DEPARTMENT OF ECONOMICS
AND BUSINESS ECONOMICS
AARHUS UNIVERSITY

CREATES
Center for Research in Econometric Analysis of Time Series

To infinity and beyond: Efficient computation of ARCH(∞) models

Morten Ørregaard Nielsen and Antoine L. Noël

CREATES Research Paper 2020-13

Department of Economics and Business Economics
Aarhus University
Fuglesangs Allé 4
DK-8210 Aarhus V
Denmark

Email: oekonomi@au.dk
Tel: +45 8716 5515

To infinity and beyond: Efficient computation of ARCH(∞) models*

Morten Ørregaard Nielsen†
Queen's University and CREATES
mon@econ.queensu.ca

Antoine L. Noël
Queen's University
noela@econ.queensu.ca

November 6, 2020

Abstract

This paper provides an exact algorithm for efficient computation of the time series of conditional variances, and hence the likelihood function, of models that have an ARCH(∞) representation. This class of models includes, e.g., the fractionally integrated generalized autoregressive conditional heteroskedasticity (FIGARCH) model. Our algorithm is a variation of the fast fractional difference algorithm of Jensen and Nielsen (2014). It takes advantage of the fast Fourier transform (FFT) to achieve an order of magnitude improvement in computational speed. The efficiency of the algorithm allows estimation (and simulation/bootstrapping) of ARCH(∞) models, even with very large data sets and without the truncation of the filter commonly applied in the literature. In Monte Carlo simulations, we show that the elimination of the truncation of the filter reduces the bias of the quasi-maximum-likelihood estimators and improves out-of-sample forecasting. Our results are illustrated in two empirical examples.

JEL codes: C22, C58, C63, C87.

Keywords: Circular convolution theorem, conditional heteroskedasticity, fast Fourier transform, FIGARCH, truncation.

1 Introduction

Many autoregressive conditional heteroskedasticity (ARCH) models have a so-called ARCH(∞) representation, which is similar to the linear representation for time series models for the conditional mean. The ARCH(∞) representation, or model, appears to be introduced by Bollerslev (1986). An example is the very popular fractionally integrated GARCH (FIGARCH) model of Baillie, Bollerslev, and Mikkelsen (1996); see below for additional examples.

*We are grateful to participants at the Canadian Econometric Study Group meeting 2019 for comments. Nielsen thanks the Canada Research Chairs program, the Social Sciences and Humanities Research Council of Canada, and the Center for Research in Econometric Analysis of Time Series (CREATES) for financial support. Noël thanks the Social Sciences and Humanities Research Council of Canada for financial support.

†Corresponding author. Address: Department of Economics, 94 University Avenue, Queen's University, Kingston, Ontario K7L 3N6, Canada. Email: mon@econ.queensu.ca. Tel.: 613-533-2262. Fax: 613-533-6668.

In ARCH(∞) models, the sequence of conditional variances is a linear convolution of all previous squared innovations, where the weights are simple functions of the parameters of the model. By standard methods, the calculation of the sequence of conditional variances, and hence of the likelihood function, requires $O(T^2)$ arithmetic operations. Evaluating the likelihood function of an ARCH(∞) model requires calculation of the sequence of conditional variances, and optimizing the likelihood typically requires many iterations and hence many calculations of the sequence of conditional variances with different parameter values. Thus, for large sample sizes, the $O(T^2)$ computational cost can be prohibitive for estimation of the model, and even more so for multivariate models. Even if it is not prohibitive for estimation with a given sample, it may render bootstrap inference or simulation methods infeasible.

Sample sizes commonly applied in empirical work have grown tremendously in recent years, especially in finance. Time series of daily observations over many years are very common; for example, our daily exchange rate data in Section 5.2 has 12,300 observations. However, much longer data series are sometimes analyzed. For example, in recent empirical work, models of the ARCH(∞) class have been fitted to samples of size 100,032 (Han, 2008), 509,472 (Chortareas, Jiang, and Nankervis, 2011), 55,860 (Conrad, Rittler, and Rotfuß, 2012), and 156,672 (Naeema, Shahbazb, Saleemc, and Mustafaa, 2019).

To speed up computation it has been standard in the literature, at least since Baillie et al. (1996), to truncate the convolution at a fixed truncation number such as 1,000. That is, only a part of the time series of innovations is used to calculate the sequence of conditional variances. Of course, this is an approximation which implies that the calculation is not exact.

In this paper, we discuss efficient computation of the sequence of conditional variances, and hence the likelihood function, for any model that has an ARCH(∞) representation. We make three separate contributions.

First, we show how to apply the fast Fourier transform (FFT) of Cooley and Tukey (1965) and the circular convolution theorem to reduce the required number of arithmetic operations from $O(T^2)$ to $O(T \log T)$. Our proposed algorithm is a variation of an algorithm recently proposed by Jensen and Nielsen (2014) for the calculation of fractional differences. The variation is threefold. We (i) apply the FFT-based method to a transformation of a time series instead of the time series itself, (ii) account for possible truncation of the convolution, and (iii) apply arbitrary coefficients instead of the fractional difference coefficients. The algorithm is exact and does not rely on any approximation device. The increase in computational speed relative to linear convolution can easily be a factor of 10 or even much more for sample sizes that can be encountered in practical applications, and is sufficient to make bootstrap and simulation methods for ARCH(∞) models feasible, even with very large sample sizes.

Second, in a small Monte Carlo simulation, we quantify the estimation bias introduced by using a fixed truncation number. We show that, for the truncation number typically applied in the literature, 1,000, the bias can be substantial. However, because our algorithm is exact and sufficiently fast to eliminate the need for truncation, it does not suffer from this bias.

Third, in the Monte Carlo simulation, we also compare out-of-sample forecast performance with and without truncation. We find that the bias from truncation makes forecast less precise compared with no truncation. Moreover, the difference in forecast errors with and without truncation in fact increases with the sample size.

In two empirical applications we apply our algorithm to data sets of exchange rates. These highlight the differences obtained with and without the truncation number, and in

particular emphasize the very different computational time achieved with the new FFT-based algorithm compared with standard linear convolution.

Recently, Klein and Walther (2017) also proposed an application of the FFT and circular convolution theorem, based on Jensen and Nielsen (2014), to calculate the sequence of conditional variances for FIGARCH models. Their method and algorithm is closely related to ours. However, Klein and Walther (2017) maintain the fixed truncation number throughout and do not consider exact algorithms. They also do not consider the bias associated with the application of a fixed truncation number. We include some comparisons with their algorithm, which show that, in the absence of the fixed truncation number, its computational time is of the same order of magnitude as the standard linear convolution algorithm.

Many conditional heteroskedasticity models proposed in the literature have an ARCH(∞) representation, going back to the GARCH model of Bollerslev (1986). However, the conditional variance of the (finite-order) GARCH model is a sum of only a fixed number of terms (in the same way that an ARMA model is a sum of a fixed number of terms, but it has an MA(∞) representation). Because the GARCH model is a sum of only a *fixed* number of terms, it will not benefit noticeably from our method. In contrast, our method will be of great benefit to models where the conditional variance is *not* a sum of a *fixed* number of terms. In particular, this covers all the long-memory-type conditional variance models in the ARCH(∞) class, e.g., the FIGARCH model of Baillie et al. (1996), the long-memory GARCH model of Karanasos, Psaradakis, and Sola (2004), and the hyperbolic GARCH model of Davidson (2004). Finally, our method also applies to some models that are not even in the ARCH(∞) class. For example, it applies to the fractionally integrated asymmetric power ARCH model of Tse (1998) and the linear ARCH model of Robinson (1991) and Giraitis, Robinson, and Surgailis (2000), and it partly applies to the fractionally integrated exponential GARCH model of Bollerslev and Mikkelsen (1996); see Section 2.2 for details.

The next section defines the ARCH(∞) class and explains our proposed FFT-based algorithm for calculation of the sequence of conditional variances. Section 3 presents results on the computational time for the new algorithm and compares with the standard linear convolution algorithm and the Klein and Walther (2017) algorithm. An analysis of the effects on computational time of using a fixed truncation number is also given. In Section 4 we present the results of a small simulation study to investigate the estimation bias that results from truncation and compare out-of-sample forecast performance. Section 5 illustrates our results with two empirical applications. Finally, in Section 6 we give some concluding remarks.

