# Trabalho 2

Caio Gomes Alves

09/10/2023

## Sumário



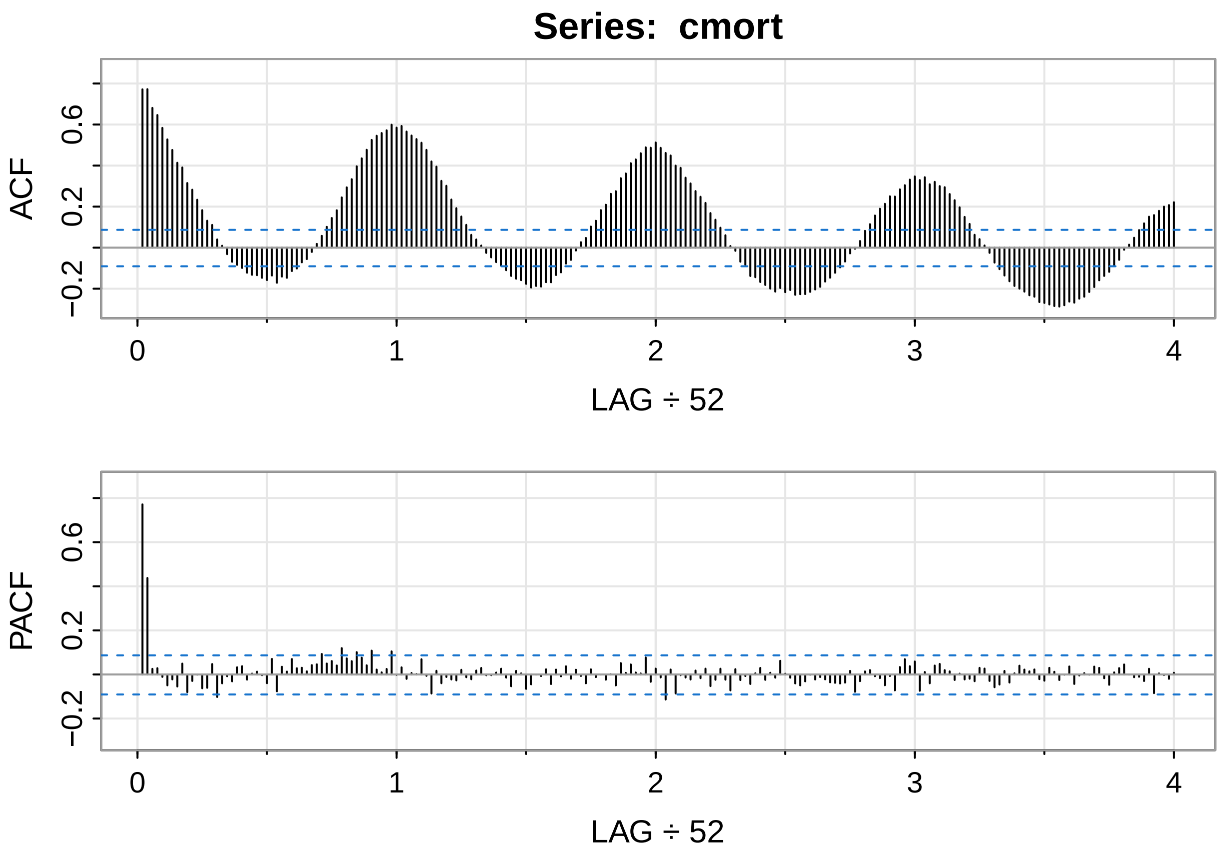
```
# Pacotes utilizados
library(astsa)
```

## Questão 10

### a)

Para verificar a ordem do modelo $AR$ a ser ajustado, verifiquemos inicialmente o **ACF** e **PACF** amostrais:

```r
acf2(cmort, plot = TRUE)
```




Podemos verificar que o **PACF** tem valores significativos para $h = 1, 2$, portanto será ajustado um modelo $AR(2)$

```r
(regr <- ar.ols(cmort, order=2, demean=FALSE, intercept=TRUE))
```

```
##
## Call:
## ar.ols(x = cmort, order.max = 2, demean = FALSE, intercept = TRUE)
##
## Coefficients:
##      1       2
## 0.4286  0.4418
##
## Intercept: 11.45 (2.394)
##
## Order selected 2  sigma^2 estimated as  32.32
```

Com os resultados obtidos, temos que o modelo $AR(2)$ é dado por:

$$X_t = 11,45 + 0,4286X_{t-1} + 0,4418X_{t-2} + W_t, \quad (1)$$

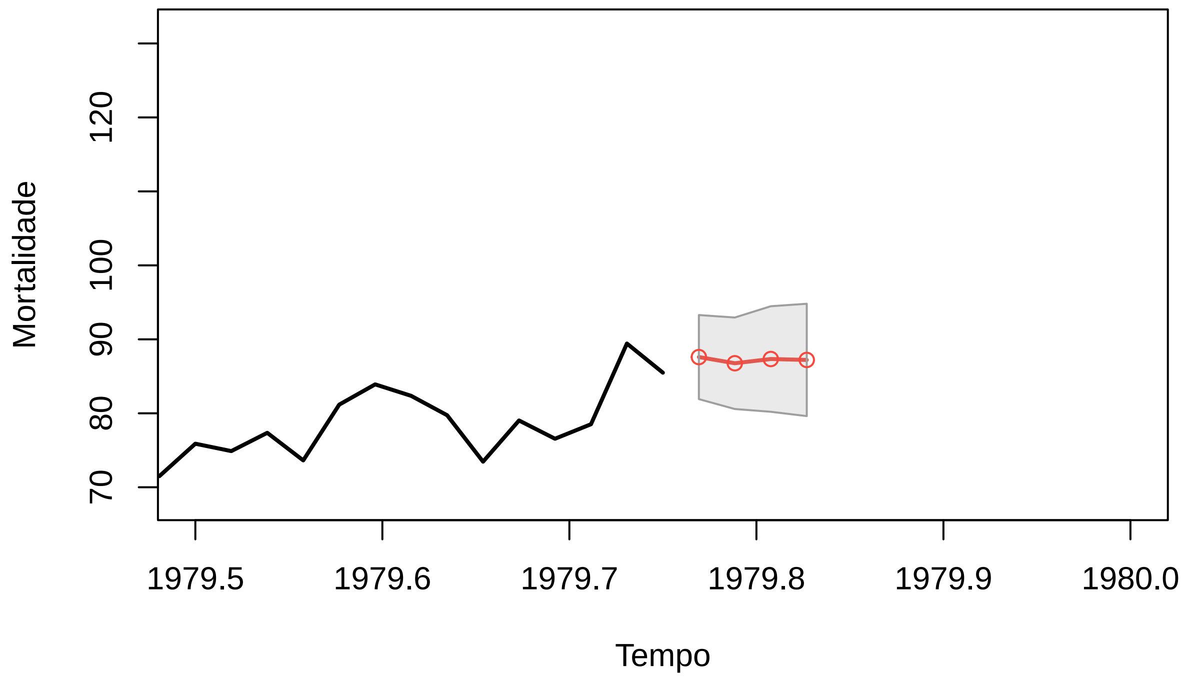
Onde $\hat{\sigma}^2_W = 32,32$.

## b)

```r
# Predições:
(fore <- predict(regr, n.ahead=4))
```

```
## $pred
## Time Series:
## Start = c(1979, 41)
## End = c(1979, 44)
## Frequency = 52
## [1] 87.59986 86.76349 87.33714 87.21350
##
## $se
## Time Series:
## Start = c(1979, 41)
## End = c(1979, 44)
## Frequency = 52
## [1] 5.684848 6.184973 7.134227 7.593357
```

```r
# Gráfico:
ts.plot(cmort, fore$pred, col=1:2, xlim=c(1979.5,1980),
        lwd=2, ylab="Mortalidade", xlab="Tempo")

# Intervalo superior:
U <- fore$pred+fore$se

# Intervalo inferior:
L <- fore$pred-fore$se

# Polígono dos intervalos de previsão:
xx <- c(time(U), rev(time(U))); yy = c(L, rev(U))
polygon(xx, yy, border = 8, col = gray(.6, alpha = .2))
lines(fore$pred, type="p", col=2)
```




## Questão 18

### a)

Como o modelo $AR(2)$ já foi ajustado no exercício anterior, vamos ajustar o modelo usando Yule-Walker:

```
# Ajuste do modelo:
(regr.yw <- ar.yw(cmort, order = 2))
```

```
##
## Call:
## ar.yw.default(x = cmort, order.max = 2)
##
## Coefficients:
##      1       2
## 0.4339  0.4376
##
## Order selected 2  sigma^2 estimated as  32.84
```

```
# Comparação dos coeficientes:
regr$ar      # Regressão linear
```

```
## , , 1
```

```
##
##            [,1]
## [1,] 0.4285906
## [2,] 0.4417874
```

```
regr.yw$ar   # Yule-Walker
```

```
## [1] 0.4339481 0.4375768
```

## b)

```
# Comparação dos erros dos coeficientes:
regr$asy.se.coef
```

```
## $x.mean
## [1] 2.393673
##
## $ar
## [1] 0.03979433 0.03976163
```

```
sqrt(diag(regr.yw$asy.var.coef))
```

```
## [1] 0.04001303 0.04001303
```

```
sqrt(length(cmort))
```

```
## [1] 22.53886
```

Temos pelo Teorema III.10 que:

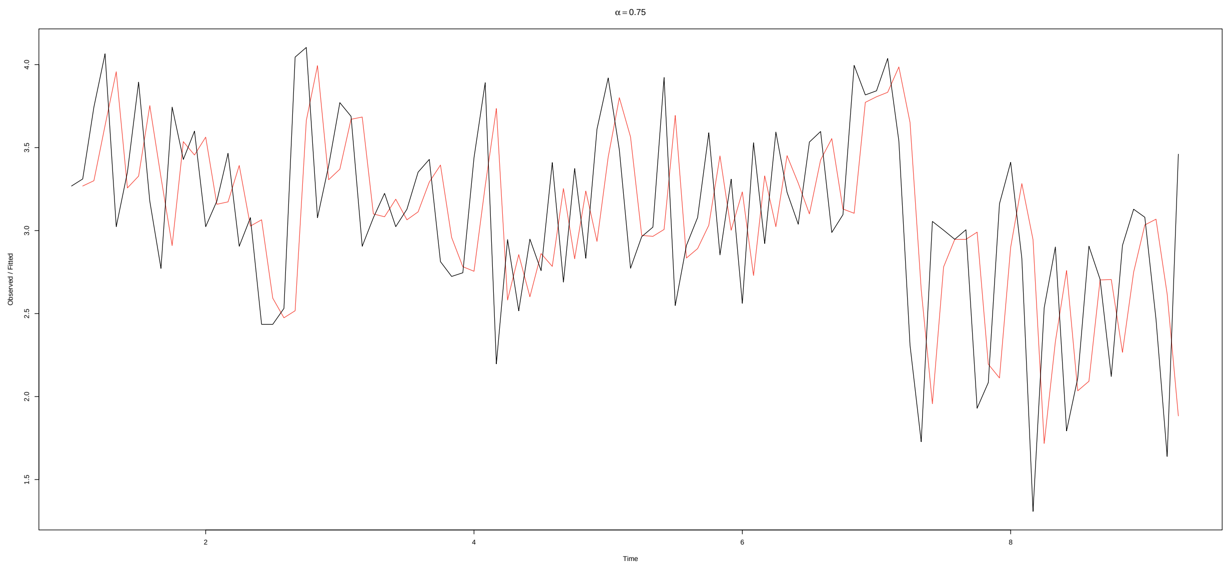
$$\begin{pmatrix} \widehat{\phi}_1 \\ \widehat{\phi}_2 \end{pmatrix} \sim N\left( \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} \frac{1}{n} \begin{pmatrix} 1-\phi_2^2 & -\phi_1(1+\phi_2) \\ -\phi_1(1+\phi_2) & 1-\phi_2^2 \end{pmatrix} \right) \tag{2}$$

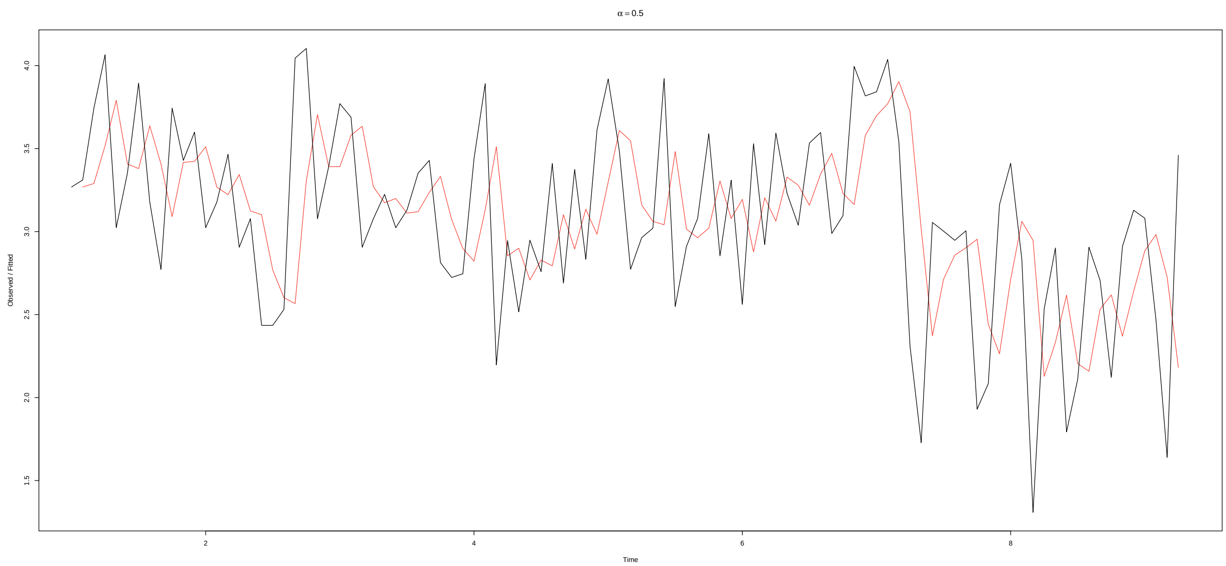
Com isso, temos que os valores dos erros dos coeficientes do modelo linear tenderão para os valores dos erros dos coeficientes estimados por Yule-Walker.

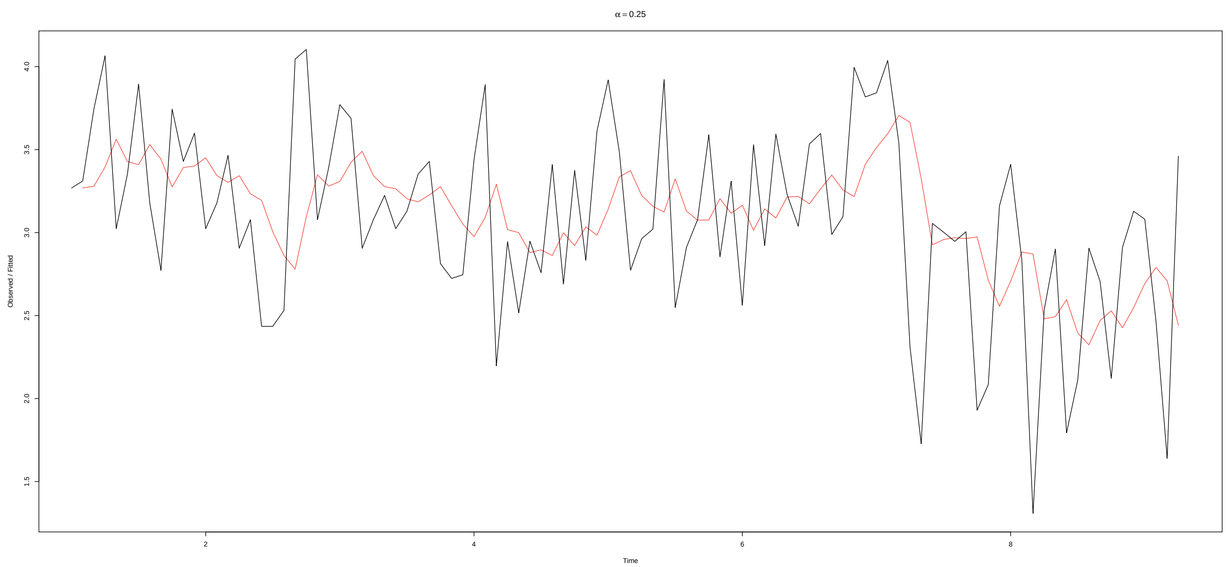
# Questão 30

```
dados <- ts(log(varve)[1:100], frequency = 12)

# Como alpha = 1-lambda,  temos que:
ajuste_1 <- HoltWinters(dados, alpha = 0.75, beta = F, gamma = F)
ajuste_2 <- HoltWinters(dados, alpha = 0.5, beta = F, gamma = F)
ajuste_3 <- HoltWinters(dados, alpha = 0.25, beta = F, gamma = F)
```

```r
# Gráficos dos ajustes:
par(mfrow = c(3, 1))
plot(ajuste_1, main = expression(alpha == 0.75))
plot(ajuste_2, main = expression(alpha == 0.5))
plot(ajuste_3, main = expression(alpha == 0.25))
```


