Modelo GARCH(1,1).

Introducción y aplicación.

Dr. Martín Lozano `https://mlozanoqf.github.io/`

12 de noviembre de 2025, 05:42 p.m.

	Fundamental	Intermedio	Especializado
Finanzas	×	✓	×
Estadística	×	✓	×
R	×	✓	×

1 Introducción.

- Estimación de la volatilidad condicional del índice S&P 500 mediante un modelo GARCH(1,1).
- Aplicar dos estrategias de estimación: un ajuste por máxima verosimilitud completa (Full MLE) y un método con *variance targeting*.
- Ejemplo práctico usando datos del S&P 500.

2 Paquetes.

```r
library(tidyquant)
library(dplyr)
library(knitr)
library(ggplot2)
library(tidyr)
library(scales)
```

3 Rendimientos.

$u_i = \frac{S_i - S_{i-1}}{S_{i-1}} \rightarrow u_i = \frac{S_i}{S_{i-1}} - 1.$

$\rightarrow u_2 = \frac{2099.60}{2076.62} - 1 \rightarrow u_2 = 0.0110660492.$

```r
S <- tq_get("^GSPC", from = "2015-07-10", to = "2020-07-10") |>
  dplyr::select(date, S = close) |>
  mutate(i = dplyr::row_number(),
         u_i = (S / lag(S)) - 1)

kable(rbind(head(S, 6), tail(S, 2)), digits = 10, format.args = list(scientific = FALSE))
```

date	S	i	u_i
2015-07-10	2076.62	1	NA
2015-07-13	2099.60	2	0.0110660492
2015-07-14	2108.95	3	0.0044531592
2015-07-15	2107.40	4	-0.0007349861
2015-07-16	2124.29	5	0.0080146804
2015-07-17	2126.64	6	0.0011061830
2020-07-08	3169.94	1258	0.0078274619
2020-07-09	3152.05	1259	-0.0056436062

4 Inicialización.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta\sigma_{n-1}^2$.

$v_i = \sigma_i^2$.

```r
garch_variance_shifted <- function(u, omega, alpha, beta) {
  n <- length(u) # 1258
  v <- rep(NA_real_, n) # inicializa el vector de varianzas condicionales v.
  if (n >= 2) v[2] <- u[1]^2 # valor inicial de v.
  if (n >= 3) { # a partir de aquí se usa la ecuación de GARCH(1,1).
    for (i in 3:n) {
      v[i] <- omega + alpha * u[i - 1]^2 + beta * v[i - 1]
    }
  }
  v
}

S <- S |>
  mutate(v_i = case_when(i <= 2 ~ NA_character_,
                         i == 3 ~ "u_i^2",
                         i >= 4 ~ "GARCH(1,1)"))

kable(rbind(head(S, 6), tail(S, 2)), digits = 10, format.args = list(scientific = FALSE))
```

date	S	i	u_i	v_i
2015-07-10	2076.62	1	NA	NA
2015-07-13	2099.60	2	0.0110660492	NA
2015-07-14	2108.95	3	0.0044531592	u_i^2
2015-07-15	2107.40	4	-0.0007349861	GARCH(1,1)
2015-07-16	2124.29	5	0.0080146804	GARCH(1,1)
2015-07-17	2126.64	6	0.0011061830	GARCH(1,1)
2020-07-08	3169.94	1258	0.0078274619	GARCH(1,1)
2020-07-09	3152.05	1259	-0.0056436062	GARCH(1,1)

5 Estimación Full MLE.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$, $v_i = \sigma_i^2$.

Función de log-verosimilitud condicional: $\sum_{i=1}^{m} \left[-ln(v_i) - \frac{u_i^2}{v_i} \right]$.

```r
u <- S$u_i |> stats::na.omit()

nll_garch_shifted <- function(par, u) {
  omega <- par[1]; alpha <- par[2]; beta <- par[3]

  if (omega <= 0 || alpha < 0 || beta < 0 || (alpha + beta) >= 1)
    return(1e12)
  v <- garch_variance_shifted(u, omega, alpha, beta)
  mask <- !is.na(v) & v > 0 # identifica entradas válidas y positivas.
  if (!any(mask)) return(1e12)
  -sum( -log(v[mask]) - (u[mask]^2) / v[mask] )} # neg-log-likelihood Gaussiana.

start <- c(omega = 4e-6, alpha = 0.2, beta = 0.7)
# Broyden-Fletcher-Goldfarb-Shanno que usa memoria limitada y cotas de los parámetros.
fit <- optim(par = start, fn = nll_garch_shifted, u = u,
             method = "L-BFGS-B", lower  = c(1e-12, 0, 0),
             upper  = c(Inf, 1 - 1e-6, 1 - 1e-6)) # maximiza vía optim.

theta_hat <- fit$par
omega_hat <- theta_hat[1]; alpha_hat <- theta_hat[2]; beta_hat <- theta_hat[3]

v_hat <- garch_variance_shifted(u, omega_hat, alpha_hat, beta_hat)
mask <- !is.na(v_hat) & v_hat > 0
ell_i <- -log(v_hat[mask]) - (u[mask]^2) / v_hat[mask]
ll_tot <- sum(ell_i)

n_obs <- sum(mask)  # 1257.
k <- 3 # estimamos 3 parámetros.
AIC <- -2 * ll_tot + 2 * k
BIC <- -2 * ll_tot + log(n_obs) * k

summary_tbl <-
  tibble::tibble(metric = c("omega", "alpha", "beta", "logLik", "n_obs", "AIC", "BIC"),
                 value  = c(theta_hat, ll_tot, n_obs, AIC, BIC))
```

6 Estimación Full MLE - Resultados.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$, $v_i = \sigma_i^2$.

La función de log-verosimilitud condicional Gaussiana: $\sum_{i=1}^{m} \left[-ln(v_i) - \frac{u_i^2}{v_i} \right]$.

```r
knitr::kable(summary_tbl, align = c("l", "r"), digits = c(0, 6),
             format.args = list(scientific = FALSE),
             col.names = c("Estimador / métrica", "Valor"))
```

Estimador / métrica	Valor
omega	0.000003
alpha	0.229733
beta	0.758404
logLik	10835.375594
n_obs	1257.000000
AIC	-21664.751188
BIC	-21649.341739

7 Estimación *variance targeting.*

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$.

$\omega = V_L(1 - \alpha - \beta)$.

$v_i = \sigma_i^2$.

La función de log-verosimilitud condicional Gaussiana: $\sum_{i=1}^{m} \left[-ln(v_i) - \frac{u_i^2}{v_i} \right]$.

```r
VL <- var(u, na.rm = TRUE)

nll_garch_VT_shifted <- function(par, u, VL) {
  alpha <- par[1]; beta <- par[2]
  if (alpha < 0 || beta < 0 || (alpha + beta) >= 1) return(1e12)
  omega <- VL * (1 - alpha - beta)
  v <- garch_variance_shifted(u, omega, alpha, beta)
  mask <- !is.na(v) & v > 0
  if (!any(mask)) return(1e12)
  -sum( -log(v[mask]) - (u[mask]^2) / v[mask] )}

fit_vt <- optim(par = c(alpha = 0.2, beta = 0.7), fn = nll_garch_VT_shifted,
                u = u, VL = VL, method = "L-BFGS-B", lower = c(0, 0),
                upper = c(1 - 1e-6, 1 - 1e-6))

alpha_vt <- fit_vt$par[1]
beta_vt  <- fit_vt$par[2]
omega_vt <- VL * (1 - alpha_vt - beta_vt)

v_vt   <- garch_variance_shifted(u, omega_vt, alpha_vt, beta_vt)
mask_vt <- !is.na(v_vt) & v_vt > 0
ell_vt <- -log(v_vt[mask_vt]) - (u[mask_vt]^2) / v_vt[mask_vt]
ll_vt  <- sum(ell_vt)

n_obs_vt <- sum(mask_vt) # 1257.
k_vt <- 2
AIC_vt <- -2 * ll_vt + 2 * k_vt
BIC_vt <- -2 * ll_vt + log(n_obs_vt) * k_vt

summary_vt <- tibble::tibble(
  metric = c("omega (target)", "alpha", "beta", "logLik", "n_obs", "AIC", "BIC"),
  value  = c(omega_vt, alpha_vt, beta_vt, ll_vt, n_obs_vt, AIC_vt, BIC_vt))
```

8 Estimación *variance targeting* - Resultados.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$.

$\omega = V_L(1 - \alpha - \beta)$.

$v_i = \sigma_i^2$.

La función de log-verosimilitud condicional Gaussiana: $\sum_{i=1}^{m} \left[-ln(v_i) - \frac{u_i^2}{v_i} \right]$.

```r
knitr::kable(summary_vt, align = c("l", "r"), digits = c(0, 6),
             format.args = list(scientific = FALSE),
             col.names = c("Estimador / métrica", "Valor"))
```

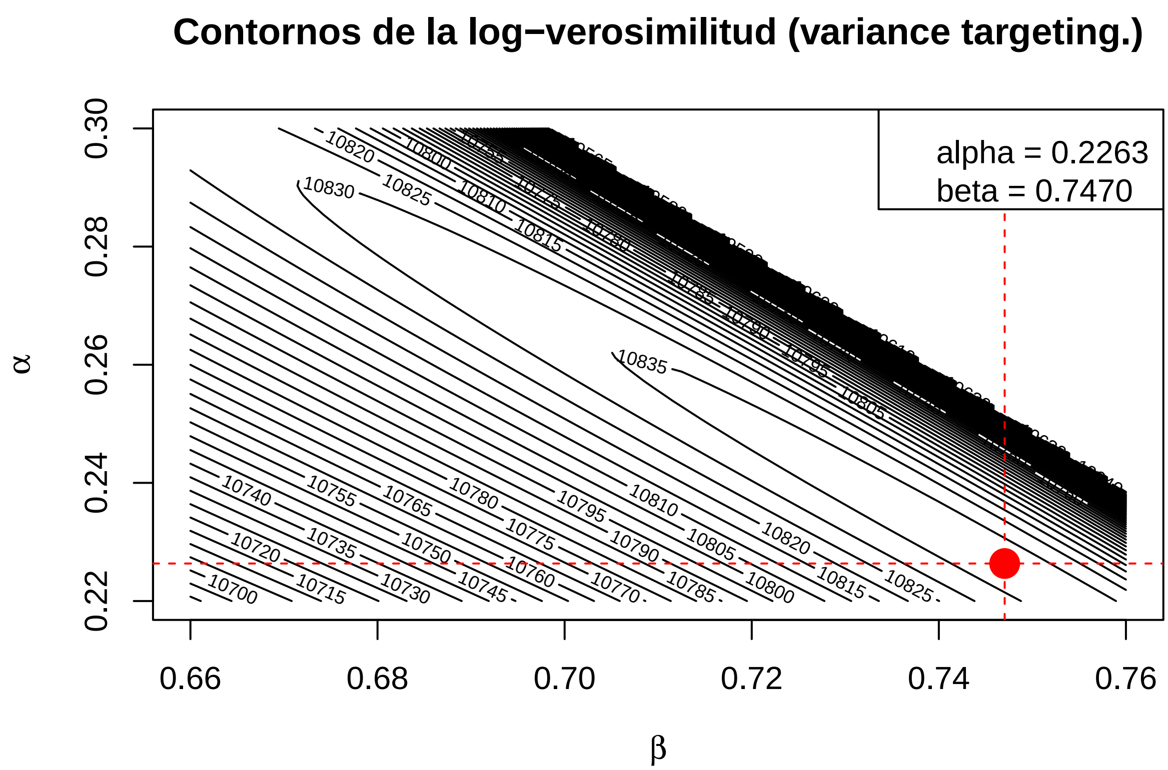
Estimador / métrica	Valor
omega (target)	0.000004
alpha	0.226349
beta	0.747038
logLik	10837.404644
n_obs	1257.000000
AIC	-21670.809288
BIC	-21660.536321

9 Visualización de la minimización *variance targeting.*

```
alphas <- seq(0.22, 0.3, length.out = 100)
betas  <- seq(0.66, 0.76, length.out = 100)

ll_surface <- outer(alphas, betas, Vectorize(function(a, b) {
  if (a < 0 || b < 0 || (a + b) >= 0.999) return(NA_real_)
  -nll_garch_VT_shifted(c(a, b), u = u, VL = VL)}))

contour(x = betas, y = alphas, z = t(ll_surface), nlevels = 50,
        xlab = expression(beta), ylab = expression(alpha),
        main = "Contornos de la log-verosimilitud (variance targeting.)")
points(beta_vt, alpha_vt, pch = 19, cex = 2, col = "red")
abline(h = alpha_vt, col = "red", lty = 2)
abline(v = beta_vt, col = "red", lty = 2)
  legend("topright",
    legend = sprintf("alpha = %.4f\nbeta = %.4f", alpha_vt, beta_vt),
    bg = "white", text.col = "black")
```



10 MLE versus *variance targeting.*

```r
comparison_tbl <- dplyr::bind_rows(
  summary_tbl |> dplyr::mutate(model = "Full MLE"),
  summary_vt  |> dplyr::mutate(
    metric = dplyr::if_else(metric == "omega (target)", "omega", metric),
    model  = "Variance targeting")) |>
  tidyr::pivot_wider(names_from = model, values_from = value) |>
  dplyr::select(metric, `Full MLE`, `Variance targeting`)

  knitr::kable(comparison_tbl,
    col.names = c("Estimador / métrica", "Full MLE", "Variance targeting"),
    align = c("l", "r", "r"), digits = c(0, 6, 6),
    format.args = list(scientific = FALSE))
```

Estimador / métrica	Full MLE	Variance targeting
omega	0.000003	0.000004
alpha	0.229733	0.226349
beta	0.758404	0.747038
logLik	10835.375594	10837.404644
n_obs	1257.000000	1257.000000
AIC	-21664.751188	-21670.809288
BIC	-21649.341739	-21660.536321

11 Resultado.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$.

$\omega = V_L(1 - \alpha - \beta)$.

$v_i = \sigma_i^2$.

```r
omega <- omega_vt
alpha <- alpha_vt
beta  <- beta_vt

S <- S |>
  mutate(v_i = lag(u_i^2))

for (i in 2:nrow(S)) {
  if (!is.na(S$u_i[i - 1]) && !is.na(S$v_i[i - 1])) {
    S$v_i[i] <- omega + alpha * (S$u_i[i - 1]^2) + beta  *  S$v_i[i - 1]
  }
}

kable(rbind(head(S, 6), tail(S, 2)), digits = 10, format.args = list(scientific = FALSE))
```

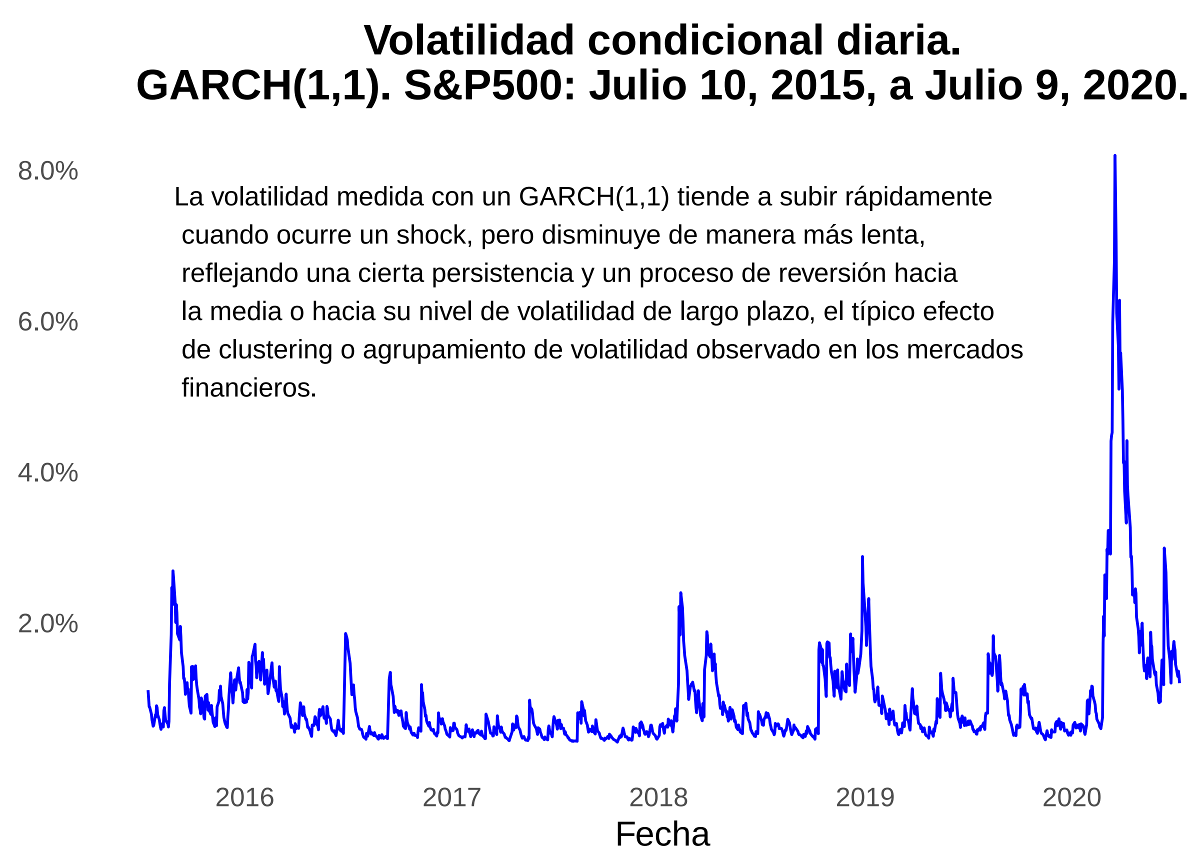
date	S	i	u_i	v_i
2015-07-10	2076.62	1	NA	NA
2015-07-13	2099.60	2	0.0110660492	NA
2015-07-14	2108.95	3	0.0044531592	0.0001224574
2015-07-15	2107.40	4	-0.0007349861	0.0000999353
2015-07-16	2124.29	5	0.0080146804	0.0000787441
2015-07-17	2126.64	6	0.0011061830	0.0000773307
2020-07-08	3169.94	1258	0.0078274619	0.0001684868
2020-07-09	3152.05	1259	-0.0056436062	0.0001437005