2 FFT algorithm for ARCH(∞) models

We consider the following generic conditional variance model for the series $(\epsilon_t)_{t=1}^T$, which is either an observable series (such as asset returns) or the error term from a (regression) model,

$$\epsilon_t = \sigma_t z_t, \quad \mathbb{E}_{t-1}(\epsilon_t^2) = \sigma_t^2, \quad \text{and} \quad z_t \sim \text{i.i.d.}(0,1). \tag{1}$$

Here, $(z_t)_{t=1}^T$ is an independently and identically distributed (i.i.d.) innovation series with mean zero and variance normalized to one, $(\sigma_t^2)_{t=1}^T$ is the conditional variance series, and $\mathbb{E}_{t-1}$ denotes the expectation conditional on the information available at time $t-1$.

The ARCH(∞) model (or representation) for the conditional variance is

$$\sigma_t^2 = c + \sum_{j=0}^{\infty} \lambda_j \epsilon_{t-j}^2, \tag{2}$$

where c and λ_j are the corresponding constant and coefficients, respectively. Of course, in practical applications, the series (ϵ_t) will be available only for $t = 1, \ldots, T$, where T denotes the sample size. In practice, therefore, the summation in (2) will need to be truncated at $j = t-1$. This results in the following ARCH(∞) model, which can be viewed, for example, as a device to approximate the likelihood function,

$$\sigma_t^2 = c + \sum_{j=0}^{t-1} \lambda_j \epsilon_{t-j}^2. \tag{3}$$

The summation in (3) is a linear convolution. By a standard linear convolution algorithm, for given series $(\epsilon_t)_{t=1}^T$ and $(\lambda_j)_{j=0}^{T-1}$, and a given value of t, the calculation (3) requires $2t$ arithmetic operations (t multiplications and t additions). Thus, the calculation of the series $(\sigma_t^2)_{t=1}^T$ requires $\sum_{t=1}^T 2t = T^2 + T$ arithmetic operations.

When the sample size is large, the computational burden of $T^2 + T$ arithmetic operations makes the calculation of the conditional variance series $(\sigma_t^2)_{t=1}^T$ very slow. This can render optimization of a likelihood function, and certainly simulations and bootstrapping, infeasible. To overcome the computational burden of the calculation in (3), Baillie et al. (1996) suggested truncating the summation in (3) at a fixed number, say n, which is typically chosen to be 1,000. This truncation has become standard in the literature. The resulting model is thus

$$\sigma_t^2 = c + \sum_{j=0}^{\min\{t-1,n\}} \lambda_j \epsilon_{t-j}^2. \tag{4}$$

The required number of arithmetic operations to calculate the series $(\sigma_t^2)_{t-1}^T$ in (4) is only of order nT. Importantly, for a fixed truncation number n, the number of operations grows only linearly with the sample size and hence avoids the squared growth required for the calculation in (3).

While the introduction of the truncation number, n, in (4) allows for much faster calculation of the conditional variance series $(\sigma_t^2)_{t=1}^T$, it also introduces an approximation. It is no longer an exact algorithm to calculate the desired series given in (3). In Section 4 we present the results of some Monte Carlo simulations to illustrate and quantify the bias in parameter estimation resulting from this approximation.

We now introduce an alternative method to calculate the conditional variance series $(\sigma_t^2)_{t=1}^T$ in (3) without truncation using frequency domain techniques. Specifically, we will apply the fast Fourier transform (FFT) combined with the so-called circular convolution theorem. As discussed in Jensen and Nielsen (2014) in the context of calculating fractional differences, this method reduces the number of arithmetic operations to order $T \log T$. To describe this method, we will need the following two definitions.

Definition 1. The discrete Fourier transform (DFT), $f = (f_j)_{j=1}^T$, of a series $a = (a_t)_{t=1}^T$ is the solution to

$$a = T^{-1} F f,$$

where F is the Fourier matrix given by $F_{jk} = w_T^{(j-1)(k-1)}$ with $w_T = e^{2\pi \mathrm{i}/T}$ and $\mathrm{i} = \sqrt{-1}$. □

Definition 2. Let a_j and b_j be two periodic sequences, meaning that $a_{j+NT} = a_j$ and $b_{j+NT} = b_j$ for $N = \pm 0, \pm 1, \pm 2, \ldots$. Then the circular convolution of $(a_j)_{j=1}^T$ and $(b_j)_{j=1}^T$ is defined as

$$(a \circledast b)_t = \sum_{j=1}^{T} a_j b_{t-j+1} = \sum_{j=1}^{t} a_j b_{t-j+1} + \sum_{j=t+1}^{T} a_j b_{T+t-j+1}, \quad t = 1, \ldots, T. \qquad \square$$

We next present two theorems. The first is a finite version of the circular convolution theorem, which shows how the circular convolution in Definition 2 can be calculated using the DFT in Definition 1. For periodic integrable functions, this result can be found in Zygmund (2003, Thm. 1.5, p. 36). The finite version has appeared in the early engineering literature as an application of the FFT; e.g. Stockham (1966, p. 230) and Cooley, Lewis, and Welch (1969, p. 32). Our version, presented in Theorem 1, is taken from Jensen and Nielsen (2014) and the proof can be found there.[1]

Theorem 1. *Let $a = (a_t)_{t=1}^T$ and $b = (b_t)_{t=1}^T$ be two sequences. Then*

$$a \circledast b = T^{-1} F(\bar{F} a \circ \bar{F} b), \tag{5}$$

where $\circ$ denotes element-wise multiplication and $\bar{F}$ is the complex conjugate of F.

To apply the result in Theorem 1 in our context, we need to transform the linear convolutions in (3) and (4) into circular convolutions. To economize on notation, we consider only (4), but allow the possibility that $n = T - 1$, in which case we obtain (3).

Our second theorem shows that the sequence of conditional variances in (3) or (4) (where the former is obtained by setting $n = T-1$ in the latter) can be calculated by extending the sequences with zeros and applying the result in Theorem 1. Specifically, we extend the vector of errors, $(\epsilon_t)_{t=1}^T$, and the vector of coefficients, $(\lambda_j)_{j=0}^n$, with zeros to length $2T-1$. Thus, let the extended vectors be denoted $\tilde{\epsilon} = [\epsilon', O'_{T-1}]$ and $\tilde{\lambda} = [\lambda', O'_{2T-1-(n+1)}]$, respectively, where O_m denotes an $m \times 1$ vector of zeros. We then prove in Theorem 2 that the linear convolution of $(\epsilon_t^2)_{t=1}^T$ and $(\lambda_j)_{j=0}^n$ in (4) can be found as the first T elements of the circular convolution of $(\tilde{\epsilon}_t^2)_{t=1}^{2T-1}$ and $(\tilde{\lambda}_j)_{j=0}^{2T-2}$.

Theorem 2. *Let $\tilde{\epsilon}^2$ and $\tilde{\lambda}$ denote the $(2T-1) \times 1$ extended vectors of squared errors and coefficients, respectively. In particular, the t'th element of $\tilde{\epsilon}^2$ is $\tilde{\epsilon}_t^2$. Then the vector of conditional variances, $(\sigma_t^2)_{t=1}^T$, in (3) or (4) can be calculated as the first T elements of the $(2T-1) \times 1$ vector $c + T^{-1} F(\bar{F}\tilde{\lambda} \circ \bar{F}\tilde{\epsilon}^2)$.*

Because of the truncation number, n, the result in Theorem 2 may appear to be different from the corresponding result in Theorem 2 of Jensen and Nielsen (2014), which does not consider truncation. Although Theorem 2 does in fact follow from Theorem 2 of Jensen and Nielsen (2014), it seems intructive to give a short proof of Theorem 2 to explicitly demonstrate the role of the extended vectors $\tilde{\epsilon}$ and $\tilde{\lambda}$.

Proof. From Theorem 1 we know that $\tilde{\lambda} \circledast \tilde{\epsilon}^2 = T^{-1} F(\bar{F}\tilde{\lambda} \circ \bar{F}\tilde{\epsilon}^2)$. Therefore, it only remains to be shown that $(\tilde{\lambda} \circledast \tilde{\epsilon}^2)_t = \sum_{j=0}^{\min\{t-1,n\}} \lambda_j \epsilon_{t-j}^2$ for $t = 1, \ldots, T$. From Definition 2, noting that the extended sequences have $2T-1$ elements and that the $\tilde{\lambda}_j$ sequence starts at index $j = 0$, we find that

$$(\tilde{\lambda} \circledast \tilde{\epsilon}^2)_t = \sum_{j=1}^{t} \tilde{\lambda}_{j-1} \tilde{\epsilon}_{t-j+1}^2 + \sum_{j=t+1}^{T+t-1} \tilde{\lambda}_{j-1} \tilde{\epsilon}_{2T+t-j}^2 + \sum_{j=T+t-1}^{2T-1} \tilde{\lambda}_{j-1} \tilde{\epsilon}_{2T+t-j}^2. \tag{6}$$

[1] In Jensen and Nielsen (2014) there is a small typo in the fifth and sixth lines of the proof, where $(t-1)(s-1)$ should be $(t-1)(u-1)$ in the exponents. However, since the last equality sets $s = u$ the result is the same.

