Markov-Switching GARCH Models in R: The MSGARCH Package

David Ardia
University of Neuchâtel
Laval University

Keven Bluteau
University of Neuchâtel

Kris Boudt
Vrije Universiteit Brussel
Vrije Universiteit Amsterdam

Denis-Alexandre Trottier
Laval University

VERY PREMIMINARY

Abstract

Markov-switching GARCH models have become popular to model the structural break in the conditional variance dynamics of financial time series. Under this approach, the state follows a hidden Markov chain. In this paper, we describe the R package **MSGARCH** which implements Markov-switching GARCH-type models very efficiently by using C++ object-oriented programming techniques. It allows the user to perform simulations as well as Maximum Likelihood and Bayesian estimation of a very large class of Markov-switching GARCH-type models. Risk management tools such as Value-at-Risk and Expected-Shortfall calculations are available. An empirical illustration of the usefulness of the R package **MSGARCH** is presented.

Keywords: GARCH, MSGARCH, Markov-switching, conditional volatility, risk management, R sofware.

1. Introduction

Modeling the volatility of financial markets is central in risk management. A seminal contribution in this field was the development of the GARCH model by Bollerslev (1986) where the volatility is a function of past asset returns. The GARCH model is today a widespread tool in risk management. However, recent studies show that estimates of GARCH models can be biased by structural breaks in the volatility dynamics (Bauwens *et al.* 2010, 2014). These structural breaks typically occur during periods of financial turmoil. Estimating a GARCH model on data displaying a structural break yields a non-stationary estimated model and implies poor risk predictions. A way to cope with this problem is provided by Markov-switching GARCH models (MSGARCH) whose parameters vary over time according to a latent discrete Markov process. These models can quickly adapt to variations in the unconditional volatility level, which improves risk predictions (Ardia 2008).

Following the seminal work of Hamilton and Susmel (1994), different parametrizations have been proposed to account for discrete changes in the GARCH parameters by Dueker (1997), Gray (1996) and Klaassen (2002). However, these parametrizations for the conditional variance process lead to computational difficulties. Indeed, the evaluation of the likelihood function for a sample of length T in the case of K states requires the integration over all K^T possible paths, rendering the estimation infeasible.

In order to avoid any difficulties related to the past infinite history of the state variable, we adopt the parametrization due to Haas *et al.* (2004b). In their model, the authors hypothesize K *separate* GARCH(1,1) processes for the conditional variance of the MSGARCH process. In addition to its appealing computational aspects, the MSGARCH model of Haas *et al.* (2004b) has conceptual advantages. In effect, one reason for specifying Markov-switching models that allow for different GARCH behavior in each regime is to capture the difference in the variance dynamics in low- and high-volatility periods.

The R package **MSGARCH** aims to provide a comprehensive set of methods for the estimation, the simulation and the forecasting of MSGARCH models. Also, methods for risk management such as Value-at-Risk and Expected-Shortfall calculations are available. The R package **MSGARCH** is available from the CRAN repository at `https://cran.r-project.org/package=MSGARCH`.

In this vignette, we describe the models and the functions/methods available in the package.

2. Model specification

We provide in this section the steps to create GARCH and MSGARCH specifications using the function `create.spec`. The R package **MSGARCH** supports single-regime models as they are the building blocks for regime-switching models.

The simplest specification we can build is a GARCH model with a symmetric Normal conditional distribution:

```
R> spec = create.spec(model = "sGARCH", distribution = "norm", do.skew = FALSE)
```

The natural regime-switching extension of this model is a two-state MSGARCH model with a symmetric Normal conditional distribution in each regime:

```
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
                      distribution = c("norm", "norm"),
                      do.skew = c(FALSE, FALSE),
                      do.mix = FALSE, do.shape.ind = FALSE)
```

Let us quickly discuss the arguments of the function `create.spec`. First, the argument `model` takes one or more single-regime specifications describing each regime conditional variance process while the argument `distribution` contains each regime respective conditional distribution. Valid models are `"sGARCH"`, `"eGARCH"`, `"gjrGARCH"`, `"tGARCH"`, and `"GAS"` (see Section 2.1 for details). All single-regime conditional variance processes are one-lag processes (*e.g.*, GARCH(1,1)). One-lag conditional variance processes have proved to be an effective specification to capture the volatility clustering observed in financial data. Moreover, it reduces the model's complexity. Available conditional distributions are `"norm"`, `"std"`, and `"ged"` (see Section 2.3 for details). Each conditional distribution can be symmetric or skewed as defined by the argument `do.skew = TRUE` (see Section 2.3.4 for details). The argument `do.mix` allows the user to create a Mixture of GARCH processes (see Section 2.2.2) instead of a MSGARCH process (see Section 2.2.1). Finally, the argument `do.shape.ind` allows us for regime-independent shape parameters (see Section 2.2.3).

The user can technically create any MSGARCH specification by selecting single-regime scedastic models and conditional distribution. Here is an example of a three-state MSGARCH process:

```
R> spec = create.spec(model = c("sGARCH", "tGARCH", "eGARCH"),
                      distribution = c("norm", "std", "ged"),
                      do.skew = c(TRUE, FALSE, TRUE),
                      do.mix = FALSE, do.shape.ind = FALSE)
```

Note however that complex models are more difficult to estimate (see Section 3 for details).

The output of the function `create.spec` is a `list` of class `MSGARCH_SPEC` containing various functions and variables. The relevant information is summarized with `print` or `summary`:

```
R> spec = create.spec()
R> print(spec)
```

```
[1] "Specification Type: Markov-Switching"
[1] "Specification Name: sGARCH_normal_sym sGARCH_normal_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in each distribution: 0 0"
[1] "Default parameters:"
alpha0 alpha1 beta alpha0 alpha1 beta   P   P
[1,]    0.1    0.1  0.8    0.1    0.1  0.8 0.5 0.5
```

2.1. Single-regime specifications

As the building block of the regime-switching models are the single-regime specifications, we quickly review the single-regime models available in the R package **MSGARCH**.

GARCH model

The GARCH model by Bollerslev (1986) can be written as:

$$\begin{aligned} y_t &= \eta h_t^{1/2} \\ h_t &\equiv \alpha_0 + \alpha_1 y_{t-1}^2 + \beta h_{t-1} \, , \end{aligned} \tag{1}$$

where $\eta \sim i.i.d.\,\mathcal{D}(0,1,\lambda)$ with $\mathcal{D}$ a distribution with zero mean, unit variance, and shape parameters λ. To ensure positivity, we require $\alpha_0 > 0$, $\alpha_1 \geq 0$, $\beta \geq 0$. Covariance-stationarity is obtained by adding the condition $\alpha_1+\beta < 1$. To create a single-regime GARCH specification we use `model = "sGARCH"` in the function `create.spec`.

EGARCH model

The Exponential GARCH (EGARCH) of Nelson (1991) can be written as:

$$\ln(h_t) \equiv \alpha_0 + \alpha_1\big(|y_{t-1}| - \mathsf{E}[|y_{t-1}|]\big) + \alpha_2 y_{t-1} + \beta \ln(h_{t-1})\,, \tag{2}$$

where the natural logarithm of conditional variance $\ln(h_t)$ is modeled instead of h_t. This model takes into consideration the leverage effect where past negative returns have a larger influence on the conditional volatility than past positive returns of the same magnitude (Black 1976; Christie 1982). The persistence of the models is captured by the coefficient β, and we set $\beta < 1$ to ensure stationarity. The creation of a single-regime EGARCH specification is done by using `model = "eGARCH"` in the function `create.spec`.

GJR model

The GJR model by Glosten *et al.* (1993) is also able to capture the asymmetry in the conditional volatility process. It can be written as:

$$h_t \equiv \alpha_0 + \alpha_1 y_{t-1}^2 + \alpha_2 y_{t-1}^2 \mathbb{I}_{y_{t-1}<0} + \beta h_{t-1}\,, \tag{3}$$

where $\mathbb{I}_{y_t \geq 0} \equiv 0$ if $y_t \geq 0$ and $\mathbb{I}_{y_t<0} \equiv 1$ otherwise. The parameter α_2 controls the degree of asymmetry. To ensure positivity, we set $\alpha_0 > 0$, $\alpha_1 \geq 0$, $\alpha_2 \geq 0$, $\beta \geq 0$ (sufficient condition). To ensure covariance-stationarity we make sure that $\alpha_1 + \alpha_2 \mathsf{E}[\eta^2 \mathbb{I}_{\eta<0}] + \beta < 1$. The single-regime GJR specification is created by using `model = "gjrGARCH"` in the function `create.spec`:

TGARCH model

Zakoian (1994) introduces the TGARCH which has the conditional volatility as dependent variable instead of the conditional variance:

$$h_t^{1/2} \equiv \alpha_0 + \alpha_1 y_{t-1} \mathbb{I}_{y_{t-1}\geq 0} + \alpha_2 y_{t-1} \mathbb{I}_{y_{t-1}<0} + \beta h_{t-1}^{1/2}\,. \tag{4}$$

For positivity we set $\alpha_0 > 0$, $\alpha_1 \geq 0$, $\alpha_2 \geq 0$ and $\beta \geq 0$. To ensure covariance-stationarity, we make sure that $\alpha_1^2 + \beta^2 - 2\beta(\alpha_1 + \alpha_2)\mathsf{E}[\eta \mathbb{I}_{\eta<0}] - (\alpha_1^2 - \alpha_2^2)\mathsf{E}[\eta^2 \mathbb{I}_{\eta<0}] < 1$ (see Francq and Zakoian 2011, Section 10.2). The single-regime TGARCH specification is created by using `model = "tGARCH"` in function `create.spec`:

GAS model

Generalized Autoregressive Score models were proposed in their full generality in Creal *et al.* (2013). It provides a general framework for modeling time variation in parametric models. The GAS model can be written as:

$$h_t \equiv \alpha_0 + \alpha_1 s_{t-1} + \beta h_{t-1}, \quad s_{t-1} \equiv S_{t-1}\nabla_{t-1}, \quad \nabla_{t-1} \equiv \frac{\partial \ln f(y_{t-1}|h_{t-1},\lambda)}{\partial h_{t-1}}\,, \tag{5}$$

where $f(y_{t-1}|h_{t-1},\lambda)$ is the likelihood of y_{t-1} given h_{t-1} and the distribution's shape parameters λ, s_{t-1} is the score function, and S_{t-1} is a scaling function for the score of the observation log-density. The scaling function in this case is defined as:

$$S_{t-1} \equiv \mathsf{E}[\nabla_{t-1}\nabla_{t-1}']^{-1}\,. \tag{6}$$

The single-regime GAS model is created by using `model = "GAS"` in the function `create.spec`.