α = 0.75
Observed / Fitted
Time
1.5
2.0
2.5
3.0
3.5
4.0
2
4
6
8


α = 0.5
Observed / Fitted
Time
1.5
2.0
2.5
3.0
3.5
4.0
2
4
6
8

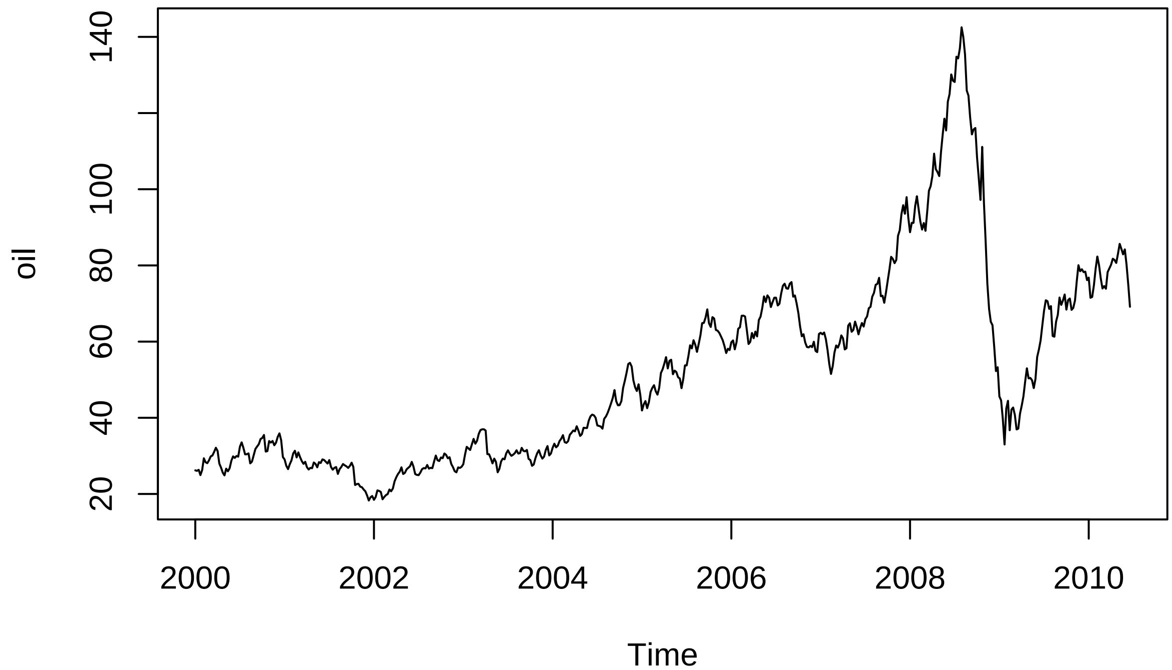

α = 0.25
Observed / Fitted
Time
1.5
2.0
2.5
3.0
3.5
4.0
2
4
6
8

Podemos verificar que conforme aumentamos $\lambda$ (e consequentemente diminuímos $\alpha$) o ajuste de médias móveis exponencialnmente ponderadas (EWMA) se torna mais suave, visto que considera pesos maiores para as observações anteriores.

# Questão 32

Inicialmente iremos plotar o gráfico de `oil` para verificar o padrão da série:

```r
plot(oil)
```

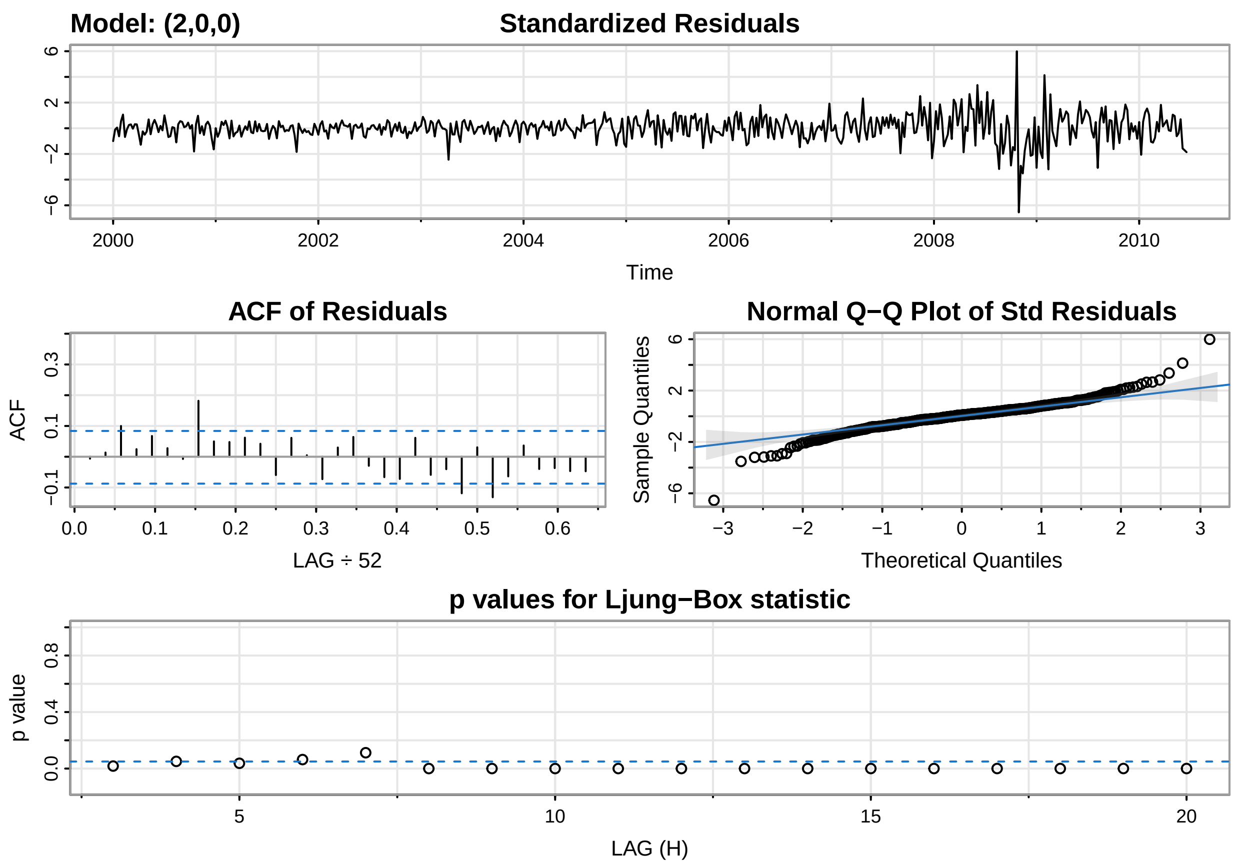



Como a série não parece ser estacionária, incluiremos o parâmetro `no.constant = F` no modelo a ser ajustado:

```r
# Modelo AR(2)
sarima(oil, 2, 0, 0, no.constant = F)
```

```
## initial  value 3.253465
## iter   2 value 3.251004
## iter   3 value 1.126206
## iter   4 value 1.111187
## iter   5 value 0.978359
## iter   6 value 0.936560
## iter   7 value 0.934941
## iter   8 value 0.934932
```

```
## iter   8 value 0.934932
## final  value 0.934932
## converged
## initial  value 0.938314
## iter   2 value 0.938313
## iter   3 value 0.938313
## iter   4 value 0.938313
## iter   5 value 0.938312
## iter   6 value 0.938312
## iter   7 value 0.938311
## iter   8 value 0.938311
## iter   9 value 0.938311
## iter  10 value 0.938310
## iter  11 value 0.938310
## iter  12 value 0.938309
## iter  13 value 0.938309
## iter  14 value 0.938308
## iter  15 value 0.938308
## iter  16 value 0.938308
## iter  17 value 0.938308
## iter  18 value 0.938308
## iter  19 value 0.938307
## iter  20 value 0.938307
## iter  21 value 0.938307
## iter  22 value 0.938307
## iter  23 value 0.938307
## iter  24 value 0.938306
## iter  25 value 0.938306
## iter  26 value 0.938306
## iter  27 value 0.938305
## iter  28 value 0.938305
## iter  29 value 0.938305
## iter  30 value 0.938305
## iter  31 value 0.938305
## iter  32 value 0.938305
## iter  33 value 0.938304
## iter  34 value 0.938304
## iter  35 value 0.938304
## iter  36 value 0.938304
## iter  37 value 0.938304
## iter  38 value 0.938304
## iter  39 value 0.938304
## iter  40 value 0.938304
## iter  41 value 0.938303
## iter  42 value 0.938303
## iter  43 value 0.938303
## iter  43 value 0.938303
## final  value 0.938303
## converged
```