When $j = t+1, \ldots, T+t-1$ we have $2T+t-j = T+1, \ldots, 2T-1$, so that $\tilde{\epsilon}^2_{2T+t-j} = 0$ by definition. Similarly, when $j = T+t+1, \ldots, 2T-1$ and $t = 1, \ldots, T$ we have $\tilde{\lambda}_{j-1} = 0$ by definition. This leaves only the first term on the right-hand side of (6). When $t-1 \leq n$, this term is equal to $\sum_{j=0}^{\min\{t-1,n\}} \tilde{\lambda}_j \tilde{\epsilon}^2_{t-j}$, which is what we needed to show. When $t-1 > n$, we split the first term on the right-hand side of (6) as

$$\sum_{j=1}^{t} \tilde{\lambda}_{j-1} \tilde{\epsilon}^2_{t-j+1} = \sum_{j=0}^{n} \tilde{\lambda}_j \tilde{\epsilon}^2_{t-j} + \sum_{j=n+1}^{t-1} \tilde{\lambda}_j \tilde{\epsilon}^2_{t-j}.$$

For $j = n+1, \ldots, t-1$ and $t = 1, \ldots, T$ we have $\tilde{\lambda}_j = 0$ by definition, so that the last summation is zero, which proves the desired result. □

Theorem 2 shows that the linear convolution of $(\epsilon_t^2)_{t=1}^T$ and $(\lambda_j)_{j=0}^n$ in (4) can be found as the first T elements of the circular convolution of $(\tilde{\epsilon}_t^2)_{t=1}^{2T-1}$ and $(\tilde{\lambda}_j)_{j=0}^{2T-2}$, which in turn can be calculated using the DFT. The power of this result lies in the fact that the required DFTs can be calculated extremely efficiently by the FFT. Indeed, the latter requires only an order $T \log T$ arithmetic operations, as shown by Cooley and Tukey (1965). This implies that, not only can the conditional variances be calculated an order of magnitude faster than using the standard linear convolution method, but also that the order of magnitude is the same with and without truncation. That is, there is no need to introduce a truncation number to speed up computation as in (4).

Most standard implementations of the FFT requires the length of the vectors to be a power of two.[2] This is easily accommodated by further extending the vectors with zeros. Thus, to apply the FFT in the calculations, the vectors ϵ and λ must be extended with zeros such that the number of elements in the vectors is equal to the smallest power of two that is at least $2T-1$. This ensures that both Theorem 2 and the FFT can be applied.

The implementation described in the previous paragraph implies that conditional variance series for a range of sample sizes can be calculated in the same amount of time. For example, consider $T = 1{,}025$ and $T = 2{,}048$. In both cases, the smallest power of two that is at least $2T-1$ is 11, and consequently one must extend with 3,071 zeros when $T = 1{,}025$ and 2,048 zeros when $T = 2{,}048$. In both situations, the extended vectors have 4,096 elements, and thus it will take the same amount of time to compute the conditional variances using the FFT-based method in Theorem 2.

In Listings 1 and 2 we present our Matlab codes to implement the standard linear convolution algorithm and the FFT-based algorithm in Theorem 2, respectively. The former is a simple application of Matlab's `filter` function. Implementations of the FFT-based method in Theorem 2 for R and Ox can be found in Listings 3 and 4, respectively. All codes are downloadable from the authors' websites.

Listing 1: Matlab code to calculate ARCH(∞) conditional variances by LC method

```
function [sigma2_arch] = arch_lc(cst, epsilon, lambda)
    sigma2_arch = cst + filter(lambda, 1, epsilon.^2);
end
```

[2] Some modern implementations require only that the length is a product of powers of small prime numbers, such as $2^k 3^l 5^m$, and the subsequent discussion is easily adapted to such cases.

Listing 2: Matlab code to calculate ARCH(∞) conditional variances by FFT method

```
function [sigma2_arch] = arch_fft(cst, epsilon, lambda)
     T = size(epsilon, 1);
     np2 = 2.^nextpow2(2*T-1);
     sigma2_arch = ifft(fft(epsilon.^2, np2).*fft(lambda, np2));
     sigma2_arch = cst + sigma2_arch(1:T);
end
```

Listing 3: R code to calculate ARCH(∞) conditional variances by FFT method

```
arch_fft <- function(cst, epsilon, lambda){
     iT <- length(epsilon)
     np2 <- nextn(2*iT-1, 2)
     sigma2_arch <- fft(fft(c(lambda, rep(0, np2-iT))) * fft(c(epsilon^2,
        rep(0, np2-iT))), inverse = T) / np2;
     sigma2_arch <- cst + sigma2_arch[1:iT]
     return(Re(sigma2_arch))
}
```

Listing 4: Ox code to calculate ARCH(∞) conditional variances by FFT method

```
arch_fft(const cst, const epsilon, const lambda)
{
     decl T, sigma2_arch;
     T = rows(epsilon);
     sigma2_arch = fft(cmul(fft(lambda'~zeros(1,T)), fft((epsilon.^2)'~zeros
        (1,T)), 2);
     return cst + sigma2_arch'[0:T-1];
}
```

2.1 Example: FIGARCH model

We now consider an example that we will apply extensively in the remainder. The most well-known ARCH(∞) model for the conditional variance is probably the FIGARCH(p,d,q) model of Baillie et al. (1996). For that model, the conditional variance specification is

$$\beta(L)\sigma_t^2 = \omega + \Big(\beta(L) - \phi(L)(1-L)^d\Big)\epsilon_t^2, \tag{7}$$

where $\beta(L) = 1 - \sum_{i=1}^{p} \beta_i L^i$ and $\phi(L) = 1 - \sum_{i=1}^{\max\{p,q\}} \phi_i L^i$ are polynomials in the lag operator, L, whose roots all lie outside the complex unit circle. The fractional difference operator $(1-L)^d$ is defined in terms of the fractional coefficients $\pi_j(u) = u(u+1)\dots(u+j-1)/j!$ from the binomial expansion of $(1-z)^{-u}$, so that, for a generic series $(x_t)_{t=1}^{T}$, we have $(1-L)^d x_t = \sum_{j=0}^{t-1} \pi_j(-d) x_{t-j}$.

The specification (7) can be rearranged to get an ARCH(∞) representation. For simplicity, suppose $p = q = 1$, so that $\beta(L) = 1 - \beta L$ and $\phi(L) = 1 - \phi L$. This is the most commonly applied variant of the FIGARCH model. Then (7) can be written in the form (2)

with $c = \omega/(1-\beta)$ and

$$\lambda_j = \begin{cases} 0 & \text{for } j = 0, \\ \phi - \beta + d & \text{for } j = 1, \\ \beta\lambda_{j-1} + \phi\pi_{j-1}(-d) - \pi_j(-d) & \text{for } j \geq 2; \end{cases} \tag{8}$$

see Baillie et al. (1996) for details. Let $g_1 = \phi - \beta + d$ and $g_j = \phi\pi_{j-1}(-d) - \pi_j(-d)$ for $j \geq 2$. Then the λ_j coefficients in (8) can be rewritten as

$$\lambda_j = \begin{cases} 0 & \text{for } j = 0, \\ \sum_{i=0}^{j-1} \beta^i g_{j-i} & \text{for } j \geq 1. \end{cases} \tag{9}$$

The last term in (9) is clearly a linear convolution. To speed up computation, the series of λ_j coefficients should also be calculated using our FFT-based algorithm.

It follows that efficient calculation of the conditional variances in the FIGARCH$(1, d, 1)$ model should use our FFT-based algorithm twice: once to calculate the conditional variances given the λ_j coefficients, and once to calculate the λ_j coefficients given the model parameters. The former calculation is described in detail in Theorem 2. Similarly, $(\lambda_j)_{j=1}^n$ can be found as the first n elements of the $(2n-1) \times 1$ vector $n^{-1}F(\bar{F}\tilde{\beta} \circ \tilde{g})$, where $\tilde{\beta}$ and $\tilde{g}$ denote the vectors $(\beta^j)_{j=0}^{n-1}$ and $(g_j)_{j=1}^n$ extended with zeros to length of the smallest power of two that is at least $2n-1$. Matlab implementations of the calculation of $(\lambda_j)_{j=0}^n$ are shown in Listings 5 and 6 for the linear convolution algorithm and the FFT-based algorithm, respectively.

Listing 5: Matlab code to calculate λ_j coefficients of FIGARCH$(1, d, 1)$ by LC method

```
function [lambda] = lambda_lc(phi, beta, d, nTrunc)
    lambda = zeros(nTrunc+1,1);
    k = (1:nTrunc)';
    pij = cumprod((k(1:end)-d-1)./(k(1:end)));
    g = zeros(nTrunc,1);
    g(1) = phi - beta + d;
    g(2:end) = phi*pij(1:end-1) - pij(2:end);
    lambda(1) = 0;
    lambda(2:end) = filter(beta.^(0:nTrunc-1), 1, g);
end
```

Listing 6: Matlab code to calculate λ_j coefficients of FIGARCH$(1, d, 1)$ by FFT method

```
function [lambda] = lambda_fft(phi, beta, d, nTrunc)
    lambda = zeros(nTrunc+1,1);
    k = (1:nTrunc)';
    pij = cumprod((k(1:end)-d-1)./(k(1:end)));
    g = zeros(nTrunc,1);
    g(1) = phi - beta + d;
    g(2:end) = phi*pij(1:end-1) - pij(2:end);
    np2 = 2.^nextpow2(2*nTrunc-1);
    lambda(1) = 0;
    tmp = ifft(fft(beta.^(0:nTrunc-1)', np2).*fft(g, np2));
```

```
    lambda(2:end) = tmp(1:nTrunc);
end
```

2.2 Application to non-ARCH(∞) models

Our FFT-based methodology can also be applied with great benefit to some models that are not even in the class of ARCH(∞) models. In particular, the linear ARCH model of Robinson (1991) and Giraitis et al. (2000) is given by $\sigma_t = c + \sum_{j=1}^{\infty} \lambda_j \epsilon_{t-j}$, c.f. (2). Clearly, our FFT-based algorithms described above and presented in Listings 2 and 6 can easily be adopted to this model by simply replacing σ_t^2 and ϵ_{t-j}^2 with σ_t and ϵ_{t-j}, respectively. Similarly, the fractionally integrated asymmetric power ARCH model of Tse (1998) is given as in (7), but with σ_t^2 and ϵ_t^2 replaced by σ_t^δ and $h(\epsilon_t)$, respectively, where $h(\epsilon) = (|\epsilon| - \gamma\epsilon)^\delta$ and $\delta > 0$. Again, our FFT-based algorithms apply with minimal changes to this model.

Finally, the popular fractionally integrated exponential GARCH (FIEGARCH) model of Bollerslev and Mikkelsen (1996) is given as in (7), but with σ_t^2 replaced by $\log(\sigma_t^2)$ and ϵ_t^2 replaced by $h(\epsilon_t/\sigma_t)$, where $h(z) = \gamma_1 z + \gamma_2(|z| - \mathbb{E}|z|)$. Thus, because σ_t enters non-linearly on the right-hand side of the conditional variance equation, the sequence of conditional variances for the FIEGARCH model cannot be written as a convolution and our methodology does not apply to the calculation of the conditional variance sequence. However, the λ_j coefficients for this model are identical to those for the FIGARCH model, and the calculation of these can therefore take advantage of our FFT-method as in Listing 6. At least this will reduce the computational time of calculating the λ_j coefficients for this model from $O(T^2)$ to $O(T \log T)$ compared with standard linear convolution.

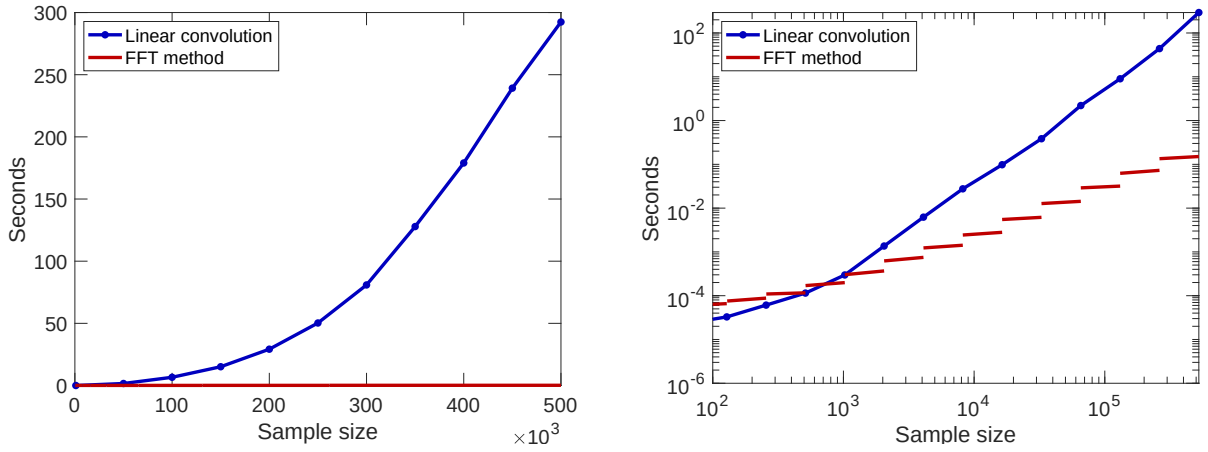
3 Computational time

In this section, we compare the computational time of our new algorithm in Theorem 2 and Listings 2,6 with the algorithm in Klein and Walther (2017) as well as the standard linear convolution implementation in Listings 1,5. All computations were performed in Matlab on a desktop with an Intel Core i5 (5250U) 1.6GHz processor running macOS Catalina 10.15.6.

In our first numerical experiment, we compare the difference in computational time for standard linear convolution and our FFT-based algorithm as a function of sample size. Specifically, for a given vector of coefficients, $[d, \phi, \beta, \omega] = [0.46, 0.27, 0.65, 0.02]$, we calculate the conditional variances for the FIGARCH$(1, d, 1)$ model using the method in Listings 1,5 (blue line) and the FFT-based algorithm in Listings 2,6 (red line). That is, we calculate the λ_j coefficients in (9) followed by the conditional variances in (3), i.e. without truncation, with random numbers for ϵ_t. This is done a large number of times (10,000 for $T <$ 4,096 and 1,000 for $T \geq$ 4,096), and the median time in milliseconds across repetitions is reported in Figure 1 as a function of sample size. In Panel (a) the axes are linear and in Panel (b) the axes are logarithmic. In particular, the figure clearly shows the different orders of magnitude of the computational time for the two methods (T^2 vs. $T \log T$). The figure also demonstrates how the computational time for the FFT-based algorithm is essentially a step function of the sample size, because of the padding to the smallest power of two that is at least $2T - 1$.

We also notice from Figure 1(b) that for $T = 500$ to $T =$ 1,000, the two methods perform equally well, while linear convolution does better for smaller sample sizes and the FFT-based algorithm does better for larger sample sizes. Thus, to obtain the fastest possible

Figure 1: Computational time



(a) Linear scale

(b) Log-log scale

algorithm across all sample sizes, we recommend implementing the linear convolution method for smaller sample sizes and the FFT-based algorithm for larger sample sizes. This can easily be achieved by combining the implementations in Listings 1 and 2 using an `if` statement, and similarly for Listings 5 and 6.

In our next experiment, we focus on the effect of truncation on computational speed, and compare linear convolution (LC), our FFT-based algorithm, and the algorithm of Klein and Walther (2017, henceforth KW).[3] The setup is the same as in Figure 1, except for the introduction of the truncation number. To reflect samples sizes in modern empirical work (see references in the introduction), we consider $T = 3{,}000$ to $T = 500{,}000$ and truncation numbers from $n = 1{,}000$ to $n = 5{,}000$, as well as no truncation, $n = T - 1$. For each (T, n) pair, we repeated the calculation 10,000 times (for fixed n) or 1,000 times (for $n = T - 1$), except for KW and LC with $T = 500{,}000$ and $n = T - 1$, where we only used 100 repetitions. In Table 1 we report the resulting median time across repetitions in milliseconds.

