, \; k \in \mathbb{Z},$$

and the *copula cross-covariance kernel*

$$\gamma_k^U(\tau_1, \tau_2) := \Big(I\{F(X_t) \leq \tau_1\}, I\{F(X_{t-k}) \leq \tau_2\}\Big), \quad \tau_1, \tau_2 \in [0,1], \; k \in \mathbb{Z},$$

where $I\{A\}$ denotes the indicator function of the event $\{A\}$, $\bar{\mathbb{R}} := \mathbb{R} \cup \{-\infty, \infty\}$ the extended real line and F the marginal distribution function (i.e., the distribution function of any X_t with strict stationarity being assumed). Obviously these measures exist without the necessity to make assumptions about moments. Also, when the underlying process is not Gaussian, and the quantile-based measures of serial dependence are considered to be functions with arguments q_1, q_2, or τ_1, τ_2 respectively, they provide a much richer picture about the pairwise dependence than would the autocovariances. As in the approach of Hong (1999), a complete description of the joint distributions (or copulas) of the pairs (X_t, X_{t-k}) is available. A particular advantage of the copula cross-covariance kernel is its invariance to monotone transformations. This allows to disentangle the serial features from the marginal features. For a full list of the properties and advantages of these dependence measures the interested reader is referred to Hong (2000); Li (2008, 2012, 2013, 2014); Hagemann (2013); Lee and Rao (2012); Kley (2014); Dette, Hallin, Kley, and Volgushev (2015) and Kley, Volgushev, Dette, and Hallin (2016).

Under summability conditions on (γ_k) and (γ_k^U) the representations of (γ_k) and (γ_k^U) in the "frequency domain" take the form of the *Laplace spectral density kernel*

$$\mathfrak{f}_{q_1, q_2}(\omega) := \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \gamma_k(q_1, q_2) e^{-ik\omega}, \quad q_1, q_2 \in \bar{\mathbb{R}}, \; \omega \in \mathbb{R}, \tag{1}$$

and the *copula spectral density kernel*

$$\mathfrak{f}_{q_{\tau_1}, q_{\tau_2}}(\omega) := \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \gamma_k^U(\tau_1, \tau_2) \mathrm{e}^{-ik\omega}, \quad \tau_1, \tau_2 \in [0, 1], \ \omega \in \mathbb{R}, \tag{2}$$

where $q_\tau := F^{-1}(\tau)$. By the relation

$$\gamma_k(q_1, q_2) = \int_{-\pi}^{\pi} \mathrm{e}^{ik\omega} \mathfrak{f}_{q_1, q_2}(\omega) \mathrm{d}\omega,$$

and a similar representation for $\gamma_k^U(\tau_1, \tau_2)$, the representations in the "frequency domain" are seen to be equivalent to the "time domain" quantities.

Sometimes considering the cumulated Laplace or copula spectral density kernels, which can be defined as

$$\mathfrak{F}_{q_1, q_2}(\omega) := \int_0^\omega \mathfrak{f}_{q_1, q_2}(\lambda) \mathrm{d}\lambda, \quad q_1, q_2 \in \bar{\mathbb{R}}, \ \omega \in [0, 2\pi], \tag{3}$$

and

$$\mathfrak{F}_{q_{\tau_1}, q_{\tau_2}}(\omega) := \int_0^\omega \mathfrak{f}_{q_{\tau_1}, q_{\tau_2}}(\lambda) \mathrm{d}\lambda, \quad \tau_1, \tau_2 \in [0, 1], \ \omega \in [0, 2\pi], \tag{4}$$

is more convenient.

The quantities such as γ_k and γ_k^U, and their spectral representations (1)–(4) naturally come into the picture when the *clipped processes* $(I\{X_t \leq q\})_{t \in \mathbb{Z}}$ and $(I\{F(X_t) \leq \tau\})_{t \in \mathbb{Z}}$ are investigated. Such binary processes have been considered earlier in the literature by, e.g., Kedem (1980). Observe that the quantile-based spectral quantities can be interpreted in terms of an orthogonal increment process of a spectral representation of the clipped process which exists for every strictly stationary process (more precisely, it suffices if the joint distributions of (X_t, X_{t-k}) depend only on k, but not on t); no assumptions about moments are necessary.

Recently, there has been a surge of interest in that type of concept, with the introduction, under the names of *Laplace-, quantile-* and *copula* spectral density and spectral density *kernels*, of various quantile-related spectral concepts, along with the corresponding sample-based periodograms and smoothed periodograms (cf. Li 2008, 2012, 2013, 2014; Hagemann 2013; Lee and Rao 2012; Kley 2014; Dette *et al.* 2015; Kley *et al.* 2016).

Despite the vast amount of theoretical work, a software solution was so far not publicly available.

1.2. Estimators for the quantile-based spectral analysis of time series

In this section, various estimators (the so-called quantile periodograms) for the Laplace and copula spectra defined in Section 1.1 are briefly considered. For the upcoming definitions denote by $X_0, \ldots, X_{n-1}$ an observed time series of the process $(X_t)_{t \in \mathbb{Z}}$, by

$$\hat{F}_n(x) := \frac{1}{n} \sum_{t=0}^{n-1} I\{X_t \leq x\}$$

the *empirical distribution function* of $X_0, \ldots, X_{n-1}$, and by $\Re z$ and $\Im z$ the real and imaginary part of $z = \Re z + i \Im z \in \mathbb{C}$, respectively. Further,

$$\rho_\tau(x) := x(\tau - I\{x \leq 0\}) = (1 - \tau)|x| I\{x \leq 0\} + \tau |x| I\{x > 0\},$$

denotes the so-called *check function* (cf. Koenker 2005).

Definition 1 (Quantile-regression based periodograms)
For $\omega \neq 0 \mod 2\pi$ and $\tau_1, \tau_2 \in (0,1)$ the Laplace periodogram $\hat{L}_n^{\tau_1,\tau_2}(\omega)$, *and the* rank-based Laplace periodogram $\hat{L}_{n,R}^{\tau_1,\tau_2}(\omega)$ *are defined as*

$$\hat{L}_n^{\tau_1,\tau_2}(\omega) := (2\pi n)^{-1}\hat{b}_n^{\tau_1}(\omega)\hat{b}_n^{\tau_2}(-\omega), \quad \hat{L}_{n,R}^{\tau_1,\tau_2}(\omega) := (2\pi n)^{-1}\hat{b}_{n,R}^{\tau_1}(\omega)\hat{b}_{n,R}^{\tau_2}(-\omega),$$

where, for $\omega \neq 0 \mod \pi$, and $\tau \in (0,1)$,

$$(\hat{a}_n^{\tau}(\omega), \hat{b}_n^{\tau}(\omega)) := \operatorname{argmin}_{a \in \mathbb{R}, b \in \mathbb{C}} \sum_{t=0}^{n-1} \rho_\tau(nX_t - a - 2\cos(\omega t)\Re b + 2\sin(\omega t)\Im b), \qquad (5)$$

$$(\hat{a}_{n,R}^{\tau}(\omega), \hat{b}_{n,R}^{\tau}(\omega)) := \operatorname{argmin}_{a \in \mathbb{R}, b \in \mathbb{C}} \sum_{t=0}^{n-1} \rho_\tau(n\hat{F}_n(X_t) - a - 2\cos(\omega t)\Re b + 2\sin(\omega t)\Im b).$$

and for $\omega_\pi = 2\pi(j + 1/2)$, $j \in \mathbb{Z}$, and $\tau \in (0,1)$,

$$(\hat{a}_n^{\tau}(\omega_\pi), \hat{b}_n^{\tau}(\omega_\pi)) := \operatorname{argmin}_{a \in \mathbb{R}, b \in \mathbb{R}} \sum_{t=0}^{n-1} \rho_\tau(nX_t - a - \cos(\omega_\pi t)b),$$

$$(\hat{a}_{n,R}^{\tau}(\omega_\pi), \hat{b}_{n,R}^{\tau}(\omega_\pi)) := \operatorname{argmin}_{a \in \mathbb{R}, b \in \mathbb{R}} \sum_{t=0}^{n-1} \rho_\tau(n\hat{F}_n(X_t) - a - \cos(\omega_\pi t)b).$$

Note that, for $\omega = 0 \mod \pi$, the estimates need to be adapted, because the regressor that yields the imaginary part of the estimate vanishes. For $\omega \in 2\pi\mathbb{Z}$ an adaptation is also possible (cf. Kley 2014), but since it is not required for the definition of the smoothed estimates this is omitted here, for the sake of brevity. Observe that the rank-based periodograms obtained their name due to the fact that $n\hat{F}_n(X_t)$ is the rank of X_t among $X_0, \ldots, X_{n-1}$

The Laplace periodograms can be traced back to Katkovnik (1998), who, in the field of signal processing, suggested L_p estimators in a harmonic linear model. Li (2008) proved asymptotic normality of the Laplace periodograms for $\tau_1 = \tau_2 = 0.5$ and later extended the approach to arbitrary quantiles with $0 < \tau_1 = \tau_2 < 1$ (Li 2012). Dette *et al.* (2015) introduced the estimator with distinct quantile levels τ_1 and τ_2 (not necessarily equal), and also considered the rank-based version.

Another estimator is based on the discrete Fourier transform of clipped processes and can be defined as follows:

Definition 2 (Periodograms based on clipped time series)
For $\omega \in \mathbb{R}$ and $q_1, q_2 \in \mathbb{R}$, the clipped time series periodogram *is defined as*

$$I_n^{q_1,q_2}(\omega) := (2\pi n)^{-1}d_n^{q_1}(\omega)d_n^{q_2}(-\omega), \quad d_n^q(\omega) := \sum_{t=0}^{n-1} I\{X_t \leq q\}e^{-i\omega t}.$$

For $\omega \in \mathbb{R}$ and $\tau_1, \tau_2 \in [0,1]$ the copula rank periodogram *(for short CR periodogram) is defined as*

$$I_{n,R}^{\tau_1;\tau_2}(\omega) := (2\pi n)^{-1}d_{n,R}^{\tau_1}(\omega)d_{n,R}^{\tau_2}(-\omega), \quad d_{n,R}^{\tau}(\omega) := \sum_{t=0}^{n-1} I\{\hat{F}_n(X_t) \leq \tau\}e^{-i\omega t}.$$

Note the similarity between all the quantile periodograms and the cross-periodograms in multivariate time series analysis (cf., e.g., Brillinger 1975, p. 235): Each periodogram is a product of two frequency representation objects computed at frequencies (ω and $-\omega$) that sum to zero.

The estimator based on the discrete Fourier transformation of clipped time series was introduced by Hong (2000), who used it for a test of pairwise independence. Hagemann (2013) analyzed the special case of $\tau_1 = \tau_2$ in the presence of serial dependence. The case of distinct quantile levels τ_1 and τ_2 (not necessarily equal) was discussed in Kley *et al.* (2016), where weak convergence to a Gaussian process was established. Lee and Rao (2012) investigated the distributions of Cramér-von Mises type statistics, based on empirical joint distributions.

As in the traditional case the new periodograms are not consistent estimators (cf. the positive variances of the limit distributions in Theorems 3.2 and 3.4 in Dette *et al.* 2015 or Proposition 3.4 in Kley *et al.* 2016).