12 Gráfico.

GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta\sigma_{n-1}^2$.

```r
y_top <- max(100 * sqrt(S$v_i), na.rm = TRUE) * 0.95
x_left <- min(S$date, na.rm = TRUE) + 50

ggplot(S, aes(x = date, y = 100 * sqrt(v_i))) +
  geom_line(color = "blue") +
  labs(title = "Volatilidad condicional diaria.\nGARCH(1,1). S&P500: Julio 10, 2015, a Julio 9, 2020.",
    y = NULL, x = "Fecha") +
  scale_y_continuous(labels = label_percent(accuracy = 0.1, scale = 1)) +
  annotate("text", x = x_left, y = y_top,
    label = "La volatilidad medida con un GARCH(1,1) tiende a subir rápidamente \n cuando ocurre un shock, pero disminuye de manera más lenta,\n reflejando una cierta persistencia y un proceso de reversión hacia \n la media o hacia su nivel de volatilidad de largo plazo, el típico efecto \n de clustering o agrupamiento de volatilidad observado en los mercados \n financieros.",
    hjust = 0, vjust = 1, size = 3.5, color = "black") +
  theme_minimal(base_size = 13) +
  theme(plot.title = element_text(face = "bold", hjust = 0.5),
        axis.title.y = element_text(angle = 0, vjust = 0.5)) +
  theme(panel.background = element_rect(fill = "white", color = NA),
        plot.background = element_rect(fill = "white", color = NA), panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```

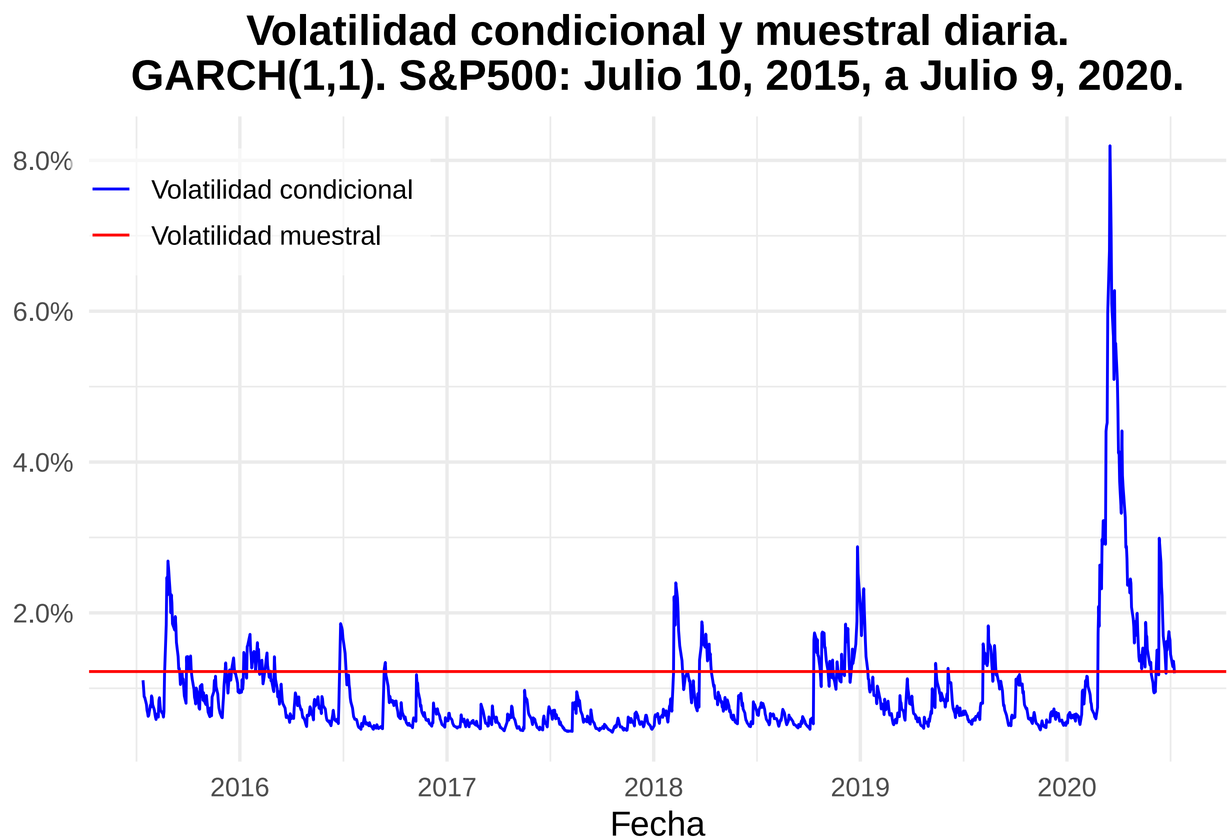


13 Comparación.

Azul, GARCH(1,1): $\sigma_n^2 = \omega + \alpha u_{n-1}^2 + \beta \sigma_{n-1}^2$.

Rojo, Volatilidad no condicional: $s = \frac{1}{n-1} \sum_{i=1}^{n} (u_i - \bar{u})^2$.

```r
ggplot(S, aes(x = date, y = 100 * v_i^.5, colour = "Volatilidad condicional")) +
  geom_line() +
  geom_hline(aes(yintercept = 1.221865, colour = "Volatilidad muestral")) +
  scale_colour_manual(
    values = c("Volatilidad condicional" = "blue", "Volatilidad muestral" = "red"),
    name = NULL) +
  labs(
    title = "Volatilidad condicional y muestral diaria.\nGARCH(1,1). S&P500: Julio 10, 2015,
     a Julio 9, 2020.",
    y = NULL, x = "Fecha") +
  scale_y_continuous(labels = scales::label_percent(accuracy = 0.1, scale = 1)) +
  theme_minimal(base_size = 13) +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    axis.title.y = element_text(angle = 0, vjust = 0.5),
    legend.position = c(0.3, 0.95),
    legend.justification = c("right", "top"),
    legend.background = element_rect(fill = scales::alpha("white", 0.6), colour = NA))
```



14 Conclusión.

- El modelo *variance targeting* presenta mayor log-verosimilitud y menores valores de AIC y BIC.
- Por tanto, se selecciona el GARCH(1,1) con *variance targeting*.