Lista 3 - Séries Temporais

Mateus Risso - 236237

2025-04-29

```r
#2)b
#Yt = c + phi1.Y{t-1} + phi2.Y{t-2} + et
estimar_ar2 <- function(y) {
  verossimilhanca_neg <- function(params) {
    c <- params[1]
    phi1 <- params[2]
    phi2 <- params[3]
    sigma <- exp(params[4])
    n <- length(y)
    soma <- 0

    for (t in 3:n) {
      residuo <- y[t] - (c + phi1*y[t-1] + phi2*y[t-2]) #Calcula Resíduo
      soma <- soma + dnorm(residuo, mean = 0, sd = sigma, log = TRUE)
    }

    return(-soma)  # Minimização atráves de soma negativa
  }

  # Definindo valores iniciais
  iniciais <- c(mean(y), 0.5, 0.1, log(sd(y)))
  opt <- optim(iniciais, verossimilhanca_neg, method = "L-BFGS-B",
               lower = c(-Inf, -1, -1, -Inf),
               upper = c(Inf, 1, 1, Inf))
  params <- opt$par
  params[4] <- exp(params[4])  #Transformar sigma novamente

  list(
    intercepto = params[1],
    phi1 = params[2],
    phi2 = params[3],
    sigma = params[4],
    log_verossimilhanca = -opt$value
  )
}
#FIM
# Dados Simulados
set.seed(236237)
y <- numeric(100)
y[1:2] <- 10  # Definindo valores iniciais

# Simular AR(2)
```

```r
#y_t = 2 + 0.4*y[t-1] + 0.6*t[t-2] + et
for (t in 3:100) {
  y[t] <- 2 + 0.4*y[t-1] + 0.6*y[t-2] + rnorm(1)
}

# Modelo
resultado <- estimar_ar2(y)
print(resultado)
```

```
## $intercepto
## [1] 2.514908
##
## $phi1
## [1] 0.002338774
##
## $phi2
## [1] 1
##
## $sigma
## [1] 1.19775
##
## $log_verossimilhanca
## [1] -156.7395
```

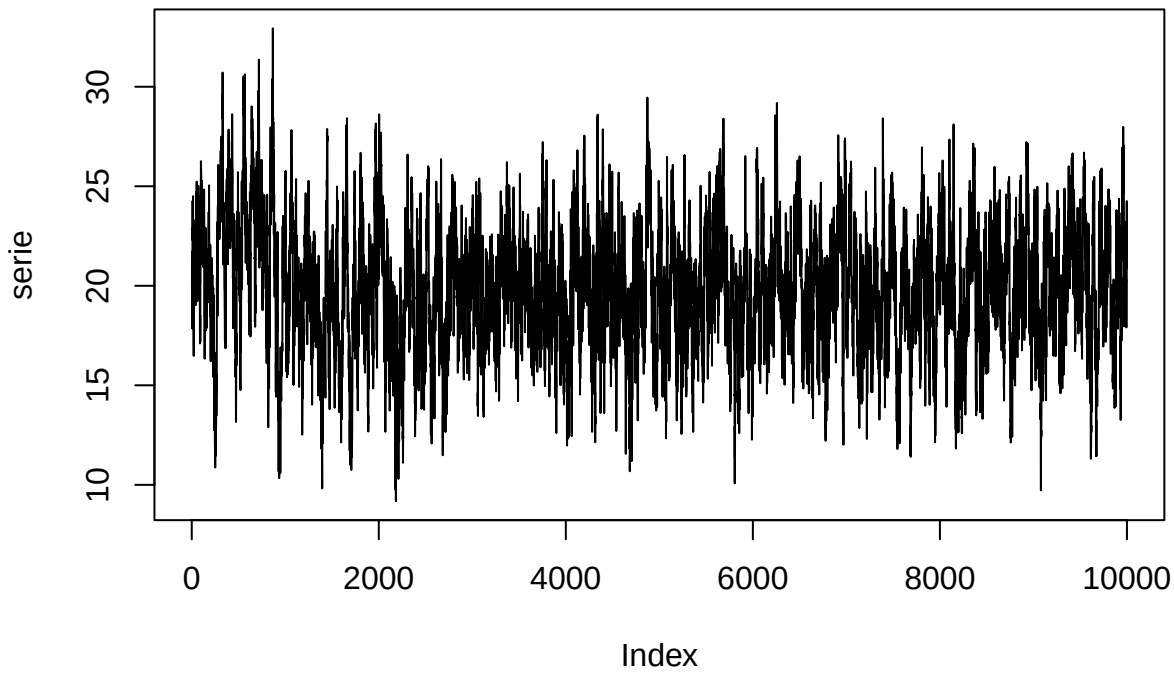
```r
set.seed(236237)
n <- 10000
c <- 2
phi1 <- 0.6
phi2 <- 0.3
sigma <- 1.5

#Condições estacionaridade
#phi1 + phi2 < 1
#phi2 - phi1 < 1
#abs(phi2) < 1

simular_ar2 <- function(n, c, phi1, phi2, sigma, y0 = NULL) {
  y <- numeric(n)
  y[1:2] <- if (is.null(y0)) c/(1-phi1-phi2) else y0[1:2]

  for (t in 3:n) {
    y[t] <- c + phi1*y[t-1] + phi2*y[t-2] + rnorm(1, 0, sigma)
  }

  return(y)
}
serie <- simular_ar2(n, c, phi1, phi2, sigma)
plot(serie, type="l")
```




```r
# Função para estimar AR(2) por Yule-Walker
estimar_ar2_yw <- function(y) {
  # Autocorrelação
  gamma <- acf(y, plot = FALSE, lag.max = 2)$acf
  gamma0 <- gamma[1]
  gamma1 <- gamma[2]
  gamma2 <- gamma[3]

  # Phi_1 e Phi_2
  phi1_hat <- (gamma1*(1 - gamma2)) / (1 - gamma1^2)
  phi2_hat <- (gamma2 - gamma1^2) / (1 - gamma1^2)

  # EStimativa C e Var do erro
  c_hat <- mean(y) * (1 - phi1_hat - phi2_hat)
  sigma2_hat <- gamma0 * (1 - phi1_hat*gamma1 - phi2_hat*gamma2)

  return(list(
    intercepto = c_hat,
    phi1 = phi1_hat,
    phi2 = phi2_hat,
    sigma = sqrt(sigma2_hat)
  ))
}

# Maxima Verossimilhança
estimar_ar2_mle <- function(y) {
```

```r
  neg_loglik <- function(params) {
    c <- 2
    phi1 <- 0.6
    phi2 <- 0.3
    sigma <- 1.5
    n <- 10000
    soma <- 0

    for (t in 3:n) {
      residuo <- y[t] - (c + phi1*y[t-1] + phi2*y[t-2])
      soma <- soma + dnorm(residuo, mean = 0, sd = sigma, log = TRUE)
    }

    return(-soma)
  }

  # #Valores iniciais usando yw
  yw <- estimar_ar2_yw(y)
  iniciais <- c(yw$intercepto, yw$phi1, yw$phi2, log(yw$sigma))

  opt <- optim(iniciais, neg_loglik, method = "L-BFGS-B",
               lower = c(-Inf, -1, -1, -Inf),
               upper = c(Inf, 1, 1, Inf))

  params <- opt$par
  params[4] <- exp(params[4])

  list(
    intercepto = params[1],
    phi1 = params[2],
    phi2 = params[3],
    sigma = params[4],
    log_verossimilhanca = -opt$value
  )
}

set.seed(236237)
n <- 10000
c <- 2
phi1 <- 0.6
phi2 <- 0.3
sigma <- 1.5

y <- numeric(n)
y[1:2] <- c / (1 - phi1 - phi2)  # Valores Iniciais
for (t in 3:n) {
  y[t] <- c + phi1*y[t-1] + phi2*y[t-2] + rnorm(1, 0, sigma)
}

#Verossimilhança e YW
yw_est <- estimar_ar2_yw(y)
mle_est <- estimar_ar2_mle(y)
```

```r
resultados <- data.frame(
  Parâmetro = c("c", "phi1", "phi2", "sigma"),
  Verdadeiro = c(c, phi1, phi2, sigma),
  Yule_Walker = c(yw_est$intercepto, yw_est$phi1, yw_est$phi2, yw_est$sigma),
  MLE = c(mle_est$intercepto, mle_est$phi1, mle_est$phi2, mle_est$sigma)
)

# Mostrar resultados
print(resultados)
```

```
##   Parâmetro Verdadeiro Yule_Walker       MLE
## 1         c        2.0   1.8862783 1.8862783
## 2      phi1        0.6   0.6192009 0.6192009
## 3      phi2        0.3   0.2855858 0.2855858
## 4     sigma        1.5   0.4780127 0.4780127
```

```r
#os parametros convergem para o verdadeiro
```

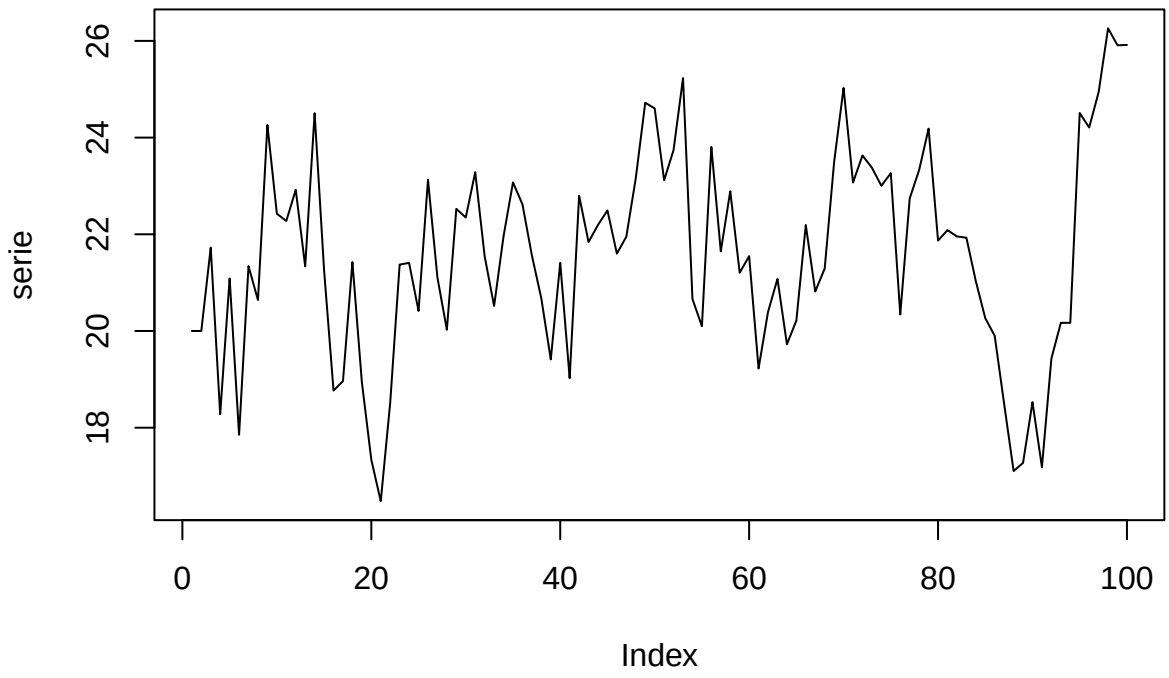
```r
#8
set.seed(236237)
n <- 100
c <- 2
phi1 <- 0.6
phi2 <- 0.3
sigma <- 1.5

#Condições estacionaridade
#phi1 + phi2 < 1
#phi2 - phi1 < 1
#abs(phi2) < 1

simular_ar2 <- function(n, c, phi1, phi2, sigma, y0 = NULL) {
  y <- numeric(n)
  y[1:2] <- if (is.null(y0)) c/(1-phi1-phi2) else y0[1:2]

  for (t in 3:n) {
    y[t] <- c + phi1*y[t-1] + phi2*y[t-2] + rnorm(1, 0, sigma)
  }

  return(y)
}
serie <- simular_ar2(n, c, phi1, phi2, sigma)
plot(serie, type="l")
```




```r
# Função para estimar AR(2) por Yule-Walker
estimar_ar2_yw <- function(y) {
  # Autocorrelação
  gamma <- acf(y, plot = FALSE, lag.max = 2)$acf
  gamma0 <- gamma[1]
  gamma1 <- gamma[2]
  gamma2 <- gamma[3]

  # Phi_1 e Phi_2
  phi1_hat <- (gamma1*(1 - gamma2)) / (1 - gamma1^2)
  phi2_hat <- (gamma2 - gamma1^2) / (1 - gamma1^2)

  # EStimativa C e Var do erro
  c_hat <- mean(y) * (1 - phi1_hat - phi2_hat)
  sigma2_hat <- gamma0 * (1 - phi1_hat*gamma1 - phi2_hat*gamma2)

  return(list(
    intercepto = c_hat,
    phi1 = phi1_hat,
    phi2 = phi2_hat,
    sigma = sqrt(sigma2_hat)
  ))
}

# Maxima Verossimilhança
estimar_ar2_mle <- function(y) {
```

```r
  neg_loglik <- function(params) {
    c <- 2
    phi1 <- 0.6
    phi2 <- 0.3
    sigma <- 1.5
    n <- 100
    soma <- 0

    for (t in 3:n) {
      residuo <- y[t] - (c + phi1*y[t-1] + phi2*y[t-2])
      soma <- soma + dnorm(residuo, mean = 0, sd = sigma, log = TRUE)
    }

    return(-soma)
  }

  # #Valores iniciais usando yw
  yw <- estimar_ar2_yw(y)
  iniciais <- c(yw$intercepto, yw$phi1, yw$phi2, log(yw$sigma))

  opt <- optim(iniciais, neg_loglik, method = "L-BFGS-B",
               lower = c(-Inf, -1, -1, -Inf),
               upper = c(Inf, 1, 1, Inf))

  params <- opt$par
  params[4] <- exp(params[4])

  list(
    intercepto = params[1],
    phi1 = params[2],
    phi2 = params[3],
    sigma = params[4],
    log_verossimilhanca = -opt$value
  )
}

set.seed(236237)
n <- 100
c <- 2
phi1 <- 0.6
phi2 <- 0.3
sigma <- 1.5

y <- numeric(n)
y[1:2] <- c / (1 - phi1 - phi2)  # Valores Iniciais
for (t in 3:n) {
  y[t] <- c + phi1*y[t-1] + phi2*y[t-2] + rnorm(1, 0, sigma)
}

#Verossimilhança e YW
yw_est <- estimar_ar2_yw(y)
mle_est <- estimar_ar2_mle(y)
```

```r
resultados <- data.frame(
  Parâmetro = c("c", "phi1", "phi2", "sigma"),
  Verdadeiro = c(c, phi1, phi2, sigma),
  Yule_Walker = c(yw_est$intercepto, yw_est$phi1, yw_est$phi2, yw_est$sigma),
  MLE = c(mle_est$intercepto, mle_est$phi1, mle_est$phi2, mle_est$sigma)
)

# Mostrar resultados
print(resultados)
```

```
##   Parâmetro Verdadeiro Yule_Walker       MLE
## 1         c        2.0   6.7294264 6.7294264
## 2      phi1        0.6   0.5073836 0.5073836
## 3      phi2        0.3   0.1805765 0.1805765
## 4     sigma        1.5   0.7723280 0.7723280
```

```r
#os parametros divergem do verdadeiro
```