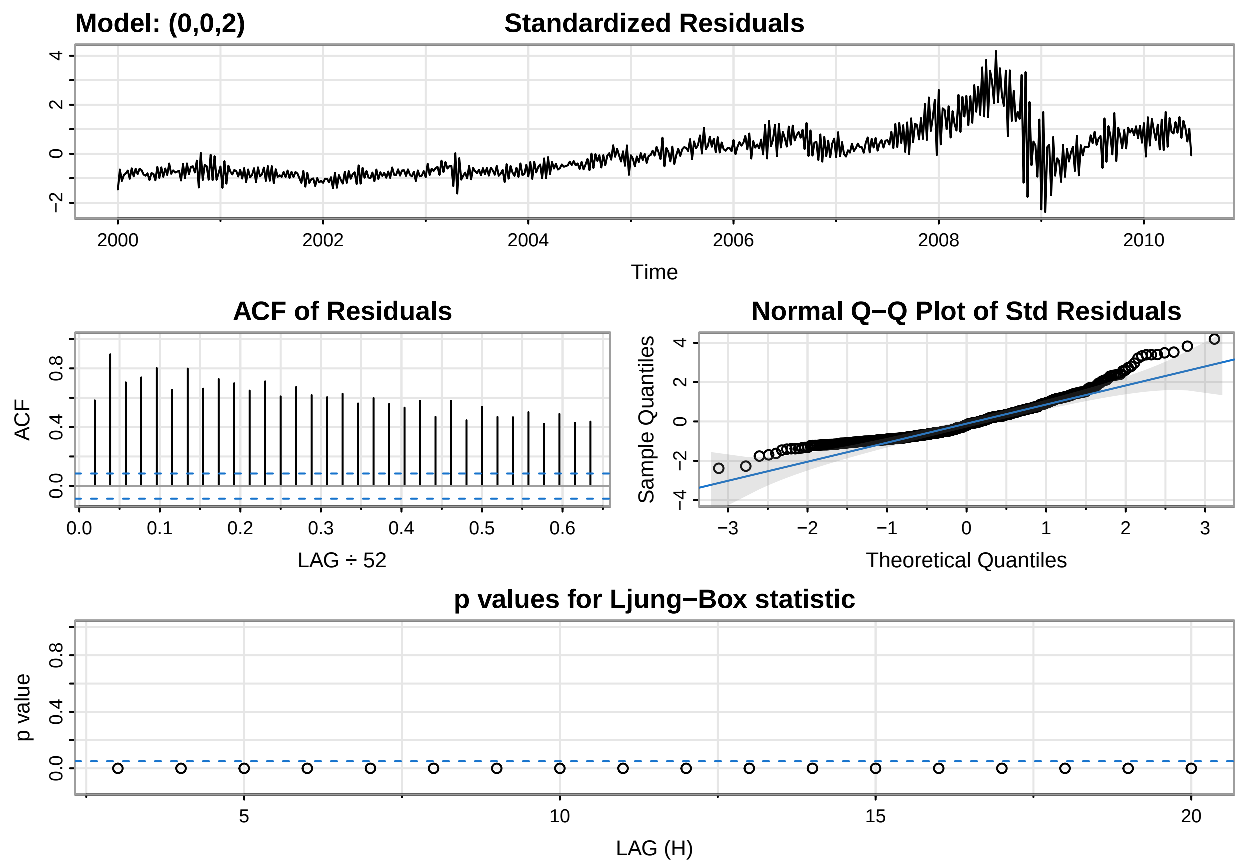



```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period = S),
##     xreg = xmean, include.mean = FALSE, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ar2    xmean
##       1.1747  -0.1810  51.2198
## s.e.  0.0423   0.0423  14.1440
##
## sigma^2 estimated as 6.476:  log likelihood = -1284.7,  aic = 2577.39
##
## $degrees_of_freedom
## [1] 542
##
## $ttable
##       Estimate      SE t.value p.value
## ar1     1.1747  0.0423 27.7936   0e+00
## ar2    -0.1810  0.0423 -4.2774   0e+00
## xmean  51.2198 14.1440  3.6213   3e-04
##
## $AIC
## [1] 4.729163
##
```

```
## $AICc
## [1] 4.729244
##
## $BIC
## [1] 4.760728
```

```r
# Modelo MA(1)
sarima(oil, 0, 0, 2, no.constant = F)
```

```
## initial  value 3.253470
## iter   2 value 2.602713
## iter   3 value 2.280393
## iter   4 value 2.196753
## iter   5 value 2.188700
## iter   6 value 2.187603
## iter   7 value 2.172641
## iter   8 value 2.171780
## iter   9 value 2.169759
## iter  10 value 2.168899
## iter  11 value 2.167935
## iter  12 value 2.167491
## iter  13 value 2.166916
## iter  14 value 2.166880
## iter  15 value 2.166877
## iter  15 value 2.166877
## final  value 2.166877
## converged
## initial  value 2.149597
## iter   2 value 2.148823
## iter   3 value 2.147301
## iter   4 value 2.146676
## iter   5 value 2.146355
## iter   6 value 2.145974
## iter   7 value 2.145854
## iter   8 value 2.145849
## iter   9 value 2.145849
## iter   9 value 2.145849
## iter   9 value 2.145849
## final  value 2.145849
## converged
```




```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period = S),
##     xreg = xmean, include.mean = FALSE, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ma1     ma2    xmean
##       1.6326  0.8117  52.0259
## s.e.  0.0274  0.0212   1.2547
##
## sigma^2 estimated as 72.58:  log likelihood = -1942.81,  aic = 3893.62
##
## $degrees_of_freedom
## [1] 542
##
## $ttable
##       Estimate     SE t.value p.value
## ma1     1.6326 0.0274 59.6825       0
## ma2     0.8117 0.0212 38.3691       0
## xmean  52.0259 1.2547 41.4638       0
##
## $AIC
## [1] 7.144253
##
```

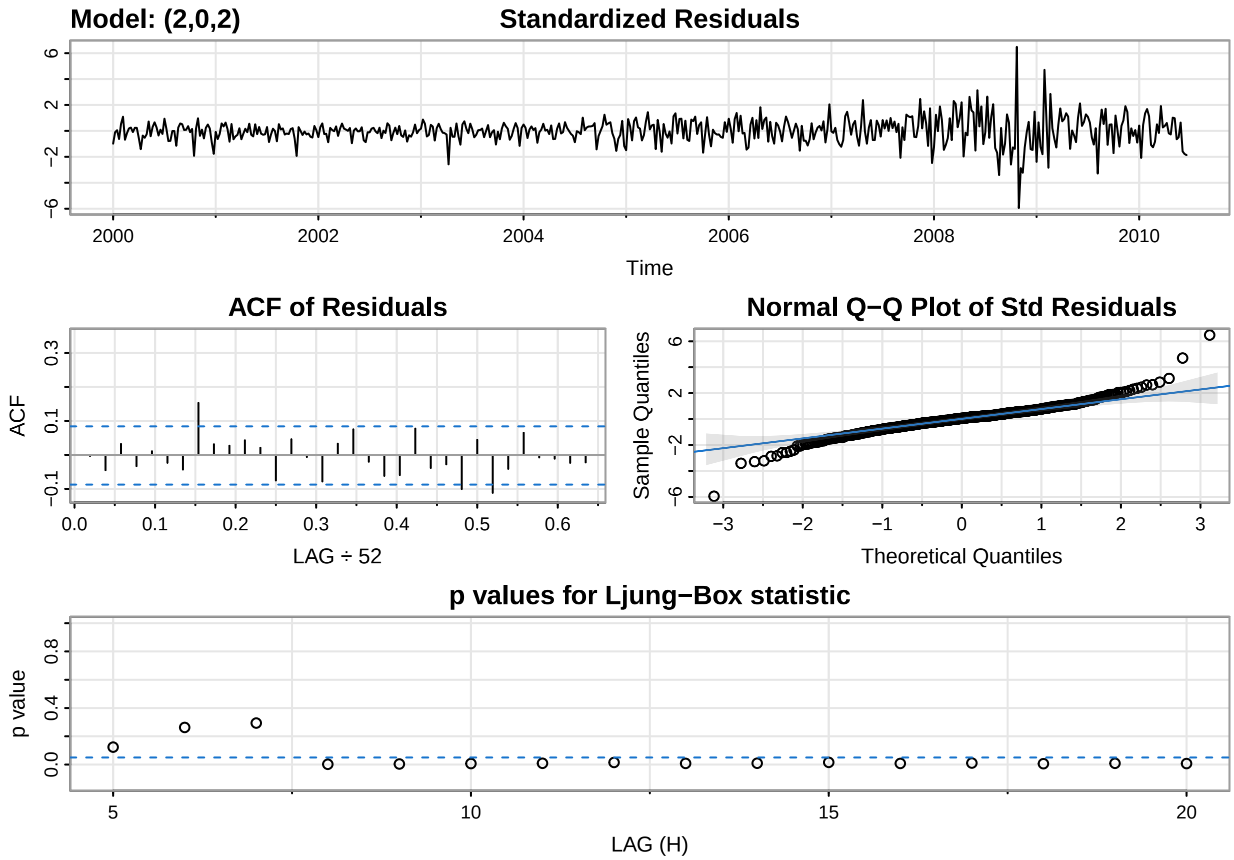
```
## $AICc
## [1] 7.144335
##
## $BIC
## [1] 7.175819
```

```r
# Modelo ARIMA(2, 0, 2)
sarima(oil, 2, 0, 2, no.constant = F)
```

```
## initial  value 3.253465
## iter   2 value 3.052391
## iter   3 value 2.924348
## iter   4 value 1.336416
## iter   5 value 1.263589
## iter   6 value 1.257847
## iter   7 value 1.175421
## iter   8 value 1.161907
## iter   9 value 1.144283
## iter  10 value 1.122834
## iter  11 value 1.073550
## iter  12 value 0.983322
## iter  13 value 0.979856
## iter  14 value 0.958447
## iter  15 value 0.938048
## iter  16 value 0.935086
## iter  17 value 0.934631
## iter  18 value 0.934576
## iter  19 value 0.934562
## iter  20 value 0.934304
## iter  21 value 0.933664
## iter  22 value 0.932791
## iter  23 value 0.932545
## iter  24 value 0.932501
## iter  25 value 0.929550
## iter  26 value 0.929025
## iter  27 value 0.928686
## iter  28 value 0.928085
## iter  29 value 0.927272
## iter  30 value 0.926342
## iter  31 value 0.926020
## iter  32 value 0.925681
## iter  33 value 0.925536
## iter  34 value 0.925159
## iter  35 value 0.925144
## iter  36 value 0.924030
## iter  37 value 0.923894
## iter  38 value 0.923751
## iter  39 value 0.923741
## iter  40 value 0.923736
## iter  41 value 0.923735
## iter  42 value 0.923735
## iter  43 value 0.923735
## iter  44 value 0.923733
## iter  45 value 0.923730
```

```
## iter  45 value 0.923730
## final  value 0.923730
## converged
## initial  value 0.927329
## iter   2 value 0.927329
## iter   3 value 0.927327
## iter   4 value 0.927319
## iter   5 value 0.927314
## iter   6 value 0.927306
## iter   7 value 0.927301
## iter   8 value 0.927299
## iter   9 value 0.927295
## iter  10 value 0.927288
## iter  11 value 0.927276
## iter  12 value 0.927263
## iter  13 value 0.927258
## iter  14 value 0.927257
## iter  15 value 0.927255
## iter  16 value 0.927254
## iter  17 value 0.927243
## iter  18 value 0.927240
## iter  19 value 0.927240
## iter  20 value 0.927239
## iter  21 value 0.927238
## iter  22 value 0.927237
## iter  23 value 0.927236
## iter  24 value 0.927236
## iter  25 value 0.927235
## iter  26 value 0.927235
## iter  27 value 0.927235
## iter  28 value 0.927234
## iter  29 value 0.927232
## iter  30 value 0.927231
## iter  31 value 0.927231
## iter  32 value 0.927231
## iter  33 value 0.927230
## iter  34 value 0.927229
## iter  35 value 0.927229
## iter  36 value 0.927229
## iter  37 value 0.927228
## iter  38 value 0.927228
## iter  39 value 0.927227
## iter  40 value 0.927226
## iter  41 value 0.927226
## iter  42 value 0.927226
## iter  43 value 0.927226
## iter  44 value 0.927225
## iter  45 value 0.927225
## iter  46 value 0.927225
## iter  47 value 0.927225
## iter  48 value 0.927224
## iter  49 value 0.927224
## iter  50 value 0.927224
## iter  51 value 0.927224
```

```
## iter  52 value 0.927223
## iter  53 value 0.927223
## iter  54 value 0.927223
## iter  55 value 0.927223
## iter  56 value 0.927223
## iter  57 value 0.927223
## iter  57 value 0.927223
## iter  57 value 0.927223
## final  value 0.927223
## converged
```