1.3. Smoothing the quantile periodograms

To achieve consistency of the estimators we convolve the sequence of periodograms (indexed with the Fourier frequencies) with a sequence of weighting functions W_n. The smoothed periodograms are defined as follows:

Definition 3 (Smoothed quantile-regression based periodograms)
For $\omega \in \mathbb{R}$ and $\tau_1, \tau_2 \in (0,1)$ the smoothed Laplace periodogram $\hat{\mathfrak{f}}_n(\tau_1, \tau_2; \omega)$ *and* smoothed rank-based Laplace periodogram $\hat{\mathfrak{f}}_{n,R}(\tau_1, \tau_2; \omega)$ *are defined as*

$$\hat{\mathfrak{f}}_n(\tau_1, \tau_2; \omega) := \frac{2\pi}{n} \sum_{s=1}^{n-1} W_n(\omega - 2\pi s/n) \hat{L}_n^{\tau_1, \tau_2}(2\pi s/n),$$

$$\hat{\mathfrak{f}}_{n,R}(\tau_1, \tau_2; \omega) := \frac{2\pi}{n} \sum_{s=1}^{n-1} W_n(\omega - 2\pi s/n) \hat{L}_{n,R}^{\tau_1, \tau_2}(2\pi s/n).$$

Definition 4 (Smoothed periodograms based on clipped time series)
For $\omega \in \mathbb{R}$ and $q_1, q_2 \in \mathbb{R}$ the smoothed clipped time series periodogram *is defined as*

$$\hat{G}_n(q_1, q_2; \omega) := \frac{2\pi}{n} \sum_{s=1}^{n-1} W_n(\omega - 2\pi s/n) I_n^{q_1, q_2}(2\pi s/n).$$

For $\omega \in \mathbb{R}$ and $\tau_1, \tau_2 \in [0,1]$ the smoothed copula rank periodogram *is defined as*

$$\hat{G}_{n,R}(\tau_1, \tau_2; \omega) := \frac{2\pi}{n} \sum_{s=1}^{n-1} W_n(\omega - 2\pi s/n) I_{n,R}^{\tau_1; \tau_2}(2\pi s/n).$$

When the weight functions are such that with $n \to \infty$ only the weights in a shrinking neighborhood of zero will be positive, the estimators will be consistent (cf. Theorem 3.7 in Kley 2014). Under suitable assumptions, scaled versions of $\hat{G}_n(\cdot, \cdot; \omega)$ and $\hat{G}_{n,R}(\cdot, \cdot; \omega)$ converge weakly to complex-valued Gaussian processes (cf. Theorem 3.5 and 3.6 in Kley *et al.* 2016). A comprehensive description of all estimators and their asymptotic properties is available in Kley (2014).

2. Conceptual design of the framework

2.1. An analysis of functional requirements

The **quantspec** software project was triggered by the development of the quantile-based methods for spectral analysis (cf. Section 1, Dette *et al.* 2015 and Kley *et al.* 2016). The primary aim has been to make these new methods accessible to a wide range of users.

Before going into the programming-specific details of the project, a conceptual design, non-specific to any programming language, was developed. By this procedure, additional insight and a thorough documentation of the computational characteristics of quantile-based spectral analysis could be gained. The conceptual design serves as a blueprint for implementations in (possibly) various environments and can easily be transformed into an implementation plan including the details specific to the respective programming environment.

Aiming for a software system that is most flexible in the ways in which it can be used, that can easily be extended in functionality and also for the ease of its maintenance an object-oriented design was chosen. This type of design also contributes to a structure of the system that can be better understood, both by users and developers. The general structure of the system for performing quantile-based spectral analysis is described using class diagrams of the unified modeling language (UML). In these diagrams, all necessary components and their interrelations are described in a formal manner.

To understand the specification of the framework, the essential elements of the UML are described very briefly. Readers familiar with UML can skip this paragraph. Recall that in an object-oriented design the components of the system are objects encapsulating both data and behavior of a specific "real-world" entity. The structure of each object can thus be described by a meaningful name (the *class name*), a collection of data fields (in R these are called *slots*) and implementations of the behavior (in R these implementations are called *methods*). In a class diagram each class (i.e., the composite of class name, data files and implemented behavior) is represented as a rectangle subdivided into three blocks. The name of the class is given in the top block, the data fields in the middle block and the methods in the bottom block. Note that in the unified modeling language the data fields and methods are specified in a standardized format. In this format the first symbol is an abbreviation used to specify the visibility of the class member. Here the symbols "+" for public and "−" for private members are used, indicating that the member is intended to be seen (and used) from outside the object or from inside the object only, respectively. For a data field the name is then followed by a colon and the type of the field. For a method the parameters are given in parenthesis; optional parameters are denoted by two dots. In the class diagram, relationships between classes are marked as lines connecting them. Currently two different types of relationships are modeled. A line with a triangular shaped tip at one end is used to declare a *generalization* relationship (sometimes also coined inheritance or "is a" relationship). The class at the end of the line with the triangle is called the *superclass* or the parent, while the class on the other end is called the *subclass* or the child. In particular this type of relationship implies that an object that is an instance to the subclass and therefore provides all the subclasses' data fields and methods will also provide the data fields and methods of the superclass (and the superclasses' superclass if there are such, etc.). The second type of relationship used in this framework is that of an *aggregation* (sometimes called "has a" relationship). A line with a hollow diamond at one end is used to denote this kind of relationship, where objects to the class at the end

of the line with the diamond are the ones having objects of the class at the other end of the line as a part of them. At each end of the line the so-called cardinalities are denoted in the form of two numbers with dots in between them. The left number is the minimum number of objects of that type in the relationship that need to exist, the right number is the maximum number. A star is used to denote an unknown positive integer. If min and max cardinality coincide they are displayed as one number without the dots.

For the class diagrams in this manuscript the classes are arranged in a way such that (whenever possible) generalization relationships are displayed with the superclass on top and the subclasses in the bottom. Aggregations are shown with aggregated classes to the left and/or the right of the class representing "the whole".

A graphical representation of the framework for quantile-based spectral analysis is not given in one holistic diagram, but in two class diagrams that are on display in Figures 1 and 2. The structure of the framework is presented in two, thematically organized class diagrams rather than in one, because the 13 classes and their relations could not be fitted easily onto one page without breaking the above mentioned layout guidelines. On the other hand it was easy to group the classes by two topics.

In the next sections, all classes of the framework and their relations are going to be thoroughly described and motivated.

2.2. The base class 'QSpecQuantity' and its successors

Many of the quantities important for the quantile-based spectral analysis of a stationary time series (i.e., the estimators of Definitions 1–4 and the model quantities (1)–(4)) are of the functional form,

$$Q_b : F \times T_1 \times T_2 \to \mathbb{C}, \quad b = 1, \ldots, B,$$

where $F \subset \mathbb{R}$ is a set of frequencies (e.g., $F = [0, 2\pi)$) and $T_1, T_2 \subset \bar{\mathbb{R}}$ are sets of levels. To provide a common interface to these objects the abstract class 'QSpecQuantity' was introduced. Its data fields (i.e., an array `values`, a vector of reals `frequencies` and a list with two vectors of reals `levels`), are designed to store the sets

$$\text{frequencies} := \{\omega_1, \ldots, \omega_J\} \subset F,$$
$$\text{levels[[1]]} := \{q_{1,1}, \ldots, q_{1,K_1}\} \subset T_1,$$
$$\text{levels[[2]]} := \{q_{2,1}, \ldots, q_{2,K_2}\} \subset T_2,$$

and the family

$$\text{values} := \left(Q_b(\omega_j, q_{1,k_1}, q_{2,k_2})\right)_{j=1,\ldots,J; k_1=1,\ldots,K_1; k_2=1,\ldots,K_2; b=1,\ldots,B}.$$

Note that the handling of a family of B quantile spectral quantities is necessary when bootstrapping replicates are present. The special case of only one function Q can easily be handled by setting $B = 1$.

There are four classes inheriting the data structure and the method `show`[1] from the abstract class 'QSpecQuantity'. Two such classes, 'QuantilePG' and 'SmoothedPG', implement the computation of the various quantile periodograms and smoothed quantile periodograms, respectively. A more detailed description has to include the other related classes and is

---

[1]The function `show` is used for printing objects of this class, and all superclasses, to the console.
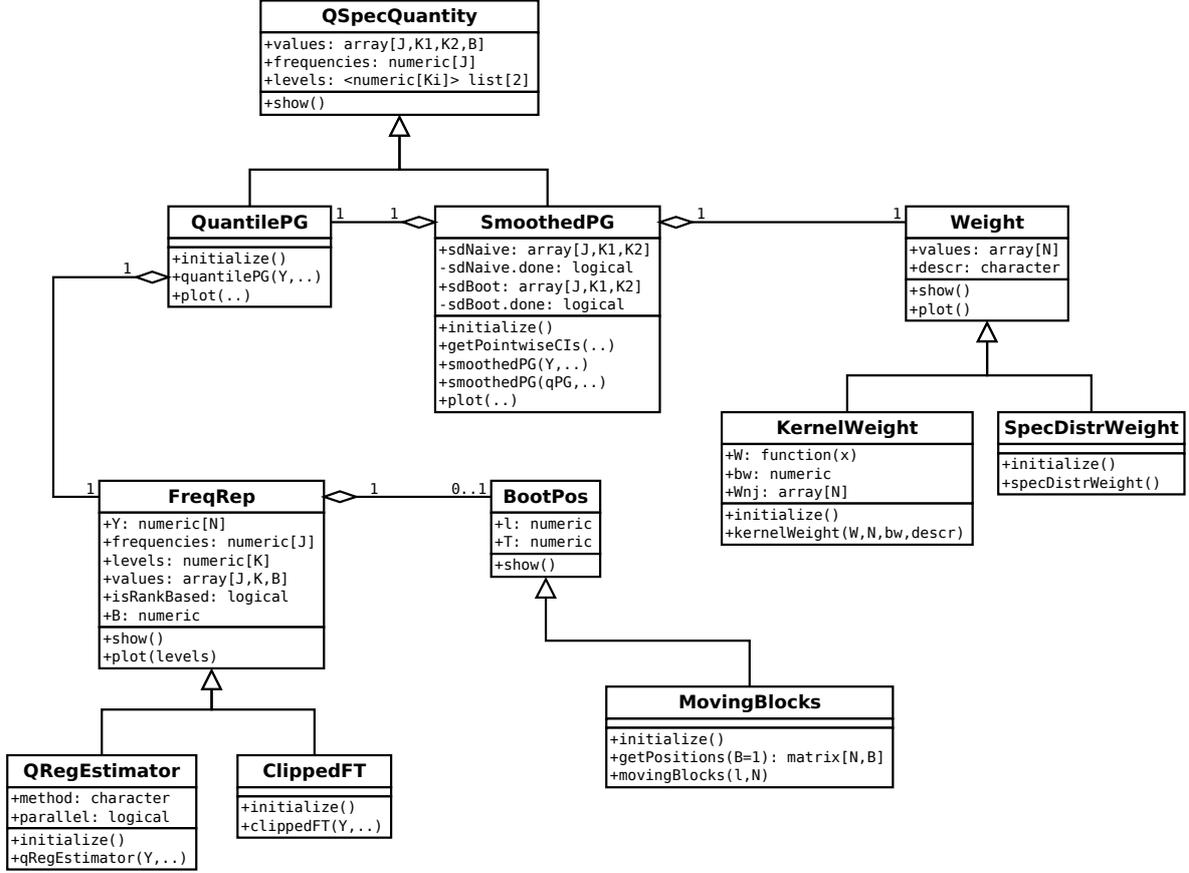
Figure 1: Classes implementing the quantile-based periodograms and smoothed periodograms.

therefore deferred to a separate section (i.e., Section 2.3). A graphical representation of the relevant parts of the framework can be seen in Figure 1. The other two of the classes generalizing the abstract class 'QSpecQuantity' are referred to by the names 'QuantileSD' and 'IntegrQuantileSD'. These two classes implement the model quantities (1)–(4). The graphical representation can be seen in Figure 2. A detailed description is deferred to Section 2.4.

