Ejemplos

Lizbeth Naranjo Albarrán y Luz Judith Rodriguez Esparza

Paper: *Modelos ocultos de Markov:*

una aplicación de estimación Bayesiana para series de tiempo financieras

Authors: Lizbeth Naranjo Albarrán & Luz Judith Rodríguez Esparza

Journal: Mixba'al

Year: 2023

https://github.com/lizbethna/HMMBayes.git

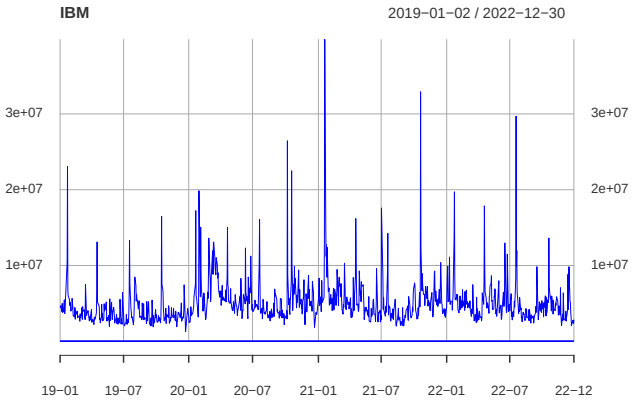
Este archivo muestra las instrucciones para correr los códigos de R y Stan.

GARCH

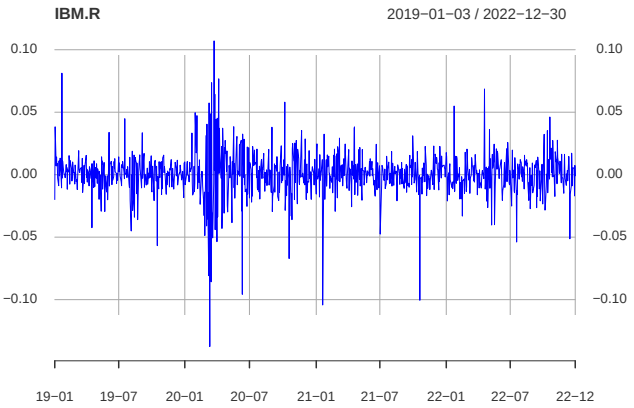
```
library(ggplot2)
library(rstan) # RStan
library(quantmod) # Quantitative Financial Modelling Framework
```

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

```
IBM.R <- as.vector(coredata(IBM.R));
datos <- list("rend"=IBM.R, "N"=length(IBM.R), "sigma1"=(IBM.R[1]^2))
param  = c("mu","alpha0","alpha1","beta1") # parametros a estimar

fit_garch <- stan("ts_garch.stan", data=datos,
            chains=2, warmup=1000, iter=2000, thin=2,
            verbose=FALSE)
```

```
SAMPLING FOR MODEL 'ts_garch' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000272 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 2.72 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 4.59451 seconds (Warm-up)
Chain 1:                2.77702 seconds (Sampling)
Chain 1:                7.37153 seconds (Total)
Chain 1:
```

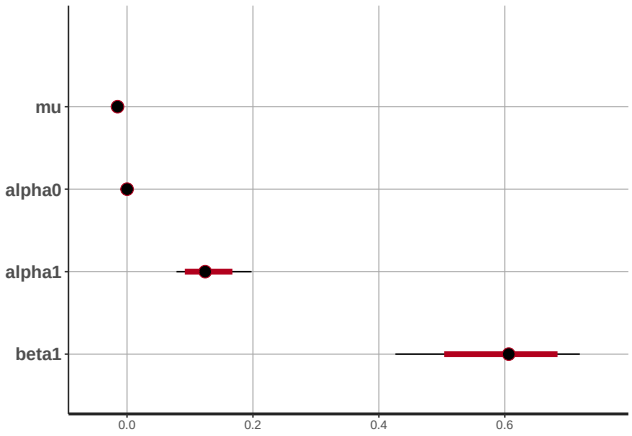
```
SAMPLING FOR MODEL 'ts_garch' NOW (CHAIN 2).
Chain 2:
Chain 2: Gradient evaluation took 0.000139 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.39 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2:
Chain 2:  Elapsed Time: 7.64351 seconds (Warm-up)
Chain 2:                2.53095 seconds (Sampling)
Chain 2:                10.1745 seconds (Total)
Chain 2:
```

Resultados

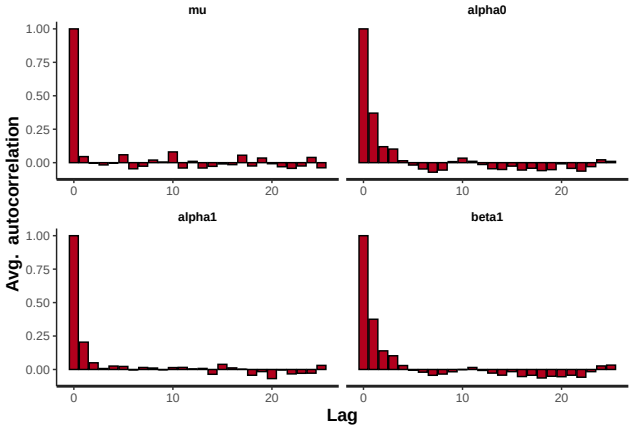
```
print(fit_garch, pars=param)
```

```
Inference for Stan model: ts_garch.
2 chains, each with iter=2000; warmup=1000; thin=2;
post-warmup draws per chain=500, total post-warmup draws=1000.

        mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
mu     -0.01       0 0.00 -0.02 -0.02 -0.02 -0.01 -0.01   920 1.00
alpha0  0.00       0 0.00  0.00  0.00  0.00  0.00  0.00   450 1.01
alpha1  0.13       0 0.03  0.08  0.11  0.12  0.15  0.20   601 1.00
beta1   0.60       0 0.08  0.43  0.55  0.61  0.65  0.72   417 1.01

Samples were drawn using NUTS(diag_e) at Thu Jun  8 13:19:52 2023.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

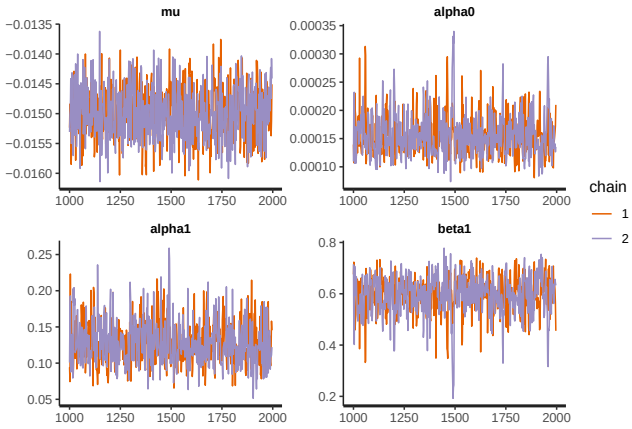
```
stan_plot(fit_garch,pars=param)
```




```
stan_ac(fit_garch,pars=param)
```




```
stan_trace(fit_garch,pars=param)
```




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

Distribución final