Estimação de parâmetros

Garch(p,q)

Sumário

1 Introdução

O pacote `rugarch` conta com 10 modelos derivados do GARCH. São eles:

- GARCH padrão ('sGARCH')
- integrated GARCH ('iGARCH')
- exponential GARCH ('eGARCH')
- GJR-GARCG ('gjrGACH')
- assymetric power ARCH ('apARCH')
- family GARCH ('fGARCH')
- component sGARCH ('csGARCH')

- multiplicative component sGARCH ('mcsGARCH')
- realized GARCH ('realGARCH')
- fractionally integrated GARCH ('fiGARCH')

Com 7 distribuições condicionais a disposição.

2 Estimação

Para estimar o modelo GARCH utilizamos duas funções: `ugarchspec` e `ugarchfit`.

2.1 `ugarchspec`

A função `ugarchspec` possui os seguintes parâmetros:

2.1.1 `variance.model`

Uma lista contendo:

- model: string com um dos modelos comentados anteriormente
- garchOrder: vetor contendo a ordem do modelo (p, q)
- submodel: se o modelo for "fGARCH", pode usar como submodelo: "GARCH","TGARCH", "AVGARCH","NGARCH","NAGARCH","APARCH","GJRGARCH" e "ALL GARCH"
- external.regressors: matriz $(n \times k)$ de variáveis exógenas.
- variance.targeting: se "TRUE" vai usar *variance targeting* para o ω. Se for dado um valor numérico, esse valor será usado no lugar da variância incondicional para calcular o intercepto.

2.1.2 `mean.model`

- armaOrder: vetor contendo a ordem do modelo ARMA(p, q)
- include.mean: Lógico (incluir a média ou não)
- archm: Lógico (incluir a volatilidade do ARCH na regressão da média).
- archpow: usar desv.pad (1) ou variância (2) no ARCH na regressão da média.
- external.regressors: matriz $(n \times k)$ de variáveis exógenas.
- archex: (integer) Whether to multiply the last 'archex' external regressors by the conditional standard deviation.

2.1.3 `distribution.model`

A distribuição condicional para as inovações: “norm” (Normal), “snorm” (Skew-Normal), “std” (t), “sstd” (skew-t), “ged” (GED), “sged” (skew-GED), “nig” (Normal Inversa), “ghyp” (Generalized Hyperbolic) e “jsu” (Johnson’s SU).

2.1.4 `start.pars`

Lista com parâmetros iniciais para a rotina de otimização (só usada caso a otimização padrão não convirja).

2.1.5 `fixed.pars`

Lista de parâmetros a serem mantidos fixos na otimização.

2.2 `ugarchfit`

2.2.1 Argumentos

Como argumentos da função, os principais são `data` e `spec`. Além disso, pode-se passar também:

- `out.sample`: inteiro indicando o número de períodos antes do último para a predição “Out Of Sample”.
- `solver`: método computacional para otimização.
- `solver.control`: argumentos passados para a otimização.
- `fit.control`:
- `numderiv.control`: argumentos passados para o cálculo da rotina numérica. Como por exemplo `hess` para a hessiana e `grad` para o jacobiano.

2.2.2 Métodos

A classe `uGARCHfit` apresenta 2 slots: `fit` contendo as informações do ajuste do modelo, e `model` cotendo informações do modelo ajustado.

Além disso, os métodos para a classe são:

- `coef`: extrai os coeficientes.
- `cofint`: extrai os intervalos de confiança.
- `vcov`: extrai a matriz de covariância dos coef.
- `infocriteria`: calcula os critérios de informação.
- `sigma`: extrai as volatilidades.
- `fitted`: extrai os valores ajustados.
- `residuals`: extrai os resíduos.
- `plot`: ajusta alguns gráficos (argumento `which` para escolher qual).
- `show`: mostra o sumário do modelo.

Além disso, outros argumentos podem ser passados como: `nyblom` (stability-test), `gof` (bondade de ajuste), `newsimpact` (curva de "news impact"), `signbias` (sign bias of Engle), `likelihood` (verossimilhança), `getspec` (retorna o spec), `uncvariance` (variância incondicional), `uncmean` (média incondicional), `persistence` (calcula a persistencia), `halflife` (calcula meia-vida), `convergence` (código de otimização), `quantile` (quantis condicionais), `pit` (probabilidade condicional integral).

3 Aplicação

3.1 Simulação

Nessa parte, realizo a estimação de algumas séries simuladas pela minha função criada. A comparação da estimação será feita via método *rolling windows*.

