Trabalho 2 de Análise de Série Temporal

Carlos Lopes, Cristiane Fractal, Cristovão Moreira Freitas Junior, Fábio Karpusca Marin, Renato Bastos Pope

20/04/2021

Apresentação do caso:

Despesas de telefonia móvel – Banco X Os dados apresentados são referentes a despesas somadas de todos os contratos de prestação de serviço de telefonia móvel, de janeiro de 2015 até março de 2021. Ao longo desse período foram assinados 12 contratos, com vigência de 5 anos cada, para uma média de 18.000 linhas ativas entre 2015 e 2019, e 25.000 linhas ativas a partir de 2020. Até o final de 2020 os contratos faturavam os seguintes serviços:

- Assinatura mensal da linha e cessão de comodato do dispositivo;
- Ligações telefônicas, por minuto e por destinação;
- SMS;
- Pacote de dados;
- Roaming nacional e internacional;
- Outros serviços de valor agregado.

A partir de 2021 os contratos passarão a faturar os seguintes serviços:

- Assinatura mensal da linha incluindo pacote de dados;
- Ligações nacionais, SMS e roaming nacional são ilimitados, sem custo adicional;
- Roaming internacional e outros serviços de valor agregado são cobrados à parte.

Importar a base de dados

```
Dados_Telefonia_Movel <- read_excel("Dados Telefonia Movel.xlsx")
db <- Dados_Telefonia_Movel
```

Converter a base de dados em série temporal:

```
db_ts <- ts(db$`Telefonia Móvel`, start=c(2015, 1), end=c(2021, 3), frequ
ency = 12)
```

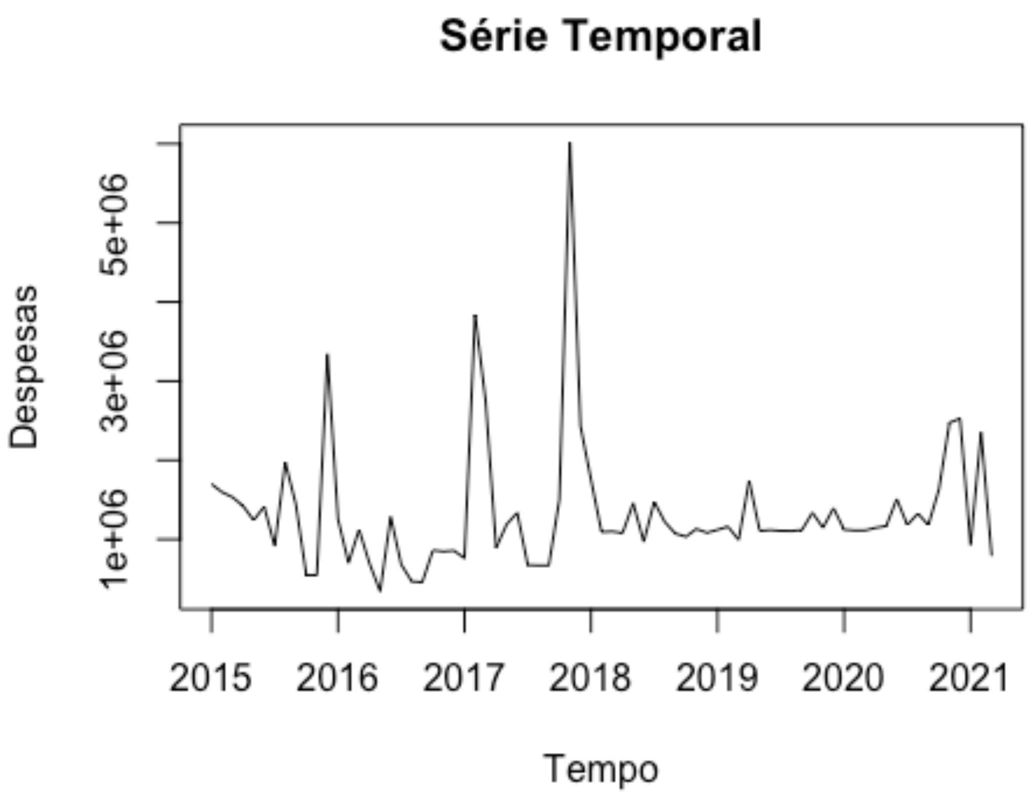
Análise estatística da série:

```
summary(db_ts)
```

```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   342538  950886 1130180 1339072 1450561 6012284
```

Plotar o gráfico da série temporal:

```
plot(db_ts, xlab="Tempo", ylab="Despesas", ylim=c(342000, 6013000), type=
"l", main="Série Temporal")
```



Definir o tamanho da amostra de treinamento e da amostra de teste:

```
amostra_validacao <- 20
amostra_treino <- length(db_ts) - amostra_validacao
```

Programar a amostra de treinamento:

```
treinamento_ts <-window(db_ts, start=c(2015, 1), end=c(2015, amostra_trei
no))
```

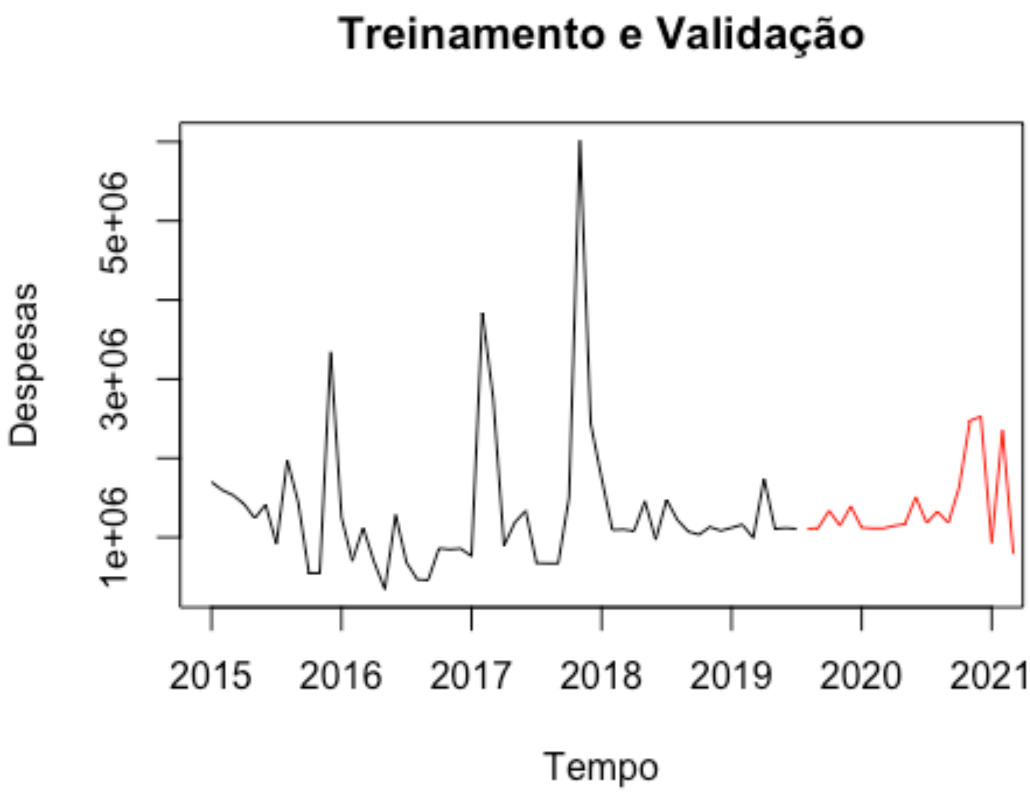
Programar a amostra de validação:

```
validacao_ts <- window(db_ts, start=c(2015, amostra_treino + 1), end=c(20
15, amostra_treino + amostra_validacao))
```

PLotagem do gráfico do treinamento com validação:

```
plot(treinamento_ts, xlab="Tempo", ylab="Despesas", xaxt="n" , ylim=c(342
000, 6013000), xlim=c(2015, 2021), type="l", main="Treinamento e Validaçã
o")
```

```r
axis(1, at=seq(2015, 2021,1), labels=format(seq(2015, 2021,1)))
lines(validacao_ts, bty="l", col="red")
```



