

betategarch: Simulation, estimation and forecasting of Beta-Skew-t-EGARCH Models

by Genaro Sucarrat

Abstract This paper illustrates the usage of the `betategarch` package, a package for the simulation, estimation and forecasting of Beta-Skew-t-EGARCH models. The Beta-Skew-t-EGARCH model is a dynamic model of the scale or volatility of financial returns. The model is characterised by its robustness to jumps or outliers, and by its exponential specification of volatility. The latter enables richer dynamics, since parameters need not be restricted to be positive to ensure positivity of volatility. In addition, the model also allows for heavy tails and skewness in the conditional return (i.e. scaled return), and for leverage and a time-varying long-term component in the volatility specification. More generally, the model can be viewed as a model of the scale of the error in a dynamic regression.

Introduction

It is well known that financial returns are characterised by volatility clustering: Large returns in absolute value are likely to be followed by other large returns in absolute value, and small returns in absolute value are likely to be followed by other small returns in absolute value. This characteristic is usually modelled by specifications in the Autoregressive Conditional Heteroscedasticity (ARCH) class of models by Engle (1982). If y_t denotes the financial return at t such that

$$y_t = \sigma_t \varepsilon_t, \quad \varepsilon_t \sim \text{IID}(0, \sigma_\varepsilon^2), \quad \sigma_t^2 = h(\sigma_{t-1}, y_{t-1}, \dots), \quad t = 1, 2, \dots,$$

then the scale or volatility $\sigma_t > 0$ is said to follow an ARCH process. Arguably, the most popular ARCH specification is the first order Generalised ARCH (GARCH) of Bollerslev (1986), where σ_t^2 is modelled in an ARMA(1,1)-like manner, $\sigma_t^2 = \omega + \phi_1 \sigma_{t-1}^2 + \kappa_1 y_{t-1}^2$ with $\sigma_\varepsilon^2 = 1$, see Francq and Zakoian (2010) for a recent survey of GARCH models. If the financial return in question is predictable, then y_t can be interpreted as de-meaned return, i.e. the unpredictable part of return. But more generally y_t can be viewed as the error-term in a dynamic regression. Three characteristics that are often exhibited by financial returns are leverage (i.e. volatility asymmetry), conditional fat-tailedness and conditional skewness. The former means volatility tends to be higher after negative returns – this is typically attributed to leverage (hence the name), whereas conditional fat-tailedness means the standardised conditional return (i.e. ε_t) is more fat-tailed than the Gaussian. Conditional skewness means the standardised return is not symmetric. For stock returns the skewness is typically negative, which means the probability of a large negative return is greater than a large positive return, even after controlling or adjusting for the recent level of volatility.

Several *R* packages provide facilities for the estimation and forecasting of univariate GARCH models that contains one or more of these features. Arguably, the three most important packages are `tseries` by Trapletti and Hornik (2013), `fGarch` by Wuertz et al. (2013) and `rugarch` by Ghalanos (2013). The `tseries` package probably has the fastest GARCH optimiser, but does not offer state-of-the-art specifications with leverage and fat-tailed skewed densities. This, by contrast, is provided by the `fGarch` and `rugarch` packages. The former has been considered by many – including myself – as the benchmark package in *R* for quite a while, since it provides a wide range of GARCH models coupled with a variety of densities. But unfortunately `fGarch` does not offer the possibility to estimate Exponential GARCH (EGARCH) models, i.e. models where the dynamics is specified in terms of $\ln \sigma_t^2$ rather than in terms of σ_t^2 . EGARCH models are attractive, since they enable richer and more flexible dynamics (parameters can be negative), and since the autocorrelation function of returns depends on the conditional density (this is non the case for non-exponential ARCH models). The `rugarch` package, which despite its relative recent origin (available on CRAN since September 2011) already offers an impressive range of GARCH specifications and variations of these, fills this gap to some extent by providing Nelson's (1991) EGARCH model. Another package that partially fills this gap is `AutoSEARCH` by Sucarrat (2012), which offers estimation and automated General-to-Specific model selection of log-ARCH-X models. However, to the best of my knowledge there are no other *R* packages that provide facilities for additional EGARCH models. The `betategarch` package thus contributes to the *R* world by offering utilities for the simulation, estimation and forecasting of Beta-t-EGARCH models.

48 The Beta-t-EGARCH model was first proposed by Harvey (2013) and Harvey and Chakravarty
 49 (2008), but it can also be viewed as an unrestricted version of the Generalised Autoregressive Score
 50 (GAS) model of Creal et al. (2013). The code which **betategarch** is based upon was originally developed
 51 for Harvey and Sucarrat (2013), which extends the Beta-t-EGARCH model to the skewed case. The
 52 Beta-Skew-t-EGARCH model has a number of attractions. First, the model is robust to jumps or
 53 outliers, and fares very well empirically for a variety of financial returns when compared with other
 54 GARCH models, see Harvey and Sucarrat (2013). Second, the model accommodates the most important
 55 characteristics of time-varying financial volatility: Leverage, conditional fat-tailedness, conditional
 56 skewness and a decomposition of volatility into a short-term and a long-term component. Third, the
 57 unconditional moments of return (i.e. y_t) exist, which is important in long-term forecasting and for the
 58 computation of the autocorrelation function of returns. By contrast, this is generally not the case for
 59 Nelson's (1991) EGARCH when coupled with a t density: A necessary condition for the unconditional
 60 moments of the first-order specification to exist when ε_t is t is that the ARCH parameter is negative,
 61 see condition (A1.6) and the subsequent discussion in Nelson (1991, p. 365). Moreover, in the presence
 62 of leverage the ARCH parameter must be even more negative for the unconditional moments to
 63 exist. This is why Nelson (1991) proposed his model with a Generalised Error Distribution (GED)
 64 instead of a t . Fourth, the asymptotic properties are much easier to derive than those of Nelson's (1991)
 65 EGARCH, see Straumann and Mikosch (2006) and Wintenberger (2012) for a detailed description of the
 66 difficulties. Finally, since the conditional score drives the dynamics of the model, the Beta-t-EGARCH
 67 acquires some attractive theoretical properties. In particular, a simple transformation of the score is
 68 Beta-distributed (hence the name).