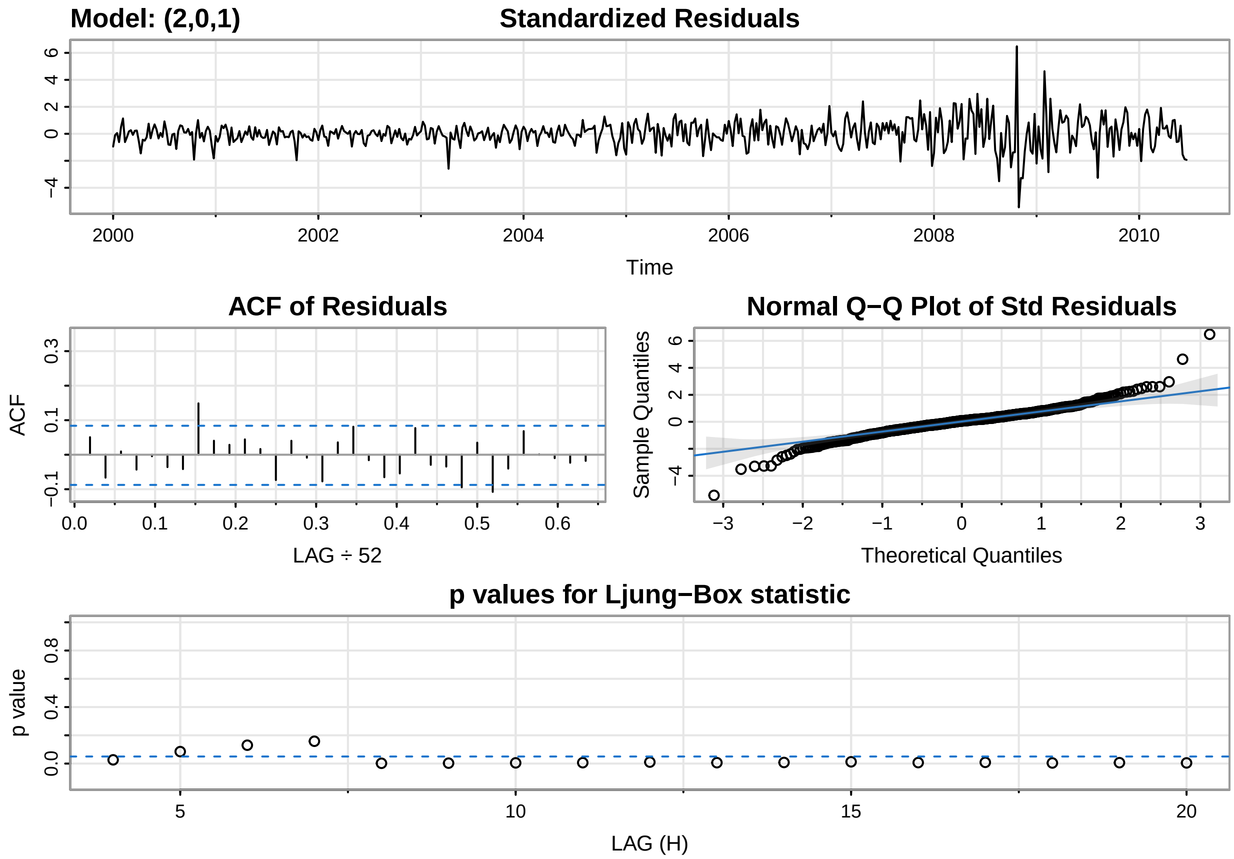

```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period = S),
##     xreg = xmean, include.mean = FALSE, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ar2      ma1      ma2    xmean
##       1.9121  -0.9138  -0.7615  -0.0719  50.9410
## s.e.  0.0380   0.0376   0.0588   0.0464   9.8262
##
## sigma^2 estimated as 6.333:  log likelihood = -1278.66,  aic = 2569.32
##
```

```
## $degrees_of_freedom
## [1] 540
##
## $ttable
##       Estimate     SE  t.value p.value
## ar1     1.9121 0.0380  50.3742   0.000
## ar2    -0.9138 0.0376 -24.2773   0.000
## ma1    -0.7615 0.0588 -12.9454   0.000
## ma2    -0.0719 0.0464  -1.5488   0.122
## xmean  50.9410 9.8262   5.1842   0.000
##
## $AIC
## [1] 4.714341
##
## $AICc
## [1] 4.714545
##
## $BIC
## [1] 4.761689
```

```r
# Como no modelo anterior o termo de MA(2) não foi
# significativo,  será ajustado um modelo ARIMA(2, 0, 1)
sarima(oil, 2, 0, 1, no.constant = F)
```

```
## initial  value 3.253465
## iter   2 value 3.060470
## iter   3 value 2.759178
## iter   4 value 1.851986
## iter   5 value 1.441980
## iter   6 value 1.326461
## iter   7 value 0.971002
## iter   8 value 0.951581
## iter   9 value 0.941932
## iter  10 value 0.941798
## iter  11 value 0.941710
## iter  12 value 0.941606
## iter  13 value 0.941236
## iter  14 value 0.941226
## iter  15 value 0.941194
## iter  16 value 0.940712
## iter  17 value 0.939637
## iter  18 value 0.937997
## iter  19 value 0.936893
## iter  20 value 0.936256
## iter  21 value 0.935610
## iter  22 value 0.934912
## iter  23 value 0.934888
## iter  24 value 0.933578
## iter  25 value 0.933173
## iter  26 value 0.933149
## iter  27 value 0.932485
## iter  28 value 0.931376
## iter  29 value 0.930301
## iter  30 value 0.929645
```

```
## iter  31 value 0.928478
## iter  32 value 0.928461
## iter  33 value 0.927284
## iter  34 value 0.927008
## iter  35 value 0.926738
## iter  36 value 0.926428
## iter  37 value 0.926161
## iter  38 value 0.926115
## iter  39 value 0.925978
## iter  40 value 0.925968
## iter  41 value 0.925968
## iter  42 value 0.925967
## iter  43 value 0.925967
## iter  44 value 0.925967
## iter  45 value 0.925967
## iter  45 value 0.925967
## iter  45 value 0.925967
## final  value 0.925967
## converged
## initial  value 0.929627
## iter   2 value 0.929626
## iter   3 value 0.929622
## iter   4 value 0.929602
## iter   5 value 0.929600
## iter   6 value 0.929590
## iter   7 value 0.929574
## iter   8 value 0.929539
## iter   9 value 0.929500
## iter  10 value 0.929478
## iter  11 value 0.929478
## iter  12 value 0.929477
## iter  13 value 0.929473
## iter  14 value 0.929471
## iter  15 value 0.929470
## iter  16 value 0.929470
## iter  17 value 0.929469
## iter  18 value 0.929469
## iter  19 value 0.929469
## iter  20 value 0.929469
## iter  21 value 0.929469
## iter  22 value 0.929469
## iter  23 value 0.929468
## iter  24 value 0.929468
## iter  25 value 0.929468
## iter  26 value 0.929468
## iter  27 value 0.929468
## iter  28 value 0.929468
## iter  28 value 0.929468
## final  value 0.929468
## converged
```




```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period = S),
##     xreg = xmean, include.mean = FALSE, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ar2      ma1    xmean
##       1.9000  -0.9019  -0.8081  50.1336
## s.e.  0.0437   0.0434   0.0592   9.8627
##
## sigma^2 estimated as 6.361:  log likelihood = -1279.88,  aic = 2569.76
##
## $degrees_of_freedom
## [1] 541
##
## $ttable
##       Estimate     SE  t.value p.value
## ar1     1.9000 0.0437  43.4772       0
## ar2    -0.9019 0.0434 -20.7989       0
## ma1    -0.8081 0.0592 -13.6604       0
## xmean  50.1336 9.8627   5.0831       0
##
## $AIC
## [1] 4.715162
```

```
##
## $AICc
## [1] 4.715298
##
## $BIC
## [1] 4.754619
```