2.3. Implementation of the quantile-based (smoothed) periodograms

The components relevant to the implementation of the quantile-based spectral statistics are presented in Figure 1. As alluded to in the previous section the two classes 'QuantilePG' and 'SmoothedPG' will do the job, in conjunction with the superclass 'QSpecQuantity' from which they inherit the data structure to store the computed values.

To better understand the implementation surrounding 'QuantilePG', observe that the quantile-based periodograms defined in Definitions 1 and 2 share the common structure of an outer product (scaled with $(2\pi n)^{-1}$). To compute either one of the four periodograms

$$\hat{L}_{n,R}^{\tau_1,\tau_2}(\omega), \ \hat{L}_n^{\tau_1,\tau_2}(\omega), \ I_{n,R}^{\tau_1,\tau_2}(\omega), \quad \tau_1 \in T_1, \ \tau_2 \in T_2, \ \omega \in F,$$
$$I_n^{q_1,q_2}(\omega) \quad q_1 \in Q_1, \ q_2 \in Q_2, \ \omega \in F,$$

it suffices to perform the same operation to one of the frequency representation objects

$$\hat{b}_{n,R}^{\tau}(\omega),\ \hat{b}_n^{\tau}(\omega),\ d_{n,R}^{\tau}(\omega),\quad \tau \in T_1 \cup T_2,\ \omega \in F,$$
$$d_n^q(\omega)\quad q \in Q_1 \cup Q_2,\ \omega \in F, \tag{6}$$

respectively. In the framework this fact is incorporated by introducing the abstract class 'FreqRep' and its two subclasses 'ClippedFT' and 'QRegEstimator', where the actual computations are implemented via the method initialization(). The class 'FreqRep' serves as a common interface to the quantities in (6). It provides data fields to store various information, including

- the observations Y from which the quantities were computed,

- the frequencies and levels for which the computation was performed,

- the result of the computation, which is stored in an array values.

Further more, a flag isRankBased indicates whether, prior to the main computations, the observations (X_t) were transformed to pseudo-observations $(\hat{F}_n(X_t))$. Performing this extra step will yield $\hat{b}_{n,R}^{\tau}(\omega)$ instead of $\hat{b}_n^{\tau}(\omega)$ or $d_{n,R}^{\tau}(\omega)$ instead of $d_n^{\tau}(\omega)$, respectively. The class 'BootPos' allows to perform a block bootstrap procedure by "shuffling" the observations and repeatedly doing the computations on these bootstrapped observations. Currently only one method, the 'MovingBlocks' bootstrap, is implemented.

Now, turning attention to the class 'SmoothedPG', recall that the various smoothed periodograms are all defined similarly, in the sense that computing the smoothed periodogram for $\omega_j := 2\pi j/n,\ j = 1, \ldots, n-1$ basically means to do a discrete convolution of the sequence of quantile periodograms computed at ω_s with a sequence of appropriately chosen weight functions $W_n(\omega_s)$. Hence, everything needed for the smoothed periodogram is these two ingredients, which is reflected in the framework by two aggregation relationships involving the class 'SmoothedPG'. The first such relationship links 'SmoothedPG' to the 'QuantilePG' to be smoothed. The second such relationship links 'SmoothedPG' to a class 'Weight', which provides a common interface to different weight functions. Currently two implementations are included. Employing weights of type 'KernelWeight', defined by a kernel W and a scale parameter bw (bandwidth), will yield an estimator for the quantile (i.e., Laplace or copula) spectral density. An alternative is to use weights of type 'SpecDistrWeight', which yields estimators for the integrated quantile (i.e., Laplace or copula) spectral density.

2.4. Implementation of the quantile-based spectral measures

The classes 'QuantileSD' and 'IntegrQuantileSD' were introduced to the framework to make the quantities (1)–(4) available to the user.

To obtain access to a quantity of the form (1) or (2), an instance of 'QuantileSD' can be created. In this case, R independent copies of a time series of length N are obtained by calling the function ts. The function ts is a parameter to specify the model for which to obtain the model quantity and it handles the simulation process. Then, for each of these time series a 'QuantilePG' object is created and their values are averaged: first across the R independent copies, saving the result to meanPG and an estimation of the standard error to stdError. After that the averages are averaged again for each combination of levels across
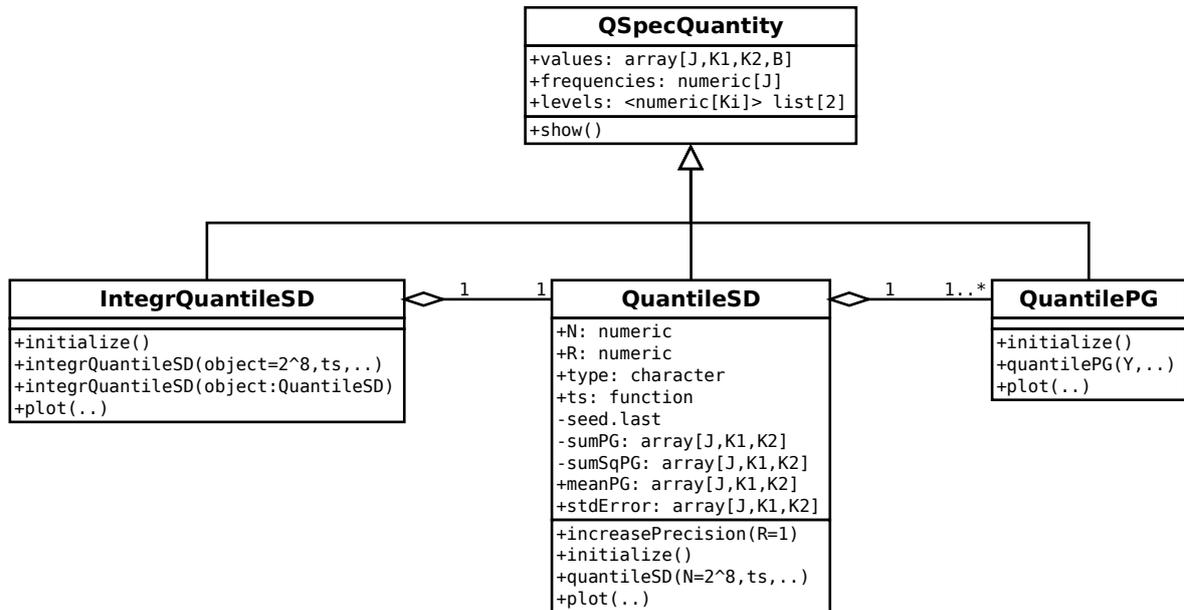
Figure 2: Classes for the numerical computation of (integrated) copula and Laplace spectral densities via simulation.

frequencies by smoothing. The result is then made available via the data field `values` of the superclass. By a call to the method `increasePrecision` the number of independent copies can be increased at any time to yield a less fluctuating average.

To obtain access to quantities of the form (3) or (4), an instance of '`IntegrQuantileSD`' can be created. For the computation an object of type '`QuantileSD`' is created and subsequently the integral is approximated via a Riemann sum.

3. Reference implementation: The R package quantspec

3.1. Overview

The **quantspec** package (Kley 2016) is intended to be used both by theoretically oriented statisticians and also by data analysts, who work on a more applied basis. In order to address this broad group of potential users the R system for statistical computing (R Core Team 2016) was chosen as a platform. The R system is particularly well suited for the realization of this project, because it is accessible from many operating systems, without charge, already available to the targeted audience and, in particular, allows to integrate the package's functionality among many other, well developed packages. An important example is that the function `rq` of the **quantreg** package (Koenker 2016) could be used.

Both R and the **quantspec** package are available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=quantspec`). The package's development is actively continued with the source code available from a GitHub repository (`https://github.com/tobiaskley/quantspec`). Besides the source code of the releases, which are also available from the CRAN servers, the GitHub repository additionally contains

a detailed history of all changes, including comments, that were applied to the source code since 2014-04-10, when the GitHub repository was created. The repository is organized into several branches: the `master` branch, the `develop` branch and possibly several topic branches. Since version 1.0-0 the `master` branch contains the source code of all release candidates and official releases, the `develop` branch contains the most recent updates and bug fixes that were not yet released. The topic branches contain the source code in which extensions to the package are developed.

To install a package from the source code straight from the repository use the `install_github` function of the **devtools** package (Wickham and Chang 2016). More precisely, install the **devtools** package, if it is not already installed, and call

```
R> library("devtools")
R> install_github("tobiaskley/quantspec", ref = "master")
```

Instead of using `"master"`, another branch (e.g., `"develop"` to pull the most recent updates) or a tag that is pointing to one of the releases (e.g., `"v1.0-0-rc1"`, to install the 1st release candidate to version 1.0-0) can be used as `ref`. Note that, in case you are using Windows, to use `install_github`, you may need to install the **Rtools**,[2] if you have not already done so. Note that the code in the `develop` branch is merged into the `master` branch only for a release (candidate) and after being thoroughly tested, so if you are installing from the `develop` branch you will potentially be using code that has not been fully tested. Use the optional argument `build_vignettes = FALSE` if you do not have LaTeX installed on your system; i.e., call

```
R> library("devtools")
R> install_github("tobiaskley/quantspec", ref = "master",
+    build_vignettes = FALSE)
```

3.2. R code intended for the user and its documentation

The classes of the **quantspec** package, their methods, slots, dependencies and inheritance properties are implemented as conceptually designed (cf. Section 2). Recall that the design was presented in form of class diagrams, on display in Figures 1 and 2, and that no specific programming language was assumed. All classes that are intended for the end-user possess a constructor method with the same name as the class itself but beginning with a lower case letter. The classes intended for the end-user and their constructors are listed in Table 1.

For a more detailed description of constructors and classes, documentation within the online help system of R is available. After loading the package, which is done by calling

```
R> library("quantspec")
```

the help file of the package, which provides an overview on the design, can be called by executing

```
R> help("quantspec")
```

---

[2]Available from https://CRAN.R-project.org/bin/windows/Rtools/.