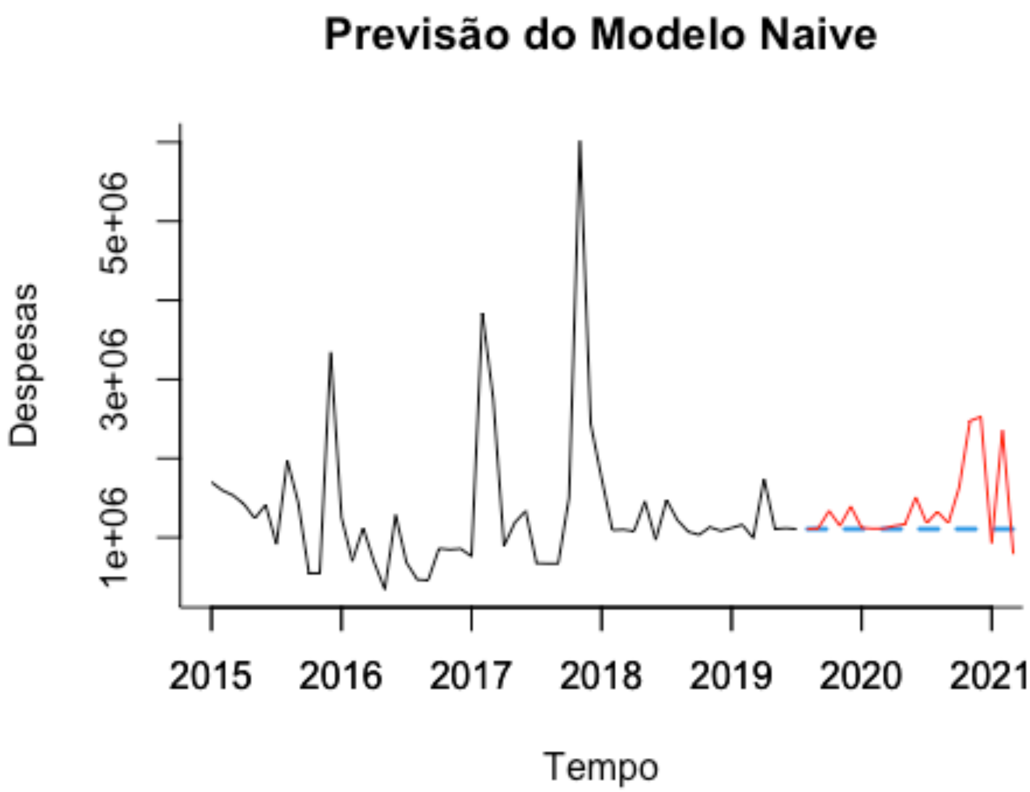
Confecção do modelo Naive:

```r
modelo_naive <- naive(treinamento_ts, level=0, h=amostra_validacao)
accuracy(modelo_naive, validacao_ts)
```

```
##                      ME      RMSE      MAE       MPE     MAPE      MASE
## Training set -10955.18 1122711.5 621341.4 -17.33006 42.30944 0.6597082
## Test set     276316.56  556863.3 325664.9  12.32497 18.23478 0.3457742
##                    ACF1 Theil's U
## Training set -0.2828524        NA
## Test set      0.0563613 0.9234698
```

Grafico da série temporal de treinamento, validação e modelo naive

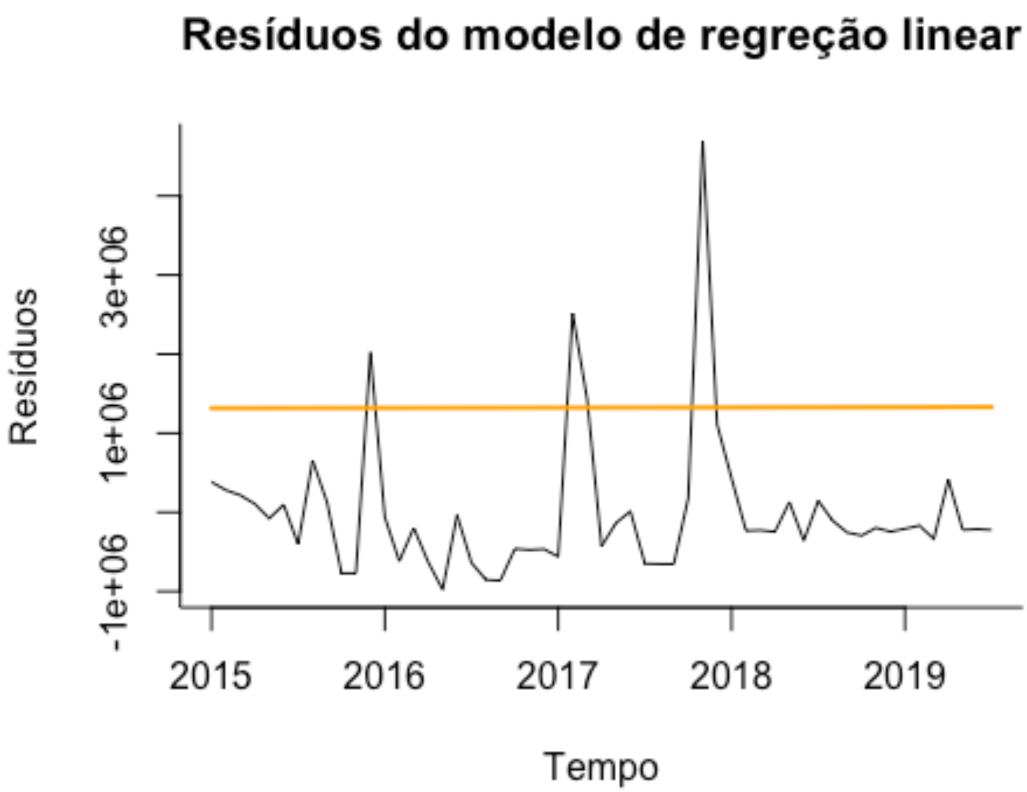
```r
plot(modelo_naive, xlab="Tempo", ylab="Despesas", xaxt="s" , ylim=c(34200
0, 6013000), xlim=c(2015, 2021), bty="l", flty=2, main="Previsão do Model
o Naive")
axis(1, at=seq(2015, 2021,1), labels=format(seq(2015, 2021,1)))
lines(validacao_ts, bty="l", col="red")
```



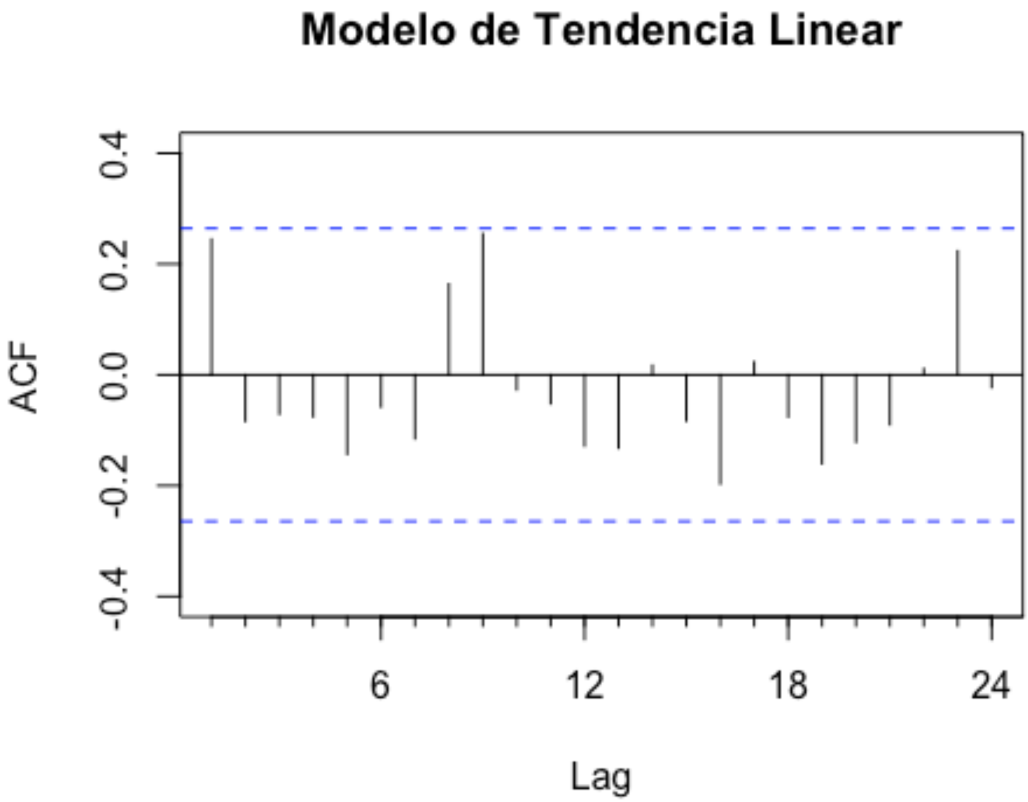
Modelo de Tendência linear:

```
modelo_tendencia_linear <- tslm(treinamento_ts ~ trend)
summary(modelo_tendencia_linear)

##
## Call:
## tslm(formula = treinamento_ts ~ trend)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -977724 -463807 -213912  128175 4687242
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1315746.6   252456.3   5.212 3.14e-06 ***
## trend           265.6     7843.4   0.034    0.973
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 923400 on 53 degrees of freedom
## Multiple R-squared:  2.163e-05,	Adjusted R-squared:  -0.01885
## F-statistic: 0.001147 on 1 and 53 DF,  p-value: 0.9731
```