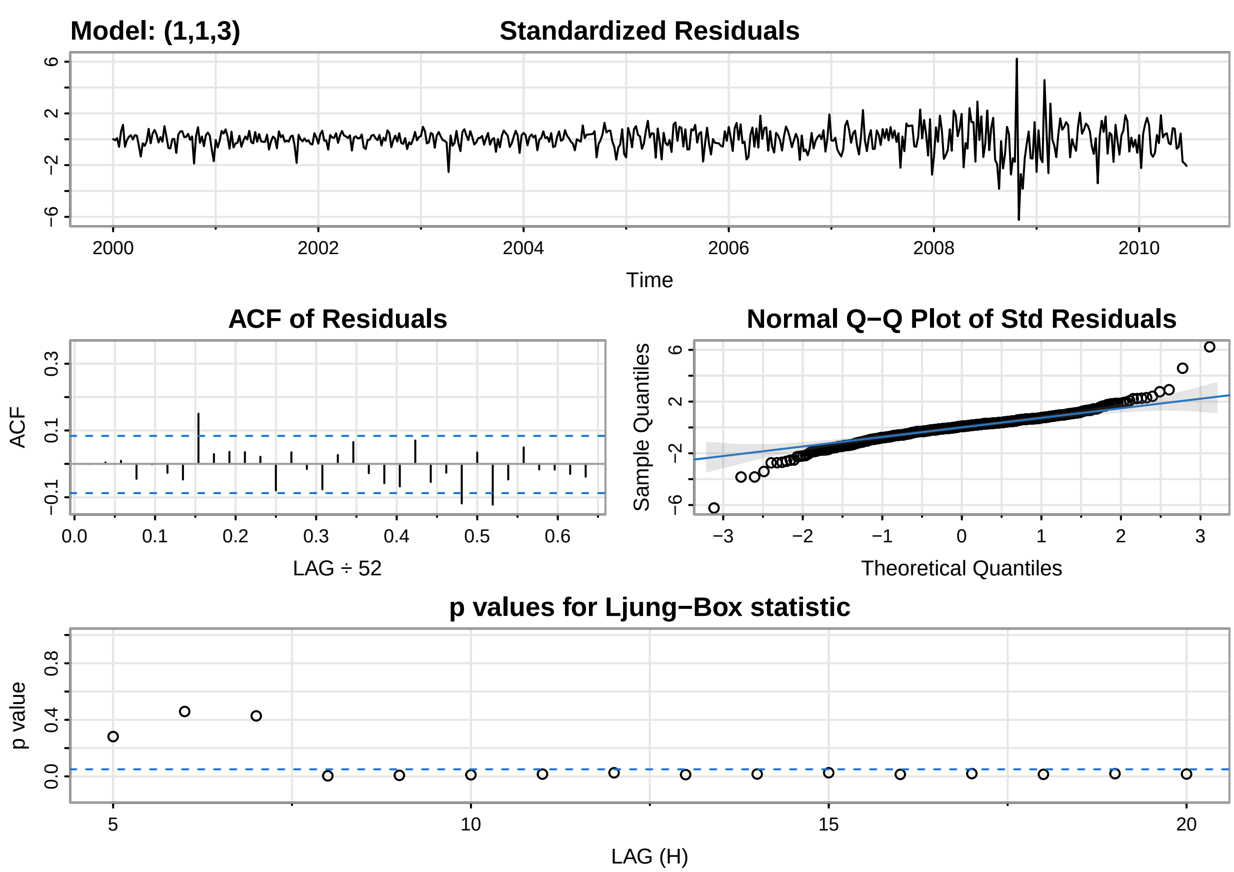
```r
# Usando auto.arima
auto.arima(oil)
```

```
## Series: oil
## ARIMA(1,1,3)(0,0,1)[52]
##
## Coefficients:
##          ar1     ma1      ma2    ma3     sma1
##       0.8793  -0.725  -0.1178  0.066  -0.0738
## s.e.  0.0539   0.069   0.0552  0.044   0.0436
##
## sigma^2 = 6.396:  log likelihood = -1274.33
## AIC=2560.66   AICc=2560.82   BIC=2586.46
```

```r
# Chegamos que o melhor modelo é o ARIMA(1, 1, 3)
sarima(oil, 1, 1, 3, no.constant = F)
```

```
## initial  value 0.952571
## iter   2 value 0.949153
## iter   3 value 0.932936
## iter   4 value 0.932471
## iter   5 value 0.932449
## iter   6 value 0.932439
## iter   7 value 0.932361
## iter   8 value 0.932231
## iter   9 value 0.931975
## iter  10 value 0.931607
## iter  11 value 0.930620
## iter  12 value 0.930517
## iter  13 value 0.930398
## iter  14 value 0.930136
## iter  15 value 0.929771
## iter  16 value 0.928704
## iter  17 value 0.928530
## iter  18 value 0.927965
## iter  19 value 0.927536
## iter  20 value 0.927402
## iter  21 value 0.927263
## iter  22 value 0.927054
## iter  23 value 0.927037
## iter  24 value 0.927007
## iter  25 value 0.926965
## iter  26 value 0.926964
## iter  27 value 0.926962
## iter  28 value 0.926961
## iter  29 value 0.926960
```

```
## iter  30 value 0.926959
## iter  31 value 0.926959
## iter  32 value 0.926959
## iter  33 value 0.926959
## iter  33 value 0.926959
## final  value 0.926959
## converged
## initial  value 0.926109
## iter   2 value 0.926109
## iter   3 value 0.926109
## iter   4 value 0.926109
## iter   5 value 0.926108
## iter   6 value 0.926108
## iter   7 value 0.926108
## iter   8 value 0.926108
## iter   9 value 0.926108
## iter   9 value 0.926108
## iter   9 value 0.926108
## final  value 0.926108
## converged
```

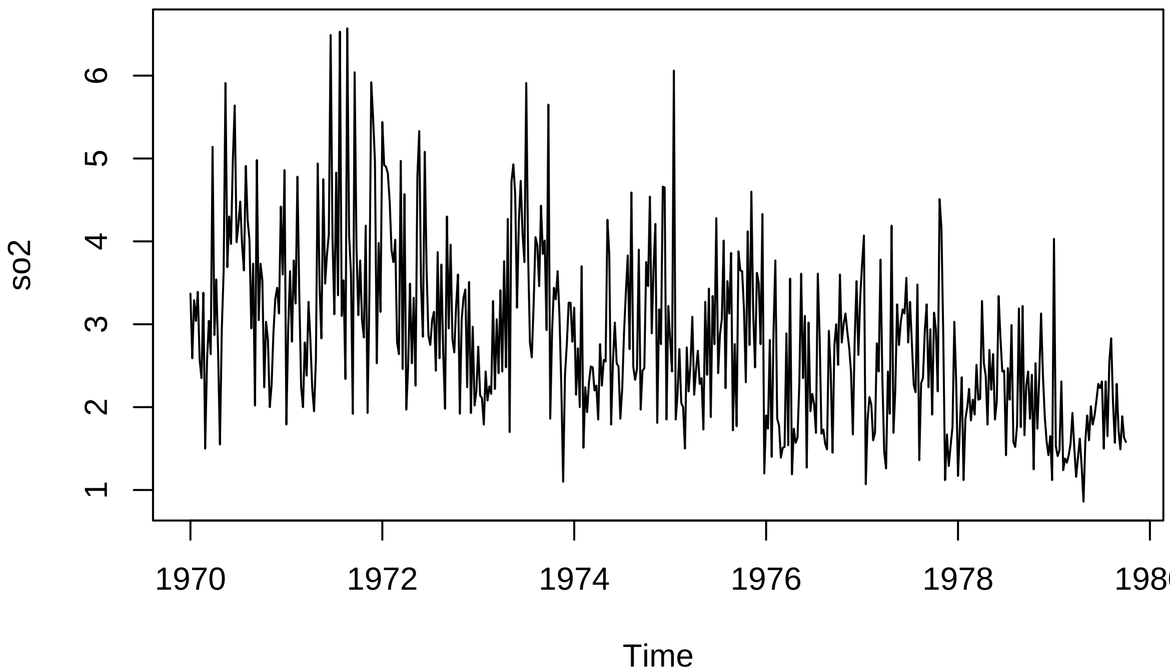



```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period = S),
```

```
##     xreg = constant, transform.pars = trans, fixed = fixed, optim.control = list(trace = trc,
##         REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ma1      ma2     ma3  constant
##       0.8834  -0.7253  -0.1186  0.0609    0.0626
## s.e.  0.0527   0.0681   0.0556  0.0445    0.2001
##
## sigma^2 estimated as 6.373:  log likelihood = -1275.71,  aic = 2563.41
##
## $degrees_of_freedom
## [1] 539
##
## $ttable
##          Estimate     SE  t.value p.value
## ar1        0.8834 0.0527  16.7732  0.0000
## ma1       -0.7253 0.0681 -10.6513  0.0000
## ma2       -0.1186 0.0556  -2.1328  0.0334
## ma3        0.0609 0.0445   1.3689  0.1716
## constant   0.0626 0.2001   0.3126  0.7547
##
## $AIC
## [1] 4.712153
##
## $AICc
## [1] 4.712358
##
## $BIC
## [1] 4.759567
```

# Questão 34

```r
# Visualização da série temporal
plot(so2)
```




```
# Para ajustar o modelo,  usaremos o auto.arima
# para encontrar os coefficientes
auto.arima(so2)
```