69 The two main functions of the **betategarch** package (version 3.1) are `tegarchSim` and `tegarch`. The
 70 first simulates from a Beta-Skew-t-EGARCH, whereas the latter estimates one. The `tegarch` function
 71 returns an object (a list) of the `tegarch` class, and a collection of S3 methods developed for this class can
 72 be applied to such objects: `coef`, `fitted`, `logLik`, `predict`, `print`, `residuals`, `summary` and `vcov`. The
 73 rest of the functions in the package are either auxiliary functions called by the two main functions, or
 74 a dataset, `nasdaq`, which is included for illustration purposes (see empirical example below). Finally, it
 75 is worth noting that the objects returned by `tegarchSim` function and the `fitted` and `predict` methods
 76 are of the class `zoo` defined in package `zoo`, see Zeileis et al. (2013). This means a large range of useful
 77 time-series methods and facilities can be applied to these objects, including plotting and printing
 78 methods.

79 The one-component Beta-Skew-t-EGARCH

80 The martingale difference version of the first order one-component Beta-Skew-t-EGARCH model (see
 81 sections 4 and 6 in Harvey and Sucarrat (2013)) is given by

$$y_t = \exp(\lambda_t)\varepsilon_t = \sigma_t\varepsilon_t, \quad \varepsilon_t \sim st(0, \sigma_\varepsilon^2, \nu, \gamma), \quad \nu > 2, \quad \gamma \in (0, \infty), \quad (1)$$

$$\lambda_t = \omega + \lambda_t^\dagger, \quad (2)$$

$$\lambda_t^\dagger = \phi_1\lambda_{t-1}^\dagger + \kappa_1 u_{t-1} + \kappa^* \text{sgn}(-y_{t-1})(u_{t-1} + 1), \quad |\phi_1| < 1. \quad (3)$$

82 The σ_t is the conditional scale or volatility, which need not equal the conditional standard deviation.
 83 In other words, ε_t is not standardised to have variance one. The conditional standard deviation is
 84 obtained as $\sigma_t\sigma_\varepsilon$, where σ_ε^2 is the variance of ε_t . The conditional error ε_t is distributed as a skewed t
 85 with zero mean, scale σ_ε^2 , degrees of freedom parameter ν and skewness parameter γ . The conditional
 86 error is defined as $\varepsilon_t = \varepsilon_t^* - \mu_{\varepsilon^*}$, where ε_t^* is an uncentred (i.e. mean not necessarily equal to zero)
 87 Skewed t variable with ν degrees of freedom, skewness parameter γ and mean μ_{ε^*} . A centred and
 88 symmetric (i.e. ordinary) t -distributed variable with mean zero is obtained when $\gamma = 1$, in which
 89 $\mu_{\varepsilon^*} = 0$, whereas a left-skewed (right-skewed) t -variable is obtained when $\gamma < 1$ ($\gamma > 1$). More
 90 details on the distribution is given below. The ω is the log-scale intercept and can be interpreted
 91 as the long-term log-volatility, ϕ_1 is the persistence parameter (the bigger, the more clustering), κ_1
 92 is the ARCH parameter (the bigger in absolute value, the greater the response to shocks), u_t is the
 93 conditional score (i.e. the derivative of the log-likelihood of y_t at t with respect to λ_t) and κ^* is the
 94 leverage parameter. A sufficient condition for stability in λ_t is $|\phi_1| < 1$.

95 Let ε^* denote an ordinary (i.e. centred and symmetric) t -distributed variable (with unit scale),
 96 and let $f(\varepsilon^*)$ denote its density. By means of the skewing method of Fernández and Steel (1998), the
 97 density of an uncentred Skewed t variable can be written as

$$f(\varepsilon^*|\gamma) = \frac{2}{\gamma + \gamma^{-1}} f\left(\frac{\varepsilon^*}{\gamma \text{sgn}(\varepsilon^*)}\right). \quad (4)$$

98 Computation of the density values and of the mean of an uncentred Skewed t variable, and random

99 number generation, can be undertaken with `dST`, `STmean` and `rST`, respectively. For example, the
 100 following code compares the empirical average of ten thousand random numbers with the analytical
 101 mean:

```
library(betategarch)
set.seed(123)
eps <- rST(10000, df=5, skew=0.7)
mean(eps)
[1] -0.69805
STmean(df=5, skew=0.7)
[1] -0.6914265
```

102 In addition, the functions `STvar`, `STskewness` and `STkurtosis` return the analytical values of the
 103 variance, skewness (the standardised 3rd moment) and kurtosis (the standardised 4th moment),
 104 respectively, of an uncentered Skewed t variable.

105 The conditional score of the martingale difference version of the Beta-Skew-t-EGARCH model (see
 106 Harvey and Sucarrat (2013, equation (32) in section 4)) is given by

$$\begin{aligned} \frac{\partial \ln f_y(y_t)}{\partial \lambda_t} &= u_t \\ &= \frac{(\nu + 1)[y_t^2 + y_t \mu_{\varepsilon^*} \exp(\lambda_t)]}{\nu \exp(2\lambda_t) \gamma^{2\text{sgn}(y_t + \mu_{\varepsilon^*} \exp(\lambda_t))} + (y_t + \mu_{\varepsilon^*} \exp(\lambda_t))^2} - 1. \end{aligned} \quad (5)$$

It is useful to note, however, that for simulation purposes the score u_t can also be written more conveniently as

$$u_t = \frac{(\nu + 1)(\varepsilon_t^{*2} - \mu_{\varepsilon^*} \varepsilon_t^*)}{\nu \gamma^{2\text{sgn}(\varepsilon_t^*)} + \varepsilon_t^{*2}} - 1,$$

107 where ε_t^* is an uncentred Skewed t variable. When the conditional distribution is symmetric (i.e.
 108 $\gamma = 1$), then $\frac{u_t + 1}{\nu + 1} \sim \text{Beta}(1/2, \nu/2)$. This explains the origin of the name Beta-t-EGARCH.