Constructor	Type of object	Quantities computed
`clippedFT()`	'ClippedFT'	$d_n^q(\omega)$, $d_{n,R}^\tau(\omega)$
`qRegEstimator()`	'QRegEstimator'	$b_n^\tau(\omega)$, $b_{n,R}^\tau(\omega)$
`quantilePG()`	'QuantilePG'	$\hat{L}_n^{\tau_1,\tau_2}(\omega)$, $\hat{L}_{n,R}^{\tau_1,\tau_2}(\omega)$,
		$I_n^{q_1,q_2}(\omega)$, $I_{n,R}^{\tau_1,\tau_2}(\omega)$
`smoothedPG()`	'SmoothedPG'	$\hat{\mathfrak{f}}_n(\tau_1,\tau_2;\omega)$, $\hat{\mathfrak{f}}_{n,R}(\tau_1,\tau_2;\omega)$,
		$\hat{G}_n(q_1,q_2;\omega)$, $\hat{G}_{n,R}(\tau_1,\tau_2;\omega)$
`quantileSD()`	'QuantileSD'	$\mathfrak{f}_{q_1,q_2}(\omega)$, $\mathfrak{f}_{q_{\tau_1},q_{\tau_2}}(\omega)$
`integrQuantileSD()`	'IntegrQuantileSD'	$\mathfrak{F}_{q_1,q_2}(\omega)$, $\mathfrak{F}_{q_{\tau_1},q_{\tau_2}}(\omega)$
`kernelWeight()`	'KernelWeight'	$W_n(u) = b_n^{-1}\sum_j W(b_n^{-1}(u+2\pi j))$
`spectrDistrWeight()`	'spectrDistrWeight'	$W_n(u) = I\{u \leq 0\}$

Table 1: Constructors of the **quantspec** package that are intended for the end-user.

on the R command line. Note that an index of all available functions can be accessed at the very bottom of the page. If for example more information on the constructor of 'QRegEstimator' and on the class itself is desired, then `?qRegEstimator` or `?QRegEstimator` should be invoked to access the corresponding help pages. Using this class to determine the frequency representation $b_{n,R}^\tau(\omega)$, for $\tau \in \{0.25, 0.5, 0.75\}$ would look as follows. In a toy example, where eight independent random variables $X_0, \ldots, X_7 \sim N(0,1)$ are generated and used to compute $b_{n,R}^\tau(\omega)$, call

```
R> Y <- rnorm(8)
R> bn <- qRegEstimator(Y, levels = c(0.25, 0.5, 0.75))
```

By default the computation is done for all Fourier frequencies $\omega = 2\pi j/n \in [0,\pi]$, $n = 8$, $j = 0, \ldots, \lfloor n/2 \rfloor$. The computed information can then be viewed by typing the name of the variable (i.e., `bn`) in the R console:

```
R> bn

QRegEstimator (J = 5, K = 3, B+1 = 1)
Frequencies:  0 0.7854 1.5708 2.3562 3.1416
Levels     :  0.25 0.5 0.75

Values:
        tau = 0.25     tau = 0.5     tau = 0.75
0       2.000+0.000i   4.000+0.00i   6.000+0.000i
0.785  -0.354+0.354i  -0.811-0.25i  -0.207+1.207i
1.571   2.750+0.250i   0.250+0.75i   0.750-0.250i
2.356   0.354-0.439i   1.414+0.00i   1.207+0.207i
3.142   1.000+0.000i   1.000+0.00i   0.500+0.000i
```

Methods other than the constructor are implemented as generic functions. To invoke the method f of an object 'obj' the call therefore is `f(obj)`. In particular all attributes mentioned in the class diagram can be accessed via getter methods. There are no setter methods, because all attributes are completely handled by internal functions. For an example, to retrieve the

attributes `frequencies` and `parallel` of the object `bn`, execute the following lines on the R shell

```
R> getFrequencies(bn)
```

```
[1] 0.0000000 0.7853982 1.5707963 2.3561945 3.1415927
```

```
R> getParallel(bn)
```

```
[1] FALSE
```

To invoke a method `f` with parameters `p1, ..., pk` of an object 'obj' the call is `f(obj, p1, ..., pk)`. An example is to invoke the accessor function `getValues`, which is equipped with parameters to get the values associated with certain `frequencies` or `levels`. An exemplary call looks like this:

```
R> getValues(bn, levels = c(0.25, 0.5))
```

```
, , 1

                   [,1]            [,2]
[1,]   2.0000000+0.0000000i   4.0000000+0.00i
[2,]  -0.3535534+0.3535534i  -0.8106602-0.25i
[3,]   2.7500000+0.2500000i   0.2500000+0.75i
[4,]   0.3535534-0.4393398i   1.4142136+0.00i
[5,]   1.0000000+0.0000000i   1.0000000+0.00i
[6,]   0.3535534+0.4393398i   1.4142136-0.00i
[7,]   2.7500000-0.2500000i   0.2500000-0.75i
[8,]  -0.3535534-0.3535534i  -0.8106602+0.25i
```

Note that the result is returned as an array of dimension `c(J, K, B + 1)`, where in the present case `B = 0` bootstrap replications were performed. For a detailed description on how to use the function `getValues` in the above mentioned case, access the online help via the command

```
R> help("getValues-FreqRep")
```

Note the format `method_name-class_name` to access the help page of a method and that the attribute `values` is part of the abstract class 'FreqRep' (cf. Figure 2).

A graphical representation of the data can easily be created by applying the `plot` command. For example, to compute and plot the frequency representations $d^\tau_{32,R}(\omega)$, from 32 simulated, standard normally distributed random variables execute the following lines on the R shell:

```
R> dn <- clippedFT(rnorm(32), levels = seq(0.05, 0.95, 0.05))
R> plot(dn, frequencies = 2 * pi * (0:64) / 32, levels = c(0.25, 0.5))
```
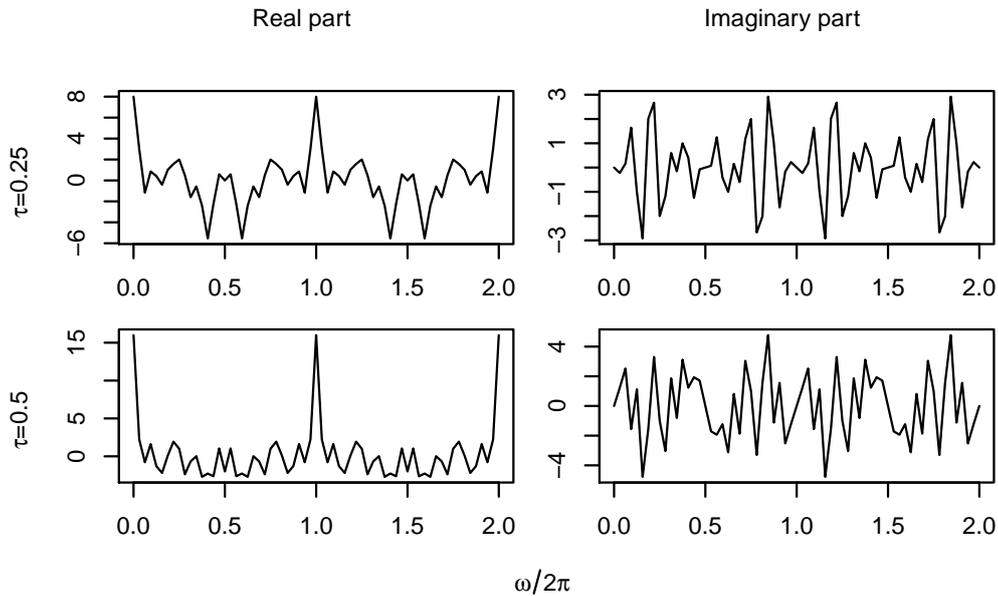
Figure 3: Plot of the 'FrequencyRepresentation' object bn.

The above script will yield the diagrams that are on display in Figure 3. Note that the $d_{32,R}^{\tau}(\omega)$ were determined for $\tau \in \{0.05, 0.1, \ldots, 0.9, 0.95\}$ and, by the default setting, for all Fourier frequencies from $[0, \pi]$. The plot, however, was parametrized to show only $\tau \in \{0.25, 0.5\}$, but all Fourier frequencies from $[0, 4\pi]$; by default all available levels and frequencies would be used. In this example two of the 19 frequencies were selected to yield a plot of a size that is appropriate to fit onto the page. Furthermore, the plot was parametrized to show $d_{n,R}(\omega)$ for all Fourier frequencies from $[0, 4\pi]$ to illustrate characteristic redundancies in the frequency representation objects, and to point out that the default values are always sufficient. The two relations

$$d_{n,R}^{\tau}(\omega) = \overline{d_{n,R}^{\tau}(2\pi - \omega)}, \quad d_{n,R}^{\tau}(\omega) = d_{n,R}^{\tau}(\omega + 2\pi j),$$

hold for any $\omega \in \mathbb{R}$ and $j \in \mathbb{Z}$, $d_{n,R}^{\tau}(\omega)$. Therefore, without additional calculations, the plot of $d_{n,R}^{\tau}(\omega)$ can be determined for any $\omega \in 2\pi j/n$, $j \in \mathbb{Z}$, as long as $d_{n,R}^{\tau}(\omega)$ is known for $\omega \in 2\pi j/n$, $j = 0, \ldots, \lfloor n/2 \rfloor$, which is what is determined by the default setting. Note that all of this happens transparently for the user, as the method getValues takes care of it. Another fact that one can presume by inspecting Figure 3 is that $d_{n,R}^{\tau}(\omega)$ appears to be uncorrelated and centered (for $\omega \neq 0 \mod 2\pi$).

3.3. Additional elements of the package

The **quantspec** package includes three demos that can be accessed via

```
R> demo("sp500", package = "quantspec")
R> demo("wheatprices", package = "quantspec")
R> demo("qar-simulation", package = "quantspec")
```

Several examples explaining how to use the various functions of the package can be found in the online help files or the folder examples in the directory where the package is installed.

The package comes with two data sets `sp500` and `wheatprices` that are used in the demos and in the examples. A package vignette amends the online help files. It contains the text of this paper. Unit tests covering all main functions were implemented using the **testthat** framework (Wickham 2011).

4. Two worked examples

4.1. Analysis of the S&P 500 stock index, 2007–2010

In this section the use of the **quantspec** package from the perspective of a data analyst is explained. To this end an analysis of the returns of the S&P 500 stock index is performed. Note that a similar analysis as well as the data set used are available in the package. Calling `demo("sp500", package = "quantspec")` will start the computations and by `sp500` the data set can be referenced to do additional analysis.

For the example the years 2007 through to 2010 were selected to have a time series that, at least to some degree, can be considered stationary. Aside from this more technical consideration, employing the new statistical toolbox will reveal interesting features in the returns collected in the financial crisis that completely escape the analysis with the traditional tools blindly applied.