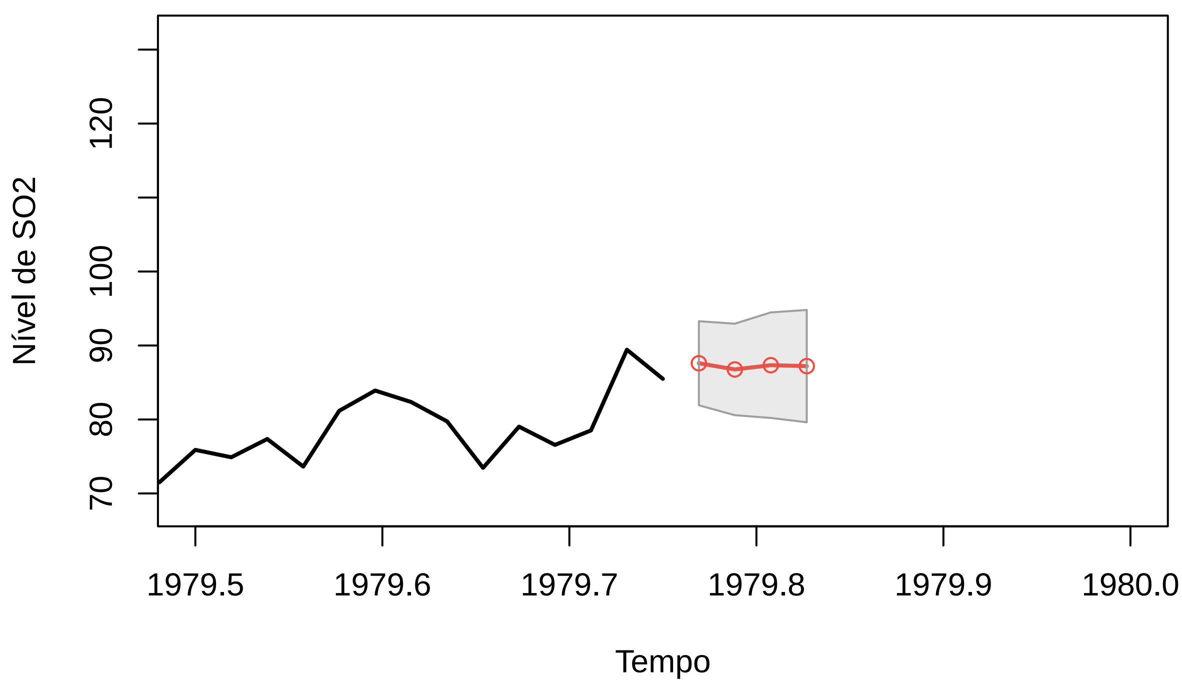
```
## Series: so2
## ARIMA(2,1,3)(2,0,0)[52]
##
## Coefficients:
##          ar1     ar2      ma1      ma2     ma3    sar1    sar2
##       0.1037  0.7169  -0.9952  -0.5167  0.5204  0.0149  0.0213
## s.e.  0.1293  0.1206   0.1450   0.2456  0.1211  0.0463  0.0509
##
## sigma^2 = 0.7707:  log likelihood = -650.83
## AIC=1317.66   AICc=1317.95   BIC=1351.49
```

```
modelo <- arima(so2, order = c(2, 1, 3))

# Previsões para n = 4:
# Predições:
(previsoes <- predict(regr, n.ahead=4))
```

```
## $pred
## Time Series:
## Start = c(1979, 41)
## End = c(1979, 44)
```

```
## Frequency = 52
## [1] 87.59986 86.76349 87.33714 87.21350
##
## $se
## Time Series:
## Start = c(1979, 41)
## End = c(1979, 44)
## Frequency = 52
## [1] 5.684848 6.184973 7.134227 7.593357
```

```r
# Gráfico:
ts.plot(cmort, previsoes$pred, col=1:2, xlim=c(1979.5,1980),
        lwd=2, ylab="Nível de SO2", xlab="Tempo")

# Intervalo superior:
U <- previsoes$pred+previsoes$se

# Intervalo inferior:
L <- previsoes$pred-previsoes$se

# Polígono dos intervalos de previsão:
xx <- c(time(U), rev(time(U))); yy = c(L, rev(U))
polygon(xx, yy, border = 8, col = gray(.6, alpha = .2))
lines(previsoes$pred, type="p", col=2)
```

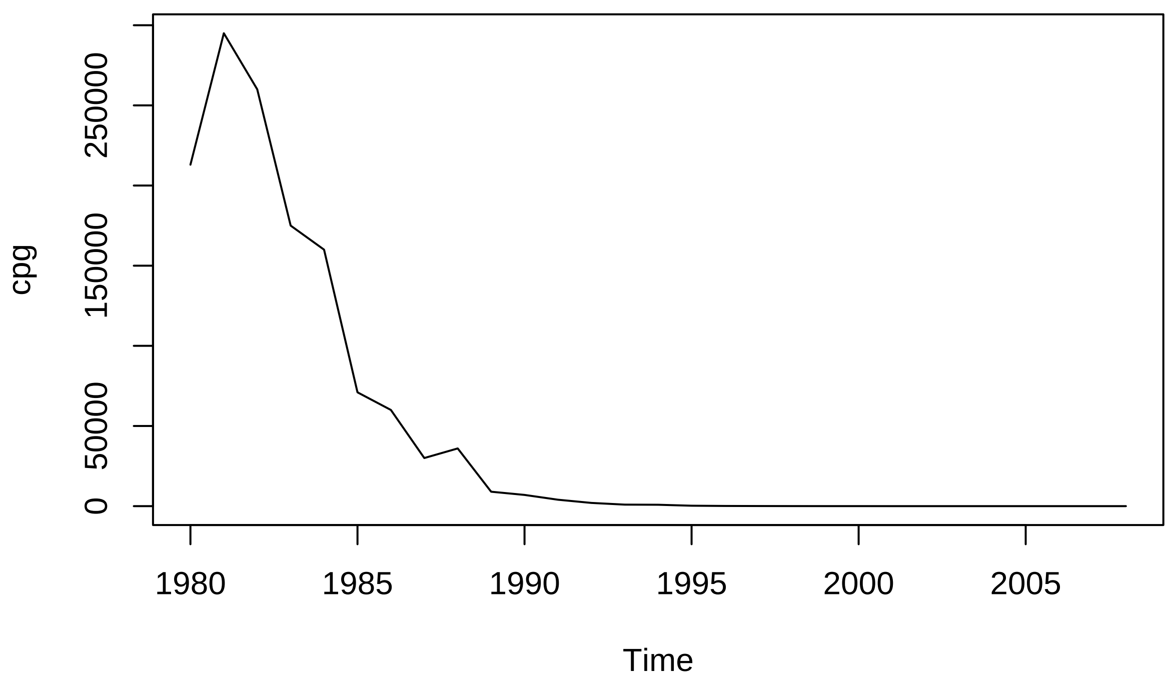



Podemos verificar que as previsões para as quatro próximas semanas é de aumento nos níveis de dióxido de enxofre, além de que pode-se verificar que as bandas de confiança aumentam para as previsões à medida que se aumenta o horizonte, como esperado.

## Questão 36

### a)

```
plot.ts(cpg)
```




Como descrito no enunciado da questão, o que se observa é que os valores começaram a decair de maneira exponencial a partir de 1982, até que atingiu valores próximos de zero em 2008.

### b)

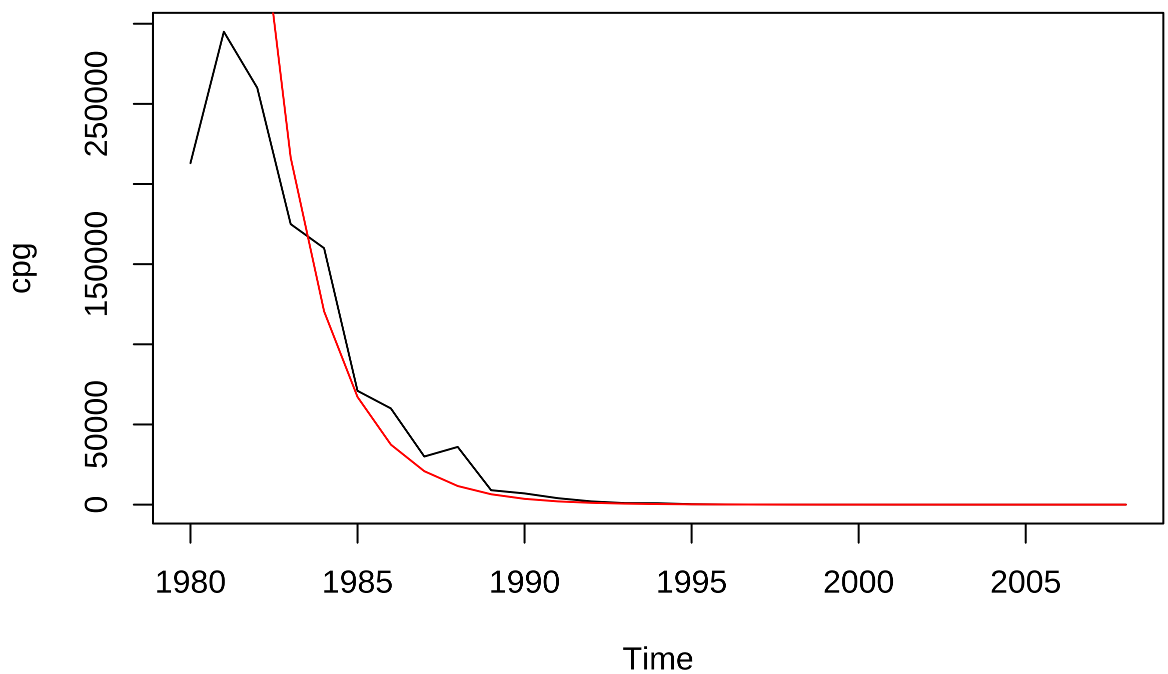
Para modelar $C_t \approx \alpha e^{\beta t}$ usaremos uma regressão linear (`lm(log(C_t) ~ t)`):

```
# Ajuste do modelo
(ajuste <- lm(log(cpg) ~ time(cpg)))
```

```
##
## Call:
## lm(formula = log(cpg) ~ time(cpg))
```

```
##
## Coefficients:
## (Intercept)    time(cpg)
##   1172.4943      -0.5851
```