109 Financial returns that follow the one-component Beta-Skew-t-EGARCH model given by (1)-(3) can
 110 be simulated with the `tegarchSim` function. For example, in order to generate two thousand returns
 111 from a specification with empirically plausible values on ω , ϕ_1 , κ_1 and ν , but without leverage and
 112 skewness, then the following code can be used:

```
y1 <- tegarchSim(2000, omega=0.1, phi1=0.95, kappa1=0.05, df=10)
```

113 Similarly, the following three commands each generate two thousand returns with, respectively,
 114 moderate leverage, strong left-skewness, and both moderate leverage and strong left-skewness:

```
y2 <- tegarchSim(2000, omega=0.1, phi1=0.95, kappa1=0.05, kappastar=0.02, df=10)
y3 <- tegarchSim(2000, omega=0.1, phi1=0.95, kappa1=0.05, df=10, skew=0.8)
y4 <- tegarchSim(2000, omega=0.1, phi1=0.95, kappa1=0.05, kappastar=0.02, df=10,
  skew=0.8)
```

115 By default the `tegarchSim` function returns the values of y_t only. However, for the full set of output
 116 one may use the `verbose` option. For example, the following code generates 100 observations using
 117 the default parameter values, stores the output in the matrix `mY` which is of class `zoo` and returns the
 118 first six observations:

```
mY <- tegarchSim(100, verbose=TRUE)
head(mY)
```

	y	sigma	stdev	lambda	lambdadagg	u	epsilon
1	0.19977534	1.0000000	1.118034	0.000000000	0.000000000	-0.9562733	0.19977534
2	-1.35118283	0.9904828	1.107393	-0.009562733	-0.009562733	0.7258681	-1.36416581
3	0.15475640	0.9981758	1.115994	-0.001825916	-0.001825916	-0.9736225	0.15503923
4	-0.04853563	0.9885947	1.105282	-0.011470845	-0.011470845	-0.9973492	-0.04909558
5	0.48034223	0.9793455	1.094942	-0.020870795	-0.020870795	-0.7415964	0.49047270
6	0.39742433	0.9731245	1.087986	-0.027243220	-0.027243220	-0.8195400	0.40840028

119 The last column named "epsilon" contains the centred Skewed t variable ε_t as defined in (1). The
 120 zeros for λ_t and λ_t^\dagger in the first row are due to the default initial value. This can be changed via the
 121 `lambda.initial` option.

122 The two-component Beta-Skew-t-EGARCH

123 Squared financial return often exhibit long-memory, see [Ding et al. \(1993\)](#), [Ding and Granger \(1996\)](#).
 124 Two-component models of volatility accommodates the long-memory property by decomposing
 125 volatility into one long-term component and one short term component. The role of the latter is to
 126 pick up temporary changes following a shock.

127 The martingale difference version of the first order two-component Beta-Skew-t-EGARCH model
 128 (see sections 2.5 and 6 in [Harvey and Sucarrat \(2013\)](#)) is given by

$$y = \exp(\lambda_t)\varepsilon_t = \sigma_t\varepsilon_t, \quad \varepsilon_t \sim st(0, \sigma_\varepsilon^2, \nu, \gamma), \quad \nu, \gamma \in (0, \infty), \quad (6)$$

$$\lambda_t = \omega + \lambda_{1,t}^\dagger + \lambda_{2,t}^\dagger, \quad (7)$$

$$\lambda_{1,t}^\dagger = \phi_1\lambda_{1,t-1}^\dagger + \kappa_1 u_{t-1}, \quad |\phi_1| < 1, \quad (8)$$

$$\lambda_{2,t}^\dagger = \phi_2\lambda_{2,t-1}^\dagger + \kappa_2 u_{t-1} + \kappa^* \text{sgn}(-y_{t-1})(u_{t-1} + 1), \quad |\phi_2| < 1, \quad \phi_1 \neq \phi_2. \quad (9)$$

129 The $\lambda_{1,t}$ and $\lambda_{2,t}$ can be interpreted as the time-varying long-term and short-term components of
 130 log-volatility, respectively. The conditional score u_t , also here given by (5), drives both the long-run
 131 and short-run components, but leverage appears only in the short-term component. This is in line
 132 with the view that shocks only matters for short term volatility, see e.g. [Engle and Lee \(1999\)](#). The
 133 model is not identifiable if $\phi_2 = \phi_1$.

134 Returns that follow a two-component Beta-Skew-t-EGARCH model given by (6)-(9) can also be
 135 simulated with the `tegarchSim` function. For example, the following code generates three thousand
 136 returns from a specification with empirically plausible values on the other parameters, but without
 137 leverage and skewness:

```
y1 <- tegarchSim(3000, omega=0.2, phi1=0.98, phi2=0.9, kappa1=0.01, kappa2=0.02, df=5)
```

138 Similarly, just as in the one-component case, the following code generates three thousand values of y_t
 139 with, respectively, leverage, skewness, and both leverage and skewness:

```
y2 <- tegarchSim(3000, omega=0.2, phi1=0.98, phi2=0.9, kappa1=0.01, kappa2=0.02,  
kappastar=0.04, df=5)
```

```
y3 <- tegarchSim(3000, omega=0.2, phi1=0.98, phi2=0.9, kappa1=0.01, kappa2=0.02,  
df=5, skew=0.95)
```

```
y4 <- tegarchSim(3000, omega=0.2, phi1=0.98, phi2=0.9, kappa1=0.01, kappa2=0.02,  
kappastar=0.04, df=5, skew=0.95)
```

140 Also here is the verbose option available for a more detailed output, and also here can the `lambda.initial`
 141 option be used to change the initial values.