2.2. Multiple-regime specifications

We present in this section the two multiple-regime specifications available in the R package **MSGARCH**.

Markov-switching GARCH

Suppose Δ_t is a Markov chain with a finite state space $S \equiv \{1, 2, ..., K\}$ with an irreducible and primitive $K \times K$ transition matrix $\mathbf{P}$ defined as:

$$\mathbf{P} \equiv \begin{bmatrix} p_{1,1} & p_{2,1} & \cdots & p_{K,1} \\ p_{1,2} & p_{2,2} & \cdots & p_{K,2} \\ \vdots & \vdots & \ddots & \vdots \\ p_{1,K} & p_{2,K} & \cdots & p_{K,K} \end{bmatrix}, \tag{7}$$

where $0 \leq p_{i,j} \leq 1$ is the probability of switching from state $\Delta_{t-1} = i$ to state $\Delta_t = j$ and $\sum_{j=1}^{K} p_{i,j} = 1$ $(i = 1, \ldots, K)$.
Let the returns of a financial asset at time t be expressed as:

$$y_t = \eta_{\Delta_t} h_{\Delta_t,t}^{1/2}, \tag{8}$$

where $\eta_{\Delta_t} \sim i.i.d.\mathcal{D}_{\Delta_t}(0, 1, \lambda_{\Delta_t})$, with $\mathcal{D}_{\Delta_t}$ a distribution with zero mean, unit variance, and shape parameters λ_{Δ_t}, and $h_{\Delta_t,t}^{1/2}$ the conditional variance, in state Δ_t at time t. For a the single-regime specification in state k, we define θ_k as the parameters of the conditional variance process, λ_k as the shape parameters of the conditional distribution $\mathcal{D}_k$, and the $T \times 1$ vector $\boldsymbol{h}_k \equiv (h_{k,1}, h_{k,2}, \ldots, h_{k,T})'$ as the resulting conditional variance vector from this specification. The MSGARCH specification is constructed following the approach by Haas *et al.* (2004b), which consists of many distinct single-regime specifications evolving in parallel. We define $\Theta \equiv [\theta_1, \theta_2, ..., \theta_K]$, $\boldsymbol{D} \equiv [\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K]$, $\Lambda \equiv [\lambda_1, \lambda_2, ..., \lambda_K]$ and $\mathbf{H} \equiv [\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_K]$.
As an example, let us use the R package **MSGARCH** to create a two-state MSGARCH model for the log-returns of the S&P 500. We create a two-state MSGARCH model, $K = 2$, from two single-regime GARCH processes each following a Normal distribution. We fit the model to the `sp500` dataset which consists of the S&P 500 index closing value log-returns ranging from 1998-01-01 to 2015-12-31.

```
R> data("sp500")
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
              distribution = c("norm", "norm"),
              do.skew = c(FALSE, FALSE),
              do.mix = FALSE, do.shape.ind = FALSE)
R> ctr.mle = list(do.init = TRUE, NP = 10*length(spec$theta0), itermax = 500,
          do.enhance.theta0 = TRUE)
R> set.seed(123)
R> fit.mle = MSGARCH::fit.mle(spec = spec, y = sp500, ctr = ctr.mle)
R> summary(fit.mle)
```

```
[1] "Specification Type: Markov-Switching"
[1] "Specification Name: sGARCH_normal_sym sGARCH_normal_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in each distribution: 0 0"
[1] "Default parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,]      0.1      0.1    0.8      0.1      0.1
beta_2   P   P
[1,]    0.8 0.5 0.5
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,] 0.001601  0.02711 0.8913  0.03795   0.1177
beta_2         P      P
[1,] 0.8778 3.212e-07 0.3704
[1] "Transition matrix:"
      t = 1  t = 2
t + 1 = 1 3.212e-07 0.3704
t + 1 = 2 1.000e+00 0.6296
[1] "Stable probabilities:"
Stable probabilities
State 1               0.2703
State 2               0.7297
[1] "Unconditional volatility:"
State 1 State 2
[1,]    0.14   2.892
[1] "Log-kernel: -6506.63976020117"
[1] "AIC: 13235.894384861"
[1] "BIC: 13287.2404364094"
```

Model is fitted by Maximum likelihood with the function `fit.mle` (see Section 3.0.1). The resulting parameters are collected in the vector `theta` where each parameter are labeled according to the model and their state. The function `transmat` is an helper function that builds the transition matrix from the fitted parameters for better readability.

Mixture of GARCH

Haas *et al.* (2004a) propose a general class of Mixture of GARCH models. They specify a Mixture of Normal distributions where the variance process of each Normal component is a GARCH process. They name this new class the MNGARCH models. A special case of this specification named the Full and Diagonal MNGARCH is encountered when all covariances between each component is constrained to be zero. This special case has a direct relationship with the MSGARCH model. Indeed, we can constrain the transition matrix $\mathbf{P}$ of the MSGARCH model to make the probability $p_{i,j}$ of switching from any state $\Delta_{t-1} = i$ to state $\Delta_t = j$ the same. That is, $P(\Delta_t = j|\Delta_{t-1} = i) \equiv p_j$ $(i = 1, \ldots, K)$. The transition matrix reduces then to a probability vector $\mathbf{P} \equiv [p_1, p_2, ..., p_K]$. This constraint converts the Markov-switching behavior to a Mixture behavior since the probabilities do not depend on the current state. For demonstration, lets repeat the experiment done previously, but with the argument `do.mix = TRUE`.

```
R> data("sp500")
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
              distribution = c("norm", "norm"),
              do.skew = c(FALSE, FALSE),
              do.mix = TRUE, do.shape.ind = FALSE)

R> ctr.mle = list(do.init = TRUE, NP = 10*length(spec$theta0), itermax = 500,
          do.enhance.theta0 = TRUE)
R> set.seed(123)
R> fit.mle = MSGARCH::fit.mle(spec = spec, y = sp500, ctr = ctr.mle)
R> summary(fit.mle)
```

```
[1] "Specification Type: Mixture"
[1] "Specification Name: sGARCH_normal_sym sGARCH_normal_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in each distribution: 0 0"
[1] "Default parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,]      0.1      0.1    0.8      0.1      0.1
beta_2   P
[1,]    0.8 0.5
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,] 0.0003265  0.01765 0.9099  0.02865   0.1067
beta_2     P
[1,] 0.8886 0.199
[1] "Stable probabilities:"
Stable probabilities
State 1               0.199
State 2               0.801
[1] "Unconditional volatility:"
State 1 State 2
[1,] 0.06712   2.472
[1] "Log-kernel: -6518.18297178244"
[1] "AIC: 13188.2083263841"
[1] "BIC: 13233.136121489"
```

We can observe that we have less parameters label as `P` since a Mixture of GARCH will always have less parameters than a Markov-Switching GARCH process. The `transmat` function, for a Mixture of GARCH, will output a probability vector and not a probability matrix.

Regime-independent shape parameters

Sometimes it is useful to have a regime-switching behavior only in the conditional variance and keep the same conditional distribution across regimes. We call this regime-independent shape parameters since all distributions $\mathcal{D}_k$ in $\boldsymbol{D}$ and λ_k in Λ are restricted to be the same (*i.e.*, they only differ via the conditional variance process of each regime). This can be done by setting the parameter `do.shape.ind = TRUE`. We illustrate this with a two-state MSGARCH model with two single-regime GARCH processes following the same Student-t distribution.

```
R> data("sp500")
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
             distribution = c("std", "std"),
             do.skew = c(FALSE, FALSE),
             do.mix = FALSE, do.shape.ind = TRUE)
R> ctr.mle = list(do.init = TRUE, NP = 10*length(spec$theta0), itermax = 500,
          do.enhance.theta0 = TRUE)
R> set.seed(123)
R> fit.mle = MSGARCH::fit.mle(spec = spec, y = sp500, ctr = ctr.mle)
R> summary(fit.mle)
```

```
[1] "Specification Type: Markov-Switching with Regime-Independent distribution"
[1] "Specification Name: sGARCH_student_sym sGARCH_student_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in distribution: 1"
[1] "Default parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,]      0.1      0.1    0.8      0.1      0.1
beta_2 nu_1   P   P
[1,]    0.8   10 0.5 0.5
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1 alpha1_1 beta_1 alpha0_2 alpha1_2
[1,] 0.000298  0.02191   0.88  0.02331   0.1016
beta_2  nu_1         P      P
[1,] 0.8938 15.73 4.918e-06 0.1838
[1] "Transition matrix:"
      t = 1  t = 2
t + 1 = 1 4.918e-06 0.1838
t + 1 = 2 1.000e+00 0.8162
[1] "Stable probabilities:"
Stable probabilities
State 1              0.1553
State 2              0.8447
[1] "Unconditional volatility:"
State 1 State 2
[1,] 0.05512   2.232
[1] "Log-kernel: -6505.99800787707"
[1] "AIC: 13121.7705309498"
[1] "BIC: 13179.5348389418"
```

As we can see, the output only contains one parameter `nu` with no regime indication instead of two parameters `nu_1` and `nu_2`.