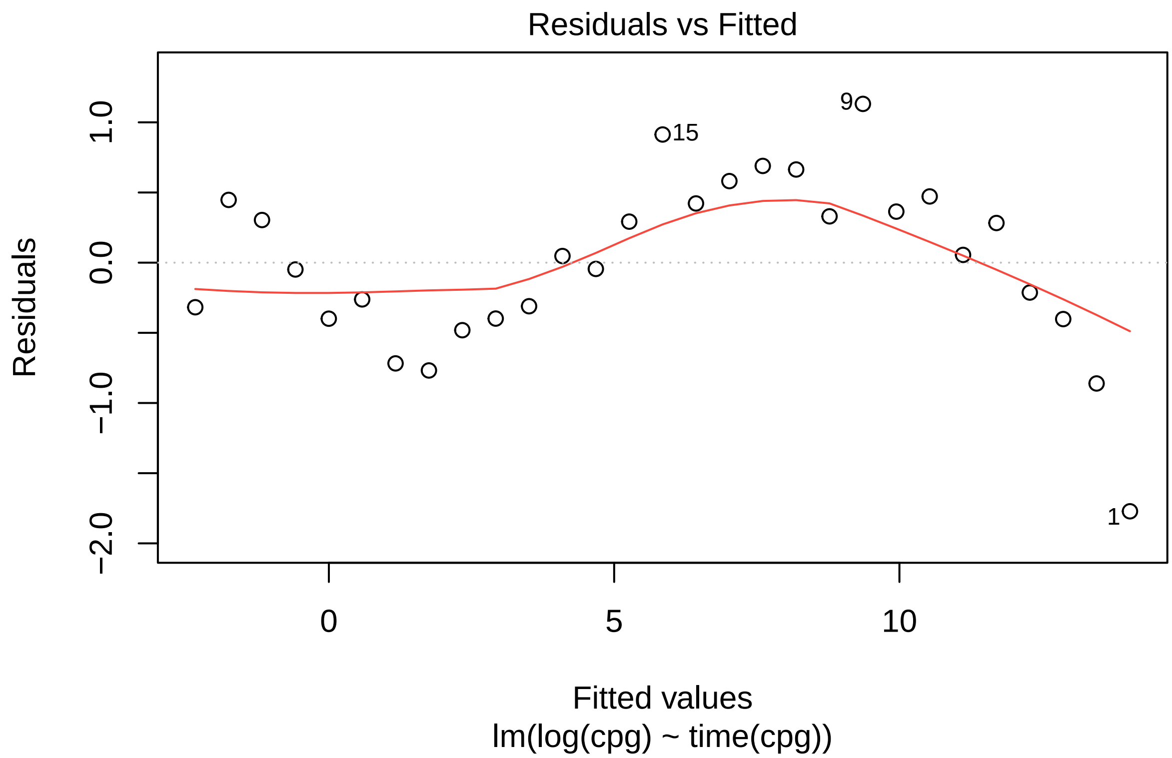
```r
grade <- seq(1980, 2008, by = 1)
pontos <- exp(predict(ajuste, newdata = as.data.frame(grade)))

plot.ts(cpg)
lines(ts(pontos, start = c(1980)), col = "red")
```




### c)

Podemos verificar os resíduos plotando-os contra os valores ajustados:

```r
plot(ajuste, which = 1)
```

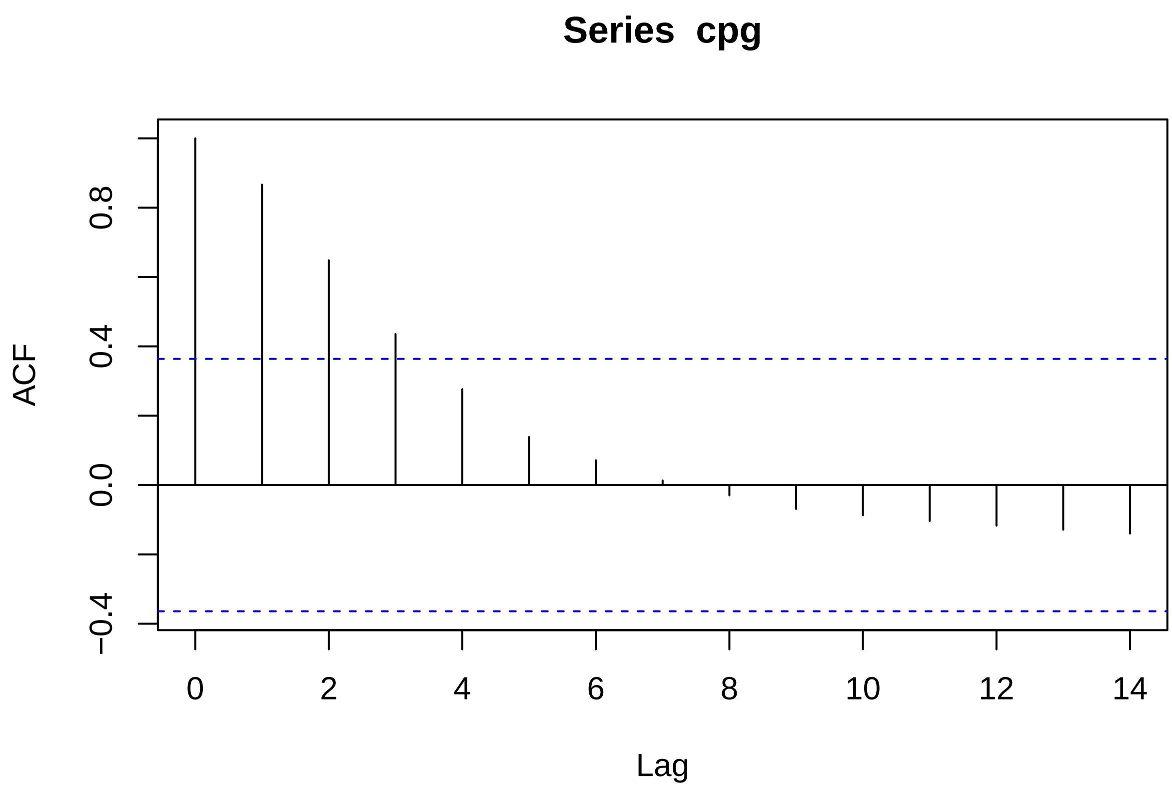



Percebe-se que os resíduos possuem um comportamento não-aleatório que não foi captado pelo modelo linear da forma $C_t \approx \alpha e^{\beta t}$.

## d)

Ajustando os dados novamente utilizando os erros como autocorrelacionados, escolhe-se ajustar com os modelos $AR(p)$. Conferindo o ACF, temos que:

```
acf(cpg)
```

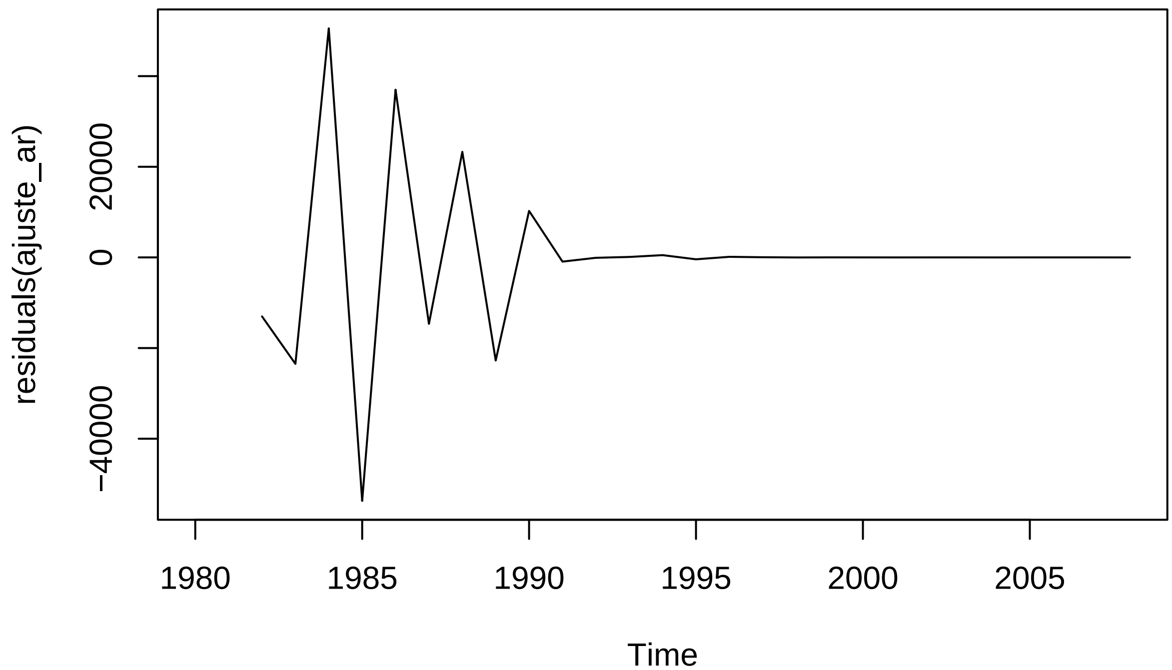



Utilizará-se o modelo $AR(2)$ para modelar os erros autocorrelacionados:

```r
(ajuste_ar <- ar.yw(cpg,order.max = 2,demean = F))
```

```
##
## Call:
## ar.yw.default(x = cpg, order.max = 2, demean = F)
##
## Coefficients:
##       1        2
##  1.2091  -0.3928
##
## Order selected 2  sigma^2 estimated as  2.142e+09
```

```r
# Plotando os resíduos:
plot(residuals(ajuste_ar))
```




Vemos que os resíduos do modelo com os erros autocorrelacionados tem um comportamento que tende a zero, conforme a série progride.