Ejemplos

Lizbeth Naranjo Albarrán y Luz Judith Rodriguez Esparza

Paper: *Modelos ocultos de Markov: una aplicación en series de tiempo*

Authors: Lizbeth Naranjo Albarrán & Luz Judith Rodríguez Esparza

Journal: Mixba'al

Year: 2023

https://github.com/lizbethna/HMMaplica

Este archivo muestra las instrucciones para correr los códigos de R y Stan.

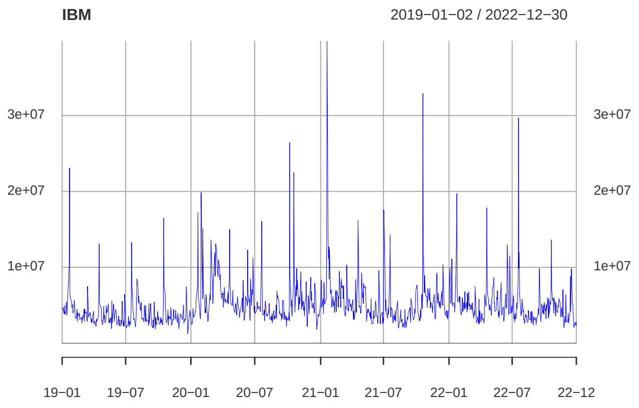
Markov switching GARCH

```
library(ggplot2)
library(rstan) # RStan
library(quantmod) # Quantitative Financial Modelling Framework
```

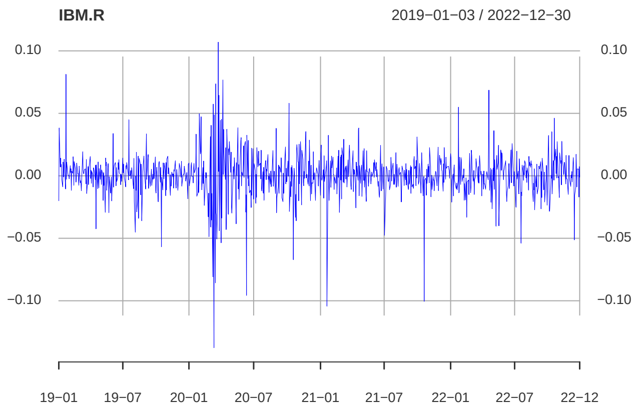
```
plot_statepath <- function(zstar) {
  K <- length(unique(as.vector(zstar)))
  x <- index(zstar)
  t <- 1:dim(zstar)[1]
  opar <- par(no.readonly = TRUE)
  zcol <- (1:K)[zstar]

  layout(matrix(c(1, 2), nrow = 2, ncol = 1), heights = c(0.95, 0.05))
  plot(x = x, y = zstar,
    xlab = bquote(t), ylab = bquote(hat(z)[t]),
    main = bquote("Secuencia mas probable de estados ocultos"),
    ylim = c(1, K), type = 'l', col = 'gray')

  points(x=x, y=zstar,
         pch = 21, bg = zcol, col = zcol, cex = 0.7)

  par(mai = c(0, 0, 0, 0))
  plot.new()
  legend(x = "center",
        legend = c('Trayectoria mas probable', paste('Estado', 1:K)),
        pch = c(NA, rep(21, K)),
        lwd = c(2, rep(NA, K)),
        col = c('lightgray', 1:K),
        pt.bg = c('lightgray', 1:K),
        bty = 'n', cex = 0.7,
         horiz = TRUE)
  par(opar)
}
```

Datos

```
IBM <- getSymbols("IBM",src='yahoo',
           from = "2019-01-01", to = "2022-12-31", auto.assign = FALSE)   # Obtener los datos
IBM.R <- na.omit(ROC(Ad(IBM)));   # Obtener los retornos

plot(IBM, format.labels="%y-%m", col="blue", lwd=0.5)
```



```
plot(IBM.R, format.labels="%y-%m", col="blue", lwd=0.5)
```