142 Estimation and inference

143 One-component and two-component specifications can be estimated with the `tegarch` function. For
 144 example, the following code generates 5000 values of y_t with default parameter values, estimates
 145 a one-component specification with leverage and skewness, and then prints the most important
 146 information:

```
set.seed(123)  
y <- tegarchSim(5000)  
onecompmod <- tegarch(y)  
onecompmod
```

```
Date: Wed Dec 04 19:53:52 2013  
Message (nlminb): relative convergence (4)
```

Coefficients:

	omega	phi1	kappa1	kappastar	df	skew
Estimate:	-0.002491487	0.92076991	0.014298775	-0.003284977	9.371817	0.99867519
Std. Error:	0.018374338	0.04263685	0.005027258	0.003574193	1.087466	0.01979645

```
Log-likelihood: -7635.482215
```

```
BIC: 3.064414
```

147 Estimation without leverage or skewness or both can be achieved by setting the `asym` and `skew` options
 148 to `FALSE`. For alternative summaries of the estimation results the `summary` method can be used, either

149 with its verbose option set to FALSE (default) or TRUE (more information is returned). The latter returns,
 150 amongst other, the initial values used, the upper and lower bounds used and the numerically estimated
 151 Hessian. For additional inference on the parameters the covariance-matrix of the parameter estimates
 152 can be extracted with the vcov method:

```
vcov(onecompmod)
      omega      phi1      kappa1      kappastar      df
omega  3.376163e-04  4.821495e-06 -3.369968e-06  3.781322e-06  1.202316e-02
phi1   4.821495e-06  1.817901e-03 -1.321280e-04  7.384223e-05  1.460351e-05
kappa1 -3.369968e-06 -1.321280e-04  2.527332e-05 -5.374307e-06 -5.498031e-04
kappastar 3.781322e-06  7.384223e-05 -5.374307e-06  1.277486e-05  2.261583e-05
df      1.202316e-02  1.460351e-05 -5.498031e-04  2.261583e-05  1.182581e+00
skew    1.808787e-06 -6.327049e-05  3.869190e-06 -7.473051e-06  2.075498e-04
      skew
omega  1.808787e-06
phi1   -6.327049e-05
kappa1  3.869190e-06
kappastar -7.473051e-06
df      2.075498e-04
skew    3.918993e-04
```

153 In order to estimate a two-component Beta-Skew-t-EGARCH model the components option has to be
 154 set to 2:

```
twocompmod <- tegarch(y, components=2)
```

155 To estimate a two-component model without skewness the skew argument must be set to FALSE.
 156 Leverage, however, cannot be turned off in the two-component model for identifiability reasons.

157 Forecasting volatility

158 Forecasts of volatility – i.e. either the conditional standard deviation or the conditional scale – can
 159 be generated with the predict method applied to a tegarch object. The formula for n -step ahead
 160 forecasts of conditional scale σ_t is

$$E_t(\sigma_{t+n}) = \exp(\omega + \phi_1^n \lambda_t^\dagger) \cdot \prod_{i=1}^n E_t \left[\exp(\phi_1^{n-i} g_{t+i-1}) \right] \quad (10)$$

161 for the one-component model, where g_t is an IID term equal to $\kappa_1 u_t + \kappa^* \text{sgn}(-\varepsilon)(u_t + 1)$. The t
 162 subscript in the conditional expectation operator $E_t(\cdot)$ means the set of conditioning information
 163 contains all values up to and including period t . Accordingly, for $i = 1$ the value of $E_t[\exp(\phi_1^{n-i} g_{t+i-1})]$
 164 is $\exp(\phi_1^{n-1} g_t)$. For $i > 1$, however, the expectations are not available in explicit form, so they are
 165 estimated by computing the sample mean of a large number of simulated IID variates. The default
 166 number of IID variates is 10 000, but this can be changed via the `n.sim` option. Another default option
 167 is that the predict method only returns forecasts of the conditional standard deviation

$$E_t(\sigma_{t+n})\sigma_\varepsilon. \quad (11)$$

168 But if the verbose option is changed to TRUE, then forecasts of the conditional scale are also returned.
 169 Similarly, the initial values used are by default those of the last observation in the estimation sample.
 170 But this can be altered via the `initial.values` option, e.g. for counterfactual or scenario analysis
 171 purposes.

172 The scale formula for the two-component specification is

$$E_t(\sigma_{t+n}) = \exp(\omega + \phi_1^n \lambda_{1,t}^\dagger + \phi_2^n \lambda_{2,t}^\dagger) \cdot \prod_{i=1}^n E_t \left[\exp(\phi_1^{n-i} g_{1,t+i-1} + \phi_2^{n-i} g_{2,t+i-1}) \right], \quad (12)$$

173 where $g_{1,t} = \kappa_1 u_t$ and $g_{2,t} = \kappa_2 u_t + \kappa^* \text{sgn}(-\varepsilon)(u_t + 1)$. Again, forecasts of conditional standard
 174 deviations are given by $E_t(\sigma_{t+n})\sigma_\varepsilon$, and the expectations in the last term on the right are estimated by
 175 simulation for $i > 1$.

176 As an example, the following code generates forecasts up to 7-period ahead for both scale and
 177 standard deviation for the one-component model estimated above:

```
set.seed(123)
predict(onecompmod, n.ahead=7, verbose=TRUE)
```

```

      sigma   stdev
1  1.012363  1.141463
2  1.014470  1.143838
3  1.012704  1.141847
4  1.011081  1.140017
5  1.009589  1.138335
6  1.008217  1.136788
7  1.006955  1.135365

```

178 The returned object is of class zoo, so a convenient time-series plotting method is readily available.
 179 Similarly, the command `predict(twocompmo, n.ahead=7, verbose=TRUE)` generates a corresponding
 180 set of forecasts for the two-component model estimated above.