2.3. Distributions

We present here the conditional distributions and their functionalities available in the R package **MSGARCH**. There are two functions directly related to the conditional distribution: `pdf`, the probability density function (PDF) and `cdf`, the cumulative density function (CDF). We refer the reader to the documentation manual for further detail.

The Normal distribution

The PDF of the standardized Normal distribution can be written as:

$$f_{\mathcal{N}(0,1)}(z) \equiv \frac{1}{\sqrt{2\pi}}\; e^{-\frac{1}{2}z^2}\,, \tag{9}$$

where $z \equiv \frac{x-\mu}{\sigma}$. The Normal distribution is completely described by its first two moments: the mean and the variance. The distribution is symmetric. To create any specification with a symmetric Normal distribution we use `distribution = "norm"` in the function `create.spec`.

The Student-t distribution

The PDF of the standardized Student-t distribution can be written as:

$$f_{\mathcal{S}(0,1,\nu)}(z) \equiv \sqrt{\frac{\nu}{\nu-2}}\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{z^2}{\nu}\right)^{-\frac{\nu+1}{2}}\,, \tag{10}$$

where Γ is the Gamma function and $\nu > 2$ is the shape parameter. It is completely described by the shape parameter ν. The kurtosis of a Student-t distribution is higher for lower ν. For $\nu = \infty$, the Student-t distribution is equivalent to the Normal distribution. The distribution is symmetric. To create any specification with a symmetric Student-t distribution we use `distribution = "std"` in the function `create.spec`.

The GED distribution

The PDF of the standardized GED distribution can be written as:

$$f_{\mathcal{GED}(0,1,\nu)}(z) \equiv \frac{\nu e^{-\frac{1}{2}|z/\lambda|^{\nu}}}{\lambda 2^{(1+1/\nu)}\Gamma(1/\nu)}\,, \tag{11}$$

where $\lambda \equiv [2^{-2/\nu}\Gamma(1/\nu)/\Gamma(3/\nu)]^{1/2}$. As in the Student-$t$ distribution, the GED distribution is described completely by the shape parameter ν. As ν decreases the density gets flatter. Special cases are the Normal distribution when $\nu = 2$ and the Laplace distribution when $\nu = 1$. The distribution is symmetric. The CDF of the standardized GED distribution can be written as:

$$F_{\mathcal{GED}(0,1,\nu)}(z) \equiv \begin{cases} \frac{1}{2} - \frac{1}{2}F_{\mathcal{GAM}(1,1/\nu)}\left(\frac{1}{2}\left(\frac{|z|}{\lambda}\right)^{\nu}\right) & \text{if} \quad z \leq 0 \\ \frac{1}{2} + \frac{1}{2}F_{\mathcal{GAM}(1,1/\nu)}\left(\frac{1}{2}\left(\frac{z}{\lambda}\right)^{\nu}\right) & \text{if} \quad z \geq 0\,, \end{cases} \tag{12}$$

where $F_{\mathcal{GAM}(1,1/\nu)}(\cdot)$ is the Gamma distribution CDF with parameter $\beta = 1$ and $\alpha = 1/\nu$. To create any specification with a symmetric GED distribution we use `distribution = "ged"` in the function `create.spec`.

Skewed distributions

Fernández and Steel (1998) provide a simple way to include skewness into a unimodal standardized distribution. Trottier and Ardia (2016) derive the moments of the standardized Fernandez-Steel skewed distributions which are needed in the estimation of the GJR, EGARCH, and TGARCH models. We refer the reader to Trottier and Ardia (2016) for details. Parameter ξ ($0 < \xi < \infty$) describes the degree of asymmetry. To create any specification with skewed distribution we use the argument `do.skew = TRUE` in the function `create.spec`.

3. Estimation

In the R package **MSGARCH**, estimation of single-regime and Markov-switching GARCH models can be either done by Maximum Likelihood (ML) or via Markov chain Monte Carlo (MCMC) simulation. In both cases, the key is the function `kernel` which is the sum of the likelihood and the prior. More precisely $\text{kernel}(\Theta) = L(\mathbf{y}|\Theta, \Lambda, \mathbf{P}) + \text{prior}(\Theta) + \text{prior}(\Lambda) + \text{prior}(\mathbf{P})$ where L is the likelihood of $\mathbf{y}$ given the parameter Θ, Λ, and $\mathbf{P}$. We follow Ardia (2008) and use non-informative truncated Normal priors. Moreover, the prior ensures that the Θ makes the conditional variance processes positive and stationary, that Λ respects the parameters bounds of all the conditional distributions, and that the sum of columns of $\mathbf{P}$ are equal to one in the case of a Markov-switching models. If any of these conditions is not respected, the prior returns `-1e10`. For details on the ML or Bayesian estimation via MCMC techniques, we refer the reader to Ardia (2008).

Maximum likelihood estimation

Obtaining the ML estimator of Markov-switching specifications using a standard optimization technique can be a difficult task in practice. The R package **MSGARCH** allows the user to find good starting values for the optimization with Differential Evolution (Price *et al.* 2006) implemented in the R package **DEoptim** (Ardia *et al.* 2015). The resulting best member of the final population is used as a starting value in fitted parameters as initialization in sequential least-squares quadratic programming algorithm (Kraft 1988) implemented in the function `slsqp` of the R package **nloptr** Johnson (2014).

```
R> data("sp500")
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
             distribution = c("std", "std"),
             do.skew = c(FALSE, FALSE),
             do.mix = FALSE, do.shape.ind = FALSE)
R> ctr.mle = list(do.init = TRUE, NP = 10*length(spec$theta0),
         itermax = 500, do.enhance.theta0 = TRUE)
R> set.seed(123)
R> fit.mle = MSGARCH::fit.mle(spec = spec, y = sp500, ctr = ctr.mle)
R> summary(fit.mle)
```

```
[1] "Specification Type: Markov-Switching"
[1] "Specification Name: sGARCH_student_sym sGARCH_student_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in each distribution: 1 1"
[1] "Default parameters:"
alpha0_1 alpha1_1 beta_1 nu_1 alpha0_2
[1,]      0.1      0.1    0.8   10      0.1
alpha1_2 beta_2 nu_2   P   P
[1,]      0.1    0.8   10 0.5 0.5
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1 alpha1_1 beta_1  nu_1 alpha0_2
[1,] 0.004169  0.05999 0.8968 3.351  0.02612
alpha1_2 beta_2 nu_2         P      P
[1,]   0.1037 0.8917   25 2.884e-05 0.3958
[1] "Transition matrix:"
      t = 1  t = 2
t + 1 = 1 2.884e-05 0.3958
t + 1 = 2 1.000e+00 0.6042
[1] "Stable probabilities:"
Stable probabilities
State 1               0.2836
State 2               0.7164
[1] "Unconditional volatility:"
State 1 State 2
[1,]  0.3106   2.383
[1] "Log-kernel: -6503.83678207374"
[1] "AIC: 13080.1083894088"
[1] "BIC: 13144.2909538444"
```

The argument `do.init` indicates if there is a pre-optimization with the R package **DEoptim**. As shown in the example above, we allow the user to directly control some of the argument of `DEoptim` in the list `ctr`. The argument `NP` sets the number of vector of parameters in the population while `itermax` sets the maximum number of iterations (number of populations generated). Please refer to the **DEoptim** documentation for more details. Finally, `do.enhance.theta0` uses the volatilities of rolling windows of `y` and adjust the default parameters so that the unconditional volatility of each regime is set to different quantiles of the volatilities obtained with rolling windows on `y`. In our experience, this provided good starting values for the standard optimization.

Bayesian estimation

To perform Bayesian estimation we use the adaptive Metropolis-Hastings sampler described in Vihola (2012) and available in the R package **adaptMCMC** (Andreas 2012).

```
R> data("sp500")
R> spec = create.spec(model = c("sGARCH", "sGARCH"),
              distribution = c("norm", "norm"),
              do.skew = c(FALSE, FALSE),
              do.mix = FALSE, do.shape.ind = FALSE)
R> set.seed(123)
R> ctr.bay = list(N.burn = 20000, N.mcmc = 10000, N.thin = 10,
          do.enhance.theta0 = TRUE)
R> fit.bay= MSGARCH::fit.bayes(spec = spec, y = sp500, ctr = ctr.bay)
R> tail(fit.bay$theta, 5)

[1] "Specification Type: Markov-Switching"
[1] "Specification Name: sGARCH_student_sym sGARCH_student_sym"
[1] "Number of parameters in each variance model: 3 3"
[1] "Number of parameters in each distribution: 1 1"
[1] "Default parameters:"
alpha0_1 alpha1_1 beta_1 nu_1 alpha0_2
[1,]      0.1      0.1    0.8   10      0.1
alpha1_2 beta_2 nu_2   P   P
[1,]      0.1    0.8   10 0.5 0.5
[1] "Bayesian posterior mean:"
alpha0_1 alpha1_1   beta_1     nu_1 alpha0_2
0.01146  0.08429  0.90644  9.86376  0.55834
alpha1_2   beta_2     nu_2        P        P
0.38338  0.49062 10.11514  0.97443  0.67645
[1] "Posterior variance-covariance matrix"
   alpha0_1   alpha1_1     beta_1
alpha0_1  1.306e-05  2.052e-05 -2.920e-05
alpha1_1  2.052e-05  1.097e-04 -1.100e-04
beta_1   -2.920e-05 -1.100e-04  1.233e-04
nu_1     -5.745e-05 -5.996e-05  6.688e-05
       nu_1   alpha0_2   alpha1_2
alpha0_1 -5.745e-05  7.725e-05  6.534e-05
alpha1_1 -5.996e-05  1.389e-04  1.365e-04
beta_1    6.688e-05 -1.002e-04 -1.218e-04
nu_1      1.716e-02 -3.352e-03 -6.777e-03
    beta_2       nu_2          P
alpha0_1 -9.290e-05 -0.0000229  3.352e-05
alpha1_1 -3.238e-04 -0.0001215  5.504e-05
beta_1    2.584e-04  0.0001008 -6.261e-05
nu_1      1.597e-02 -0.0045914  3.114e-04
          P
alpha0_1  6.665e-05
alpha1_1  2.033e-04
beta_1   -1.899e-04
nu_1     -1.810e-02
[ reached getOption("max.print") -- omitted 6 rows ]
[1] "Posterior mean transition matrix:"
    t = 1  t = 2
```

```
t + 1 = 1 0.97443 0.6764
t + 1 = 2 0.02557 0.3236
[1] "Posterior mean stable probabilities:"
Stable probabilities
State 1              0.96357
State 2              0.03643
[1] "Posterior mean unconditional volatility:"
State 1 State 2
[1,]   1.112   2.105
[1] "Acceptance rate: 0.986"
[1] "AIC: 13006.9565090122"
[1] "BIC: 13071.1390734477"
[1] "DIC: 13003.1027841308"
```