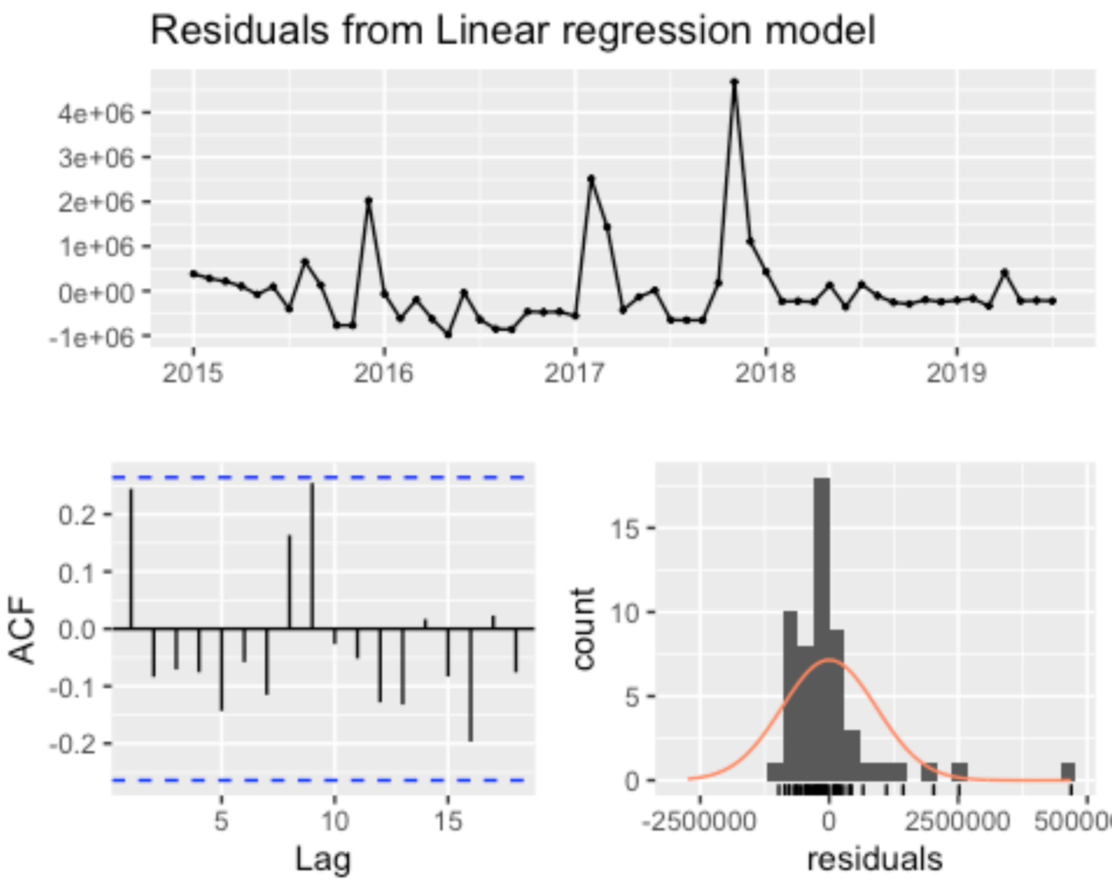
```
plot(modelo_tendencia_linear$residuals, xlab="Tempo", ylab="Resíduos", bty="l", main="Resíduos do modelo de regreção linear")
lines(modelo_tendencia_linear$fitted.values, lwd="2", col="orange")
```




```
Acf(modelo_tendencia_linear$residuals, main="Modelo de Tendencia Linear")
```

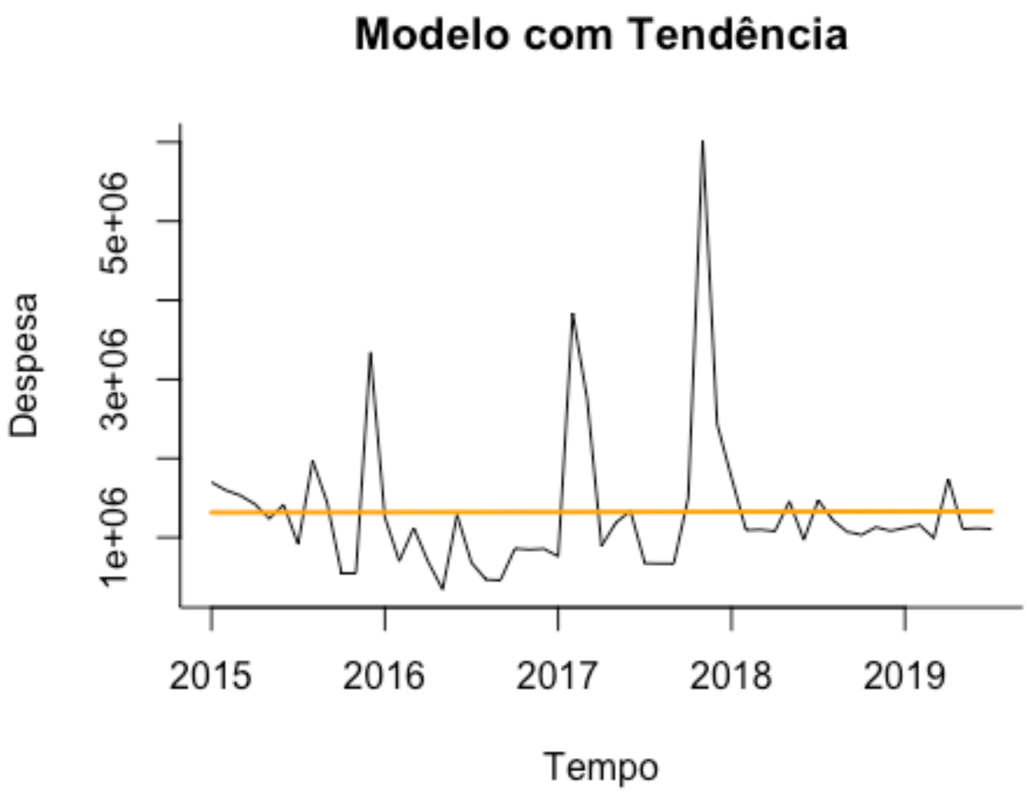



```
checkresiduals(modelo_tendencia_linear, test="LB")
```

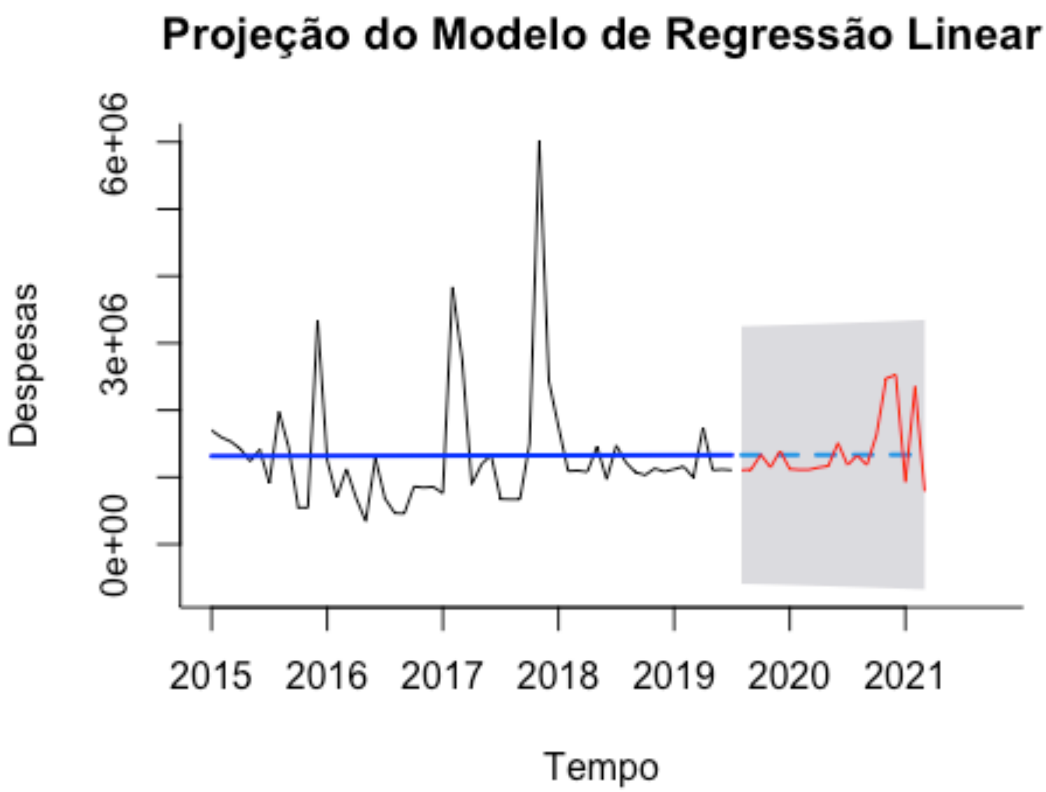



```
##
##  Ljung-Box test
##
## data:  Residuals from Linear regression model
## Q* = 13.368, df = 9, p-value = 0.1467
##
## Model df: 2.   Total lags used: 11
```

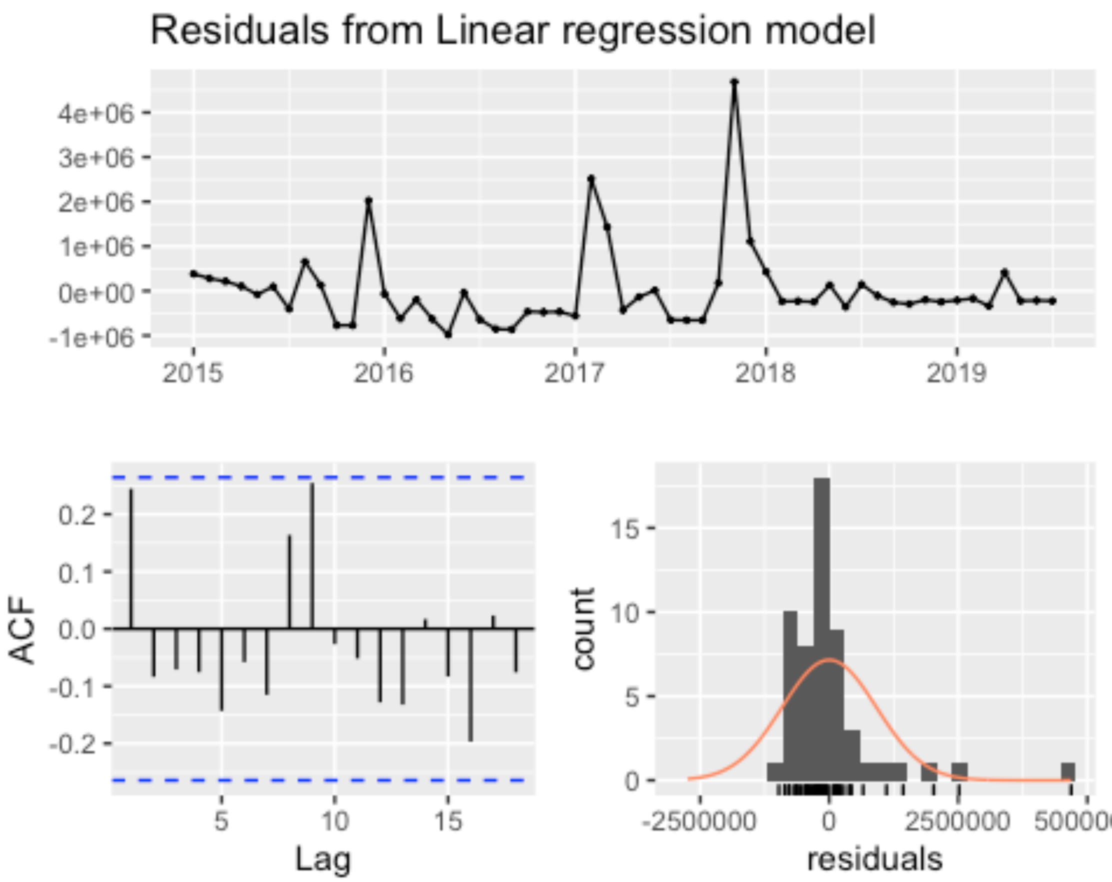
```
plot(treinamento_ts, xlab="Tempo", ylab="Despesa", bty="l", main="Modelo
com Tendência")
lines(modelo_tendencia_linear$fitted.values, lwd="2", col="orange")
```




```
modelo_tendencia_linear_proj <- forecast(modelo_tendencia_linear, h=amost
ra_validacao, level=0.95)

plot(modelo_tendencia_linear_proj, xlab="Tempo", ylab="Despesas", xaxt="n
" , xlim=c(2015, 2021.75), bty="l", flty=2, main="Projeção do Modelo de R
egressão Linear")
axis(1, at=seq(2015, 2021, 1), labels=format(seq(2015,2021,1)))
lines(validacao_ts, col="red")
lines(modelo_tendencia_linear_proj$fitted, lwd="2", col="blue")
```




```
checkresiduals(modelo_tendencia_linear, test="LB")
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Linear regression model
## Q* = 13.368, df = 9, p-value = 0.1467
##
## Model df: 2.   Total lags used: 11
```

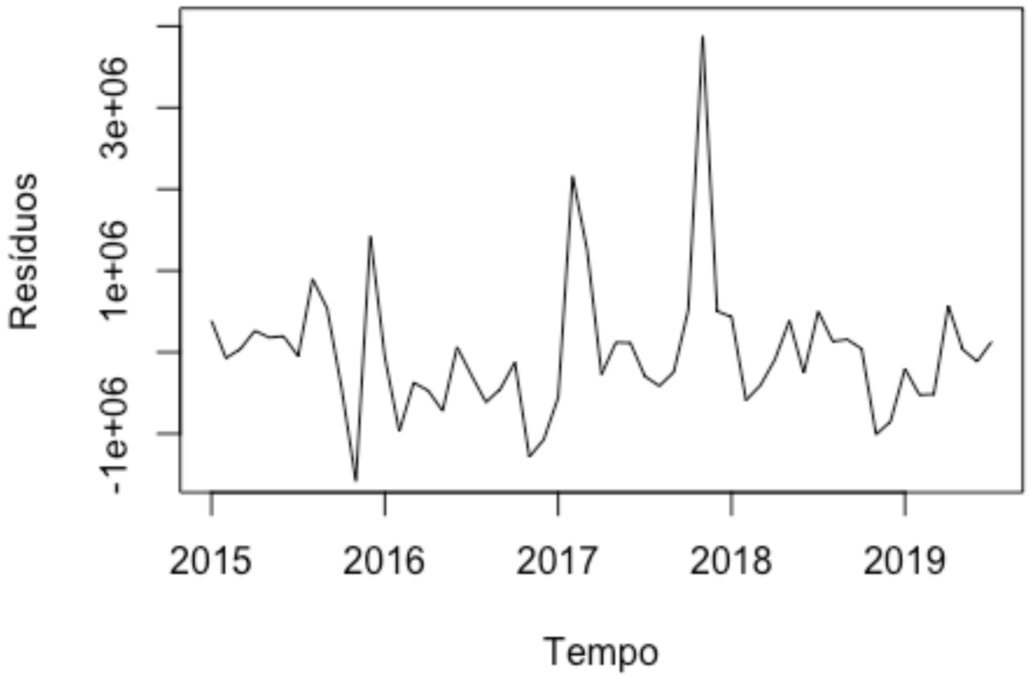
```
accuracy(modelo_tendencia_linear_proj, validacao_ts)
```

```
##                        ME     RMSE      MAE        MPE     MAPE      M
ASE
## Training set 3.600772e-11 906451.5 543025.5 -31.190011 49.06136 0.5765
564
## Test set     4.962528e+04 485395.9 338812.5  -5.631544 22.73450 0.3597
336
##                    ACF1 Theil's U
## Training set 0.24473472        NA
## Test set     0.05374474 0.7712741
```

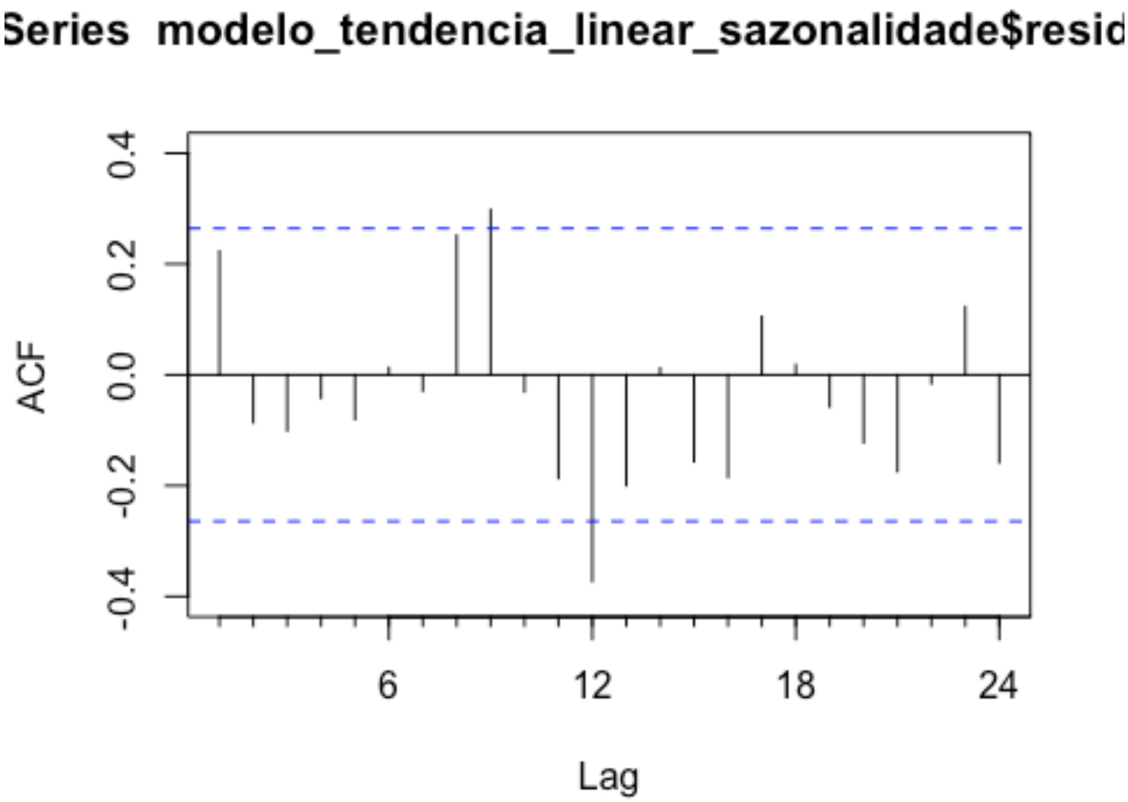
Pelo baixo valor de R Square e AR-Squared, o Modelo de Tendência Linear não foi satisfatório. Partimos para o próximo modelo.

Modelo de Tendencia linear com Sazonalidade

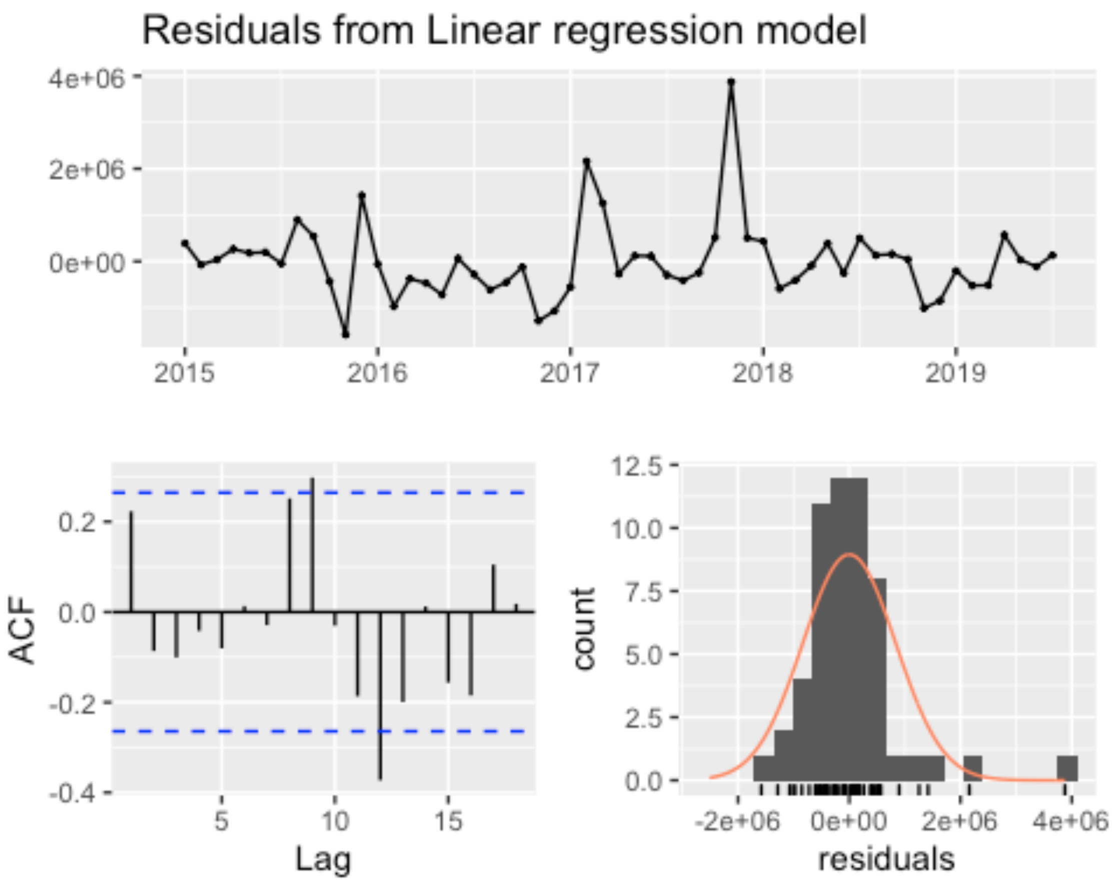
```
modelo_tendencia_linear_sazonalidade <- tslm(treinamento_ts ~ season+tren
d)
summary(modelo_tendencia_linear_sazonalidade)

##
## Call:
## tslm(formula = treinamento_ts ~ season + trend)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1581776  -442666   -73520   229384  3876568
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1310468.7   470336.4   2.786  0.00797 **
## season2      359198.3   600187.6   0.598  0.55274
## season3      181747.3   600352.1   0.303  0.76359
## season4     -152623.7   600626.1  -0.254  0.80065
## season5     -252309.5   601009.5  -0.420  0.67676
## season6      -95486.5   601502.1  -0.159  0.87463
## season7     -349930.6   602103.7  -0.581  0.56423
## season8     -235364.9   636588.6  -0.370  0.71344
## season9     -406980.1   636743.7  -0.639  0.52619
## season10    -332293.8   637002.1  -0.522  0.60465
## season11     815159.4   637363.6   1.279  0.20793
## season12     610086.8   637828.1   0.957  0.34429
## trend           288.2     8112.9   0.036  0.97183
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 948900 on 42 degrees of freedom
## Multiple R-squared:  0.1632, Adjusted R-squared:  -0.07589
## F-statistic: 0.6826 on 12 and 42 DF,  p-value: 0.7582

plot(modelo_tendencia_linear_sazonalidade$residuals, xlab="Tempo", ylab="
Resíduos",ylim=c(-1500000, 4000000), type="l")
```




```
Acf(modelo_tendencia_linear_sazonalidade$residuals)
```

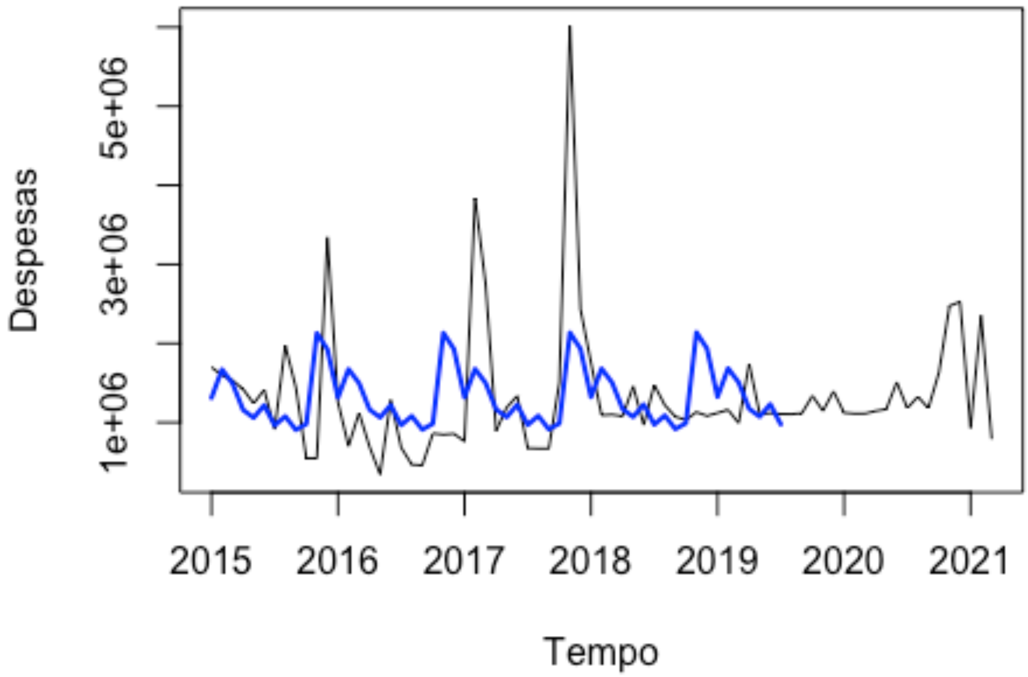



```
checkresiduals(modelo_tendencia_linear_sazonalidade, test="LB", main="Tes
te de Ljung-Box")
```

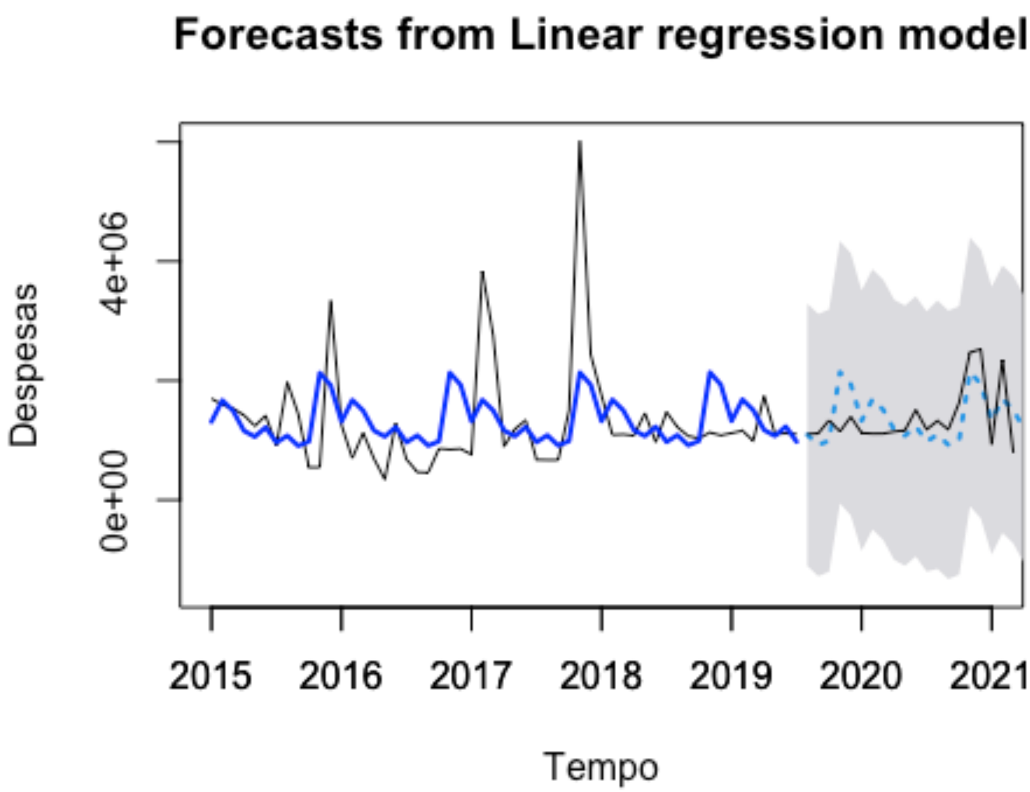



```
##
##  Ljung-Box test
##
## data:  Residuals from Linear regression model
## Q* = 35.139, df = 3, p-value = 1.139e-07
##
## Model df: 13.   Total lags used: 16
```

```
plot(db_ts, xlab="Tempo", ylab="Despesas", ylim=c(342000, 6013000), type=
"l")
lines(modelo_tendencia_linear_sazonalidade$fitted.values, lwd=2, col="Blu
e")
```




```
modelo_tendencia_linear_sazonalidade_proj <- forecast(modelo_tendencia_li
near_sazonalidade, h = 55, level=0.95)

plot(modelo_tendencia_linear_sazonalidade_proj, xlab="Tempo", ylab="Despe
sas", xaxt="s" , ylim=c(-1500000, 6013000), xlim=c(2015, 2021), type="l",
flty=3)
axis(1, at=seq(2015, 2021), labels=format(seq(2015, 2021)))
lines(validacao_ts)
lines(modelo_tendencia_linear_sazonalidade_proj$fitted, lwd=2, col="blue"
)
```



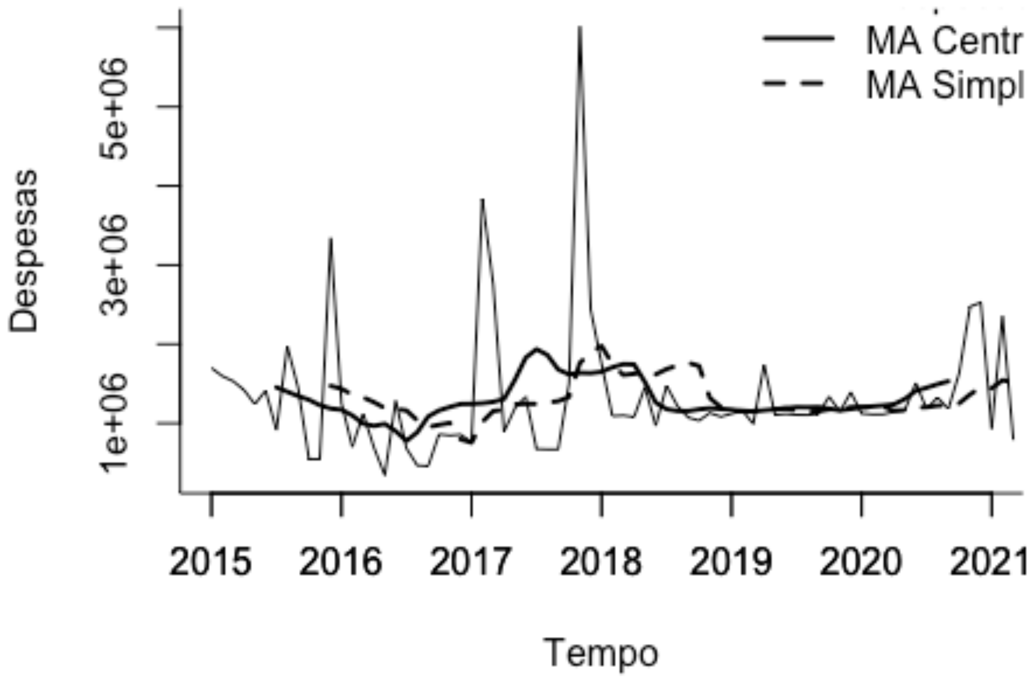

```
accuracy(modelo_tendencia_linear_sazonalidade_proj, validacao_ts)

##                         ME     RMSE      MAE        MPE     MAPE
MASE
## Training set  5.607526e-11 829202.6 542067.0 -26.497013 48.46750 0.575
5387
## Test set     -3.218108e+03 457840.2 384434.7  -6.974305 29.96604 0.408
1729
##                   ACF1 Theil's U
## Training set 0.2231391        NA
## Test set     0.1232444 0.8003244
```

O Erro Médio Absoluto Percentual (MAPE) deste modelo foi superior ao do modelo Naive. Baseado neste dado, passaremos para um Modelo de Média móvel.

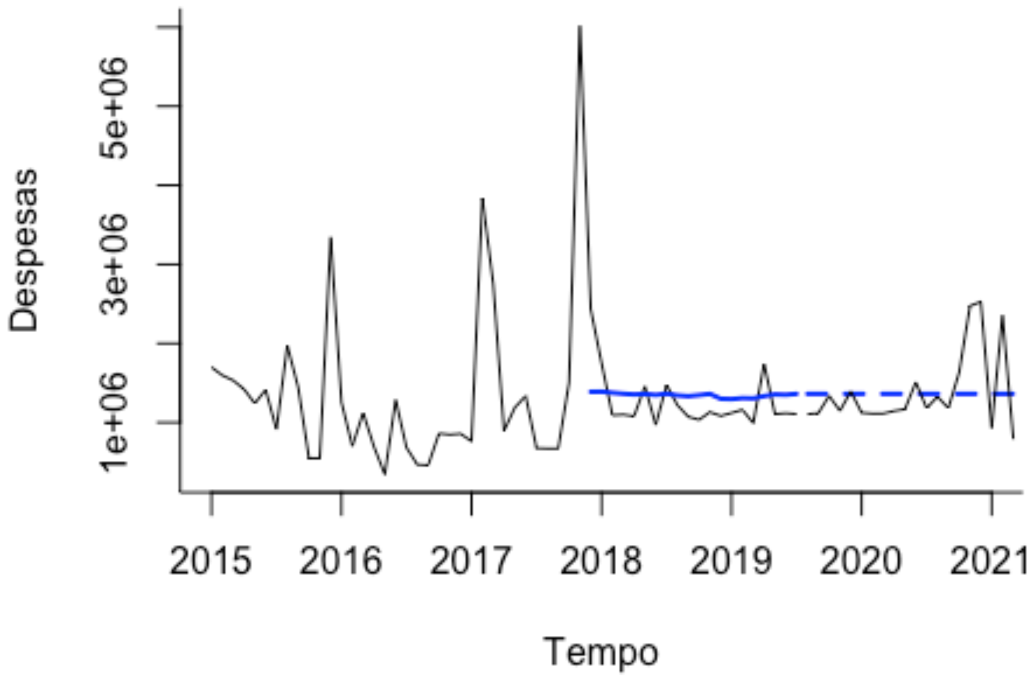
Modelo de Média Móvel

```
ma_simples <- rollmean(db_ts, k=12, align="right")
ma_centrada <- ma(db_ts, order=12)

plot(db_ts, ylim=c(342000, 6013000), ylab="Despesas", xlab="Tempo", bty="
l", xaxt="s", xlim=c(2015,2021))
axis(1, at=seq(2015, 2021, 1), labels=format(seq(2015, 2021, 1)))
lines(ma_centrada, lwd=2)
```

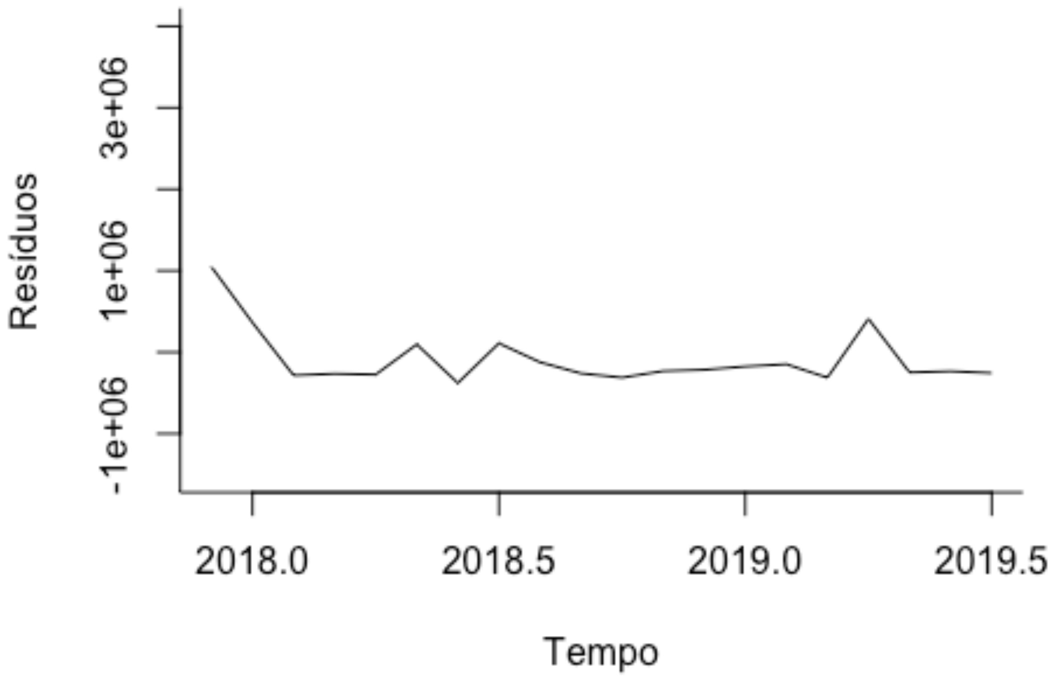
```
lines(ma_simples, lwd=2, lty=2)
legend(2019,7000000, c("Despesas", "MA Centrada", "MA Simples"), lty=c(1,
1,2), lwd=c(1,2,2), bty="n")
```