In Table 1 we first note that our FFT-based method is faster than the other methods for all sample sizes and truncation numbers considered. While the computational times of both the KW and LC algorithms are very sensitive to the choice of truncation number, the FFT-based method is much less so. For example, from $n = 1{,}000$ to $n = T - 1$, the computational time of the KW and LC algorithms increase by a factor of 7–8 for $T = 3{,}000$ and several hundred for $T = 500{,}000$. On the other hand, the computational time of the FFT-based method only approximately doubles for either sample size. Hence, with our FFT-based algorithm, there is really no need for the truncation number and the associated approximation.

Using the exact filter to calculate the conditional variances, i.e. without truncation ($n = T - 1$), the difference in computational time between the KW and LC algorithms on the one hand and our FFT-based algorithm on the other hand is very substantial. For $T = 10{,}000$, which is a rather common sample size in finance, the FFT-based algorithm is 15–20 times faster than the KW and LC algorithms. For the largest sample size considered, $T = 500{,}000$,

[3]The Klein and Walther (2017) Matlab code was downloaded from the publisher's website.

Table 1: Computational time for different sample sizes and truncation numbers

	$n = 1{,}000$			$n = 2{,}000$		
T	KW	LC	FFT	KW	LC	FFT
3,000	0.48	0.64	**0.38**	1.54	1.72	**0.52**
5,000	0.75	0.99	**0.63**	1.56	2.44	**0.77**
10,000	1.40	1.85	**1.19**	2.19	4.25	**1.32**
25,000	2.94	4.46	**2.59**	3.80	9.70	**2.75**
50,000	6.51	8.80	**6.47**	7.48	18.8	**6.65**
100,000	15.5	17.7	**13.3**	16.5	37.1	**13.5**
500,000	88.0	88.5	**79.6**	90.4	183	**79.8**
	$n = 5{,}000$			$n = T - 1$		
T	KW	LC	FFT	KW	LC	FFT
3,000	—	—	—	4.05	4.70	**0.68**
5,000	—	—	—	12.1	14.2	**1.30**
10,000	11.1	23.9	**1.92**	37.6	51.4	**2.61**
25,000	12.6	53.4	**3.33**	205	352	**6.28**
50,000	16.2	102	**7.17**	868	1,611	**14.2**
100,000	25.3	201	**14.1**	3,440	6,610	**34.0**
500,000	99.8	988	**80.0**	147,506	293,788	**192**

Notes: Median computational time in milliseconds across 10,000 repetitions (fixed n) or 1,000 repetitions ($n = T - 1$) for different sample sizes and truncation numbers (only 100 repetitions for KW and LC with $T = 500{,}000$ and $n = T - 1$). KW is the algorithm of Klein and Walther (2017), LC is linear convolution (Listings 1,5), and FFT is our proposed algorithm (Listings 2,6). The fastest algorithm for each (T, n) pair is highlighted in bold.

the FFT-based algorithm is over 750 times faster than the KW algorithm and 1500 times faster than the standard LC algorithm.

The numbers reported in Figure 1 and Table 1 are difficult to extrapolate directly to the time required for, e.g., estimation of a model or a simulation study. This is because, in estimation, each evaluation of the likelihood function and its derivatives involves several convolutions, and this is repeated for many iterations to obtain convergence. Practical examples of such timings and extrapolations to simulations are given at the end of Section 5.2.

4 Estimation bias

Since we are able to compute the sequence of conditional variances, and hence the likelihood function, of any model with an ARCH(∞) representation very quickly using our FFT-based algorithm, we can explore the effects on the estimators from approximating the calculations. That is, we can simulate the estimation bias that results from using a fixed truncation number in the calculation of the conditional variances in (4). As an alternative to (4), Baillie et al. (1996) suggested terminating the summation in (4) at a fixed n and replacing unobserved pre-sample values of ϵ^2 by the unconditional sample variance, $T^{-1}\sum_{t=1}^{T}\epsilon_t^2$. We also consider

Table 2: Simulated bias and MCSE for FIGARCH$(1,d,1)$ model—DGP 1

		$n=1{,}000$ (P)		$n=1{,}000$		$n=T-1$	
T	Par.	Bias	MCSE	Bias	MCSE	Bias	MCSE
5,000	d	0.0808	0.0923	0.0036	0.0462	−0.0042	0.0486
	ϕ	−0.0244	0.0434	−0.0014	0.0349	−0.0036	0.0356
	β	0.0511	0.0790	0.0016	0.0597	−0.0078	0.0601
	ω	-3.1×10^{-5}	3.7×10^{-5}	1.5×10^{-5}	3.7×10^{-5}	9.7×10^{-6}	3.4×10^{-5}
10,000	d	0.0648	0.0505	0.0125	0.0310	−0.0013	0.0331
	ϕ	−0.0139	0.0250	0.0010	0.0237	−0.0017	0.0246
	β	0.0474	0.0420	0.0130	0.0338	−0.0028	0.0361
	ω	-1.6×10^{-5}	2.3×10^{-5}	2.0×10^{-5}	2.5×10^{-5}	5.2×10^{-6}	2.1×10^{-5}
25,000	d	0.0504	0.0289	0.0203	0.0207	−0.0006	0.0208
	ϕ	−0.0070	0.0146	0.0020	0.0143	−0.0007	0.0151
	β	0.0407	0.0256	0.0212	0.0218	−0.0012	0.0226
	ω	1.8×10^{-5}	2.3×10^{-5}	4.1×10^{-5}	2.4×10^{-5}	3.3×10^{-6}	1.5×10^{-5}
50,000	d	0.0436	0.0200	0.0247	0.0157	−0.0001	0.0143
	ϕ	−0.0040	0.0104	0.0020	0.0102	−0.0005	0.0107
	β	0.0372	0.0184	0.0254	0.0164	−0.0005	0.0157
	ω	5.3×10^{-5}	2.5×10^{-5}	6.9×10^{-5}	2.5×10^{-5}	2.4×10^{-6}	1.2×10^{-5}
100,000	d	0.0393	0.0149	0.0279	0.0127	4.0×10^{-5}	0.0101
	ϕ	−0.0019	0.0074	0.0020	0.0074	−0.0003	0.0077
	β	0.0352	0.0142	0.0283	0.0130	−0.0002	0.0112
	ω	9.8×10^{-5}	2.8×10^{-5}	1.1×10^{-4}	2.8×10^{-5}	1.8×10^{-6}	1.1×10^{-5}

Notes: Simulations are based on 10,000 replications. "(P)" denotes fixed truncation with pre-sample values equal to the unconditional sample variance. DGP 1 has $[d,\phi,\beta,\omega]=[0.4;0.2;0.6;0.0001]$.

this possibility, and denote the results by "(P)" (for pre-sample) in the following tables.[4]

Specifically, in this section we simulate the bias of the estimated coefficients in the baseline FIGARCH$(1,d,1)$ model for a range of sample sizes, $T\in\{5{,}000;10{,}000;25{,}000;50{,}000;100{,}000\}$, and either fixed truncation, $n=1{,}000$, or no truncation, $n=T-1$. For each sample size, we simulate 10,000 samples from the FIGARCH$(1,d,1)$ model given by (1) and (7) with $z_t\sim$ i.i.d.$\mathcal{N}(0,1)$. We consider two different parameter values in the data generating process (DGP). First, DGP 1 has $\theta=[d,\phi,\beta,\omega]'=[0.4;0.2;0.6;0.0001]'$. Second, we consider DGP 2 with $\theta=[0.4;0.28;0.68;0.0001]'$ to investigate the effect of a larger value of β on the bias.

Given observations $(\epsilon_t)_{t=1}^T$ and truncation number n, the quasi-maximum-likelihood estimator is

$$\hat{\theta}=\arg\max_{\theta}\ \log L(\theta)\quad\text{with}\quad \log L(\theta)=-\frac{1}{2}\sum_{t=1}^{T}\left(\log(\sigma_t^2)+\frac{\epsilon_t^2}{\sigma_t^2}\right), \tag{10}$$

where σ_t^2 is calculated from (4) and (9). The maximization is unconstrained, but inequality constraints on the parameters that guarantee non-negativity of the conditional variances are

[4]We are grateful to an anonymous referee for this suggestion.

Table 3: Simulated bias and MCSE for FIGARCH$(1, d, 1)$ model—DGP 2

		$n = 1{,}000$ (P)		$n = 1{,}000$		$n = T - 1$	
T	Par.	Bias	MCSE	Bias	MCSE	Bias	MCSE
5,000	d	0.1123	0.1098	0.0087	0.0465	−0.0046	0.0497
	ϕ	−0.0502	0.0578	−0.0029	0.0325	−0.0009	0.0337
	β	0.0577	0.0622	0.0061	0.0397	−0.0047	0.0423
	ω	-4.2×10^{-5}	2.7×10^{-5}	1.1×10^{-5}	3.1×10^{-5}	8.8×10^{-6}	3.0×10^{-5}
10,000	d	0.0886	0.0673	0.0204	0.0330	−0.0019	0.0350
	ϕ	−0.0353	0.0347	−0.0049	0.0224	−0.0007	0.0235
	β	0.0493	0.0404	0.0148	0.0276	−0.0023	0.0296
	ω	-2.6×10^{-5}	2.4×10^{-5}	1.5×10^{-5}	2.5×10^{-5}	5.4×10^{-6}	2.2×10^{-5}
25,000	d	0.0684	0.0347	0.0315	0.0228	−0.0009	0.0223
	ϕ	−0.0239	0.0178	−0.0074	0.0144	−0.0003	0.0146
	β	0.0413	0.0230	0.0227	0.0183	−0.0010	0.0186
	ω	-6.1×10^{-6}	2.3×10^{-5}	3.3×10^{-5}	2.3×10^{-5}	3.4×10^{-6}	1.6×10^{-5}
50,000	d	0.0597	0.0232	0.0373	0.0178	−0.0003	0.0154
	ϕ	−0.0193	0.0123	−0.0090	0.0106	−0.0004	0.0104
	β	0.0376	0.0165	0.0266	0.0142	−0.0005	0.0132
	ω	-4.0×10^{-5}	2.4×10^{-5}	5.8×10^{-5}	2.4×10^{-5}	2.3×10^{-6}	1.3×10^{-5}
100,000	d	0.0547	0.0176	0.0415	0.0149	-3.4×10^{-5}	0.0111
	ϕ	−0.0166	0.0100	−0.0103	0.0095	−0.0003	0.0079
	β	0.0354	0.0136	0.0291	0.0122	−0.0003	0.0097
	ω	-8.3×10^{-5}	2.8×10^{-5}	9.4×10^{-5}	2.8×10^{-5}	1.8×10^{-6}	1.1×10^{-5}

Notes: Simulations are based on 10,000 replications. "(P)" denotes fixed truncation with pre-sample values equal to the unconditional sample variance. DGP 2 has $[d, \phi, \beta, \omega] = [0.4; 0.28; 0.68; 0.0001]$.

checked post-estimation; see Baillie et al. (1996, footnote 19) and Conrad and Haag (2006).

The Monte Carlo simulation results can be found in Tables 2 and 3 for DGPs 1 and 2, respectively. For each sample size and truncation number, we report the bias and Monte Carlo standard error (MCSE) for the four parameters.

We first observe from Tables 2 and 3 that replacing unobserved pre-sample observations with unconditional variance estimates appears to induce a large bias. Indeed, the biases in the (P) column are substantially larger than in other columns, and particularly the estimates of d and β are heavily upwards biased. It would appear that adding the effect of many (up to $n - 1$) constant observations near the start of the sample, but none later, induces a type of non-stationarity which biases estimates of d and β upwards. Johansen and Nielsen (2016) analyze bias in an ARFIMA model using higher-order asymptotic theory, and show that inclusion of a level parameter (in our model this is ω) essentially eliminates all bias coming from pre-sample values, so there appears to be no need for plugging in arbitrary constant pre-sample values. A similar mechanism may apply in our setup.

Next, disregarding the (P) column, we see that the truncation at $n = 1{,}000$ induces a

Table 4: Simulated out-of-sample forecast RMSE (×1,000) for FIGARCH$(1, d, 1)$ model

	DGP 1			DGP 2		
T	$n = 1{,}000$ (P)	$n = 1{,}000$	$n = T - 1$	$n = 1{,}000$ (P)	$n = 1{,}000$	$n = T - 1$
5,000	1.9797	1.4328	1.2003	2.2205	1.5488	1.2619
10,000	1.8057	1.4575	0.9092	2.0225	1.5901	0.9440
25,000	1.8871	1.7448	0.7159	2.1068	1.9261	0.7488
50,000	2.2862	2.2152	0.5517	2.5440	2.4401	0.5827
100,000	2.5670	2.5736	0.4298	2.8311	2.8332	0.4461

Notes: We conducted 500 expanding window one-step-ahead out-of-sample forecasts and report the resulting forecast RMSE averaged across 100 simulation replications. "(P)" denotes fixed truncation with pre-sample values equal to the unconditional sample variance. DGP 1 has $[d, \phi, \beta, \omega] = [0.4; 0.2; 0.6; 0.0001]$ and DGP 2 has $[d, \phi, \beta, \omega] = [0.4; 0.28; 0.68; 0.0001]$.

substantial bias that increases with the sample size, T. This is expected since a larger sample size implies that a fixed truncation affects a larger proportion of the sample. The parameters d and β seem particularly affected by the truncation. The biases in these estimators are likely several parameter standard errors in magnitude for the larger sample sizes.

Comparing across Tables 2 and 3 we note that biases are substantially larger for DGP 2 (in Table 3) compared with DGP 1 (in Table 2). It appears that, as the β parameter increases, it is more difficult to identify a given d from highly persistent autoregressive-type dynamics, thus increasing the bias of both d and β.[5]

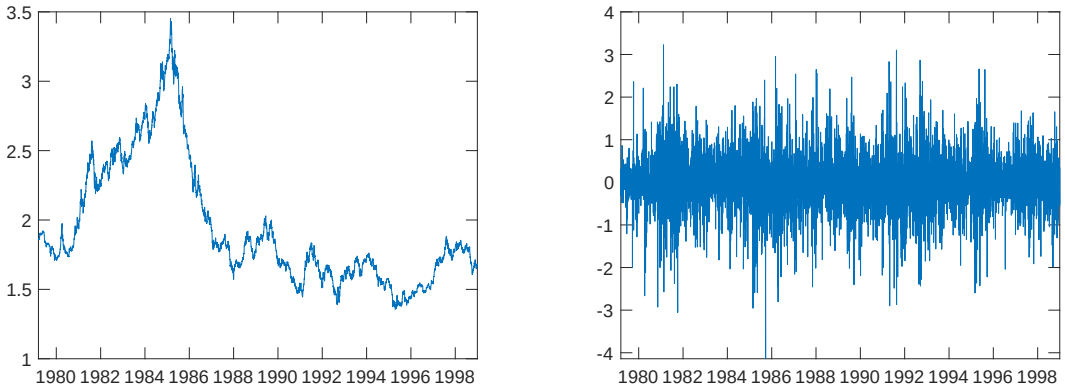
Finally, we conduct a small out-of-sample forecast comparison. For each sample, we estimate the model based on the first $T-k-1$ observations and calculate the one-step ahead forecast of observation $T - k$. We repeat this for $k = 1, \ldots, 500$ and calculate the forecast root-mean-squared-error (RMSE) for these 500 one-step-ahead forecasts. We then average the RMSE across 100 simulation replications and report the results (×1,000) in Table 4 for a range of sample sizes, truncation numbers, and the two DGPs.

In Table 4 we first note that truncation implies less precise forecasts. For each DGP and for each sample size, the smallest forecast RMSE is obtained by using the untruncated filter. The difference in forecast RMSE between truncation at $n = 1{,}000$ and no truncation is always in favor of no truncation and varies from a reduction of about 16% (for DGP 1 and $T = 5{,}000$) to over 80% (for DGP 2 and $T = 100{,}000$). More generally, the results in Table 4 reflect those in Tables 2 and 3, in the sense that (i) setting pre-sample values equal to the unconditional variance estimate increases forecast RMSE, (ii) forecast RMSE is higher for DGP 2 compared with DGP 1, and (iii) the difference in forecast RMSE between truncation at $n = 1{,}000$ and no truncation increases with the sample size.

In conclusion, the results of our Monte Carlo simulations in Tables 2–4 clearly illustrate the consequences of the approximation implied by truncating the convolution as in (4). The resulting bias in the estimators is substantial, and it is worse for larger sample sizes. Furthermore, the truncation and associated bias result in less precise out-of-sample forecasts. Importantly, however, these problems are easily avoidable by application of our proposed

[5]We are grateful to an anonymous referee for suggesting a second DGP with a larger value of β based on this motivation.

Figure 2: Daily DEM-USD exchange rate 3/13/1979–12/31/1998



(a) spot rate

(b) return

FFT-based algorithm, which is exact and does not rely on any approximation or truncation.