The function `fit.bayes` takes up to five controls arguments in `ctr`. The argument `N.mcmc` is the number of draws to keep, `N.burn` is number of discarded draws, and `N.thin` is the thinning factor. The main purpose of `N.burn` and `N.thin` is to diminishes the auto-correlation in the MCMC chain. The argument `N.burn` also serves as pre-optimization step; this is why it is set to a large value in the example. One alternative is to use a custom starting parameters `theta0` in the `ctr` argument or to set `do.enhance.theta0 = TRUE`. For example, we could set `theta0` as the ML estimator obtained with `fit.mle`. The total length of the chain is: `N.mcmc / N.thin`. The chain is converted to a **coda** object meaning that all function for MCMC analysis available in the R package **coda** (Plummer *et al.* 2006) are available.

4. Other functionalities

Many functionalities are available in the R package **MSGARCH**, which allow the user to filter (functions `ht` and `Pstate`), to simulate (functions `sim` and `simahead`), to compute the predictive density (function `pred`) and the probability integral transform (function `pit`), or to compute risk measures such as the Value-at-Risk (VaR) or Expected-shortfall (ES) (function `risk`). We refer the reader to the documentation manual for details.

In all cases, the object from the ML or Bayesian fit can be used as an input. In the case of the MCMC estimation, the functions return the aggregated value over MCMC draws, hence the *true* predictive distribution, and the VaR or ES which integrate the parameter uncertainty.

Finally, to perfom in-sample model selection, information criterions such as the Aikaike (AIC) criterion (Akaike 1974), the Bayesian information criterion (BIC) (Schwarz *et al.* 1978), and the deviance information criterion (DIC) (Gelman *et al.* 2014) are available. These are all measures of the relative quality of statistical models for a given set of data, where lower values are preferred.

5. Empirical illustration

We illustrate the package's usage on daily log-returns of the Swiss market index for a period ranging from November 12, 1990, to October 20, 2000. The data set is also used in Mullen *et al.* (2011) in the case of a MSGARCH model estimated by ML. In our empirical stuy, we consider a single-regime GJR model with a skewed Student-t distribution and a two-state Markov-switching GJR model with skewed Student-t distributions in each regime. Figure 1 displays the time series of log-returns.

[Insert Figure 1 about here.]

We first estimate both models by ML with the pre-optimization argument `do.init = TRUE` and the argument `do.enhance.theta0 = TRUE`:

```
R> data("SMI")
R> plot(y, xlab = "Date", ylab = "Log-return")
R> SMI = as.matrix(y)
R> date = as.Date(rownames(SMI))
R> date = c(date, date[length(date)] + 1)
R> spec1 = create.spec(model = c("gjrGARCH"),
               distribution = c("std"),
               do.skew = c(TRUE),
               do.mix = FALSE, do.shape.ind = FALSE)
R> ctr.mle1 = list(do.init = TRUE, NP = 10*length(spec1$theta0),
           itermax = 500, do.enhance.theta0 = TRUE)
R> set.seed(123)
R> fit.mle1 = MSGARCH::fit.mle(spec = spec1, y = SMI, ctr = ctr.mle1)
R> summary(fit.mle1)

[1] "Specification Type: Single-Regime"
[1] "Specification Name: gjrGARCH_student_skew"
[1] "Number of parameters in variance model: 4"
[1] "Number of parameters in distribution: 2"
[1] "Default parameters:"
alpha0_1 alpha1_1 alpha2_1 beta_1 nu_1 xi_1
[1,]      0.1     0.05      0.1    0.8   10    1
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1 alpha1_1 alpha2_1 beta_1  nu_1
[1,]  0.03933  0.04298   0.1143 0.8702 8.138
xi_1
[1,] 0.8554
[1] "Unconditional volatility:"
State 1
[1,]   1.285
[1] "Log-kernel: -3376.27190288128"
[1] "AIC: 6743.30840993793"
[1] "BIC: 6778.25268600307"
```

The results indicate a high level of volatility persistence in the conditional variance process together with skewness and fat tails in the conditional distribution. A plot of the conditional variance process can be generated using the following code:

```
R> ht = MSGARCH::ht(fit.mle1)
R> plot(ht, date = date)
```

Results are displayed in Figure 2.

[Insert Figure 2 about here.]

Let us now perform the ML estimation of the Markov-swiching model. This is achieved with the following code:

```
R> spec2 = create.spec(model = c("gjrGARCH", "gjrGARCH"),
              distribution = c("std", "std"),
              do.skew = c(TRUE, TRUE),
              do.mix = FALSE, do.shape.ind = FALSE)
R> ctr.mle2 = list(do.init = TRUE, NP = 50*length(spec2$theta0),
           itermax = 500, do.enhance.theta0 = TRUE)

R> set.seed(123)
R> fit.mle2 = MSGARCH::fit.mle(spec = spec2, y = SMI, ctr = ctr.mle2)
R> summary(fit.mle2)
```

```
[1] "Specification Type: Markov-Switching"
[1] "Specification Name: gjrGARCH_student_skew gjrGARCH_student_skew"
[1] "Number of parameters in each variance model: 4 4"
[1] "Number of parameters in each distribution: 2 2"
[1] "Default parameters:"
alpha0_1 alpha1_1 alpha2_1 beta_1 nu_1 xi_1
[1,]      0.1     0.05      0.1    0.8   10    1
alpha0_2 alpha1_2 alpha2_2 beta_2 nu_2 xi_2
[1,]      0.1     0.05      0.1    0.8   10    1
P   P
[1,] 0.5 0.5
[1] "DEoptim initialization: TRUE"
[1] "Fitted Parameters:"
alpha0_1  alpha1_1 alpha2_1 beta_1  nu_1
[1,]   0.2229 7.648e-06   0.2139 0.5404 5.945
xi_1 alpha0_2 alpha1_2 alpha2_2 beta_2
[1,] 0.8521    0.083 0.006211   0.1393 0.8774
nu_2   xi_2      P       P
[1,] 20.01 0.8582 0.9981 0.00313
[1] "Transition matrix:"
     t = 1   t = 2
t + 1 = 1 0.998052 0.00313
t + 1 = 2 0.001948 0.99687
[1] "Stable probabilities:"
Stable probabilities
```

```
State 1              0.5464
State 2              0.4536
[1] "Unconditional volatility:"
State 1 State 2
[1,]  0.8101   1.423
[1] "Log-kernel: -3364.58904570219"
[1] "AIC: 6687.68084240918"
[1] "BIC: 6769.21748656117"
```

From the results, we first note that the first regime of the MSGARCH model exhibits less persistence in the conditional variance. We also observe that parameter `alpha2_1` is larger in the first regime, implying a larger leverage effect in the less persistent state. The estimated degrees of freedom suggests that the first regime is more fat-tailed than the second regime, but the unconditional volatility of the first regime is much lower than that of the second regime. Both conditional distributions are negatively skewed. The transtion matrix indicates that the regime does not switch very often. This can be observed by computing the filtered probabilities:

```
R> state = Pstate(fit.mle2)
R> plot(state, date = date)
```

Result is displayed in Figure 3.

[Insert Figure 3 about here.]