O código da função criada se encontra abaixo:

```
garch_sim <- function(n, alpha, beta) { # Erros Normais
  # alpha[1] -> omega ; demais alphas compoem a somatoria dos retornos
  n_burnin <- 500
  order_max <- max(length(alpha) - 1, length(beta))
  epsilon <- rnorm(n + order_max + n_burnin)
  sigma2 <- rep(0, n + order_max + n_burnin)
  ret <- rep(0, n + order_max + n_burnin)

  for (i in (order_max + 1):(n + order_max + n_burnin)) {
    sum_alpha <- 0
```

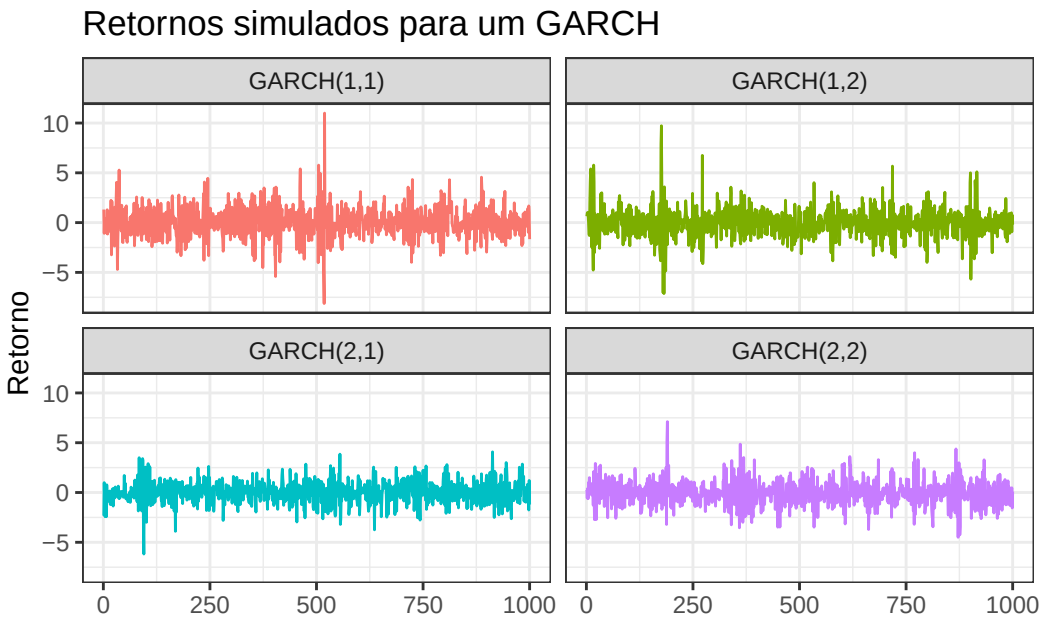
```
    sum_beta <- 0
    for (p in 2:(length(alpha))) {sum_alpha <- sum_alpha +
      alpha[p]*(ret[i - p + 1])^2}
    for (q in 1:length(beta)) {sum_beta <- sum_beta +
      beta[q]*sigma2[i - q]}
    sigma2[i] <- alpha[1] + sum_alpha + sum_beta
    ret[i] <- sqrt(sigma2[i])*epsilon[i]
  }
  return(list(returns = ret[-c(1:(n_burnin + order_max))],
              volatility = sqrt(sigma2[-c(1:(n_burnin + order_max))])))
}
```

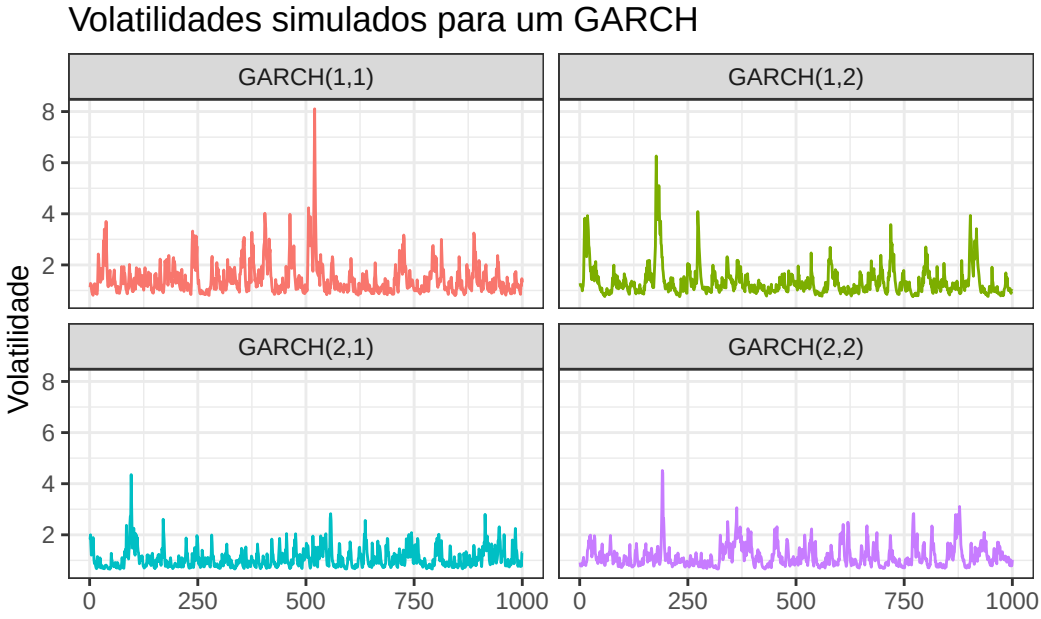
A partir da função, simulei quatro amostras de 1000 observações seguindo os seguintes modelos:

- Modelo 1: $GARCH(1,1) : \omega = 0.3, \alpha = 0.4, \beta = 0.5$
- Modelo 2: $GARCH(1,2) : \omega = 0.2, \alpha = 0.3, \beta_1 = 0.4, \beta_2 = 0.2$
- Modelo 3: $GARCH(2,1) : \omega = 0.3, \alpha_1 = 0.4, \alpha_2 = 0.1, \beta = 0.3$
- Modelo 4: $GARCH(2,2) : \omega = 0.3, \alpha_1 = 0.2, \alpha_2 = 0.3, \beta_1 = 0.1, \beta_2 = 0.2$

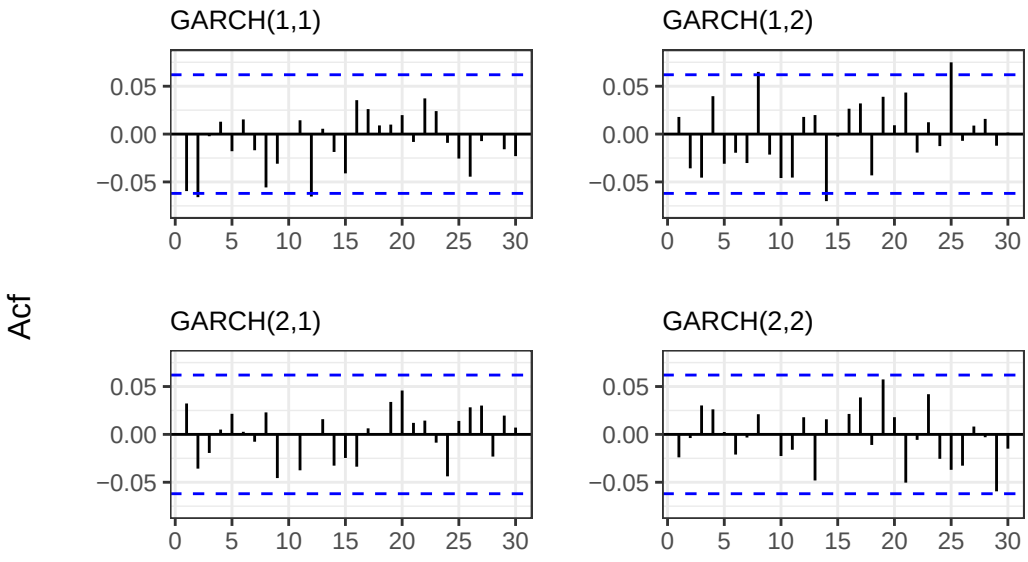
```
set.seed(236106)
n <- 1000

# Garch(1, 1)
amostra1 <- garch_sim(n, alpha = c(.3, .4), beta = .5)

# Garch(1, 2)
amostra12 <- garch_sim(n, alpha = c(.2, .3), beta = c(.4, .2))

# Garch(2, 1)
amostra21 <- garch_sim(n, alpha = c(.3, .4, .1), beta = .3)

# Garch(2, 2)
amostra22 <- garch_sim(n, alpha = c(.3, .2, .3), beta = c(.1, .2))
```