181 Computational challenges

182 Estimation of the Beta-Skew-t-EGARCH model is by exact Maximum Log-likelihood (ML). The
 183 expressions of the first and second derivatives are not available in explicit form, so the procedure is all
 184 numerical. This leads to four computational challenges. The first is simply that the model is highly
 185 nonlinear, which is readily apparent by simply looking at the expression for the score (equation (5)).
 186 Moreover, the degree of non-linearity is compounded in the two-component specification. But since
 187 no positivity constraints on the ARCH, GARCH and leverage parameters (i.e. $\omega, \phi_1, \phi_2, \kappa_1, \kappa_2, \kappa$) are
 188 needed, the numerical challenges are in fact not as large as those of, say, the two-component GARCH
 189 model of [Engle and Lee \(1999\)](#). There, positivity constraints on all parameters are necessary. As in
 190 all highly nonlinear estimation problems a set of good initial values is essential. By default, these are
 191 0.02, 0.95, 0.05, 0.01, 10, 0.98 for one-component specifications, and 0.02, 0.95, 0.9, 0.001, 0.01, 0.005,
 192 10, 0.98 for two-component specifications. But if the user wishes to do so they can be changed via
 193 the `initial.values` option in the `tegarch` function. The `summary` method with option `verbose=TRUE`
 194 returns (amongst other) the initial values used in estimation.

195 The second computational challenge is that dynamic stability or stationarity – in practice that $|\phi_1| <$
 196 1 (and $|\phi_2| < 1$ in the two-component case) – is required for estimation, whereas empirically ϕ_1 (and ϕ_2)
 197 is often close to, but just below, 1. In order to avoid explosive recursions during estimation it is therefore
 198 desirable to restrict ϕ_1 (and ϕ_2) such that $|\phi_1| < 1$ (and $|\phi_2| < 1$). My experience suggests the `nlm`
 199 function is an excellent choice for this type of problems. The `optim` function with the L-BFGS-U option
 200 provides a similar bounding facility, but in practice it does not work as well as `nlm` (it is less precise
 201 and fails more often in my experience). As for bounds, the skewing parameter γ is only defined on
 202 strictly positive values, and on theoretical grounds the degrees of freedom parameter ν must be greater
 203 than 2. For these reasons the default lower bounds on ϕ_1, ϕ_2, ν and γ are `-1+.Machine$double.eps`,
 204 `-1+.Machine$double.eps`, `2+.Machine$double.eps` and `.Machine$double.eps`, and their default up-
 205 per bounds are `1-.Machine$double.eps`, `1-.Machine$double.eps`, `+Inf` and `+Inf` (i.e. ν and γ are
 206 unbounded from above). The other parameters are unbounded (i.e. their default upper and lower
 207 bounds are `+Inf` and `-Inf`, respectively). If the user wishes to do so the bounds can be changed via the
 208 lower and upper options in the `tegarch` function. Additional control options can also be passed on to
 209 the `nlm` optimiser (see `nlm` documentation) by introducing them as arguments in the `tegarch`
 210 function.

211 A third computational challenge is due to the presence of the sign function `sgn` in the skewed
 212 density (4), which means the log-likelihood is not differentiable in γ at the skewness change point.
 213 The log-likelihood is continuous, so the problem is numerical and not theoretical. In fact, consistency
 214 and asymptotic normality results often holds regardless (see e.g. [Zhu and Galbraith \(2010\)](#)). Most
 215 of the time the user will not encounter problems due to this characteristic, neither in simulation
 216 nor in empirical practice. Occasionally, however, the numerically estimated gradients (analytical
 217 ones are not available in explicit form) may explode when they iterate towards the proximity of the
 218 skewness change point. The `nlm` function together with its bounding facility resolves this problem
 219 to a large extent. But for extra protection against NA and `+/-Inf` values the log-likelihood functions
 220 (`tegarchLog1` and `tegarchLog2`) check for NA and `+/-Inf` values at each iteration: Whenever such
 221 values are produced, then a small value on the log-likelihood is returned. By default this small value
 222 is the log-likelihood associated with the initial values, but this can be changed via the `log1.penalty`
 223 option in the `tegarch` function.

224 The fourth computational challenge is easily resolved. The expressions for the density function of
 225 the t -distribution and the moments of a t -distributed variable contain the gamma function in both the
 226 numerator and the denominator. When the argument of the gamma function is 172 or larger (i.e. 344
 227 degrees of freedom or higher), then a value of `Inf/Inf = NaN` is returned, which can be detrimental

228 to estimation. This issue is not particular to *R* but a consequence of how the Gamma function is
 229 implemented numerically in computer languages in general. Luckily the issue can be easily resolved
 230 by noting that $\beta(x, y) = \Gamma(x) * \Gamma(y) / \Gamma(x+y)$. The *t*-density and the moments of
 231 *t*-variables are thus readily re-written in terms of the beta function. The **fGarch** package has been
 232 using this parametrisation of the *t*-distribution for some time (always?), and I suspect it is for the same
 233 reason.

234 Empirical example: Daily volatility of the Nasdaq 100 stock 235 market index

236 This section illustrates the use of the package in an empirical example (it is available as a demo by
 237 typing `demo(betategarch)`). The financial return in question is the daily log-return (in percent) of
 238 the Nasdaq 100 composite index (adjusted closing values). The study by [Harvey and Sucarrat \(2013\)](#)
 239 suggests stock market indices are particularly prone to be characterised by leverage, conditional
 240 fat-tailedness, skewness and a long-term component, so a stock market index is therefore specially
 241 suited to illustrate the usefulness of the Beta-Skew-t-EGARCH model. Also, the Nasdaq index was not
 242 included in [Harvey and Sucarrat \(2013\)](#) study. The source of the data is <http://finance.yahoo.com/>
 243 and the period in question is 3 January 2001 to 15 October 2013, i.e. a total of 3215 observations.

244 The following code loads the data, defines *y* to equal daily return in terms of a **zoo** object and plots
 245 *y*:

```
data(nasdaq)
y <- zoo(nasdaq[, "nasdaqret"], order.by=as.Date(nasdaq[, "day"], "%Y-%m-%d"))
plot(y, main="The Nasdaq 100 index (daily)", xlab="", ylab="Log-return in %")
```

246 The plot appears as the upper graph in Figure 1 and shows that the return series is clearly characterised
 247 by time-varying volatility.