For a start, use the following R script to plot the data using the **zoo** package (Zeileis and Grothendieck 2005), the autocovariances of the returns and the autocovariances of the squared returns.

```
R> library("zoo")
R> plot(sp500, xlab = "time t", ylab = "", main = "")
R> acf(coredata(sp500), xlab = "lag k", ylab = "", main = "")
R> acf(coredata(sp500)^2, xlab = "lag k", ylab = "", main = "")
```

The three plots are displayed in Figure 4. Inspecting them, it is important to observe that the returns themselves appear to be almost uncorrelated. Therefore, not much insight into the serial dependency structure of the data can be expected from traditional spectral analysis.
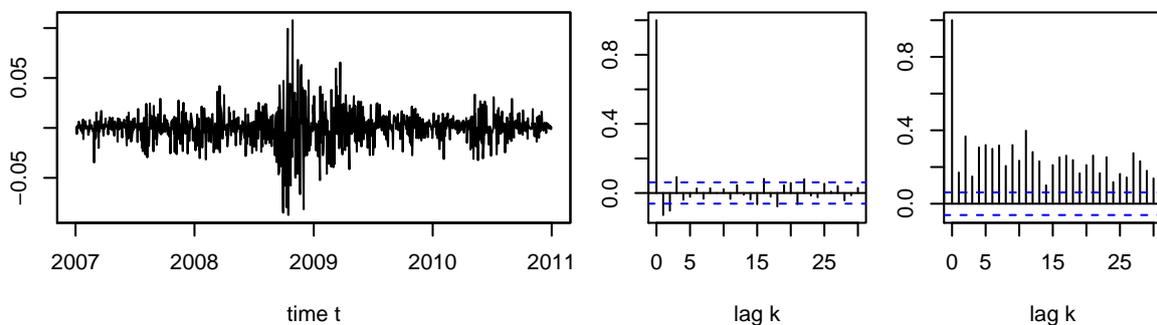

Figure 4: Returns (Y_t) of the S&P 500 returns example data (left), autocovariances $\mathsf{COV}(Y_{t+k}, Y_t)$ of the returns (middle), and autocovariances $\mathsf{COV}(Y_{t+k}^2, Y_t^2)$ of the squared returns (right).
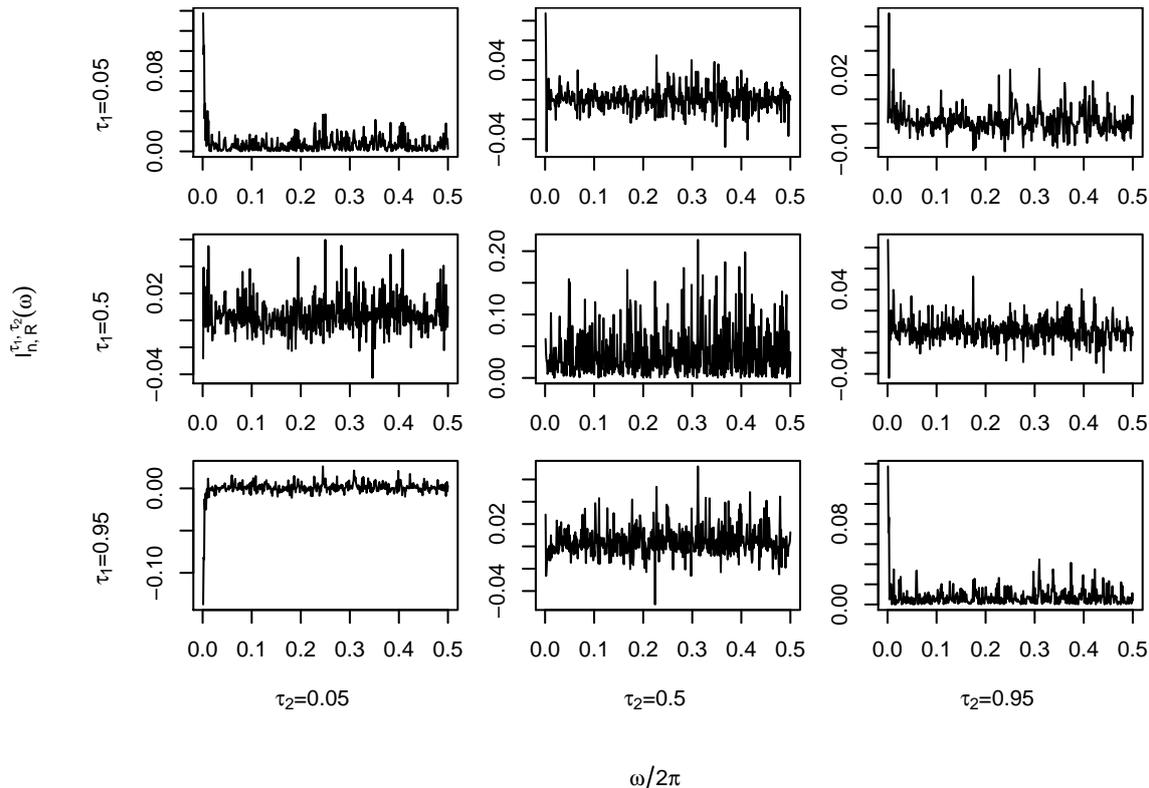
Figure 5: Plot of the 'QuantilePG' object CR, computed from the sp500 time series; $\omega \in (0, \pi]$.

The squared returns on the other hand are significantly correlated. This observation, typically taken as an argument to fit an ARCH or GARCH model, clearly proves that serial dependency exists. In what follows the copula spectral density will be estimated from the data, using quantile periodograms and smoothing them. It will be seen that using the **quantspec** package this can be done requiring only a few lines of code.

First, take a look at the CR periodogram $I_{n,R}^{\tau_1, \tau_2}(\omega)$. In the **quantspec** package it is represented as a 'QuantilePG' object and can be computed calling the constructor function quantilePG() with the parameter type = "clipped". To do the calculation for $\tau_1, \tau_2 \in \{0.05, 0.5, 0.95\}$, all Fourier frequencies ω and with 250 bootstrap replications determined from a moving blocks bootstrap with block length $\ell = 32$, it suffices to execute the first command of the following script:

```
R> CR <- quantilePG(sp500, levels.1 = c(0.05, 0.5, 0.95),
+     type = "clipped", type.boot = "mbb", B = 250, l = 32)
R> freq <- getFrequencies(CR)
R> plot(CR, levels = c(0.05, 0.5, 0.95),
+     frequencies = freq[freq > 0 & freq <= pi],
+     ylab = expression({I[list(n, R)]^{list(tau[1], tau[2])}}(omega)))
```

Using the second command it is possible to learn for which frequencies the values of the CR periodogram are available. As pointed out in the previous paragraph it was computed for all Fourier frequencies from the interval $[0, 2\pi)$, which is the default setting for quantilePG()
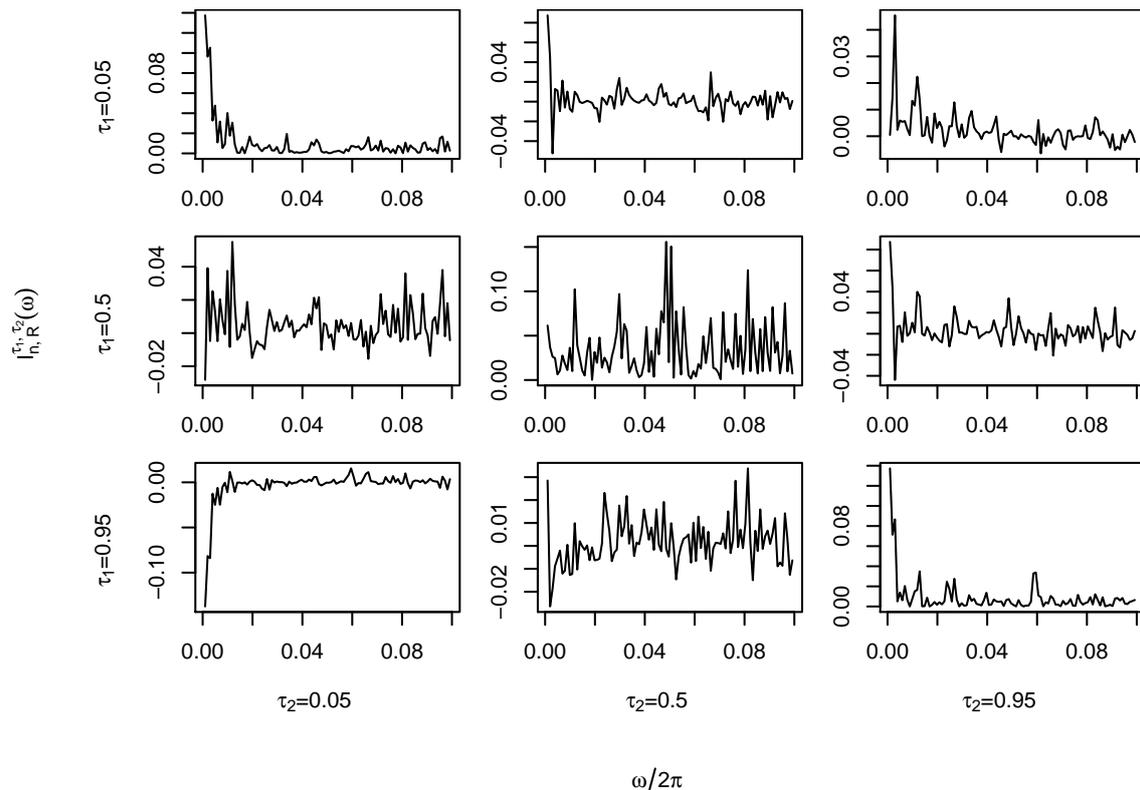
Figure 6: Plot of the 'QuantilePG' object CR, computed from the sp500 time series; $\omega \in (0, \pi/5]$.

and smoothedPG(). With the third command the graphical representation of the CR periodogram, which can be seen in Figure 5, is plotted. The plot seen here is a typical plot of any 'QSpecQuantity' object: In a configuration with K levels the plot has the form of a $K \times K$ matrix, where the subplots on and below the diagonal display the real part of the CR periodogram $I_{n,R}^{\tau_1, \tau_2}(\cdot)$, with the levels τ_1 and τ_2 denoted on the left and bottom margins of the plot. Above the diagonal the imaginary parts are shown.

To observe the larger values in the neighborhood of $\omega = 0$ and in the extreme quantile levels more closely a plot showing the CR periodogram only for frequencies $\omega \in [0, \pi/5]$ can be generated using the following script:

```
R> plot(CR, levels = c(0.05, 0.5, 0.95),
+    frequencies = freq[freq > 0 & freq <= pi/5],
+    ylab = expression({I[list(n, R)]^{list(tau[1], tau[2])}}(omega)))
```

The plot is shown in Figure 6.

In the next step the computed quantile periodogram CR can be used as the basis to determine a smoothed CR periodogram sCR. In the form of a 'SmoothedPG' object it can be generated by the constructor smoothedPG() of that class. Besides the 'QuantilePG' object CR, a 'KernelWeight' object is required, which is easily generated using the constructor kernelWeight(). As parameters the constructor kernelWeight() requires a kernel W and a bandwidth bw. The **quantspec** package comes with several kernels already implemented. The
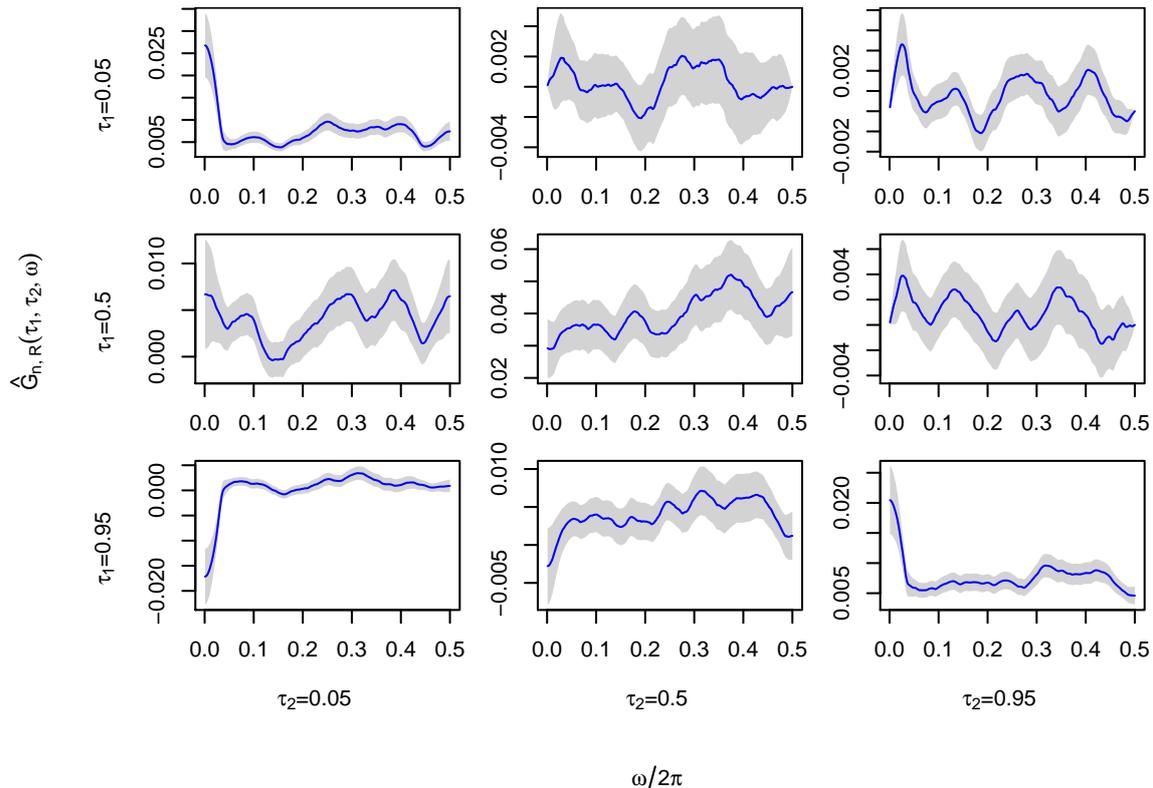
Figure 7: Plot of the 'SmoothedPG' object sCR, computed from the sp500 time series; type.scaling = "individual", ptw.CIs = 0.1.

Epanechnikov kernel for example can be referred to by the name W1.

To compute the smoothed CR periodogram from CR using the Epanechnikov kernel and bandwidth bw = 0.07 the first of the following two commands needs to be executed.

```
R> sPG <- smoothedPG(CR, weight = kernelWeight(W = W1, bw = 0.07))
R> plot(sPG, levels = c(0.05, 0.5, 0.95), type.scaling = "individual",
+    frequencies = freq[freq > 0 & freq <= pi], ptw.CIs = 0.1,
+    ylab = expression(hat(G)[list(n, R)](list(tau[1], tau[2], omega))))
```

Of course, the second line initiates plotting the smoothed CR periodogram, which is on display in Figure 7. Note that the option type.scaling can be set to yield a plot with certain subplots possessing the same scale. In Figure 7 pointwise confidence intervals are shown. By default these are determined using a normal approximation to the distribution of the estimator as is suggested by the limit theorem in Kley *et al.* (2016). An alternative is to use the quantiles of estimates computed from the block bootstrap replicates. These pointwise confidence intervals can be plotted using the option type.CIs = "boot.full", as is shown in the following script:

```
R> plot(sPG, levels = c(0.05, 0.5, 0.95), type.scaling = "real-imaginary",
+    ptw.CIs = 0.1, type.CIs = "boot.full",
+    frequencies = freq[freq > 0 & freq <= pi],
+    ylab = expression(hat(G)[list(n, R)](list(tau[1], tau[2], omega))))
```
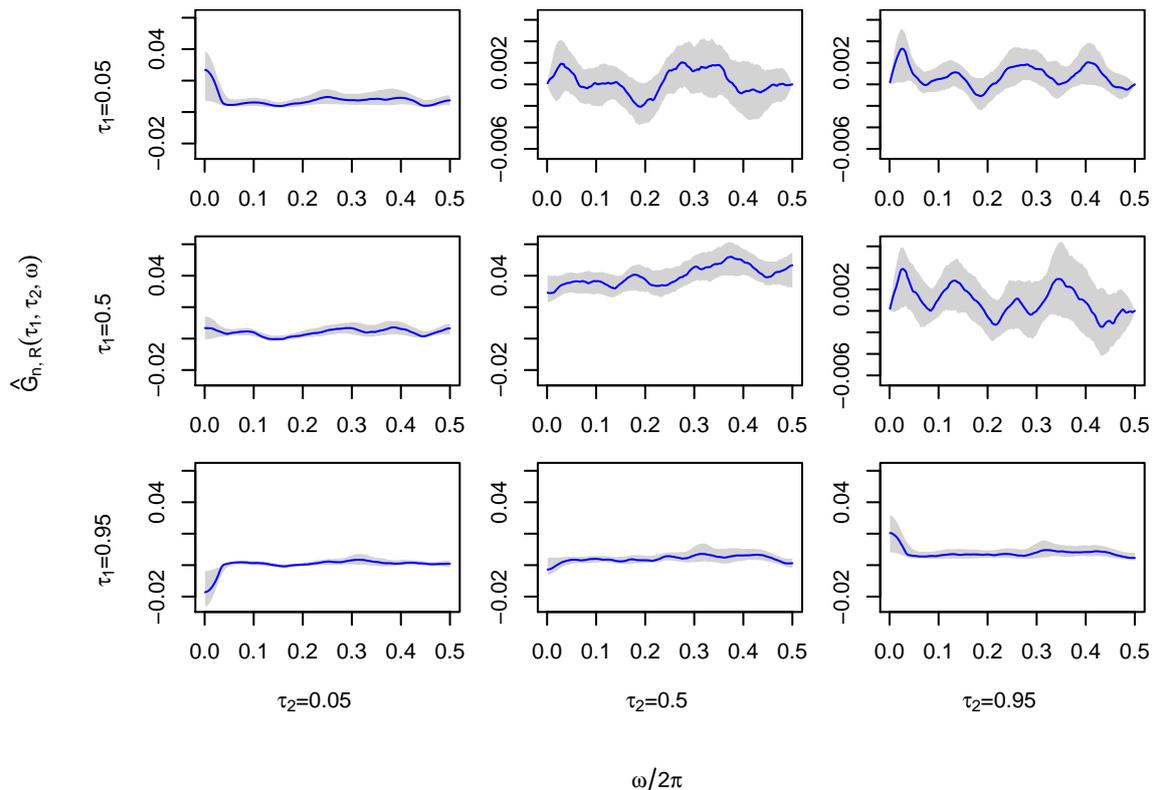
Figure 8: Plot of the 'SmoothedPG' object sCR, computed from the sp500 time series; type.scaling = "real-imaginary", ptw.CIs = 0.1, type.CIs = "boot.full".

For illustrative purposes a different type of scaling was used for the second plot. A complete description of the options is available in the online help, which can be accessed by calling

```
R> help("plot-SmoothedPG")
```

Inspecting the plots in Figures 5–8 reveals information about the dependency of the events $\{X_t \leq q_{0.05}\}$, $\{X_t \leq q_{0.5}\}$ and $\{X_t \leq q_{0.95}\}$ in the data. More precisely, the top left, middle and bottom right plots show the copula rank periodograms, in Figures 5 and 6, and smoothed copula rank periodograms, in Figures 7 and 8, that are estimates of the copula spectra $\mathfrak{f}_{q_{0.05},q_{0.05}}(\omega)$, $\mathfrak{f}_{q_{0.5},q_{0.5}}(\omega)$ and $\mathfrak{f}_{q_{0.95},q_{0.95}}(\omega)$, respectively. Each set of three plots summarizes the serial dependency structure of the three binary time series $I\{X_t \leq q_{0.05}\}$, $I\{X_t \leq q_{0.5}\}$ and $I\{X_t \leq q_{0.95}\}$ that indicate values of X_t below the 0.05-quantile, median and 0.95-quantile, respectively. The fact that, in the data example discussed here, these copula spectra are not flat can be seen as evidence that serial dependency is present; this cannot be seen from the autocorrelations. Comparing the two outer spectra allows for a comparison between the serial dependency structure in positive and negative extremes; this information completely gets lost when the observations are squared. In the analyzed S&P 500 data from 2007–2010 there seems to be only little difference between the serial dependency structure of the two types of extreme events. Note that this is not always the case. In the QAR model to be discussed in the next section, for example, the serial dependency structure of the positive and negative extremes is remarkably different (cf. Figures 9 and 10).

Additional information can be found in the off-diagonal plots. Take for example the bottom left and top right plots where the real and imaginary part of $\mathfrak{f}_{q_{0.05},q_{0.95}}(\omega)$ are shown, respectively. This quantity contains information on the *joint* serial dependence of the two binary time series $I\{X_t \leq q_{0.05}\}$ and $I\{X_t \leq q_{0.95}\}$. For example, if the imaginary part were zero for all frequencies this would correspond to a pairwise time reversibility in the sense that

$$\mathsf{P}(X_t \leq q_{0.05}, X_{t-k} \leq q_{0.95}) = \mathsf{P}(X_t \leq q_{0.05}, X_{t+k} \leq q_{0.95}),$$

for all k. This information cannot be recovered from autocovariances, because they are symmetric by definition. Further discussion of copula spectra can, for example, be found in Li (2008); Hagemann (2013); Li (2012); Dette *et al.* (2015); Kley *et al.* (2016); Birr, Volgushev, Kley, Dette, and Hallin (2014).

This concludes the introduction of the **quantspec** package for data analysts and we can continue with the presentation of how it can also make the work of a probability theorist easier.

4.2. A simulation study: Analyzing a quantile autoregressive process

In this section using the **quantspec** package from the perspective of a probability theorist is explained. The aim is twofold. On the one hand, further insight into a stochastic process shall be gained. Any strictly stationary process for which a function to simulate finite stretches of is available can be studied. On the other hand, the finite sample performance of the new spectral methods are to be evaluated. Note that the example discussed in this section and the functions to simulate QAR(1) processes are available inside the package, by calling `demo("qar-simulation", package = "quantspec")` and by referring to the function `ts1`, which implements the QAR(1) process that was discussed in Dette *et al.* (2015) and Kley *et al.* (2016). Recall that a QAR(1) process is a sequence (X_t) of random variables that fulfills

$$X_t = \theta_1(U_t)X_{t-1} + \theta_0(U_t),$$

where U_t is independent white noise with $U_t \sim \mathcal{U}[0,1]$, and $\theta_1, \theta_0 : [0,1] \to \mathbb{R}$ are model parameters (Koenker and Xiao 2006). The function `ts1` implements the model, where $\theta_1(u) = 1.9(u - 0.5)$, $u \in [0,1]$ and $\theta_0 = \Phi^{-1}$, which was discussed in Dette *et al.* (2015) and Kley *et al.* (2016). A complete list of models included in the package can be seen in the online documentation of the package by calling

```
R> help("ts-models")
```

The following, two-line script can be used to generate the graphical representation of the copula spectral density that is on display in Figure 9:

```
R> csd <- quantileSD(N = 2^9, seed.init = 2581, type = "copula",
+    ts = ts1, levels.1 = c(0.25, 0.5, 0.75), R = 100, quiet = TRUE)
R> plot(csd, ylab = expression(f[list(q[tau[1]], q[tau[2]])](omega)))
```

Figure 9 should be read in a way similar to Figures 5–8. Here, the red line represents the simulated spectrum, and the black line is an indicator for the precision of the simulation. When analyzing a time series model the recommended practice is to compute the quantile spectral density once with high precision, store it to the hard drive, and load it later whenever it is needed. The following two lines of code can be used to do this:
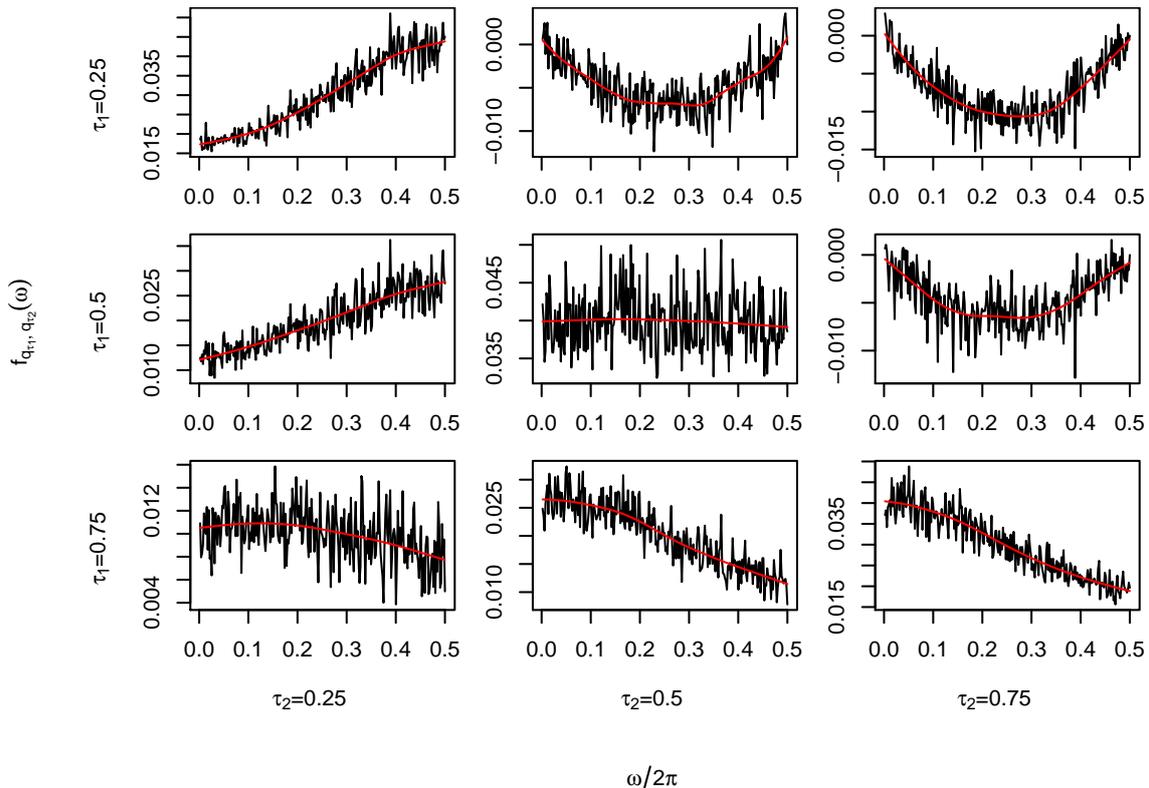
Figure 9: Plot of the copula spectral density $f_{q_{\tau_1}, q_{\tau_2}}(\omega)$ of the QAR(1) model; $\tau_1, \tau_2 \in \{0.25, 0.5, 0.75\}$, and $\omega \in [0, \pi]$; $N = 2^9$ and $R = 100$.

```
R> csd <- quantileSD(N = 2^12, seed.init = 2581, type = "copula",
+    ts = ts1, levels.1 = c(0.25, 0.5, 0.75), R = 50000)
R> save(csd, file = "csd-qar1.RData")
```

With the first configuration ($N = 2^9$ and $R = 100$) the computation time was only around three seconds. To compute the second `csd` object (with $N = 2^{12}$ and $R = 50000$) the same machine needed roughly 2.5 hours. Storing the object in a file takes about 1MB of hard disk space. Not only `values` and `stdError`s are stored within the 'QuantileSD' object; also the final state of the pseudo random number generator is stored and the method `increasePrecision` can be used to add more simulation runs at any time to yield a better approximation to the true quantile spectrum. More information on this method can be found in the online help, which is accessible via

```
R> help("increasePrecision-QuantileSD")
```

Once the computation is finished the diagram in Figure 10 can be created using the following two lines of code:

```
R> load("csd-qar1.RData")
R> plot(csd, frequencies = 2 * pi * (1:2^8) / 2^9,
+    ylab = expression(f[list(q[tau[1]], q[tau[2]])](omega)))
```
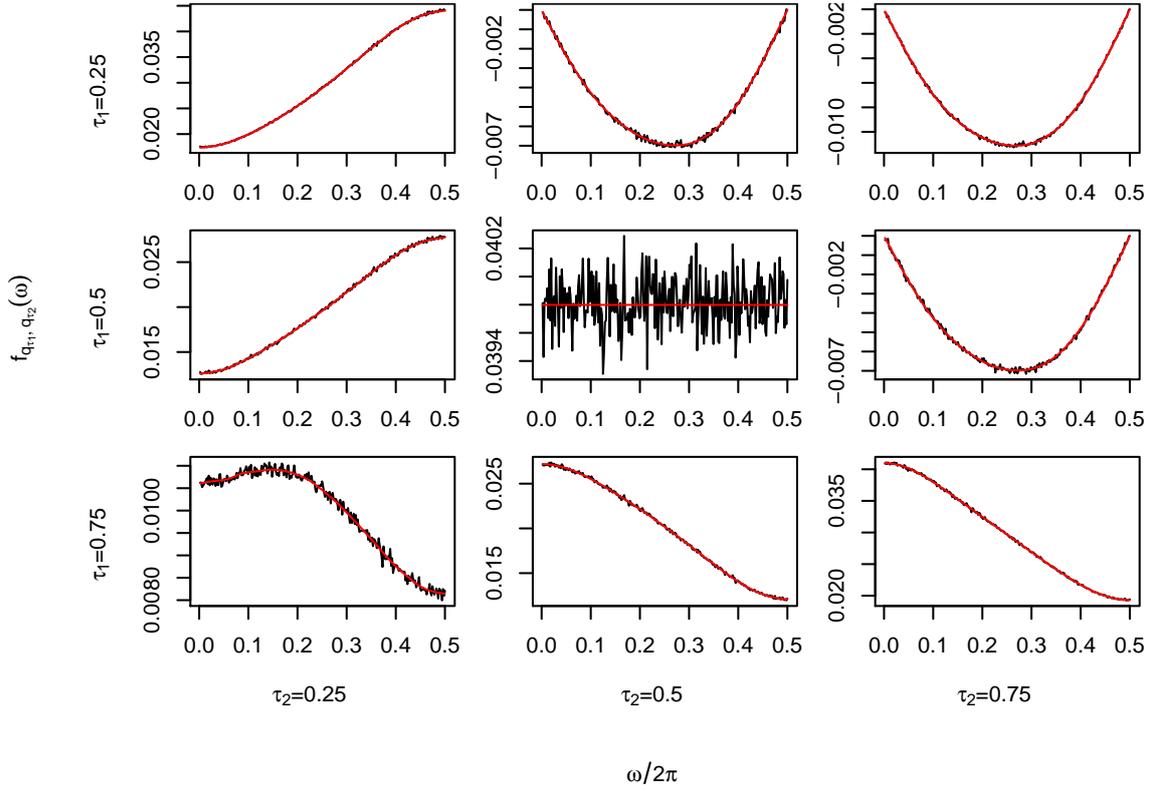
Figure 10: Plot of the copula spectral density $\mathfrak{f}_{q_{\tau_1}, q_{\tau_2}}(\omega)$ of the QAR(1) model; $\tau_1, \tau_2 \in \{0.25, 0.5, 0.75\}$, and $\omega \in [0, \pi]$; $N = 2^{12}$ and $R = 50000$.

The parameter `frequencies` was used when plotting the copula spectral density to create a plot that can be compared to the one in Figure 9. Note that by default the plot would have been created using all available frequencies which yields a grid of 8 times as many points on the x-axis ($N = 2^{12}$ vs. $N = 2^9$). Now, to get a first idea of how well the estimator performs, plot the smoothed CR periodogram computed from one simulated QAR(1) time series of length 512:

```
R> sCR <- smoothedPG(ts1(512), levels.1 = c(0.25, 0.5, 0.75),
+    weight = kernelWeight(W = W1, bw = 0.1))
R> plot(sCR, qsd = csd,
+    ylab = bquote(paste(hat(G)[list(n, R)](list(tau[1], tau[2], omega)),
+    " and ", f[list(q[tau[1]], q[tau[2]])](omega))))
```

The generated plot is on display in Figure 11. It is worth pointing out that in this example ($N = 512$) the estimator performs already quite well. Note that a different version of the constructor `smoothedPG()` was used here than in Section 4.1. When computing a smoothed quantile periodogram straight from a time series, the syntax is the same as for `quantilePG()`, but with the additional parameter `weight`.

Finally, for the simulation study, `R = 5000` independent QAR(1) time series are generated. Before the actual simulations, some variables that determine what is to be simulated are defined:
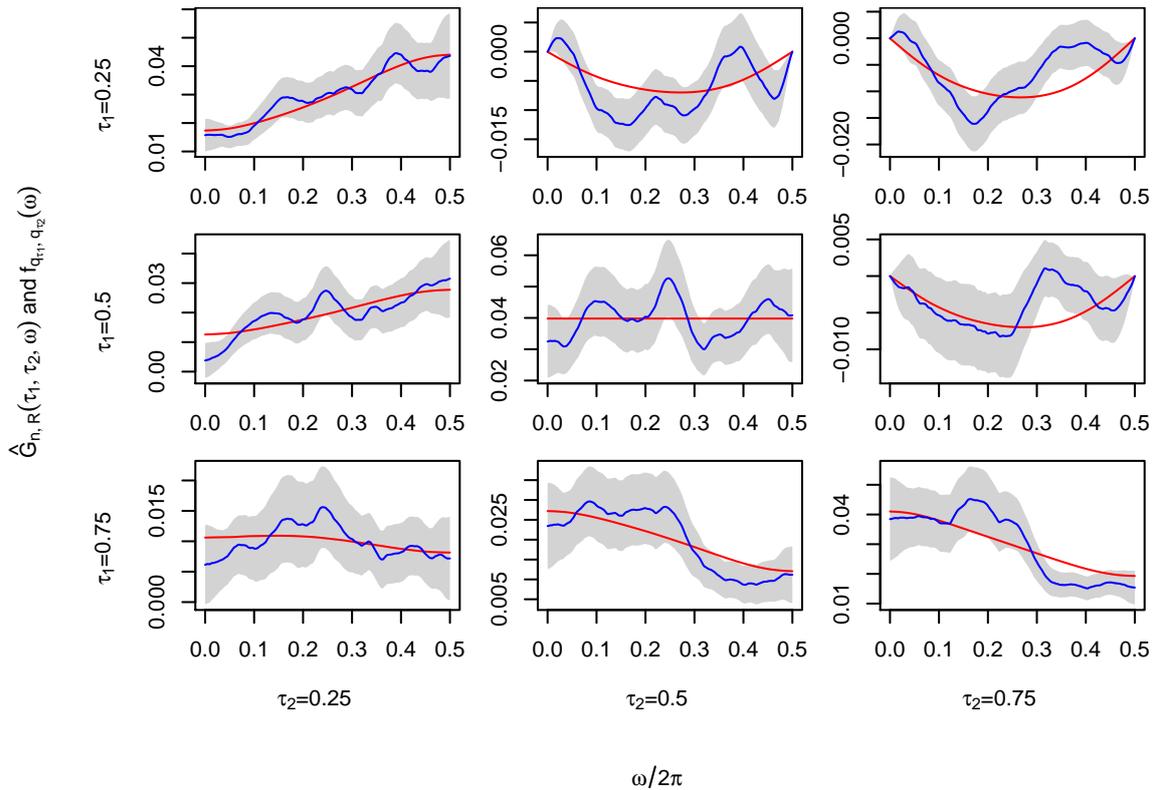
Figure 11: Plot of a smoothed CR periodogram, computed from one realization of a QAR(1) time series; $n = 512$, Epanechnikov kernel with $b_n = 0.1$.

```
R> set.seed(2581)
R> ts <- ts1
R> N <- 128
R> R <- 5000
R> freq <- 2  * pi * (1:16) / 32
R> levels <- c(0.25, 0.5, 0.75)
R> J <- length(freq)
R> K <- length(levels)
R> sims  <- array(0, dim = c(4, R, J, K, K))
R> weight <- kernelWeight(W = W1, bw = 0.3)
```

Setting the seed in the very beginning allows for reproducible results. Recall that `ts1` is a function to simulate from the QAR(1) model to be studied. `N` is the length of the time series and also the number of Fourier frequencies for which the quantile periodograms will be computed. By the parameter `freq` a subset of these Fourier frequencies is specified to be stored; a subset is used to save storage space. The estimates at these frequencies `freq` and at the specified `levels` are then stored to the array `sims`. In this example, the smoothed periodograms are computed using the Epanechnikov kernel and the (rather large) bandwidth of $b_n = 0.3$.

For the actual simulation the following `for` loop can be used:

```
R> for (i in 1:R) {
+    Y <- ts(N)
+    CR  <- quantilePG(Y, levels.1 = levels, type = "clipped")
+    LP  <- quantilePG(Y, levels.1 = levels, type = "qr")
+    sCR <- smoothedPG(CR, weight = weight)
+    sLP <- smoothedPG(LP, weight = weight)
+    sims[1, i, , , ] <- getValues(CR,  frequencies = freq)[, , , 1]
+    sims[2, i, , , ] <- getValues(LP,  frequencies = freq)[, , , 1]
+    sims[3, i, , , ] <- getValues(sCR, frequencies = freq)[, , , 1]
+    sims[4, i, , , ] <- getValues(sLP, frequencies = freq)[, , , 1]
+  }
```

Note that the flexible accessor method `getValues` is used to access the relevant subset of values for the frequencies specified (i.e., `freq`). Once the array `sims` is available, many interesting properties of the estimator can be analyzed. Examples include the bias, variance, etc. Here, using the function `getValues` again, the true copula spectral density is copied to an array `trueV`. Using the arrays `sims` and `trueV` the root integrated mean squared errors are computed as follows:

```
R> trueV <- getValues(csd, frequencies = freq)
R> SqDev <- array(apply(sims, c(1, 2),
+    function(x) abs(x - trueV)^2), dim = c(J, K, K, 4, R))
R> rimse <- sqrt(apply(SqDev, c(2, 3, 4), mean))
R> rimse

, , 1

          [,1]       [,2]       [,3]
[1,] 0.03292733 0.03543752 0.02879294
[2,] 0.03543752 0.04113916 0.03427386
[3,] 0.02879294 0.03427386 0.03014753

, , 2

          [,1]       [,2]       [,3]
[1,] 0.02688778 0.03275136 0.02691644
[2,] 0.03275136 0.03447488 0.03191837
[3,] 0.02691644 0.03191837 0.02526850

, , 3

           [,1]        [,2]        [,3]
[1,] 0.004338658 0.005889695 0.004472232
[2,] 0.005889695 0.006794010 0.005428915
[3,] 0.004472232 0.005428915 0.004917286

, , 4
```

```
          [,1]        [,2]        [,3]
[1,] 0.005133067 0.005699958 0.004575845
[2,] 0.005699958 0.006179409 0.005386339
[3,] 0.004575845 0.005386339 0.004797662
```

These numbers could now be inspected to observe, for example, that the smoothed quantile periodogram possess smaller root integrated mean squared errors than the quantile periodograms (without smoothing). Further discussion of the results is omitted, because the purpose of this chapter was to explain how to implement the simulation study, not to actually perform it.

5. Roadmap to future developments and concluding remarks

As the new methodology evolves additional features will be added to the **quantspec** package. For each new feature an entry to the issue tracker available on the GitHub repository will be made. Then the new feature will be implemented on a topic branch of the repository. For example, an extension that will allow to analyze multivariate time series is currently being developed.

Other, more complex extensions to the software include the implementation of functions to perform quantile spectral analysis for locally stationary processes, the computation and smoothing of higher order quantile periodograms for the estimation of quantile polyspectra. Procedures for graphical representations of these objects, possibly animated ones, will amend these planned parts of the package.

Summing up, it can be said that the **quantspec** package provides a comprehensive and conclusive toolbox to perform quantile-based spectral analysis. Due to the great interest in and active development of the statistical procedures that perform quantile-based spectral analysis it was deliberately designed in an object-oriented and extensible fashion. Thus it is well prepared for the many extensions that are sure to come in the near future. The source code is open and extensive documentation of the system is freely available. Comments on and contribution to the project are, of course, very much welcome.

Acknowledgments

References

Birr S, Volgushev S, Kley T, Dette H, Hallin M (2014). "Quantile Spectral Analysis for

Locally Stationary Time Series." arXiv:1404.4605 [math.ST], URL http://arxiv.org/abs/1404.4605.

Brillinger D (1975). *Time Series: Data Analysis and Theory.* Holt, Rinehart and Winston, Inc.

Dette H, Hallin M, Kley T, Volgushev S (2015). "Of Copulas, Quantiles, Ranks and Spectra: An L_1-Approach to Spectral Analysis." *Bernoulli*, **21**(2), 781–831. doi:10.3150/13-bej587.

Hagemann A (2013). "Robust Spectral Analysis." arXiv:1111.1965 [math.ST], URL http://arxiv.org/abs/1111.1965.

Hill J, McCloskey A (2013). "Heavy Tail Robust Frequency Domain Estimation." URL http://www.econ.brown.edu/fac/adam_mccloskey/Research_files/FDTTQML.pdf.

Hong Y (1999). "Hypothesis Testing in Time Series via the Empirical Characteristic Function: A Generalized Spectral Density Approach." *Journal of the American Statistical Association*, **94**(448), 1201–1220. doi:10.1080/01621459.1999.10473874.

Hong Y (2000). "Generalized Spectral Tests for Serial Dependence." *Journal of the Royal Statistical Society B*, **62**(3), 557–574. doi:10.1111/1467-9868.00250.

Katkovnik V (1998). "Robust M-Periodogram." *IEEE Transactions on Signal Processing*, **46**(11), 3104–3109. doi:10.1109/78.726825.

Kedem B (1980). *Binary Time Series.* Dekker, New York.

Kleiner B, Martin R, Thomson D (1979). "Robust Estimation of Power Spectra." *Journal of the Royal Statistical Society B*, **41**(3), 313–351.

Kley T (2014). *Quantile-Based Spectral Analysis: Asymptotic Theory and Computation.* PhD Thesis, Ruhr University Bochum. URL http://www-brs.ub.rub.de/netahtml/HSS/Diss/KleyTobias/.

Kley T (2016). **quantspec**: *Quantile-Based Spectral Analysis Functions.* R package version 1.2-1, URL http://CRAN.R-project.org/package=quantspec.

Kley T, Volgushev S, Dette H, Hallin M (2016). "Quantile Spectral Processes: Asymptotic Analysis and Inference." *Bernoulli*, **22**(3), 1770–1807.

Klüppelberg C, Mikosch T (1994). "Some Limit Theory for the Self-Normalised Periodogram of Stable Processes." *Scandinavian Journal of Statistics*, **21**(4), 485–491.

Koenker R (2005). *Quantile Regression.* Econometric Society Monographs. Cambridge University Press.

Koenker R (2016). **quantreg**: *Quantile Regression.* R package version 5.21, URL http://CRAN.R-project.org/package=quantreg.

Koenker R, Xiao Z (2006). "Quantile Autoregression." *Journal of the American Statistical Association*, **101**(475), 980–990. doi:10.1198/016214506000000672.

Lee J, Rao S (2012). "The Quantile Spectral Density and Comparison Based Tests for Nonlinear Time Series." arXiv:1112.2759 [math.ST], URL http://arxiv.org/abs/1112.2759.

Li TH (2008). "Laplace Periodogram for Time Series Analysis." *Journal of the American Statistical Association*, **103**(482), 757–768. doi:10.1198/016214508000000265.

Li TH (2012). "Quantile Periodograms." *Journal of the American Statistical Association*, **107**(498), 765–776. doi:10.1080/01621459.2012.682815.

Li TH (2013). *Time Series with Mixed Spectra: Theory and Methods*. CRC Press, Boca Raton.

Li TH (2014). "Quantile Periodogram and Time-Dependent Variance." *Journal of Time Series Analysis*, **35**(4), 322–340. doi:10.1111/jtsa.12065.

Mikosch T (1998). "Periodogram Estimates from Heavy-Tailed Data." In R Adler, R Feldman, M Taqqu (eds.), *A Practical Guide to Heavy Tails: Statistical Techniques for Analysing Heavy-Tailed Distributions*, pp. 241–258. Birkhäuser, Boston.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Wickham H (2011). "**testthat**: Get Started with Testing." *The R Journal*, **3**(1), 5–10. URL https://journal.R-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf.

Wickham H, Chang W (2016). **devtools**: *Tools to Make Developing R Code Easier*. R package version 1.10.0, URL https://CRAN.R-project.org/package=devtools.

Zeileis A, Grothendieck G (2005). "**zoo**: S3 Infrastructure for Regular and Irregular Time Series." *Journal of Statistical Software*, **14**(6), 1–27. doi:10.18637/jss.v014.i06.

Affiliation:

Tobias Kley
Ruhr-Universität Bochum
Department of Mathematics
Institute of Statistics
Building NA 3/76
44780 Bochum, Germany
E-mail: tobias.kley@ruhr-uni-bochum.de
URL: http://www.ruhr-uni-bochum.de/mathematik3/en/team/kley.html