Bayesian estimation of the MSGARCH model can also be easily performed. We use here the ML estimator as the starting values:

```
R> ctr.bay2 = list(N.burn = 5000, N.mcmc = 10000,
N.thin = 10, theta0 = fit.mle2$theta)
R> set.seed(123)
R> fit.bay2 = MSGARCH::fit.bayes(spec = spec2, y = SMI, ctr = ctr.bay2)
R> summary(fit.bay2)
```

```
[1] "Specification Type: Markov-Switching"
[1] "Specification Name: gjrGARCH_student_skew gjrGARCH_student_skew"
[1] "Number of parameters in each variance model: 4 4"
[1] "Number of parameters in each distribution: 2 2"
[1] "Default parameters:"
alpha0_1 alpha1_1 alpha2_1 beta_1 nu_1 xi_1
[1,]      0.1     0.05      0.1    0.8   10    1
alpha0_2 alpha1_2 alpha2_2 beta_2 nu_2 xi_2
[1,]      0.1     0.05      0.1    0.8   10    1
P   P
[1,] 0.5 0.5
[1] "Bayesian posterior mean:"
```

```
alpha0_1  alpha1_1  alpha2_1    beta_1      nu_1
0.223706  0.006811  0.234363  0.535547  5.951993
xi_1  alpha0_2  alpha1_2  alpha2_2    beta_2
0.848133  0.084816  0.011409  0.155874  0.866230
nu_2      xi_2         P         P
19.997764  0.862992  0.996872  0.004549
[1] "Posterior variance-covariance matrix"
alpha0_1   alpha1_1   alpha2_1
alpha0_1  8.391e-05 -4.415e-06 -1.010e-04
alpha1_1 -4.415e-06  2.955e-05  8.658e-05
beta_1       nu_1       xi_1
alpha0_1  2.623e-05 -2.188e-04  1.144e-04
alpha1_1 -7.793e-06  4.085e-05 -2.881e-05
alpha0_2   alpha1_2   alpha2_2
alpha0_1 -6.799e-05  6.192e-06 -1.172e-04
alpha1_1  3.302e-05  2.371e-06  7.230e-05
beta_2       nu_2       xi_2
alpha0_1  8.185e-05  8.469e-06  5.016e-05
alpha1_1 -4.671e-05 -6.221e-05  1.984e-05
P          P
alpha0_1 -4.778e-07 -1.360e-06
alpha1_1  2.629e-07  8.527e-07
[ reached getOption("max.print") -- omitted 12 rows ]
[1] "Posterior mean transition matrix:"
t = 1    t = 2
t + 1 = 1 0.996872 0.004549
t + 1 = 2 0.003128 0.995451
[1] "Posterior mean stable probabilities:"
Stable probabilities
State 1              0.5497
State 2              0.4503
[1] "Posterior mean unconditional volatility:"
State 1 State 2
[1,]  0.8286   1.488
[1] "Acceptance rate: 0.984"
[1] "AIC: 6689.39263252908"
[1] "BIC: 6770.92927668107"
[1] "DIC: 6672.09065870322"
```

We can test the mixing properties of the chains as follows:

```
R> coda::traceplot(fit.bay2$theta)
R> pairs(x = as.matrix(fit.bay2$theta[,c(1,3,4,7,9,10)]),
pch = 20, cex = 0.8)
```

Results are displayed in Figure 4. We observe that the chain is mixing well. We can also display the pairs plot of MCMC draws with the following code:

```
R> pairs(x = as.matrix(fit.bay2$theta[,c(1,3,4,7,9,10)]),
pch = 20, cex = 0.8)
```

We can observe in Figure 5 that there is an high positive correlation between `alpha2_1` and `alpha2_2`, an high negative correlation between `alpha2_1` and `beta_2`, and an high negative correlation between `alpha2_2` and `beta_2`.

[Insert Figure 4 and Figure 5 about here.]

Figure 6 displays the filtered probabilies in the Bayesian case, reported as a fan plot.

Finally, we can compute and compare the ML and Bayesian VaR at the 95% risk level for the two model specifications:

```
R> risk.mle1 = MSGARCH::risk(fit.mle1, level = c(0.95),
                        ES = FALSE, do.its = TRUE)
R> risk.mle2 = MSGARCH::risk(fit.mle2, level = c(0.95),
                        ES = FALSE, do.its = TRUE)
R> risk.bay1 = MSGARCH::risk(fit.bay1, level = c(0.95),
                        ES = FALSE, do.its = TRUE)
R> risk.bay2 = MSGARCH::risk(fit.bay2, level = c(0.95),
                        ES = FALSE, do.its = TRUE)

R> par(oma = c(4, 1, 1, 1))
R> plot(zoo::zoo(risk, order.by = date),plot.type = "single",
         col = tsRainbow, ylab = "VaR",xlab = "Date")
R> legend("bottomright",legend =  colnames(risk),
   lty = 3, col = tsRainbow, xpd = TRUE, horiz = TRUE,
R> inset = c(0,-0.5), bty = "n", pch = c(4, 2, 15, 19), cex = 1)
```

The Value-at-Risk at 5% for both MLE estimation of each model can be seen in Figure 7. They look similar except that the MSGARCH model often shows bigger spikes than the single-regime model when there is a large shift in volatility.

[Insert Figure 7 about here.]

6. Conclusion

This vignette introduced the R package **MSGARCH** which allows us to estimate, simulate and forecast Markov-switching GARCH models in the R statistical sofware. We detailed how to create various single-regime and regime-switching specifications with various scedastic functions and conditional distributions. We documented how to perfom Maximum Likelihood and Bayesian estimation of these models. In an empirical illustration to real financial data, we showed how to fit and compare the in-sample performance of two complicated single-regime and Markov-switching GARCH specifications.

The R language has become an important vector for knowledge transfer in quantitative finance over the last years. We hope the R package **MSGARCH** will provide risk managers and regulators with new methodologies for improving risk forecasts of their portfolios.

Finally, if you use R or **MSGARCH**, please cite the software in publications.

Computational details

The results in this paper were obtained using R 3.2.3 (R Core Team 2016) with the packages: **MSGARCH** (Bluteau *et al.* 2016), **adaptMCMC** (Andreas 2012), **DEoptim** (Ardia *et al.* 2015), **nloptr** (Johnson 2014), **Rcpp** (Eddelbuettel *et al.* 2016a; Eddelbuettel and François 2011), **RcppArmadillo** (Eddelbuettel *et al.* 2016b; Eddelbuettel and Sanderson 2014), **Rsolnp** (Ghalanos and Theussl 2016), and **xts** (Ryan and Ulrich 2015). R itself and all packages used are available from CRAN at `http://CRAN.R-project.org/`. The package **MSGARCH** is under development in GitHub at `https://github.com/keblu/MSGARCH`. Computations were performed on a Genuine Intel® quad core CPU i7–3630QM 2.40Ghz processor. Code outputs were obtained using `options(digits = 4, max.print = 40, prompt = "R> ", width = 50)`.

Acknowledgments

The authors acknowledge Google for financial support via the Google Summer of Code 2016 project "MSGARCH"; see `https://summerofcode.withgoogle.com/projects/?sp-search=KBlutea#6497774455488512`, the International Institute of Forecasters (IIF, `https://forecasters.org`) and Industrielle-Alliance.

References

Akaike H (1974). "A New Look at the Statistical Model Identification." *Automatic Control, IEEE Transactions on*, **19**(6), 716–723. `doi:10.1007/978-1-4612-1694-0_16`.

Andreas S (2012). ***adaptMCMC**: Implementation of a Generic Adaptive Monte Carlo Markov Chain Sampler.* URL `https://cran.r-project.org/package=adaptMCMC`.

Ardia D (2008). *Financial Risk Management with Bayesian Estimation of GARCH Models: Theory and Applications*, volume 612 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin. `doi:10.1007/978-3-540-78657-3`. ISBN 978-3-540-78656-6.

Ardia D, Mullen KM, Peterson BG, Ulrich J (2015). ***DEoptim**: Differential Evolution in R.* URL `https://cran.r-project.org/package=DEoptim`.

Bauwens L, Backer B, Dufays A (2014). "A Bayesian Method of Change-Point Estimation With Recurrent Regimes: Application to GARCH Models." *Journal of Empirical Finance*, **29**, 207–229. `doi:10.1016/j.jempfin.2014.06.008`.

Bauwens L, Preminger A, Rombouts J (2010). "Theory and Inference for a Markov Switching GARCH Model." *Econometrics Journal*, **13**(2), 218–244. doi:10.1111/j.1368-423X.2009.00307.x.

Black F (1976). "Studies of Stock Price Volatility Changes." In *Proceedings of the 1976 Meetings of the Business and Economics Statistics Section*, pp. 177–181. American Statistical Association.

Bluteau K, Ardia D, Trottier DA, Boudt K, Peterson B (2016). ***MSGARCH**: Markov-Switching GARCH Models in R*. R package version 0.16, URL https://cran.r-project.org/package=MSGARCH.

Bollerslev T (1986). "Generalized Autoregressive Conditional Heteroskedasticity." *Journal of Econometrics*, **31**(3), 307–327. doi:10.1016/0304-4076(86)90063-1.

Christie AA (1982). "The Stochastic Behavior of Common Stock Variances: Value, Leverage and Interest Rate Effects." *Journal of Financial Economics*, **10**(4), 407–432. doi:10.1016/0304-405x(82)90018-6.

Creal D, Koopman SJ, Lucas A (2013). "Generalized Autoregressive Score Models with Applications." *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Dueker MJ (1997). "Markov Switching in GARCH Processes and Mean-Reverting Stock-Market Volatility." *Journal of Business & Economic Statistics*, **15**(1), 26–34. doi:10.2307/1392070.

Eddelbuettel D, François R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. doi:10.18637/jss.v040.i08.

Eddelbuettel D, François R, Allaire J, Ushey K, Kou Q, Bates D, Chambers J (2016a). ***Rcpp**: Seamless R and C++ Integration.* R package version 0.12.5, URL https://cran.r-project.org/package=Rcpp.

Eddelbuettel D, François R, Bates D (2016b). ***RcppArmadillo**: **Rcpp** Integration for the **Armadillo** Templated Linear Algebra Library.* R package version 0.7.100.3.1, URL https://cran.r-project.org/package=RcppArmadillo.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with High–Performance C++ Linear Algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

Fernández C, Steel MF (1998). "On Bayesian Modeling of Fat Tails and Skewness." *Journal of the American Statistical Association*, **93**(441), 359–371. doi:10.1080/01621459.1998.10474117.

Francq C, Zakoian JM (2011). *GARCH models: Structure, statistical inference and financial applications.* John Wiley & Sons. doi:10.1002/9780470670057.

Gelman A, Carlin JB, Stern HS, Rubin DB (2014). *Bayesian Data Analysis*, volume 2. Chapman & Hall/CRC Boca Raton, FL, USA. doi:10.2307/2965436.

Ghalanos A, Theussl S (2016). ***Rsolnp**: General Non–Linear Optimization using Augmented Lagrange Multiplier Method.* R package version 1.16, URL `https://cran.r-project.org/package=Rsolnp`.