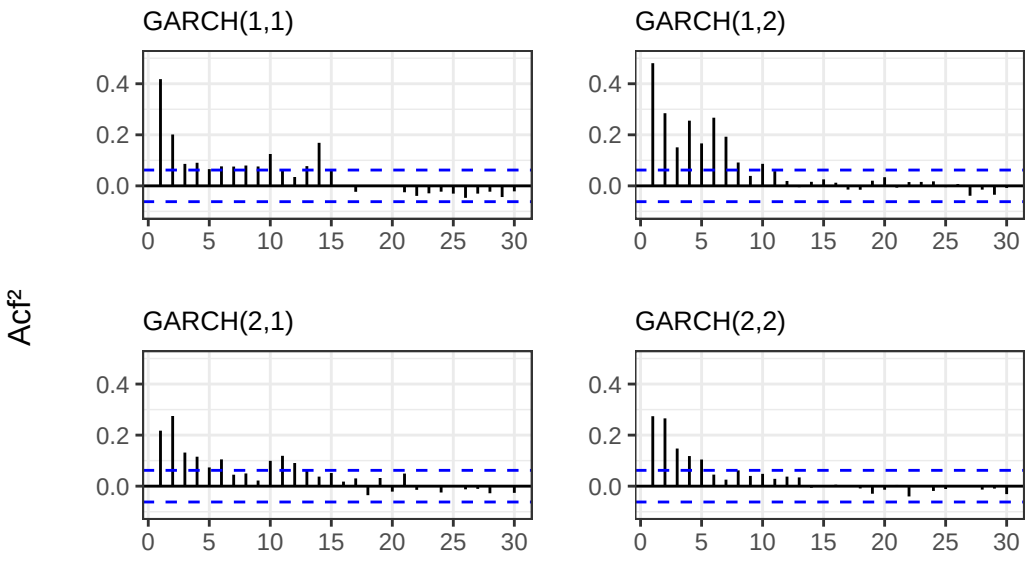
Os gráficos dos retornos e das volatilidades dos 4 modelos podem ser vistos a baixo:





As funções de autocorrelações e seus quadrados são mostradas abaixo:





Como era esperado, não há autocorrelação significativa para os retornos, mas há para os retornos ao quadrado (queremos modelar a variância condicional, não a média).

3.2 Estimação

Criaremos um ugarchspec para cada modelo, especificando sua respectiva ordem.

```
# Garch(1,1)
spec11 <- ugarchspec(variance.model = list(model = 'sGARCH',
                                           garchOrder = c(1, 1)),
                     mean.model = list(armaOrder = c(0, 0),
                                       include.mean = FALSE),
                     distribution.model = "norm")

# Garch(1,2)
spec12 <- ugarchspec(variance.model = list(model = 'sGARCH',
                                           garchOrder = c(1, 2)),
                     mean.model = list(armaOrder = c(0, 0),
                                       include.mean = FALSE),
                     distribution.model = "norm")
# Garch(2,1)
spec21 <- ugarchspec(variance.model = list(model = 'sGARCH',
                                           garchOrder = c(2, 1)),
                     mean.model = list(armaOrder = c(0, 0),
                                       include.mean = FALSE),
                     distribution.model = "norm")
# Garch(2,2)
spec22 <- ugarchspec(variance.model = list(model = 'sGARCH',
                                           garchOrder = c(2, 2)),
                     mean.model = list(armaOrder = c(0, 0),
                                       include.mean = FALSE),
                     distribution.model = "norm")
```

E ajustaremos o modelo com o `ugarchfit`:

```
# Garcha(1,1)
fit11 <- ugarchfit(spec11, amostra1$returns)

# Garcha(1,2)
fit12 <- ugarchfit(spec12, amostra12$returns)

# Garcha(2,1)
fit21 <- ugarchfit(spec21, amostra21$returns)

# Garcha(2,2)
fit22 <- ugarchfit(spec22, amostra22$returns)
```

Os coeficientes estimados foram:

Table 1: Parâmetros Estimados dos Modelos

Modelo	ω	α_1	α_2	β_1	β_2
GARCH(1, 1)	0.3968	0.4149	—	0.4313	—
GARCH(1, 2)	0.2967	0.4132	—	0.3087	0.1500
GARCH(2, 1)	0.3352	0.2635	0.2368	0.2580	—
GARCH(2, 2)	0.2918	0.2160	0.2575	0.2685	0.1014

Table 2: Parâmetros Reais dos Modelos

Modelo	ω	α_1	α_2	β_1	β_2
GARCH(1, 1)	0.3	0.4	—	0.5	—
GARCH(1, 2)	0.2	0.3	—	0.4	0.2
GARCH(2, 1)	0.3	0.4	0.1	0.3	—
GARCH(2, 2)	0.3	0.2	0.3	0.1	0.2

Além disso, os quadrados das autocorrelações dos resíduos são mostrados abaixo: