Trabalho5

May 25, 2025

[2]:
```
library(pacman)
p_load(ggplot2,dplyr,lmtest,forecast,dlm)
```

[14]:
```
options(repr.plot.width=20, repr.plot.height=10)  # Ajuste dos gráficos
options(warn=-1)
```

[15]:
```
rm(list=ls())
```

[16]:
```
setwd("C:\\Users\\Marcelo\\OneDrive\\Área de Trabalho\\ts\\trabalho5\\")
getwd()
```

'C:/Users/Marcelo/OneDrive/Área de Trabalho/ts/trabalho5'

0.0.1 1)A)

Série temporal de produção de energia, jan/1985 até jan/2018. Vinda do kaggle.

[17]:
```
data<- read.csv("Electric_Production.csv")
Y<-data$IPG2211A2N
```

0.0.2 1)B)

[18]:
```
H=12
```

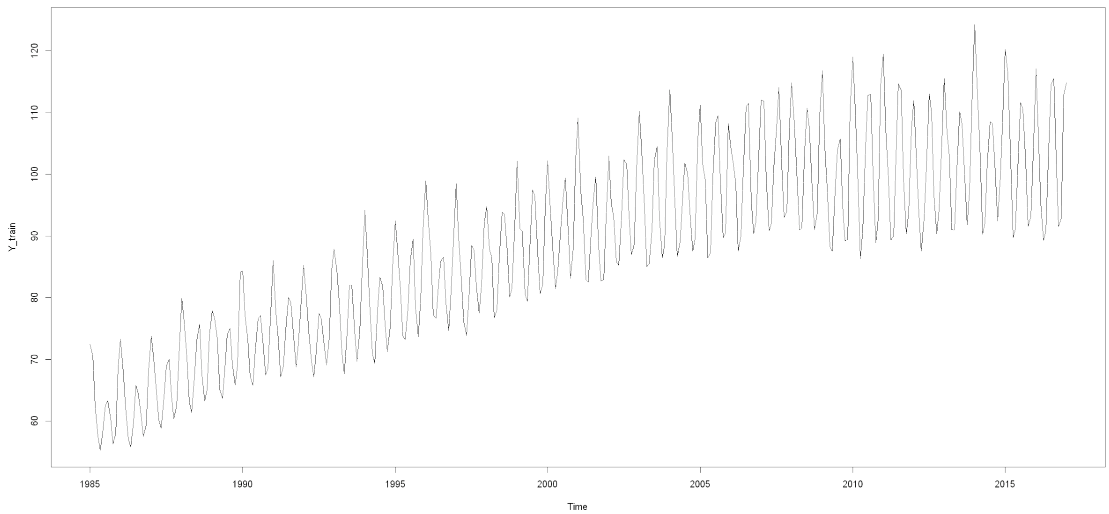
[19]:
```
n<-length(Y)
```

[20]:
```
Y_train<-ts(Y[1:(n-H)],start=c(1985,1),frequency=12)
```

[21]:
```
Y_test <- Y[(n-H+1):n]
```

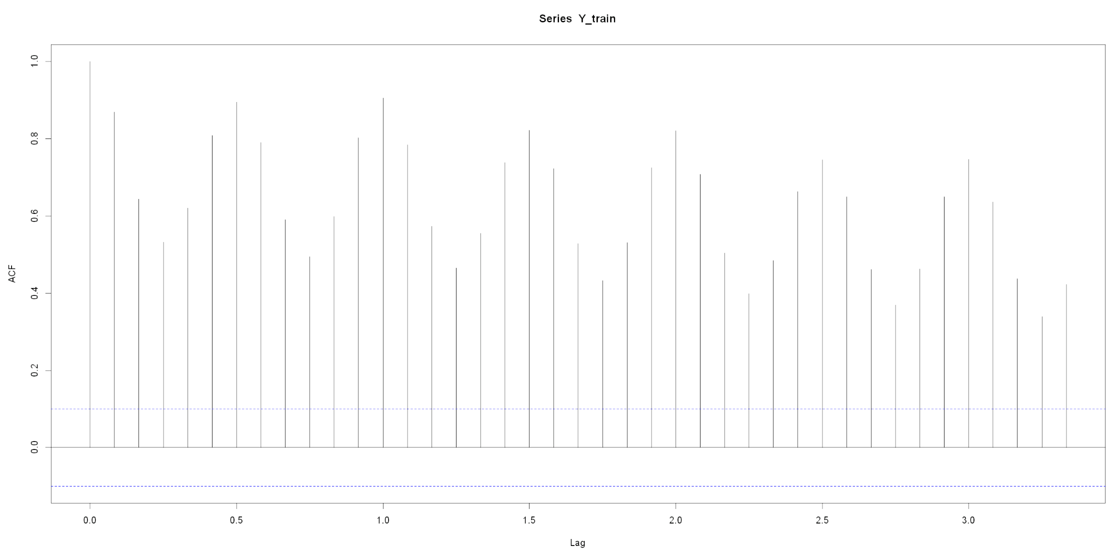
0.0.3 1)C)

[22]:
```
plot(Y_train)
```

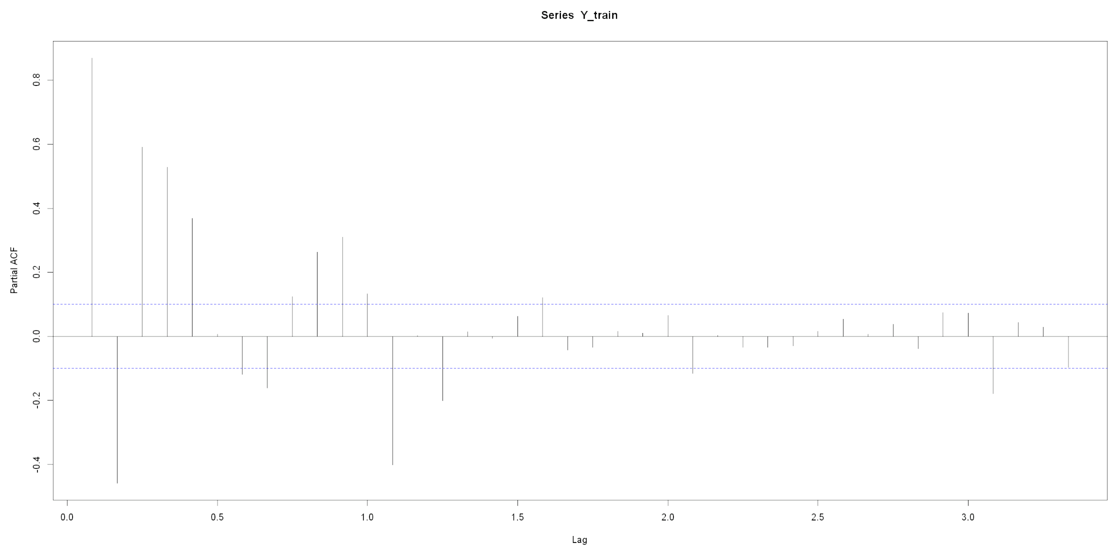


0.0.4 2)a)

[23]:
```
acf(Y_train, lag = 40)
```



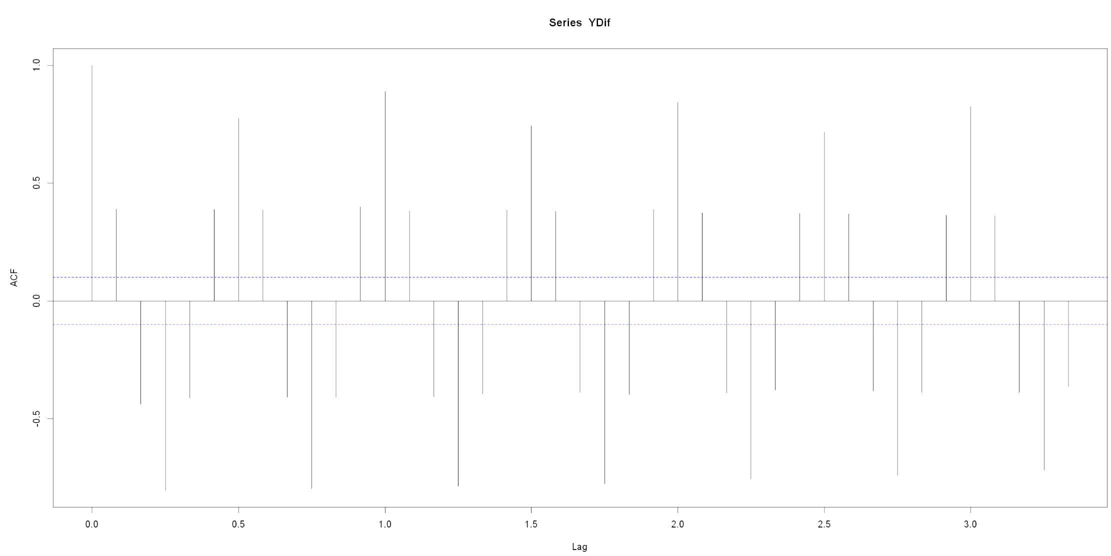
[24]:
```
pacf(Y_train, lag = 40)
```



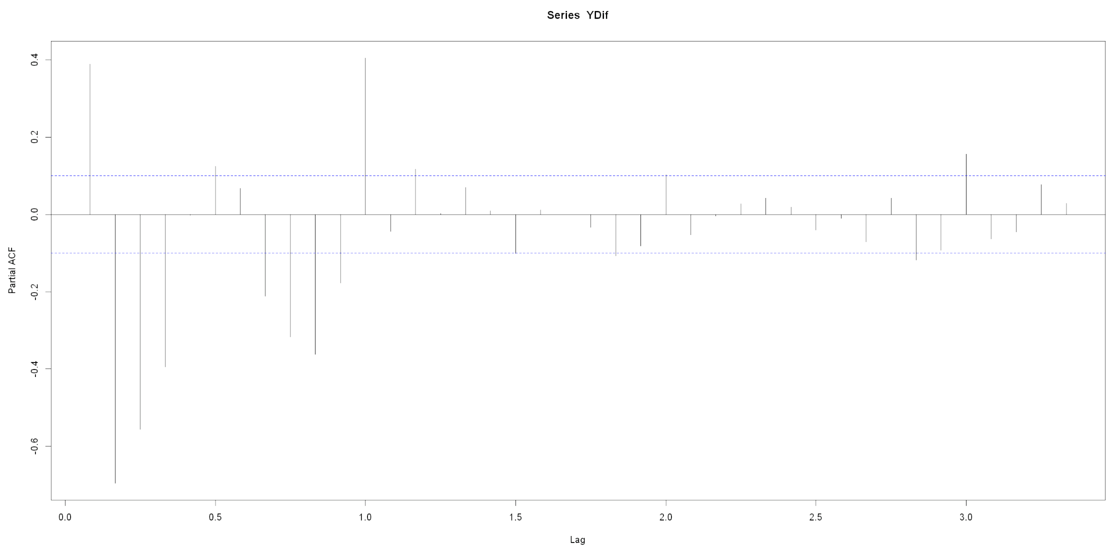
Série não estacionária

[25]:
```
log_Y<-log(Y_train)
YDif<-diff(log_Y)
```

[30]:
```
acf(YDif,lag = 40)
```

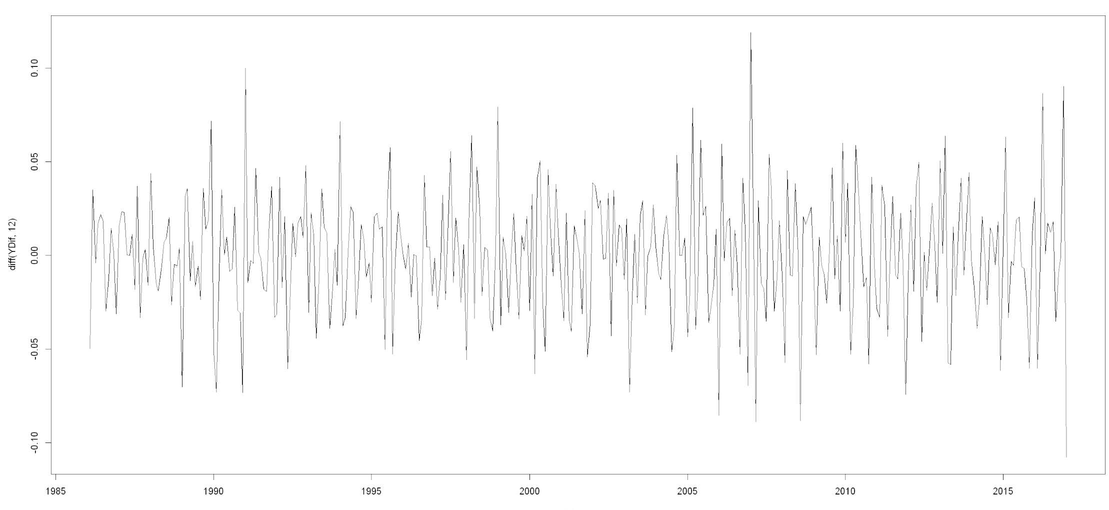


[31]:
```
pacf(YDif, lag = 40)
```



Ainda está não estacionária. Suspeito da necessidade de integrar a parte sazonal.

[39]:
```
plot(diff(YDif,12))
```



0.0.5 2)B)

[35]:
```
M1<-arima(log_Y, order = c(1, 1, 1),   seasonal = list(order = c(0, 1, 1)))
coeftest(M1)
AIC(M1)
```

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1   0.513454   0.054303   9.4554 < 2.2e-16 ***
ma1  -0.923166   0.024111 -38.2883 < 2.2e-16 ***
sma1 -0.814404   0.033575 -24.2563 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

-1701.74766764675

[37]:
```
M2<-arima(log_Y, order = c(1, 1 , 1),   seasonal = list(order = c(1, 1, 1)))
coeftest(M2)
AIC(M2)
```

```
z test of coefficients:

      Estimate Std. Error  z value Pr(>|z|)
ar1   0.518378   0.054024   9.5953   <2e-16 ***
ma1  -0.924708   0.023740 -38.9512   <2e-16 ***
sar1  0.081978   0.062869   1.3040   0.1923
sma1 -0.837271   0.035366 -23.6744   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

-1701.46383849794

[38]:
```
M3<-arima(log_Y, order = c(1, 1, 1),   seasonal = list(order = c(0, 1, 2)))
M3
coeftest(M3)
AIC(M3)
```

```
Call:
arima(x = log_Y, order = c(1, 1, 1), seasonal = list(order = c(0, 1, 2)))

Coefficients:
         ar1      ma1     sma1     sma2
      0.5203  -0.9254  -0.7262  -0.0991
s.e.  0.0538   0.0236   0.0627   0.0610

sigma^2 estimated as 0.0005643:  log likelihood = 856.16,  aic = -1702.33


z test of coefficients:
```

```
      Estimate Std. Error  z value Pr(>|z|)
ar1   0.520320   0.053843   9.6636   <2e-16 ***
ma1  -0.925422   0.023562 -39.2759   <2e-16 ***
sma1 -0.726164   0.062725 -11.5770   <2e-16 ***
sma2 -0.099113   0.060970  -1.6256    0.104
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

-1702.32698460546

[40]:
```r
M4<-arima(log_Y, order = c(2, 1, 1),   seasonal = list(order = c(1, 1, 2)))
coeftest(M4)
AIC(M4)
```

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1   0.533024   0.057342   9.2956 < 2.2e-16 ***
ar2  -0.037939   0.056249  -0.6745  0.500007
ma1  -0.919867   0.026666 -34.4953 < 2.2e-16 ***
sar1 -0.457070   0.212847  -2.1474  0.031760 *
sma1 -0.261031   0.197872  -1.3192  0.187104
sma2 -0.482504   0.156762  -3.0779  0.002084 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

-1701.74401091749

[41]:
```r
M5<-arima(log_Y, order = c(2, 1, 1),   seasonal = list(order = c(0, 1, 1)))
coeftest(M5)
AIC(M5)
```

```
z test of coefficients:

      Estimate Std. Error  z value Pr(>|z|)
ar1   0.522536   0.057522   9.0841   <2e-16 ***
ar2  -0.030988   0.056196  -0.5514   0.5813
ma1  -0.918021   0.027454 -33.4387   <2e-16 ***
sma1 -0.815885   0.033687 -24.2195   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

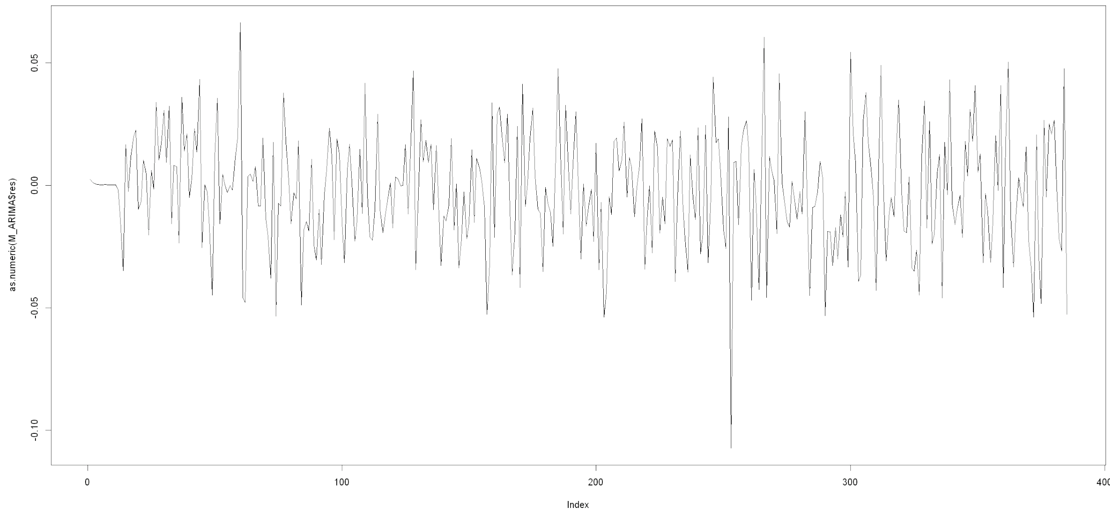
-1700.05128780756

Melhor modelo: Arima(1,1,1)(0,1,1)

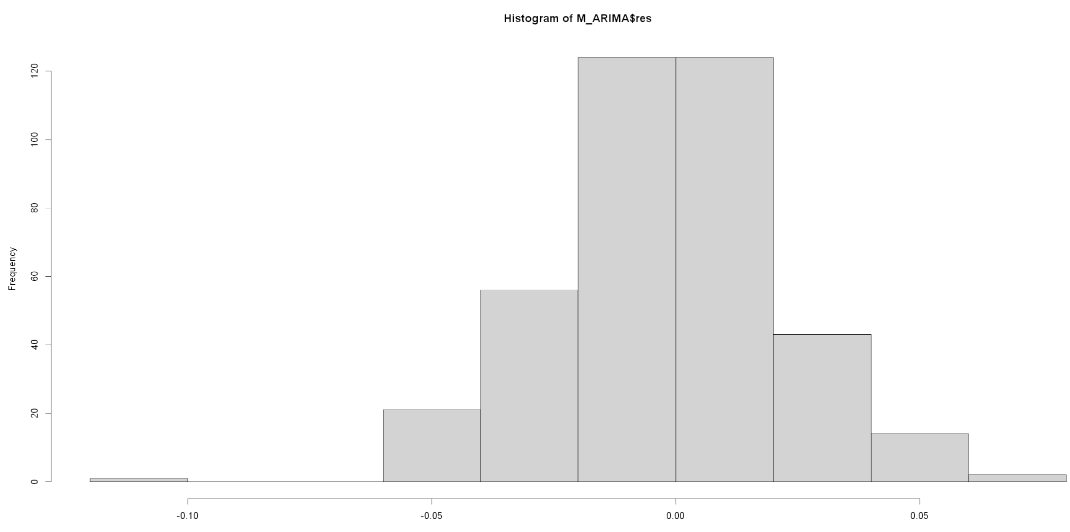
```
[45]: M_ARIMA<-M1
```

0.0.6 2)c)

```
[48]: plot(as.numeric(M_ARIMA$res),type='l')
```

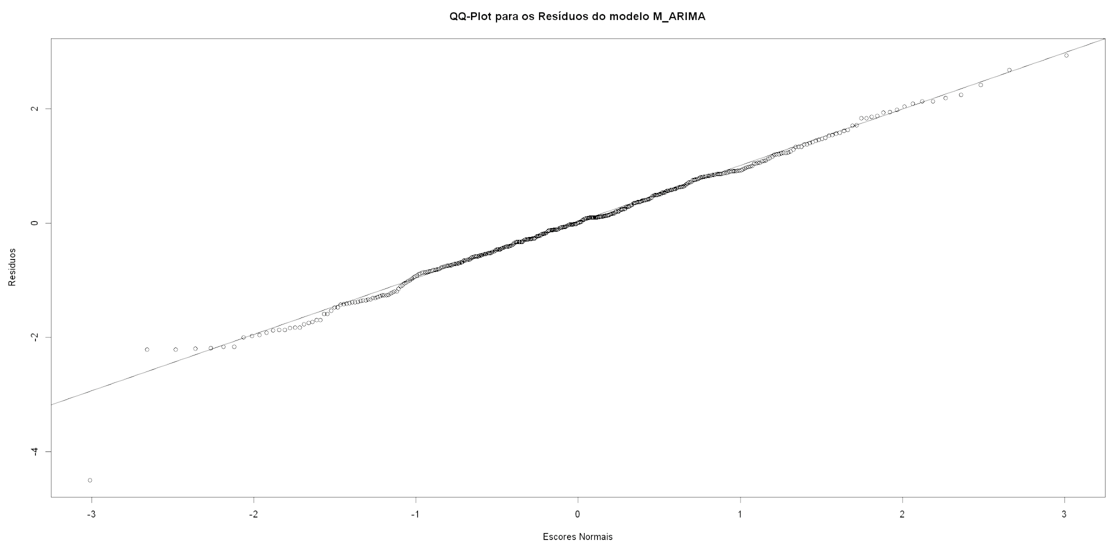



```
[50]: hist(M_ARIMA$res)
```

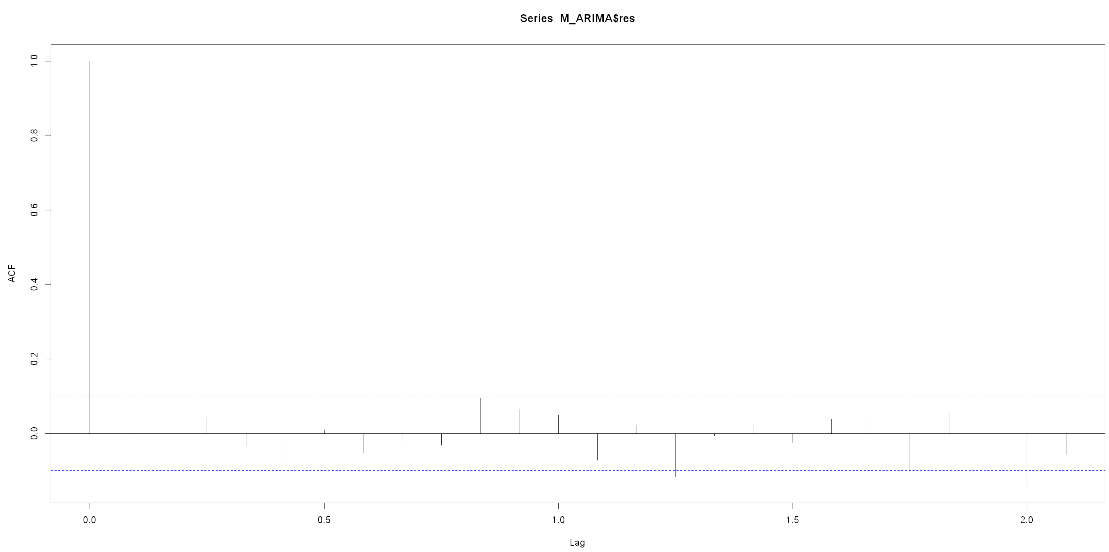



```
[51]: std_resid <- (M_ARIMA$res - mean(M_ARIMA$res))/sd(M_ARIMA$res)
```

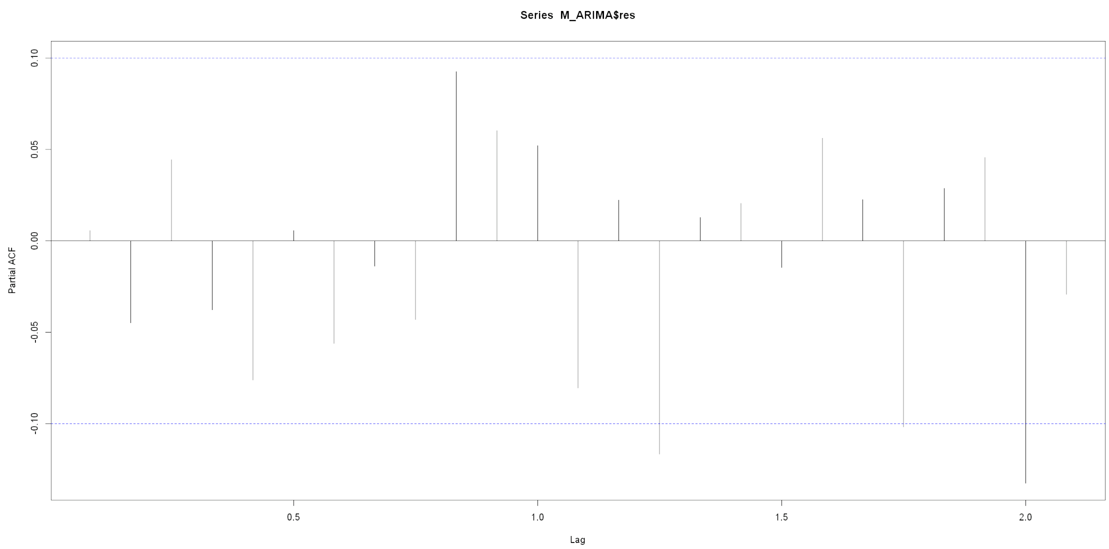
[53]:
```
qqnorm(std_resid,
       ylab="Residuos",
       xlab="Escores Normais",
       main="QQ-Plot para os Resíduos do modelo M_ARIMA")
qqline(std_resid)
```



[54]:
```
acf(M_ARIMA$res)
```



[55]:
```
pacf(M_ARIMA$res)
```



[57]:
```
shapiro.test(M_ARIMA$res)
```

```
        Shapiro-Wilk normality test

data:  M_ARIMA$res
W = 0.99287, p-value = 0.06405
```

[59]:
```
Box.test(M_ARIMA$resid, lag = 12, type = c("Box-Pierce", "Ljung-Box"))
```

```
        Box-Pierce test

data:  M_ARIMA$resid
X-squared = 12.008, df = 12, p-value = 0.445
```

Rejeitamos a hipótese de normalidade para o nível de 10%. Não rejeitamos a hipótese de independência dos resíduos. Não consegui um modelo melhor que esse, mesmo não conseguindo normalidade nos resíduos.

0.0.7 2)d)

Modelo com tendência e sazonalidade

[61]:
```
buildmod<-function(par){
    trend <- dlmModPoly(order=2,dV=exp(par[1]),dW=exp(par[2:3]))
    seasonal <- dlmModSeas(frequency=12,dV=0,dW=c(exp(par[4]),rep(0,10)))
    model<- trend+seasonal
```

```
    return(model)
}
```

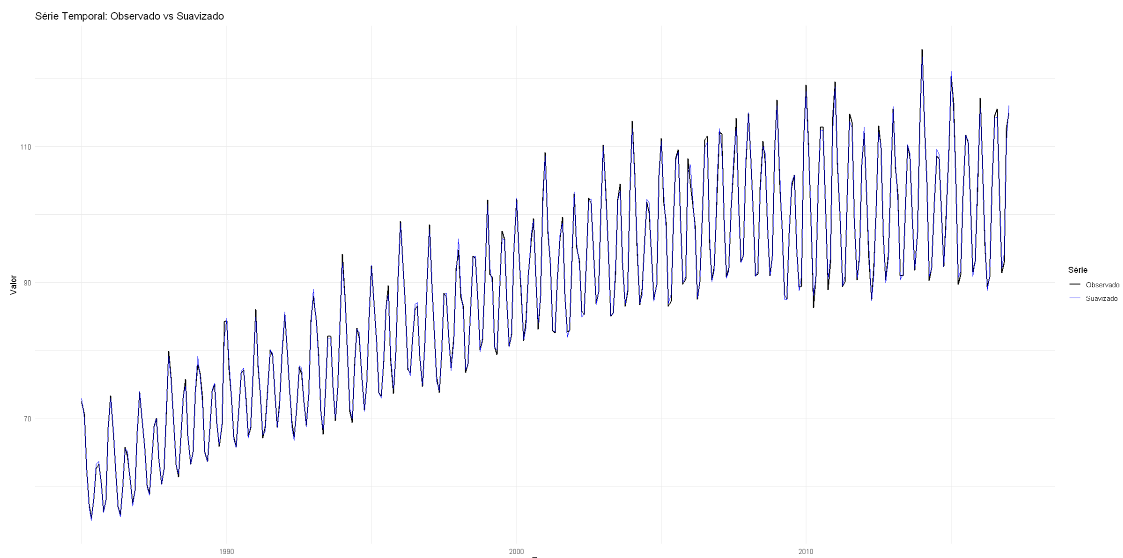
[62]:
```
fit<-dlmMLE(Y_train,parm=rep(0,4),build=buildmod)
```

[63]:
```
smoothed <- dlmSmooth(dlmFilter(Y_train, buildmod(exp(fit$par))))
```

[64]:
```
plot_data <- data.frame(
  Data = time(Y_train),
  Observado = as.numeric(Y_train),
  Suavizado = smoothed$s[-1,1] + smoothed$s[-1,3],  # Nível + sazonalidade
  Tendencia = smoothed$s[-1,1],
  Sazonalidade = smoothed$s[-1,3]
)
```

[65]:
```
ggplot(plot_data, aes(x = Data)) +
  geom_line(aes(y = Observado, color = "Observado"), linewidth = 0.7) +
  geom_line(aes(y = Suavizado, color = "Suavizado"), linewidth = 0.5) +
  labs(title = "Série Temporal: Observado vs Suavizado",
       y = "Valor",
       x = "Tempo") +
  scale_color_manual(name = "Série",
                     values = c("Observado" = "Black",
                                "Suavizado" = "blue")) +
  theme_minimal()
```

Don't know how to automatically pick scale for object of type
<ts>. Defaulting to continuous.



0.0.8 3)

Previsão e erro quadrático médio do modelo arima

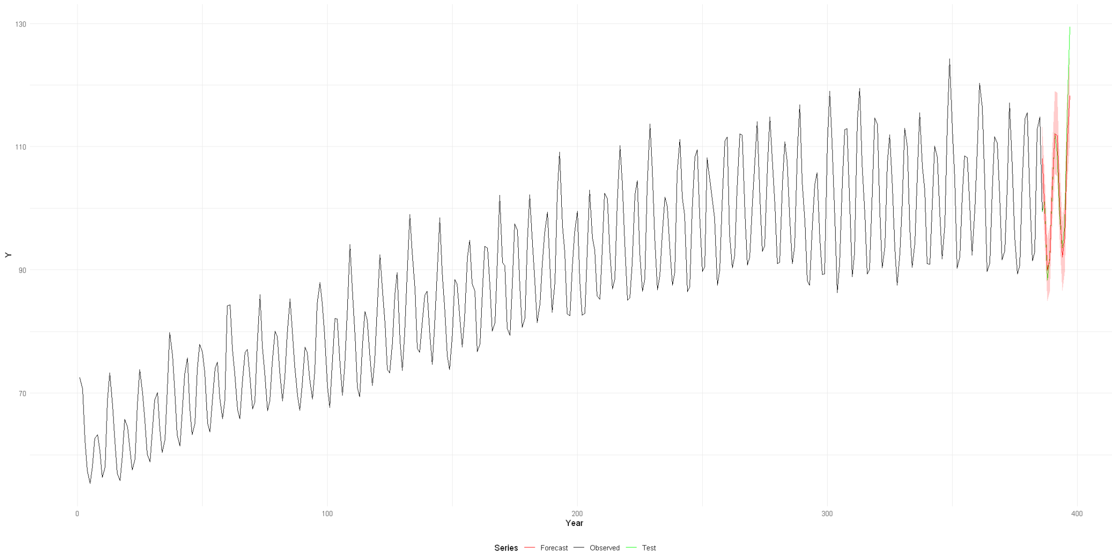
[73]:
```
Prev=forecast(M_ARIMA, H, level=c(95))
```

[74]:
```
time_index <- time(Y)
```

[75]:
```
df <- data.frame(
  Time = time_index,
  Observed = as.numeric(Y),
  Test = c(rep(NA, n-H), Y_test),
  Forecast = rep(NA, n),
  Lower = rep(NA, n),
  Upper = rep(NA, n)
)
```

[76]:
```
for(i in 1:H) {
  df$Forecast[n-H+i] <- exp(Prev$mean[i])
  df$Lower[n-H+i] <- exp(Prev$lower[i])
  df$Upper[n-H+i] <- exp(Prev$upper[i])
}
```

[77]:
```
ggplot(df, aes(x = Time)) +
  geom_line(aes(y = Observed, color = "Observed")) +
  geom_line(aes(y = Test, color = "Test"), na.rm = TRUE) +
  geom_line(aes(y = Forecast, color = "Forecast"), na.rm = TRUE) +
  geom_ribbon(aes(ymin = Lower, ymax = Upper),
              fill = "red", alpha = 0.2, na.rm = TRUE) +
  scale_color_manual(values = c("Observed" = "black",
                                "Test" = "green",
                                "Forecast" = "red")) +
  labs(x = "Year", y = "Y", color = "Series") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



[78]:
```
# Calculo do EQMP
EQMP <- sum((as.numeric(Y_test)-exp(Prev$mean))^2)/H
EQMP
```

20.4151575086912

Previsão para o modelo de espaço de estado

[80]:
```
fit<-dlmFilter(Y_train,buildmod(exp(fit$par)))
```

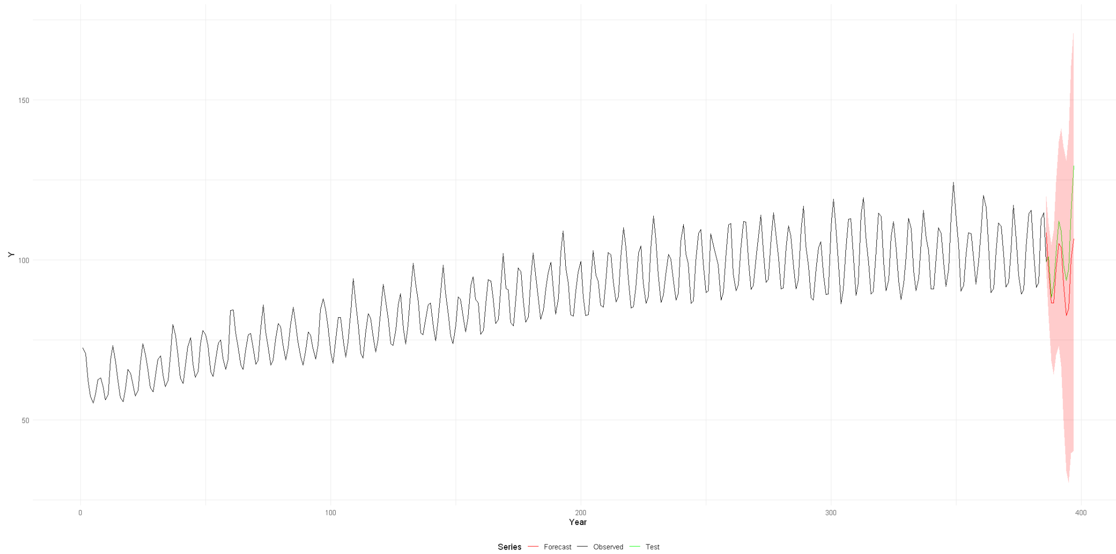
[81]:
```
forecast<-dlmForecast(fit,nAhead=12)
```

[83]:
```
q_value <- qnorm(0.975)
lower <- forecast$f - q_value * sqrt(unlist(forecast$Q))
upper <- forecast$f + q_value * sqrt(unlist(forecast$Q))
```

[84]:
```
df2<- data.frame(
  Time = time(Y),
  Observed = as.numeric(Y),
  Test = c(rep(NA, n-H), Y_test),
  Forecast = c(rep(NA, n-H),as.numeric(forecast$f)),
  Lower = c(rep(NA, n-H),lower),
  Upper = c(rep(NA, n-H),upper)
)
```

[85]:
```
ggplot(df2, aes(x = Time)) +
  geom_line(aes(y = Observed, color = "Observed")) +
  geom_line(aes(y = Test, color = "Test"), na.rm = TRUE) +
  geom_line(aes(y = Forecast, color = "Forecast"), na.rm = TRUE) +
```

```
  geom_ribbon(aes(ymin = Lower, ymax = Upper),
              fill = "red", alpha = 0.2, na.rm = TRUE) +
  scale_color_manual(values = c("Observed" = "black",
                                "Test" = "green",
                                "Fitted" = "blue",
                                "Forecast" = "red")) +
  labs(x = "Year", y = "Y", color = "Series") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



[88]:
```
EQMP <- sum((as.numeric(Y_test)-forecast$f)^2)/H
EQMP
```

105.82502639787