Glosten LR, Jagannathan R, Runkle DE (1993). "On the Relation Between the Expected Value and the Volatility of the Nominal Excess Return on Stocks." *The Journal of Finance*, **48**(5), 1779–1801. `doi:10.1111/j.1540-6261.1993.tb05128.x`.

Gray SF (1996). "Modeling the Conditional Distribution of Interest Rates as a Regime-Switching Process." *Journal of Financial Economics*, **42**(1), 27–62. `doi:10.1016/0304-405x(96)00875-6`.

Haas M, Mittnik S, Paolella MS (2004a). "A New Approach to Markov-Switching GARCH Models." *Journal of Financial Econometrics*, **2**(4), 493–530. `doi:10.1093/jjfinec/nbh020`.

Haas M, Mittnik S, Paolella MS (2004b). "Mixed Normal Conditional Heteroskedasticity." *Journal of Financial Econometrics*, **2**(2), 211–250. `doi:10.1093/jjfinec/nbh009`.

Hamilton JD, Susmel R (1994). "Autoregressive Conditional Heteroskedasticity and Changes in Regime." *Journal of Econometrics*, **64**(1), 307–333. `doi:10.1016/0304-4076(94)90067-1`.

Johnson SG (2014). *The **NLopt** Nonlinear-Optimization Package.* URL `https://cran.r-project.org/web/packages/nloptr/`.

Klaassen F (2002). "Improving GARCH Volatility Forecasts with Regime-Switching GARCH." In *Advances in Markov-Switching Models*, pp. 223–254. Springer-Verlag. `doi:10.1007/978-3-642-51182-0_10`.

Kraft D (1988). *A Software Package for Sequential Quadratic Programming.* Wiss. Berichtswesen d. DFVLR.

Mullen KM, Ardia D, Gil DL, Windover D, Cline J, *et al.* (2011). "**DEoptim**: An R Package for Global Optimization by Differential Evolution." *Journal of Statistical Software*, **40**(6), 1–26. `doi:10.18637/jss.v040.i06`.

Nelson DB (1991). "Conditional Heteroskedasticity in Asset Returns: A New Approach." *Econometrica*, pp. 347–370. `doi:10.2307/2938260`.

Plummer M, Best N, Cowles K, Vines K (2006). "CODA: Convergence Diagnosis and Output Analysis for MCMC." *R News*, **6**(1), 7–11. URL `https://cran.r-project.org/package=coda`.

Price K, Storn RM, Lampinen JA (2006). *Differential Evolution: A Practical Approach to Global Optimization.* Springer Science & Business Media.

R Core Team (2016). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. R version 3.2.3, URL `https://www.R-project.org/`.

Ryan JA, Ulrich JM (2015). ***xts**: Extensible Time Series.* R package version 0.9-7, URL https://CRAN.R-project.org/package=xts.

Schwarz G, *et al.* (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2), 461–464. doi:10.1214/aos/1176344136.

Trottier DA, Ardia D (2016). "Moments of Standardized Fernandez-Steel Skewed Distributions: Applications to the Estimation of GARCH-Type Models." *Finance Research Letters*, **18**, 311–316. doi:10.1016/j.frl.2016.05.006.

Vihola M (2012). "Robust Adaptive Metropolis Algorithm with Coerced Acceptance Rate." *Statistics and Computing*, **22**(5), 997–1008. doi:10.1007/s11222-011-9269-5.

Zakoian JM (1994). "Threshold Heteroskedastic Models." *Journal of Economic Dynamics & Control*, **18**(5), 931–955. doi:10.1016/0165-1889(94)90039-6.

Figure 1: Log-returns of the Swiss Market Index. Data range from November 12, 1990, to October 20, 2000.

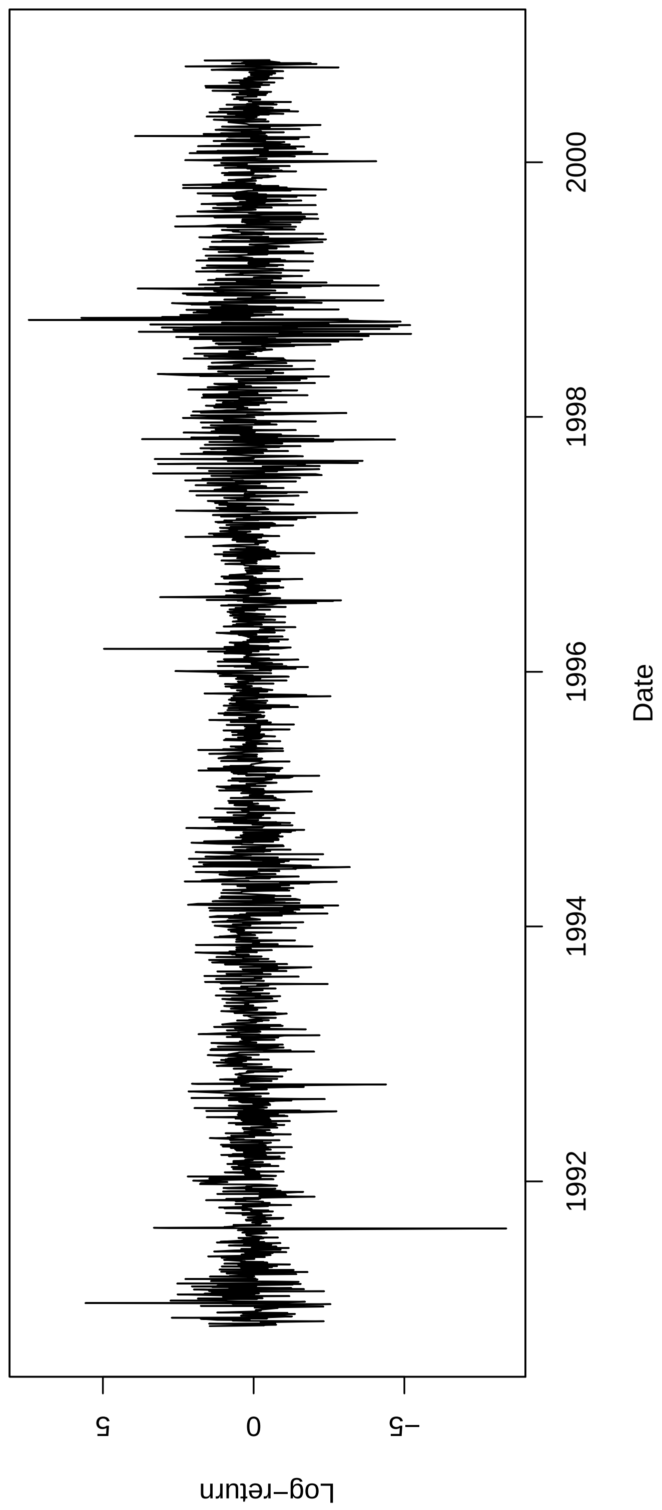


Figure 2: Conditional volatility of the single-regime GJR model with skewed Student-t innovations estimated by ML.

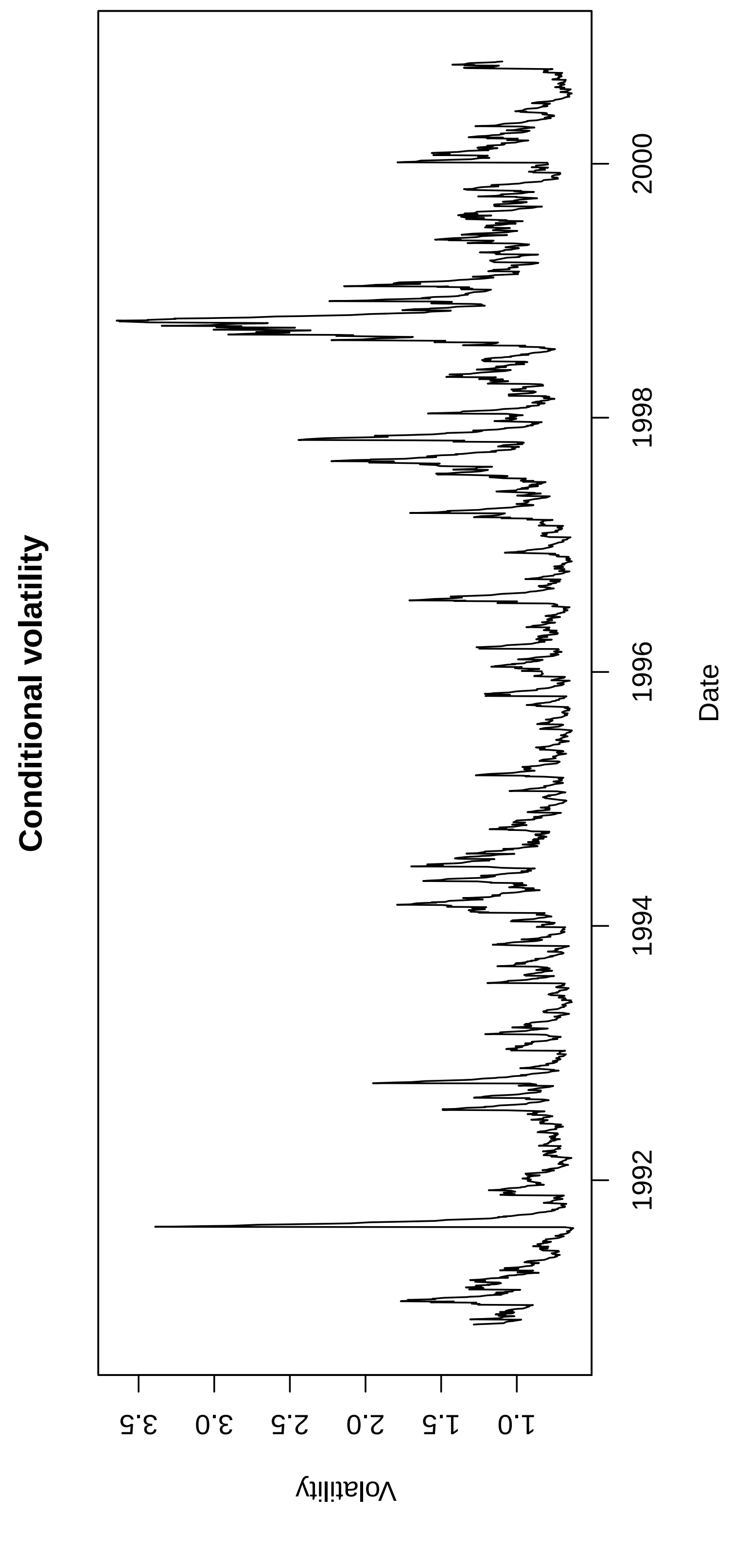


Figure 3: Filtered probabilities of the first regime obtain by ML for the two-state Markov-switching GJR model with skewed Student-*t* innovations.

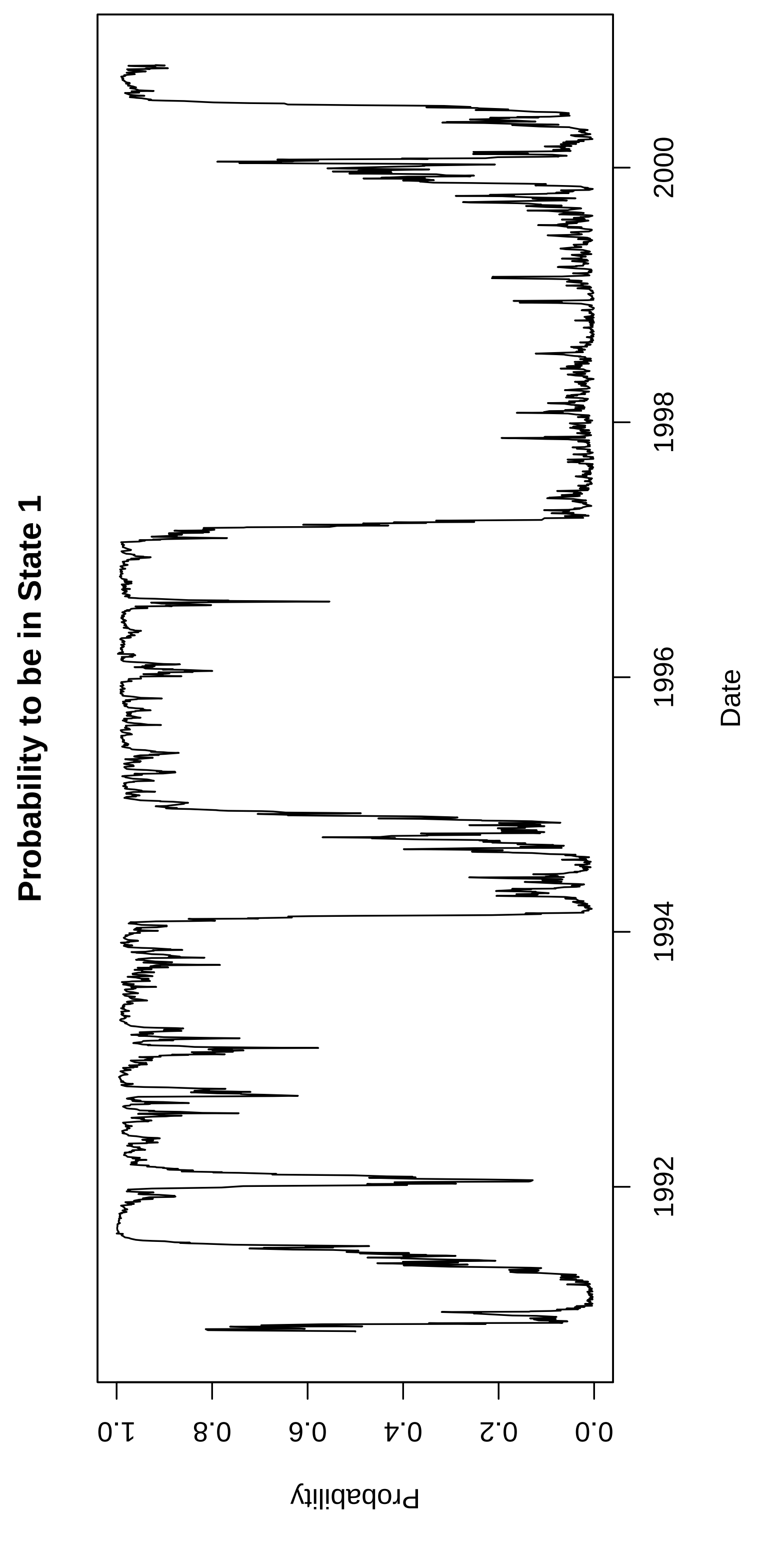


Figure 4: Trace of MCMC samples for the the two-state Markov-switching GJR model with skewed Student-*t* innovations.

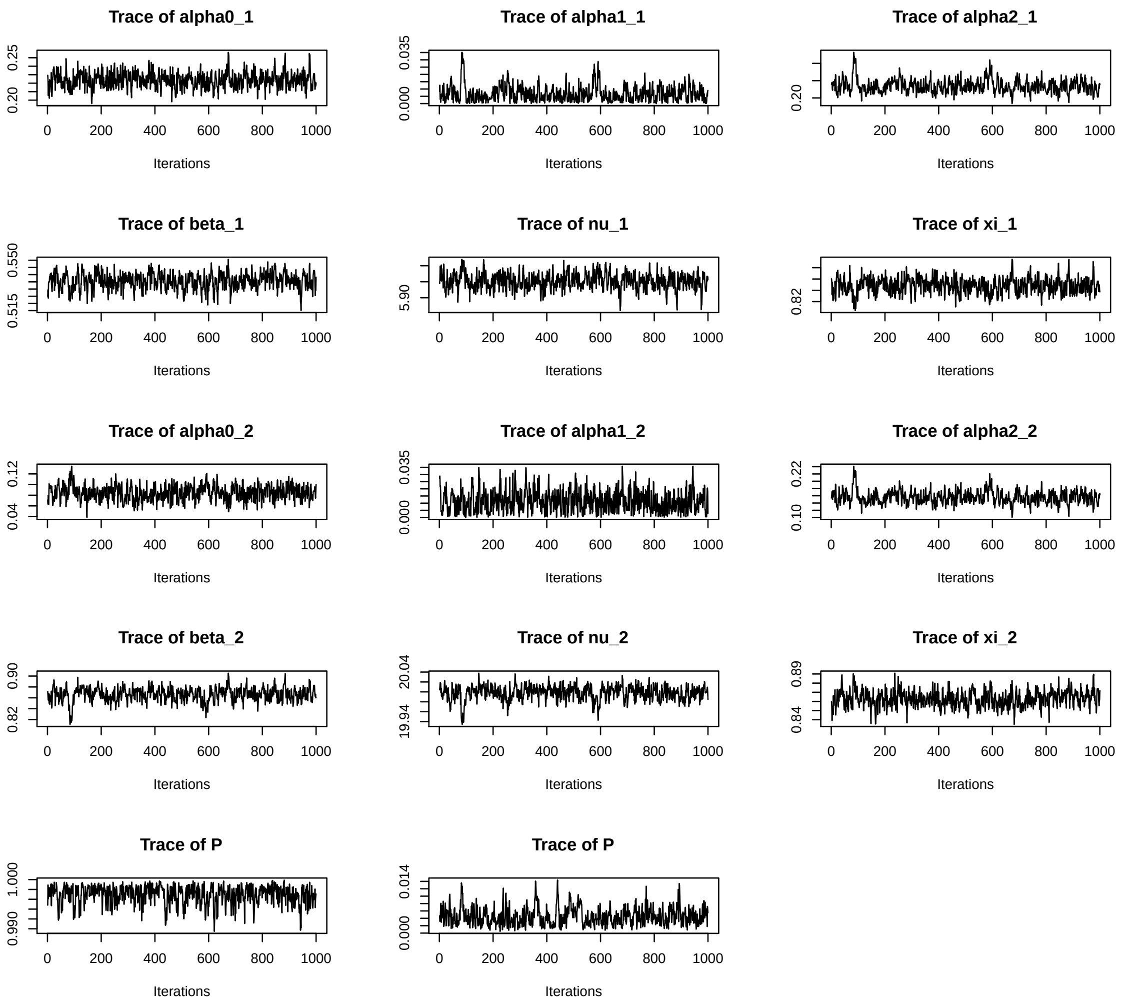


Figure 5: Pairs plot of the MCMC draws for the two-state Markov-switching GJR model with skewed Student-*t* innovations.