5 Empirical illustrations

The persistence in the volatility of nominal exchange rates has been well-documented in the literature, and exchange rates are often modeled using FIGARCH-type models. Indeed, a daily time series of Deutsche Mark-US dollar (DEM-USD) exchange rates was used as the empirical example in Baillie et al. (1996). An updated data set, covering until the end of the DEM era and start of the Euro, was subsequently analyzed in Baillie, Cecen, and Han (2000). In our first empirical illustration, we consider this data set. As a second empirical illustration, we consider a longer time series of daily US Dollar-British Pound (USD-GBP) exchange rates.

5.1 Deutsche Mark-US Dollar exchange rate from 1979 to 1998

In this subsection, we analyze the daily DEM-USD exchange rate from March 13, 1979 to December 31, 1998, for a total of 4,975 spot rate observations.[6] In Figure 2(a) we plot the spot rate and in Figure 2(b) we plot the continuously compounded percentage returns. It is clear from the figure that the spot exchange rate is nonstationary. Indeed, standard arbitrage arguments would dictate that the spot rate should be a Martingale, which justifies modeling the returns as serially uncorrelated as in (1).

Table 5 reports the quasi-maximum-likelihood estimates for the FIGARCH$(1, d, 1)$ and FIGARCH$(1, d, 0)$ models with either no truncation or with truncation at $n = 1{,}000$ (with or without setting pre-sample value equal to unconditional variance estimates). Several interesting findings appear from these results. First, regardless of the truncation number, the additional parameter in the $(1, d, 1)$ specification compared with the $(1, d, 0)$ specification is highly significant as measured by a t-test on the additional ϕ parameter. Thus, we consider mainly the FIGARCH$(1, d, 1)$ model. Second, for this model and data set, we note that the parameter estimates for d, and to a lesser extent β do vary with the truncation number. Third, with truncation at $n = 1{,}000$ the computing times for our FFT-based method are

[6]The Euro was introduced on January 1, 1999. Our data set was downloaded from the Federal Reserve H.10 historical data website, and covers the same time period as that analyzed in Baillie et al. (2000), but has a slightly different number of observations.

Table 5: FIGARCH models for DEM-USD exchange rate (1979–1998)

	$n = 1{,}000$ (P)		$n = 1{,}000$		$n = T - 1$	
(p, d, q)	$(1, d, 1)$	$(1, d, 0)$	$(1, d, 1)$	$(1, d, 0)$	$(1, d, 1)$	$(1, d, 0)$
μ	0.0033	0.0032	0.0038	0.0033	0.0041	0.0035
	(0.0087)	(0.0087)	(0.0086)	(0.0086)	(0.0086)	(0.0086)
d	0.4544	0.2860	0.4334	0.2741	0.4794	0.2854
	(0.0186)	(0.0180)	(0.0194)	(0.0176)	(0.0192)	(0.0168)
ϕ	0.2929	—	0.2984	—	0.2801	—
	(0.0214)		(0.0222)		(0.0213)	
β	0.6689	0.2143	0.6530	0.2026	0.6778	0.2130
	(0.0146)	(0.0234)	(0.0152)	(0.0236)	(0.0155)	(0.0232)
ω	0.0131	0.0519	0.0179	0.0613	0.0147	0.0511
	(0.0034)	(0.0079)	(0.0036)	(0.0086)	(0.0042)	(0.0095)
Iterations	34	16	34	16	37	17
Fcn. evals.	230	108	236	105	236	105
Time FFT	191	92	193	88	356	155
Time LC	311	174	277	150	3,356	1,225

Notes: Quasi-maximum-likelihood estimates for FIGARCH(p, d, q) models for the DEM-USD percentage returns from March 14, 1979 to December 31, 1998 for different truncation numbers and $T = 4{,}974$ return observations. Standard errors are reported in parentheses, and "(P)" denotes fixed truncation with pre-sample values equal to the unconditional sample variance. Finally, the number of iterations and the total number of function evaluations are reported, along with the computing time in milliseconds using the FFT-based method and the linear convolution (LC) method.

about two-thirds of those for the LC method, but without truncation ($n = T - 1$) the computing time for our FFT-based method is almost ten times faster than for the LC method.

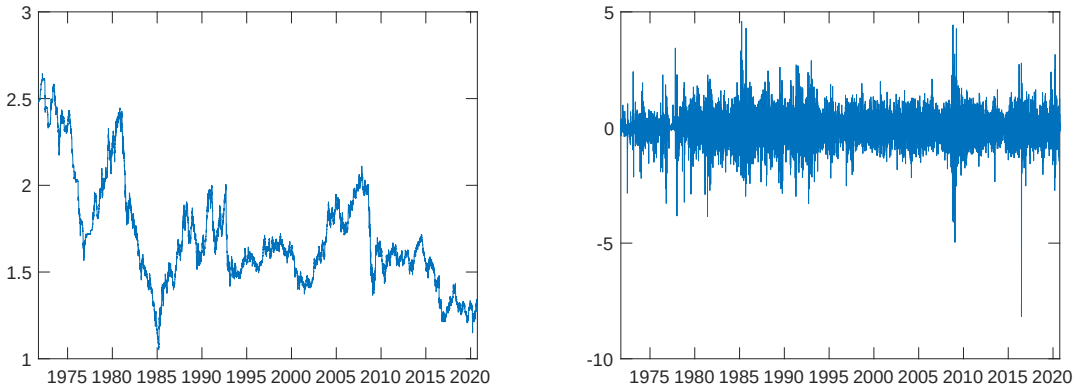
5.2 US Dollar-British Pound exchange rate from 1971 to 2020

In this subsection, we analyze the daily USD-GBP spot exchange rate from October 1, 1971 to October 2, 2020, for a total of 12,300 spot rate observations. The data series was downloaded from the FRED database and is plotted in Figure 3. As in the previous subsection, the spot exchange rate is nonstationary and we model the continuously compounded returns.

The estimation results for the USD-GBP data set are presented in Table 6, which is laid out precisely as Table 5. The results are qualitatively quite similar, except they are more pronounced in Table 6 because of the longer sample.

First of all, the FIGARCH$(1, d, 1)$ specification is again statistically superior to the FIGARCH$(1, d, 0)$ specification, regardless of truncation number. Secondly, the estimates for d, and again to a lesser extent for β, are now quite different for the three different FIGARCH$(1, d, 1)$ estimates. The estimates in the "(P)" column are larger than the others, as expected based the simulation findings in Section 4. Comparing truncation at $n = 1{,}000$, which is standard in the literature, with no truncation, the estimates of d differ by about one standard error. The parameter ω, which is usually not of primary interest, differs even more.

Figure 3: Daily USD-GBP exchange rate 10/1/1971–10/2/2020



(a) spot rate (b) return

Table 6: FIGARCH models for USD-GBP exchange rate (1971–2020)

	$n = 1{,}000$ (P)		$n = 1{,}000$		$n = T - 1$	
(p, d, q)	$(1, d, 1)$	$(1, d, 0)$	$(1, d, 1)$	$(1, d, 0)$	$(1, d, 1)$	$(1, d, 0)$
μ	−0.0004	0.0007	0.0012	0.0020	0.0016	0.0025
	(0.0048)	(0.0045)	(0.0049)	(0.0046)	(0.0053)	(0.0050)
d	0.4948	0.3257	0.4230	0.3123	0.4459	0.3136
	(0.0182)	(0.0140)	(0.0177)	(0.0132)	(0.0194)	(0.0143)
ϕ	0.2532	—	0.2633	—	0.2535	—
	(0.0177)		(0.0167)		(0.0170)	
β	0.6543	0.2562	0.5970	0.2429	0.6036	0.2377
	(0.0165)	(0.0174)	(0.0163)	(0.0170)	(0.0172)	(0.0175)
ω	0.0113	0.0311	0.0140	0.0327	0.0115	0.0250
	(0.0037)	(0.0070)	(0.0042)	(0.0073)	(0.0050)	(0.0091)
Iterations	34	20	30	20	31	19
Fcn. evals.	238	132	226	142	238	137
Time FFT	370	208	356	281	715	493
Time LC	664	422	654	466	16,581	9,272

Notes: Quasi-maximum-likelihood estimates for FIGARCH(p, d, q) models for the USD-GBP percentage returns from October 4, 1971 to October 2, 2020 for different truncation numbers and $T = 12{,}299$ return observations. Standard errors are reported in parentheses, and "(P)" denotes fixed truncation with pre-sample values equal to the unconditional sample variance. Finally, the number of iterations and the total number of function evaluations are reported, along with the computing time in milliseconds using the FFT-based method and the linear convolution (LC) method.