248 To estimate a one-component Beta-Skew-t-EGARCH, to extract its fitted values and to plot the
 249 fitted conditional standard deviations, the following code can be used:

```
nasdaq1comp <- tegarch(y)
nasdaq1stddev <- fitted(nasdaq1comp)
plot(nasdaq1stddev, main="", ylab="1-comp: St.dev.", xlab="")
```

250 The estimation results are stored in `nasdaq1comp`, so typing `nasdaq1comp` yields

```
Date: Mon Dec 09 17:42:06 2013
Message (nlminb): relative convergence (4)

Coefficients:
      omega      phi1      kappa1  kappastar      df      skew
Estimate:  1.0421017 0.996543407 0.023508613 0.032033017 10.336337 0.85670426
Std. Error: 0.2412326 0.001184624 0.003542337 0.003065121 1.646172 0.01925872

Log-likelihood: -5586.666891
BIC:              3.490447
```

251 The degrees of freedom in the skewed *t* is estimated to be 10.3, a reasonably fat-tailed conditional *t*
 252 density, whereas the skewness is estimated to be 0.86, which corresponds to pronounced negative
 253 skewness in ε_t . Also, the test statistic $(0.8567 - 1) / 0.0193 = -7.4249$ is significant at all conventional
 254 significance levels. For a model without skewness, the command `tegarch(y, skew=FALSE)` can be used.
 255 For a closer look at the standardised residuals $\hat{\varepsilon}_t / \hat{\sigma}_\varepsilon$ in the estimated model above, the residuals
 256 method can be used for extraction. For the unstandardised residuals $\hat{\varepsilon}_t$, the standardised argument
 257 must be set to FALSE. BIC is the average value of the [Schwarz \(1978\)](#) information criterion. By default,
 258 the `fitted` method returns the fitted conditional standard deviation. However, more output is available
 259 by using the verbose option, i.e. by typing `fitted(nasdaq1comp, verbose=TRUE)`. This returns a matrix
 260 with, amongst other, the fitted scale and log-scale, the estimated score and the residuals. The returned
 261 matrix is a **zoo** object that is automatically matched with the dates – if any – of returns y_t . The middle
 262 graph in Figure 1 contains the plot of the fitted conditional standard deviations of the one-component
 263 model.

264 To estimate a two-component Beta-Skew-t-EGARCH model, to extract its fitted values and to plot
 265 its fitted conditional standard deviation, the following code can be used:

```

nasdaq2comp <- tegarch(y, components=2)
nasdaq2stddev <- fitted(nasdaq2comp)
plot(nasdaq2stddev, main="", ylab="2-comp: St.dev.", xlab="")

```

266 The plot of the fitted conditional standard deviations appears as the lower graph in Figure 1. Typing
 267 `nasdaq2comp` returns

```

Date: Mon Dec 09 17:42:09 2013
Message (nlminb): relative convergence (4)

```

Coefficients:

```

          omega          phi1          phi2          kappa1          kappa2          kappastar          df
Estimate:  1.4409972 1.0000000000 0.94175462 0.022074604 0.006442775 0.049203985 9.732886
Std. Error: 0.1991004 0.0004089023 0.01940377 0.004854267 0.006545395 0.006899511 1.564813

```

```

          skew
Estimate:  0.89320223
Std. Error: 0.02094124

```

```

Log-likelihood: -5573.471563
BIC:              3.487262

```

268 Comparison of the BIC values of the two models suggest the latter provides a better fit. This provides
 269 further evidence in favour of two-component models in modelling the volatility of financial stock-
 270 market returns.

271 To generate out-of-sample forecasts up to 5-days ahead, the following code can be used:

```

set.seed(123)
predict(nasdaq1comp, n.ahead=5)

          1          2          3          4          5
0.8121401 0.8179406 0.8218067 0.8256776 0.8295533

```

272 In other words, the predicted conditional standard deviation 5 trading days after the 15th of October
 273 2013 is 0.8295533. By default, the fitted values of λ_t and λ_t^\dagger for the last day of the estimation sample are
 274 used as initial values. However, for alternative scenario and counterfactual analysis can be undertaken
 275 by providing user-specific values on λ_t and λ_t^\dagger via the `initial.values` option.

276 Conceptually the Beta-Skew-t-EGARCH model can appear complicated. So one may ask whether it
 277 performs better than conceptually simpler models like, say, an ordinary GARCH model with leverage
 278 and the two-component model of Engle and Lee (1999). An ordinary GARCH(1,1) model with leverage
 279 of the Glosten et al. (1993) type, sometimes referred to as the GJR-GARCH, coupled with a standardised
 280 skewed t conditional density is given by

$$y = \sigma_t \varepsilon_t, \quad \varepsilon_t \sim st(0, 1, \nu, \gamma), \quad (13)$$

$$\sigma_t^2 = \omega + \phi_1 \sigma_{t-1}^2 + \kappa_1 y_{t-1}^2 + \kappa^* I_{\{y_{t-1} < 0\}} y_{t-1}^2, \quad (14)$$

281 where the parameters have similar interpretations to those of the Beta-Skew-t-EGARCH specification.
 282 The model can be estimated with the `fGarch` package by means of

```

library(fGarch)
nasdaqgarch <- garchFit(data=y, cond.dist="sstd",
  include.mean=FALSE, include.skew=TRUE, leverage=TRUE)

```

283 Next, `summary(nasdaqgarch)` yields (amongst other)

```

          Estimate Std. Error t value Pr(>|t|)
omega    0.016866   0.004315   3.908 9.29e-05 ***
alpha1   0.040237   0.009882   4.072 4.67e-05 ***
gamma1   0.712143   0.183734   3.876 0.000106 ***
beta1    0.933986   0.008357  111.762 < 2e-16 ***
skew     0.898964   0.021099  42.608 < 2e-16 ***
shape    9.910543   1.562340   6.343 2.25e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

Log Likelihood:
-5591.593    normalized: -1.73922

```


Information Criterion Statistics:
 AIC BIC SIC HQIC
 3.482173 3.493511 3.482166 3.486237

284 In the table alpha1, gamma1 and beta1 correspond to κ_1 , κ^* and ϕ_1 , respectively. The skewness and
 285 degrees of freedom estimates are relatively similar to those of the Beta-Skew-t-EGARCH above. How-
 286 ever, the BIC value is lower, which means the Beta-Skew-t-EGARCH provides a better fit according to
 287 this criterion.

288 In the **rugarch** package the first order two-component model of Engle and Lee (1999) coupled with
 289 a standardised skewed t conditional density is given by

$$y = \sigma_t \varepsilon_t, \quad \varepsilon_t \sim st(0, 1, \nu, \gamma), \quad (15)$$

$$\sigma_t^2 = q_t + \phi_1(\sigma_{t-1}^2 - q_{t-1}) + \kappa_1(y_{t-1}^2 - \sigma_{t-1}^2), \quad (16)$$

$$q_t^2 = \omega + \phi_2 q_{t-1}^2 + \kappa_1(y_{t-1}^2 - \sigma_{t-1}^2). \quad (17)$$

290 In other words, **rugarch** does not provide the option to include leverage in the Engle and Lee (1999)
 291 model. Estimation can be undertaken with

```
library(rugarch)
EngleLeeSpec <- ugarchspec(variance.model = list(model = "csGARCH",
  garchOrder = c(1, 1)), mean.model=list(armaOrder=c(0,0),
  include.mean=FALSE), distribution.model="sstd")
nasdaqEngleLee <- ugarchfit(EngleLeeSpec, y, solver="nlminb")
```

292 Next, summary(nasdaqgarch) yields (amongst other)

	Estimate	Std. Error	t value	Pr(> t)
omega	0.008419	0.003911	2.1528	0.031337
alpha1	0.021006	0.014765	1.4228	0.154807
beta1	0.931799	0.044489	20.9446	0.000000
eta11	0.995956	0.002383	417.9196	0.000000
eta21	0.044147	0.013570	3.2533	0.001141
skew	0.912973	0.020984	43.5076	0.000000
shape	11.055129	2.095500	5.2757	0.000000

LogLikelihood : -5617.186

Information Criteria

```
-----
Akaike          3.4987
Bayes           3.5119
Shibata         3.4987
Hannan-Quinn   3.5035
```

293 In the table alpha1, beta1, eta11 and eta21 correspond to κ_1 , ϕ_1 , ϕ_2 and κ_2 , respectively. The skewness
 294 and degrees of freedom estimates are again relatively similar to those of the Beta-Skew-t-EGARCH
 295 above, and again the BIC value is higher. So also in this case does the Beta-Skew-t-EGARCH provide
 296 a better fit according to BIC. In fact, the log-likelihood of the Engle and Lee (1999) model is even
 297 lower than that of the GJR-GARCH, which contains fewer parameters. This suggests leverage, which
 298 is omitted from the **rugarch** implementation of the Engle and Lee (1999) model, is an important
 299 determinant of Nasdaq 100 return volatility.

300 Summary

301 This paper illustrates how the **betategarch** package can be used for the simulation, estimation and
 302 forecasting of one-component and two-component first order Beta-Skew-t-EGARCH models. The
 303 model allows for skewed and heavy-tailed t -distributed conditional errors, and leverage and a time-
 304 varying long-term component in the volatility specification. The two main functions of the package
 305 are tegarchSim and tegarch. The first simulates from a Beta-Skew-t-EGARCH model, either a one-
 306 component or two-component specification, whereas the latter function estimates one. The object (a
 307 list) returned by the second function is of class tegarch, and a collection of S3 methods can be applied
 308 to objects from this class: coef, fitted, logLik, predict, print, residuals, summary and vcov. Finally,
 309 the empirical illustration on daily Nasdaq 100 returns provides further in-sample evidence in favour
 310 of the Beta-Skew-t-EGARCH model.

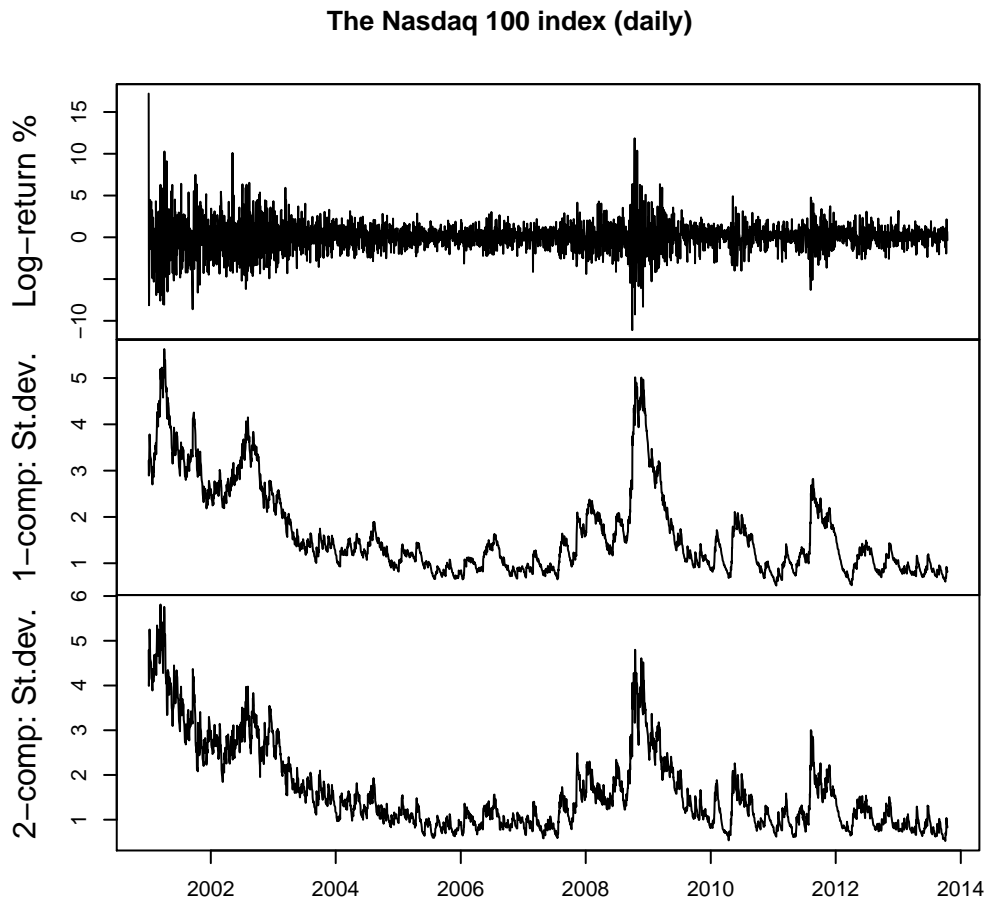


Figure 1: Nasdaq log-returns in %, fitted conditional standard deviation of the one-component model and fitted conditional standard deviation of the two-component model.

Bibliography

- 311
- 312 T. Bollerslev. Generalized autoregressive conditional heteroscedasticity. *Journal of Econometrics*, 31:
313 307–327, 1986. [p1]
- 314 D. Creal, S. J. Koopmans, and A. Lucas. Generalized Autoregressive Score Models with Applications.
315 *Journal of Applied Econometrics*, 28:777–795, 2013. [p2]
- 316 Z. Ding and C. Granger. Modelling volatility persistence of speculative returns: a new approach.
317 *Journal of Empirical Finance*, 1:83–106, 1996. [p4]
- 318 Z. Ding, R. Engle, and C. Granger. A long memory property of stock market returns and a new model.
319 *Journal of Empirical Finance*, 1:83–106, 1993. [p4]
- 320 R. Engle. Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United
321 Kingdom Inflation. *Econometrica*, 50:987–1008, 1982. [p1]
- 322 R. F. Engle and G. G. Lee. A Long-Run and a Short-Run Component Model of Stock Return Volatility.
323 Oxford University Press, Oxford, 1999. [p4, 6, 8, 9]
- 324 C. Fernández and M. Steel. On Bayesian Modelling of Fat Tails and Skewness. *Journal of the American
325 Statistical Association*, 93:359–371, 1998. [p2]
- 326 C. Francq and J.-M. Zakoïan. *GARCH Models*. Marcel Dekker, New York, 2010. [p1]

- 327 A. Ghalanos. *rugarch: Univariate GARCH models*, 2013. URL <http://CRAN.R-project.org/package=rugarch>. R package version 1.2-7. [p1]
328
- 329 L. R. Glosten, R. Jagannathan, and D. E. Runkle. On the Relation between the Expected Value and the
330 Volatility of the Nominal Excess Return on Stocks. *Journal of Finance*, 48:1779–1801, 1993. [p8]
- 331 A. C. Harvey. *Dynamic Models for Volatility and Heavy Tails*. Cambridge University Press, New York,
332 2013. [p2]
- 333 A. C. Harvey and T. Chakravarty. Beta-t-(E)GARCH. Cambridge Working Papers in Economics 0840,
334 Faculty of Economics, University of Cambridge, 2008. [p2]
- 335 A. C. Harvey and G. Sucarrat. EGARCH models with fat tails, skewness and leverage. *Computational
336 Statistics and Data Analysis*, Forthcoming, 2013. URL [http://dx.doi.org/10.1016/j.csda.2013.09.
337 022](http://dx.doi.org/10.1016/j.csda.2013.09.022). [p2, 3, 4, 7]
- 338 D. B. Nelson. Conditional Heteroskedasticity in Asset Returns: A New Approach. *Econometrica*, 59:
339 347–370, 1991. [p1, 2]
- 340 G. Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6:461–464, 1978. [p7]
- 341 D. Straumann and T. Mikosch. Quasi-Maximum-Likelihood Estimation in Conditionally Heteroscedastic
342 Time Series: A Stochastic Recurrence Equations Approach. *The Annals of Statistics*, 34:2449–2495,
343 2006. [p2]
- 344 G. Sucarrat. *AutoSEARCH: General-to-Specific (GETS) Model Selection*, 2012. URL [http://CRAN.R-
345 project.org/package=AutoSEARCH](http://CRAN.R-project.org/package=AutoSEARCH). R package version 1.2. [p1]
- 346 A. Trapletti and K. Hornik. *tseries: Time series analysis and computational finance*, 2013. URL [http://
347 CRAN.R-project.org/package=tseries](http://CRAN.R-project.org/package=tseries). R package version 0.10-32. [p1]
- 348 O. Wintenberger. QML estimation of the continuously invertible EGARCH(1,1) model. Unpublished
349 working paper, 2012. [p2]
- 350 D. Wuertz, Y. C. with contribution from Michal Miklovic, C. Boudt, P. Chausse, and others. *fGarch:
351 Rmetrics - Autoregressive Conditional Heteroskedastic Modelling*, 2013. URL [http://CRAN.R-project.
352 org/package=fGarch](http://CRAN.R-project.org/package=fGarch). R package version 3010.82. [p1]
- 353 A. Zeileis, G. Grothendieck, and J. A. Ryan. *zoo: S3 Infrastructure for Regular and Irregular Time Series
354 (Z's ordered observations)*, 2013. URL <http://CRAN.R-project.org/package=zoo>. R package version
355 1.7-10. [p2]
- 356 D. Zhu and J. W. Galbraith. A Generalized Asymmetric Student t Distribution with Application to
357 Financial Econometrics. *Journal of Econometrics*, 157:297–305, 2010. [p6]

358 Genaro Sucarrat
359 BI Norwegian Business School
360 Nydalsveien 37
361 0484 Oslo
362 Norway
363 genaro.sucarrat@bi.no