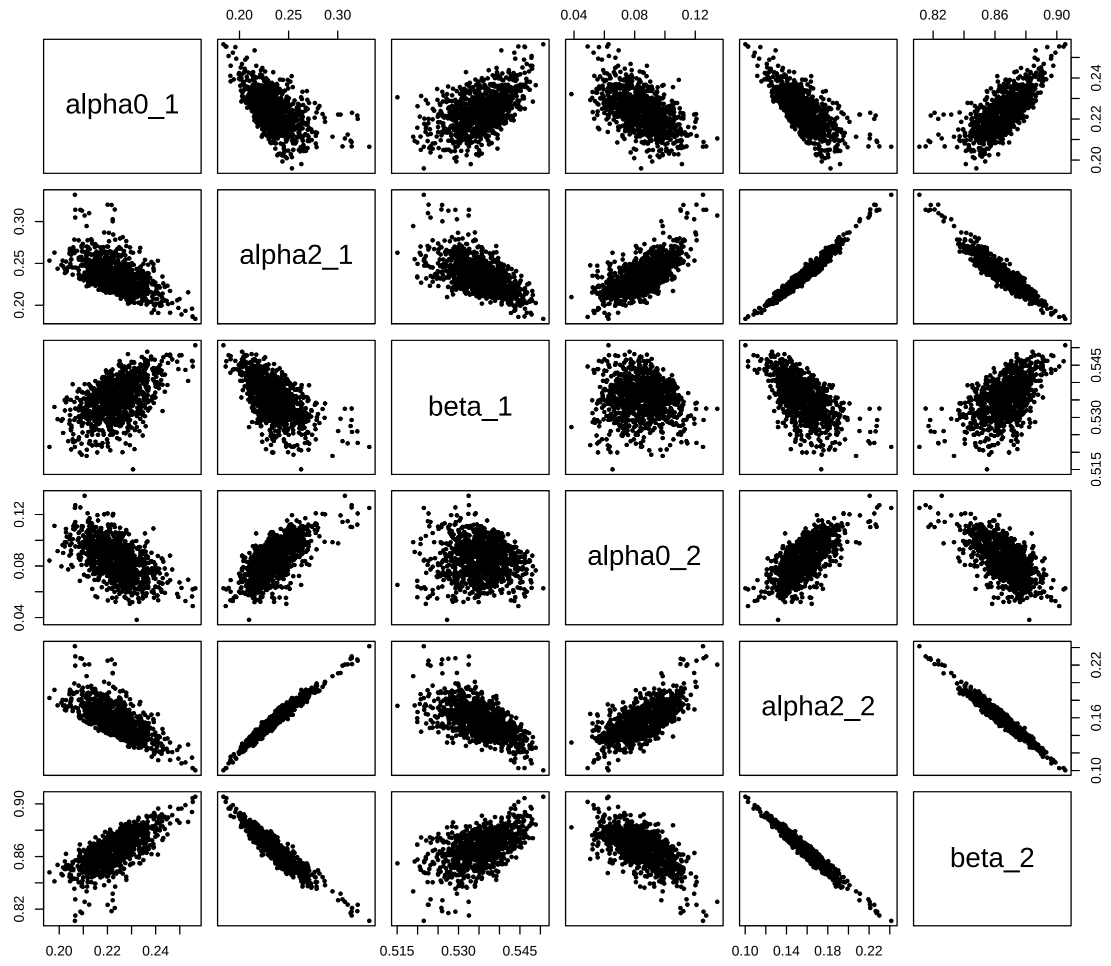


Figure 6: Filtered probabilities of the first regime obtain by MCMC for the two-state Markov-switching GJR model with skewed Student-t innovations. Blue line indicates the median.

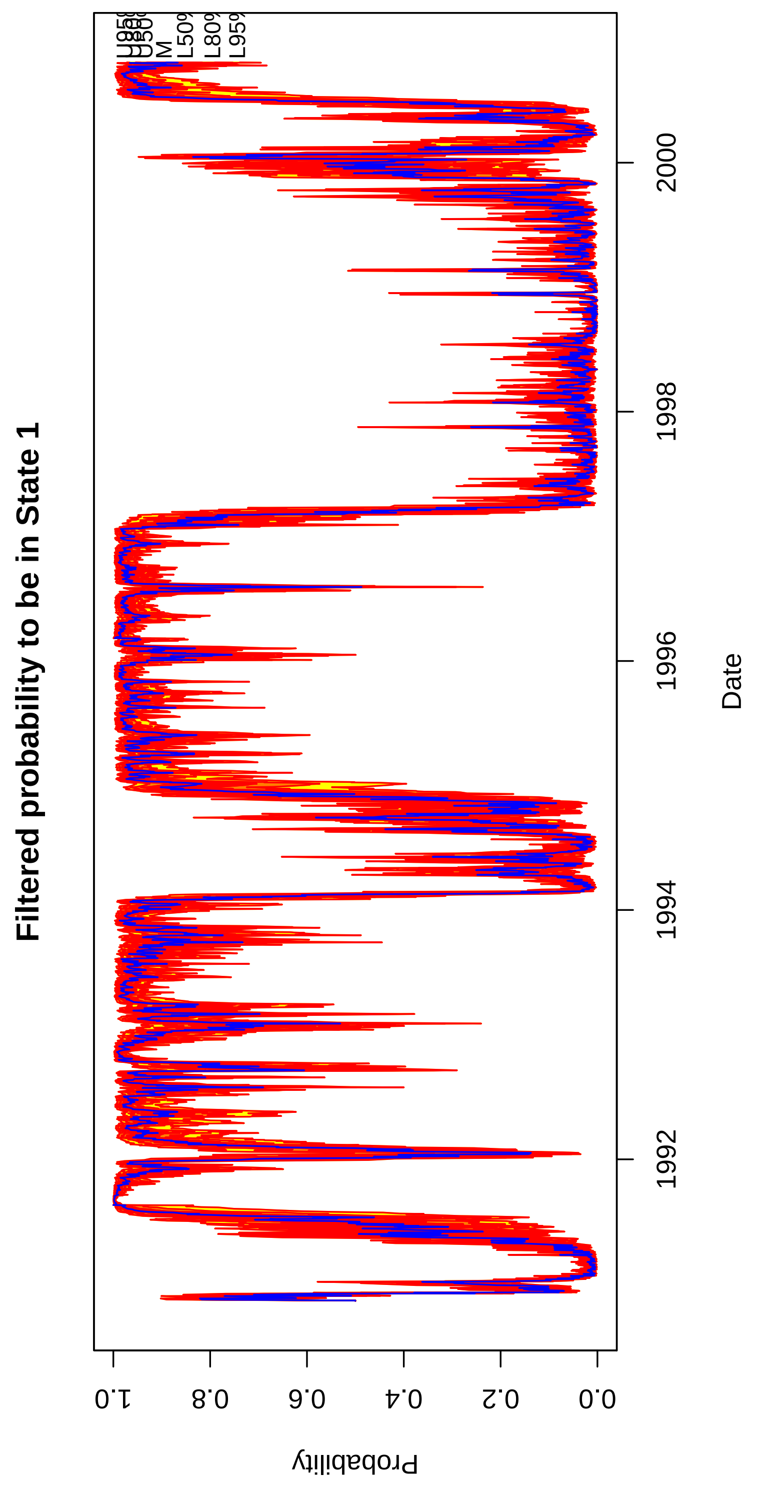
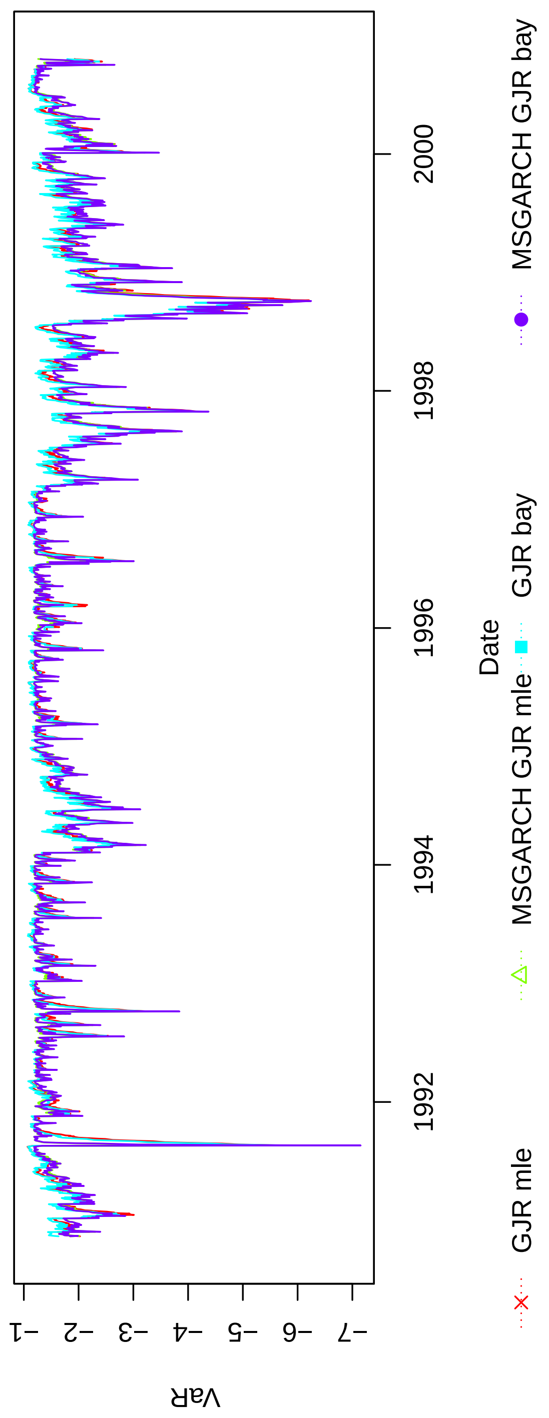


Figure 7: In-sample ML and Bayesian Value-at-Risk at the 95% risk leval for the single-regime and regime-switching models.



Affiliation:

David Ardia
Institute of Financial Analysis
University of Neuchâtel, Switzerland
&
Department of Finance, Insurance and Real Estate
Laval University, Canada
E-mail: `david.ardia@unine.ch`

Keven Bluteau (corresponding author)
Institute of Financial Analysis
Neuchatel University,Neuchatel, Switzerland
E-mail: `keven.bluteau@unine.ch`

Kris Boudt
Vrije Universiteit Brussel, Belgium
&
Vrije Universiteit Amsterdam, The Netherlands
E-mail: `kris.boudt@vub.ac.be`

Denis-Alexandre Trottier
Laval University, Canada
E-mail: `denis-alexandre.trottier.1@ulaval.ca`