The computing time using our FFT-based method is now much faster than the LC method. Without truncation, our FFT-based method is about 23 times faster than the LC method for the FIGARCH$(1, d, 1)$ model. As an order of magnitude, a simulation study or a bootstrap with 10,000 replications for the FIGARCH$(1, d, 1)$ model in Table 6 without

truncation would require 46 hours with the LC method but less than 2 hours with our FFT-based algorithm. This result confirms those found in Section 3, and could certainly be the difference between bootstrap inference or simulations being feasible or not.

6 Conclusion

In this paper we have considered the computation of the time series of conditional variances for models in the ARCH(∞) class. This class is very large and contains many commonly applied models in empirical finance such as the FIGARCH model. For models in the ARCH(∞) class, we have provided an exact algorithm for efficient computation of the conditional variances, and hence of the likelihood function. Our algorithm is based on the fast Fourier transform (FFT) and achieves an order of magnitude improvement in computational speed from $O(T^2)$ to $O(T \log T)$. The efficiency of the algorithm allows estimation, as well as simulation and/or bootstrapping of ARCH(∞) models, even with very large data sets. We have focused on univariate models, but multivariate models will achieve a proportional increase in computational speed.

Furthermore, to speed up computation it has been completely standard in the literature to resort to a truncation of the ARCH(∞) model at a fixed truncation lag. With our proposed algorithm, this is not needed. As additional contributions, we also showed that the elimination of the truncation substantially reduces both the bias of the quasi-maximum-likelihood estimators and the out-of-sample forecast errors.

We illustrated our results with two empirical examples that highlighted both the differences in the computational speed and in the parameter estimates for our FFT-based algorithm and the standard algorithm with and without truncation.

As sample sizes in economics and finance have become longer in recent years (see examples in the introduction), efficient computation has become increasingly more important. As we have shown, this is certainly true for daily time series, and when the models discussed in this paper are applied to high-frequency data this becomes crucially important. Furthermore, the gaining popularity of computationally intensive methods of inference, based on simulation, bootstrapping, or machine-learning techniques, that require estimation of (variations of) each model a large number of times, emphasizes these points even more.

Data availability statement

The daily DEM-USD exchange rate data that support the findings of this study are openly available in the Federal Reserve H.10 historical data website at https://www.federalreserve.gov/releases/H10/hist/dat89_ge.htm and https://www.federalreserve.gov/releases/H10/hist/dat96_ge.htm. The daily USD-GBP exchange rate data that support the findings of this study are openly available in the FRED database at https://fred.stlouisfed.org/series/DEXUSUK. The data files are included in the online supplementary materials together with the computer programs listed in the paper. The latter are also available on the authors' websites.

References

Baillie, R. T., T. Bollerslev, and H. O. Mikkelsen (1996). Fractionally integrated generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 74*, 3–30.

Baillie, R. T., A. Cecen, and Y. W. Han (2000). High frequency Deutsche Mark-US Dollar returns: FIGARCH representations and non linearities. *Multinational Finance Journal 4*, 247–267.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 31*, 307–327.

Bollerslev, T. and H. O. Mikkelsen (1996). Modeling and pricing long memory in stock market volatility. *Journal of Econometrics 73*, 151–184.

Chortareas, G., Y. Jiang, and J. C. Nankervis (2011). Forecasting exchange rate volatility using high-frequency data: Is the Euro different? *International Journal of Forecasting 27*, 1089–1107.

Conrad, C. and B. R. Haag (2006). Inequality constraints in the fractionally integrated GARCH model. *Journal of Financial Econometrics 4*, 413–449.

Conrad, C., D. Rittler, and W. Rotfuß (2012). Modeling and explaining the dynamics of European Union Allowance prices at high-frequency. *Energy Economics 34*, 316–326.

Cooley, J. W., P. A. W. Lewis, and P. D. Welch (1969). The fast Fourier transform and its applications. *IEEE Transactions on Education 12*, 27–34.

Cooley, J. W. and J. W. Tukey (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation 19*, 297–301.

Davidson, J. (2004). Moment and memory properties of linear conditional heteroscedasticity models, and a new model. *Journal of Business & Economic Statistics 22*, 16–29.

Giraitis, L., P. M. Robinson, and D. Surgailis (2000). A model for long memory conditional heteroscedasticity. *Annals of Applied Probability 10*, 1002–1024.

Han, Y. W. (2008). Intraday effects of macroeconomic shocks on the US Dollar-Euro exchange rates. *Japan and the World Economy 20*, 585–600.

Jensen, A. N. and M. Ø. Nielsen (2014). A fast fractional difference algorithm. *Journal of Time Series Analysis 35*, 428–436.

Johansen, S. and M. Ø. Nielsen (2016). The role of initial values in conditional sum-of-squares estimation of nonstationary fractional time series models. *Econometric Theory 32*, 1095–1139.

Karanasos, M., Z. Psaradakis, and M. Sola (2004). On the autocorrelation properties of long-memory GARCH processes. *Journal of Time Series Analysis 25*, 265–282.

Klein, T. and T. Walther (2017). Fast fractional differencing in modeling long memory of conditional variance for high-frequency data. *Finance Research Letters 22*, 274–279.

Naeema, M., M. Shahbazb, K. Saleemc, and F. Mustafaa (2019). Risk analysis of high frequency precious metals returns by using long memory model. *Resources Policy 61*, 399–409.

Robinson, P. M. (1991). Testing for strong serial correlation and dynamic conditional heteroskedasticity in multiple regression. *Journal of Econometrics 47*, 67–84.

Stockham, T. G. (1966). High-speed convolution and correlation. *Proceedings of the Spring Joint Computer Conference 28*, 229–233.

Tse, Y. K. (1998). The conditional heteroscedasticity of the yen-dollar exchange rate. *Journal of Applied Econometrics 13*, 49–55.

Zygmund, A. (2003). *Trigonometric Series, vol. I and II, 3rd rev. ed.* Cambridge, UK: Cambridge University Press.

Research Papers 2020

CREATES
Center for Research in Econometric Analysis of Time Series

2019-21: Mikkel Bennedsen, Eric Hillebrand and Siem Jan Koopman: Modeling, Forecasting, and Nowcasting U.S. CO2 Emissions Using Many Macroeconomic Predictors

2019-22: Anne G. Balter, Malene Kallestrup-Lamb and Jesper Rangvid: The move towards riskier pensions: The importance of mortality

2019-23: Duván Humberto Cataño, Carlos Vladimir Rodríguez-Caballero and Daniel Peña: Wavelet Estimation for Dynamic Factor Models with Time-Varying Loadings

2020-01: Mikkel Bennedsen: Designing a sequential testing procedure for verifying global CO2 emissions

2020-02: Juan Carlos Parra-Alvarez, Hamza Polattimur and Olaf Posch: Risk Matters: Breaking Certainty Equivalence

2020-03: Daniel Borup, Bent Jesper Christensen, Nicolaj N. Mühlbach and Mikkel S. Nielsen: Targeting predictors in random forest regression

2020-04: Nicolaj N. Mühlbach: Tree-based Synthetic Control Methods: Consequences of moving the US Embassy

2020-05: Juan Carlos Parra-Alvarez, Olaf Posch and Mu-Chun Wang: Estimation of heterogeneous agent models: A likelihood approach

2020-06: James G. MacKinnon, Morten Ørregaard Nielsen and Matthew D. Webb: Wild Bootstrap and Asymptotic Inference with Multiway Clustering

2020-07: Javier Hualde and Morten Ørregaard Nielsen: Truncated sum of squares estimation of fractional time series models with deterministic trends

2020-08: Giuseppe Cavaliere, Morten Ørregaard Nielsen and Robert Taylor: Adaptive Inference in Heteroskedastic Fractional Time Series Models

2020-09: Daniel Borup, Jonas N. Eriksen, Mads M. Kjær and Martin Thyrsgaard: Predicting bond return predictability

2020-10: Alfonso A. Irarrazabal, Lin Ma and Juan Carlos Parra-Alvarez: Optimal Asset Allocation for Commodity Sovereign Wealth Funds

2020-11: Bent Jesper Christensen, Juan Carlos Parra-Alvarez and Rafael Serrano: Optimal control of investment, premium and deductible for a non-life insurance company

2020-12: Anine E. Bolko, Kim Christensen, Mikko S. Pakkanen and Bezirgen Veliyev: Roughness in spot variance? A GMM approach for estimation of fractional log-normal stochastic volatility models using realized measures

2020-13: Morten Ørregaard Nielsen and Antoine L. Noël: To infinity and beyond: Efficient computation of ARCH(∞) models