Código Stan

```
# Markov-switching GARCH
msgarch_fit <- function(y) {
  rstan_options(auto_write = TRUE)
  options(mc.cores = parallel::detectCores())

  stan.model = 'hmm_garch.stan'
  y <- as.vector(coredata(y));
  stan.data = list(
    T = length(y),
    y = y
  )

  stan(file = stan.model,
       data = stan.data, verbose = T,
       iter = 1000, warmup = 500,
       thin = 1, chains = 1,
       cores = 1, seed = 900)
}
# Fit GARCH
fit <- msgarch_fit(IBM.R)
```

```
TRANSLATING MODEL 'hmm_garch' FROM Stan CODE TO C++ CODE NOW.
successful in parsing the Stan model 'hmm_garch'.

CHECKING DATA AND PREPROCESSING FOR MODEL 'hmm_garch' NOW.

COMPILING MODEL 'hmm_garch' NOW.

STARTING SAMPLER FOR MODEL 'hmm_garch' NOW.

SAMPLING FOR MODEL 'hmm_garch' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.001646 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 16.46 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 13.3548 seconds (Warm-up)
Chain 1:                12.1476 seconds (Sampling)
Chain 1:                25.5024 seconds (Total)
Chain 1:
```

Resultados

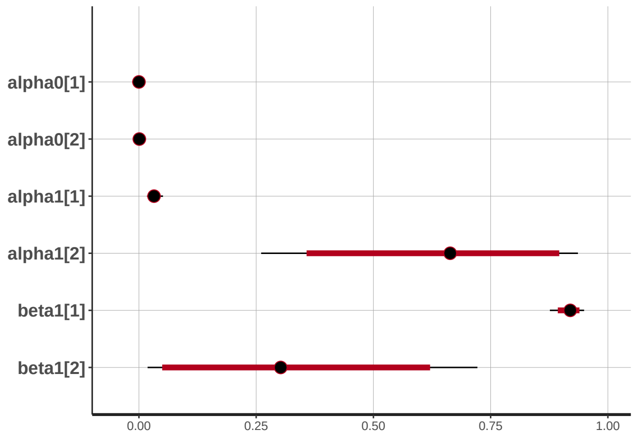
```
round(summary(fit, pars=c("alpha0","alpha1","beta1","A"))$summary,3)
```

```
           mean se_mean    sd  2.5%   25%   50%   75% 97.5%   n_eff  Rhat
alpha0[1] 0.000   0.000 0.000 0.000 0.000 0.000 0.000 0.000 329.528 1.001
alpha0[2] 0.001   0.000 0.000 0.000 0.001 0.001 0.001 0.002 293.842 0.998
alpha1[1] 0.033   0.000 0.009 0.017 0.027 0.032 0.038 0.051 309.071 0.999
alpha1[2] 0.646   0.012 0.200 0.261 0.482 0.663 0.817 0.936 296.904 0.998
beta1[1]  0.918   0.001 0.019 0.876 0.908 0.920 0.930 0.950 290.352 1.002
beta1[2]  0.321   0.012 0.210 0.018 0.139 0.303 0.496 0.722 289.117 0.998
A[1,1]    0.937   0.001 0.016 0.902 0.927 0.938 0.950 0.963 447.746 1.000
A[1,2]    0.063   0.001 0.016 0.037 0.050 0.062 0.073 0.098 447.746 1.000
A[2,1]    0.542   0.005 0.099 0.383 0.463 0.543 0.610 0.752 375.599 1.004
A[2,2]    0.458   0.005 0.099 0.248 0.390 0.457 0.537 0.617 375.599 1.004
```

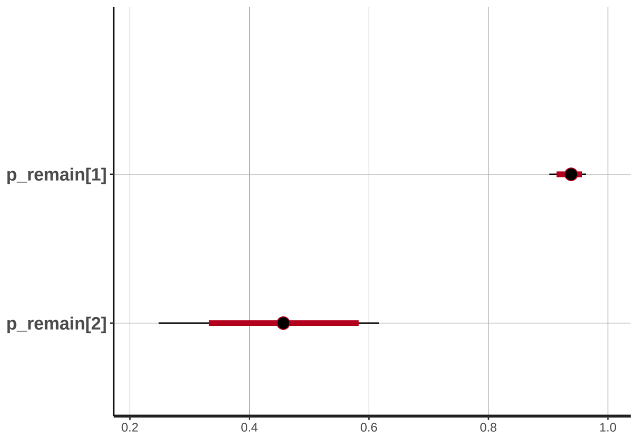
```
round(summary(fit, pars=c("alpha0","alpha1","beta1","A"))$c_summary,3)
```

```
, , chains = chain:1

           stats
parameter    mean    sd  2.5%   25%   50%   75% 97.5%
  alpha0[1] 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  alpha0[2] 0.001 0.000 0.000 0.001 0.001 0.001 0.002
  alpha1[1] 0.033 0.009 0.017 0.027 0.032 0.038 0.051
  alpha1[2] 0.646 0.200 0.261 0.482 0.663 0.817 0.936
  beta1[1]  0.918 0.019 0.876 0.908 0.920 0.930 0.950
  beta1[2]  0.321 0.210 0.018 0.139 0.303 0.496 0.722
  A[1,1]    0.937 0.016 0.902 0.927 0.938 0.950 0.963
  A[1,2]    0.063 0.016 0.037 0.050 0.062 0.073 0.098
  A[2,1]    0.542 0.099 0.383 0.463 0.543 0.610 0.752
  A[2,2]    0.458 0.099 0.248 0.390 0.457 0.537 0.617
```

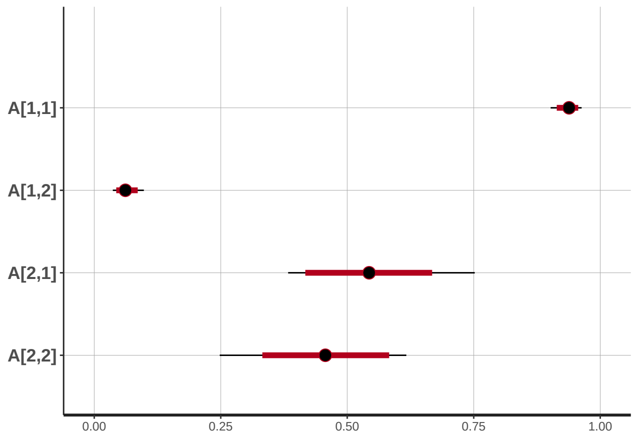
```
plot(fit,pars=c("alpha0","alpha1","beta1"))
```



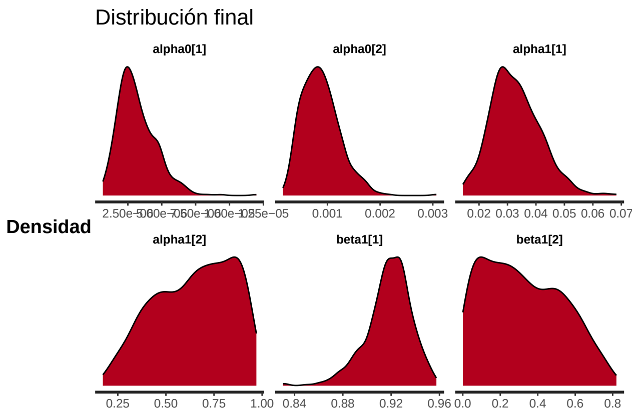
```r
plot(fit,pars="p_remain")
```



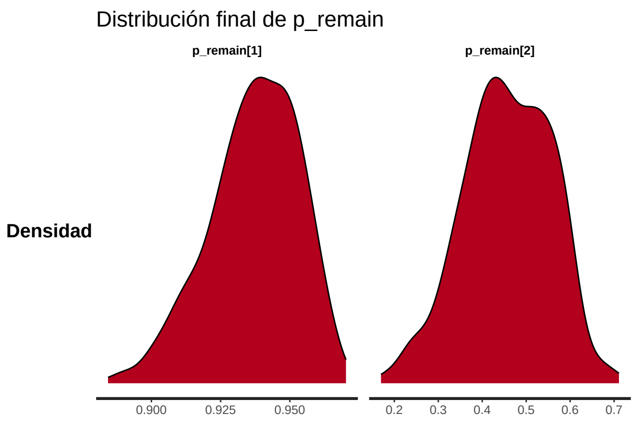
```r
plot(fit,pars="A")
```



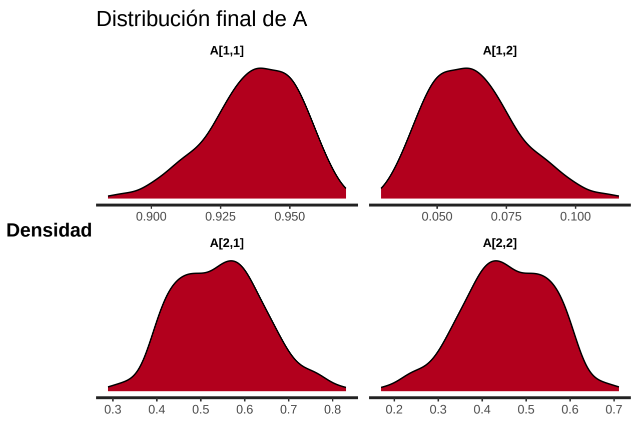
```r
stan_dens(fit,pars=c("alpha0","alpha1","beta1"), point_est = "mean", show_density = TRUE) +
  ggtitle(expression("Distribución final",alpha[0],alpha[1],beta[1])) + ylab("Densidad") +
  theme(axis.title.x=element_text(size=14), axis.title.y=element_text(size=14),
        plot.title = element_text(size=16))
```



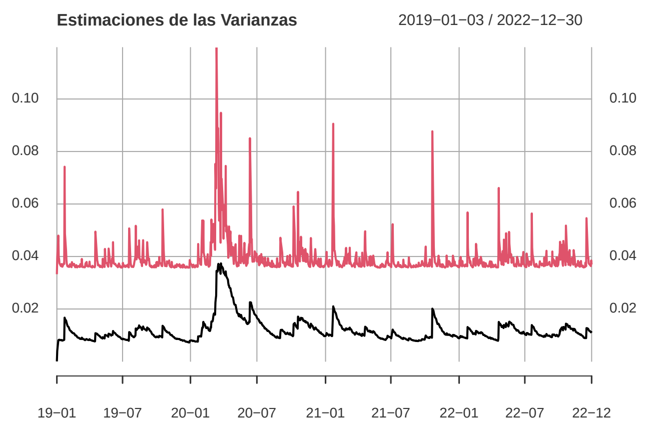
```
stan_dens(fit,pars="p_remain", point_est = "mean", show_density = TRUE) +
  ggtitle("Distribución final de p_remain") + ylab("Densidad") +
  theme(axis.title.x=element_text(size=14), axis.title.y=element_text(size=14),
        plot.title = element_text(size=16))
```



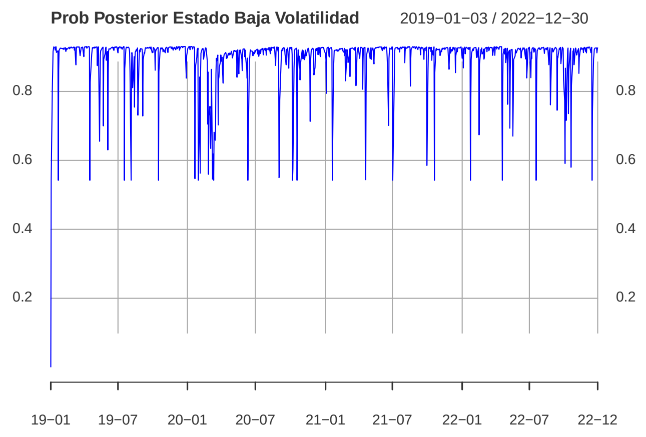
```
stan_dens(fit,pars="A", point_est = "mean", show_density = TRUE) +
  ggtitle("Distribución final de A") + ylab("Densidad") +
  theme(axis.title.x=element_text(size=14), axis.title.y=element_text(size=14),
        plot.title = element_text(size=16))
```



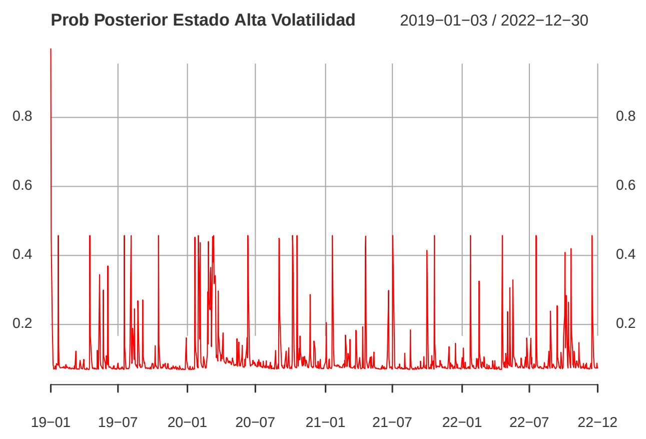
```
garch_posterior_means <- xts(apply(extract(fit, "sigma_t")[[1]], 2:3, mean),
                             index(IBM.R))
colnames(garch_posterior_means) <- c("Low-Vol State", "High-Vol State")

plot(
  garch_posterior_means,
  main = "Estimaciones de las Varianzas",
  format.labels = "%y-%m"
)
```



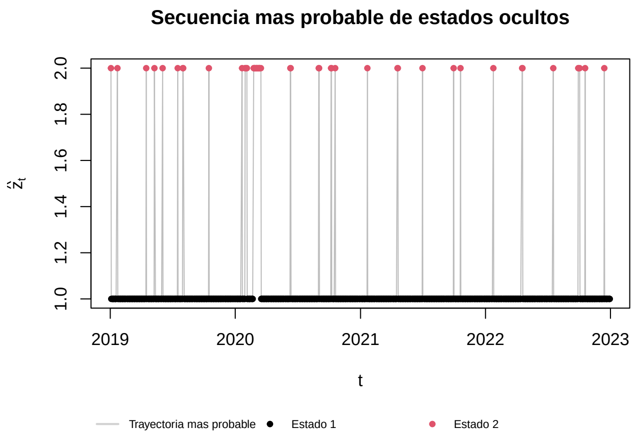
```
garch_posterior_prob1 <- xts(apply(extract(fit, "alpha")[[1]], 2:3, mean)[,1],
                              index(IBM.R))

plot(
  garch_posterior_prob1,
  main = "Prob Posterior Estado Baja Volatilidad",
  format.labels = "%y-%m",
  col="blue",lwd=1
)
```



```
garch_posterior_prob2 <- xts(apply(extract(fit, "alpha")[[1]], 2:3, mean)[,2],
                              index(IBM.R))
plot(
  garch_posterior_prob2,
  main = "Prob Posterior Estado Alta Volatilidad",
  format.labels = "%y-%m",
  col="red",lwd=1
)
```



```r
zstar <- xts(apply(extract(fit, "zstar")[[1]], 2, median),
                                    index(IBM.R))
plot_statepath(zstar)
```


Secuencia mas probable de estados ocultos
$\hat{z}_t$
t
Trayectoria mas probable
Estado 1
Estado 2