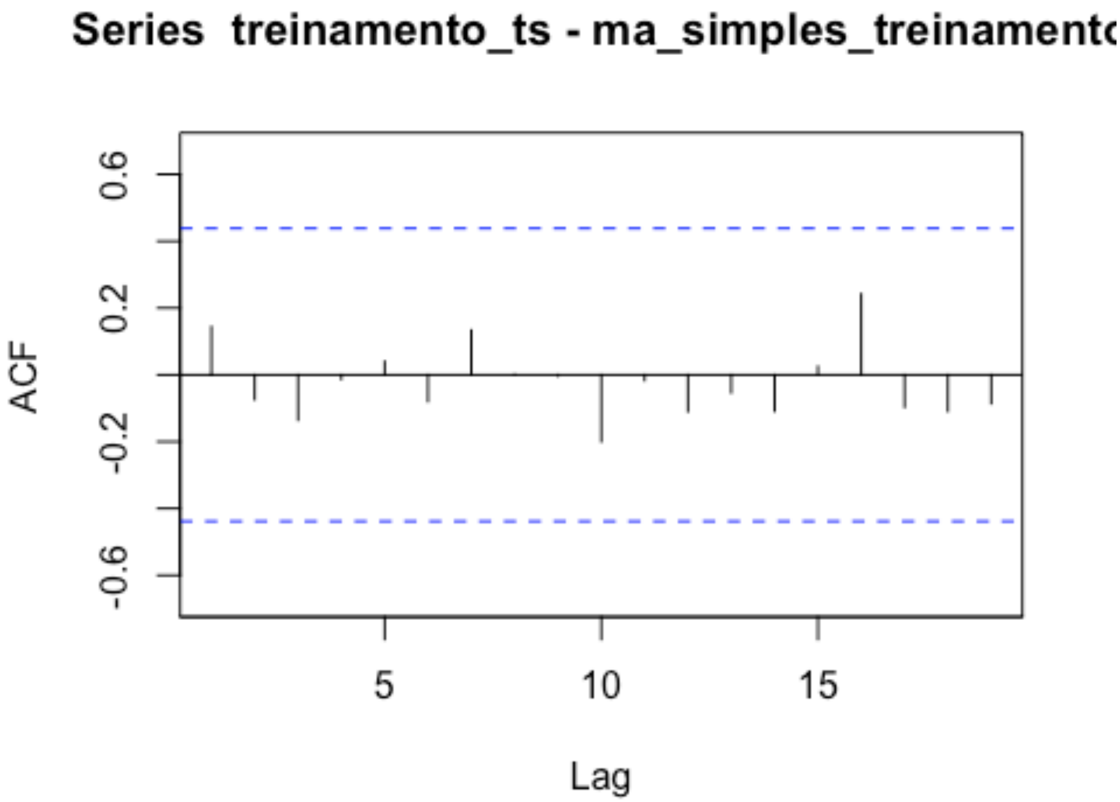

```
ma_simples_treinamento <- rollmean(treinamento_ts, k=36, align="right")
ultima_ma <- tail(ma_simples_treinamento, 1)

ma_simples_proj <- ts(rep(ultima_ma, 55), start=c(2015, 56), end = c(2015
, 75), freq=12)

plot(treinamento_ts, ylim=c(342000, 6013000), ylab="Despesas", xlab="Temp
o", bty="l", xaxt="n", xlim=c(2015,2021))
axis(1, at=seq(2015, 2021), labels=format(seq(2015, 2021)))
lines(ma_simples_treinamento, lwd=2, col="blue")
lines(ma_simples_proj, lwd=2, lty=2, col="blue")
lines(validacao_ts)
```




```
plot(treinamento_ts-ma_simples_treinamento, xlab="Tempo", ylab="Resíduos"
, ylim=c(-1500000, 4000000), bty="l")
```

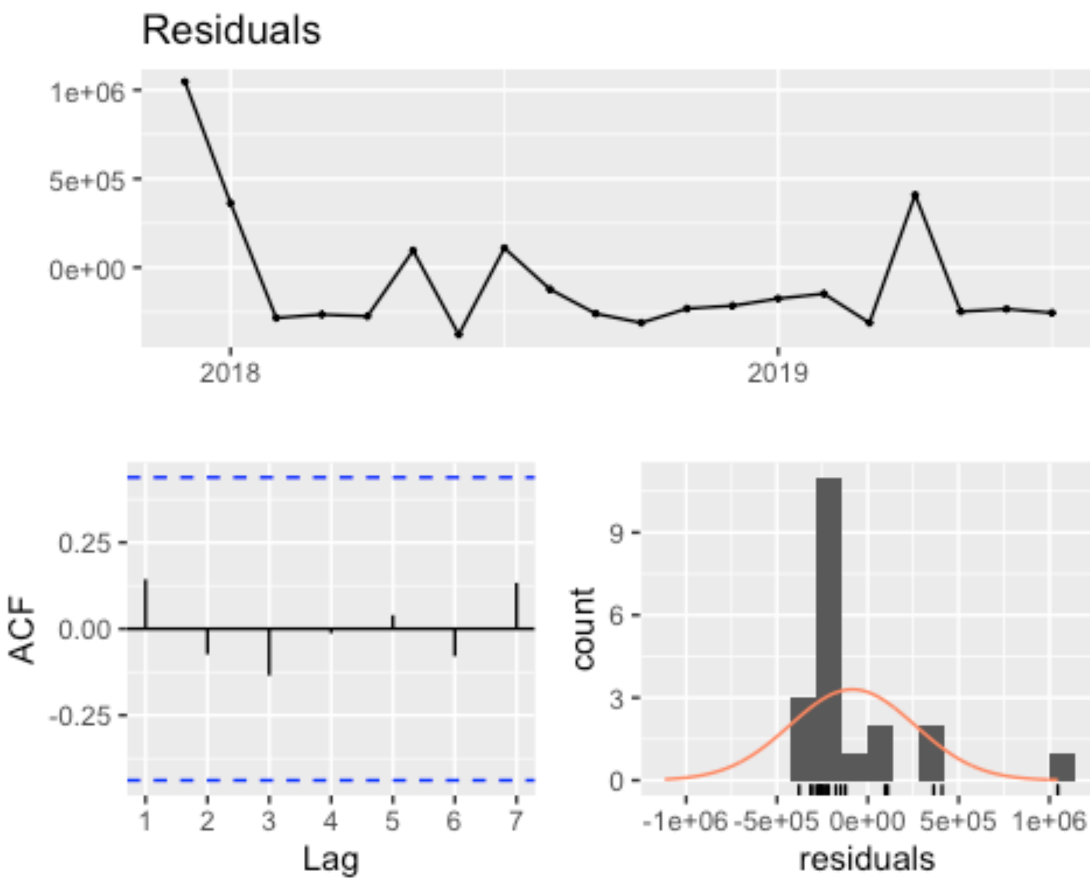



```
Acf(treinamento_ts-ma_simples_treinamento)
```




```
checkresiduals(treinamento_ts-ma_simples_treinamento, test="LB")
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees
of
## freedom for this model.
```



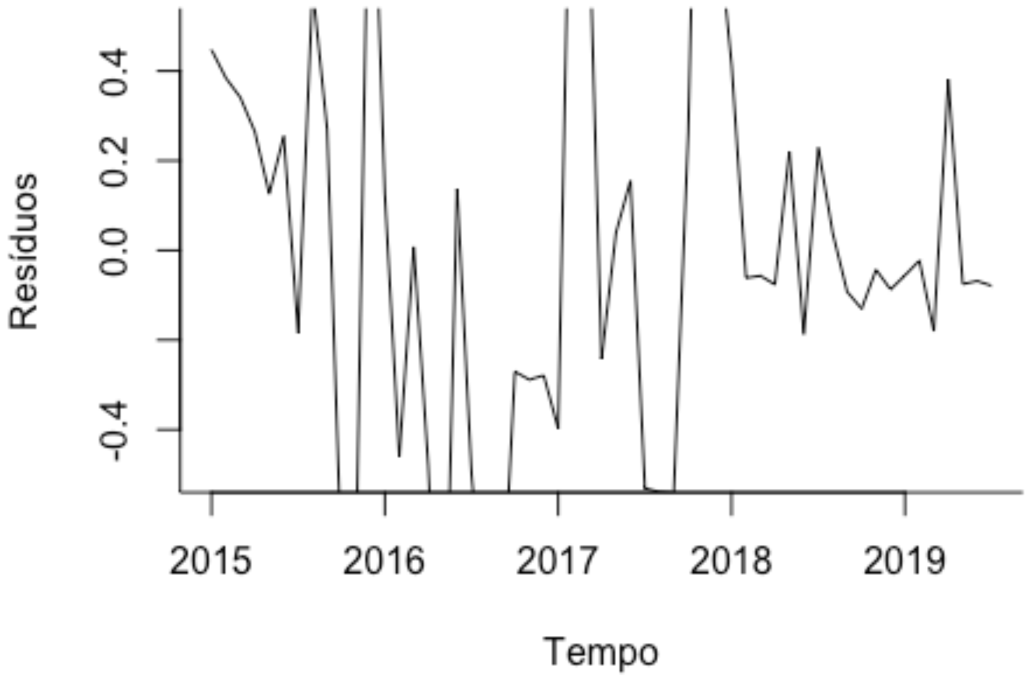

```
accuracy(ma_simples_treinamento, treinamento_ts)

##                    ME     RMSE      MAE       MPE     MAPE      ACF1 Thei
l's U
## Test set -84558.58 346045.6 286905.3 -12.22114 22.33803 0.1438404 0.76
61441

accuracy(ma_simples_proj, validacao_ts)

##                  ME   RMSE      MAE       MPE     MAPE      ACF1 Theil's
U
## Test set 20917.06 483925 350275.8 -7.912854 24.07041 0.0563613 0.76626
08
```

O MAPE do modelo de média móvel também foi superior ao MAPE do Modelo Naive. Modelo de tendência exponencial:

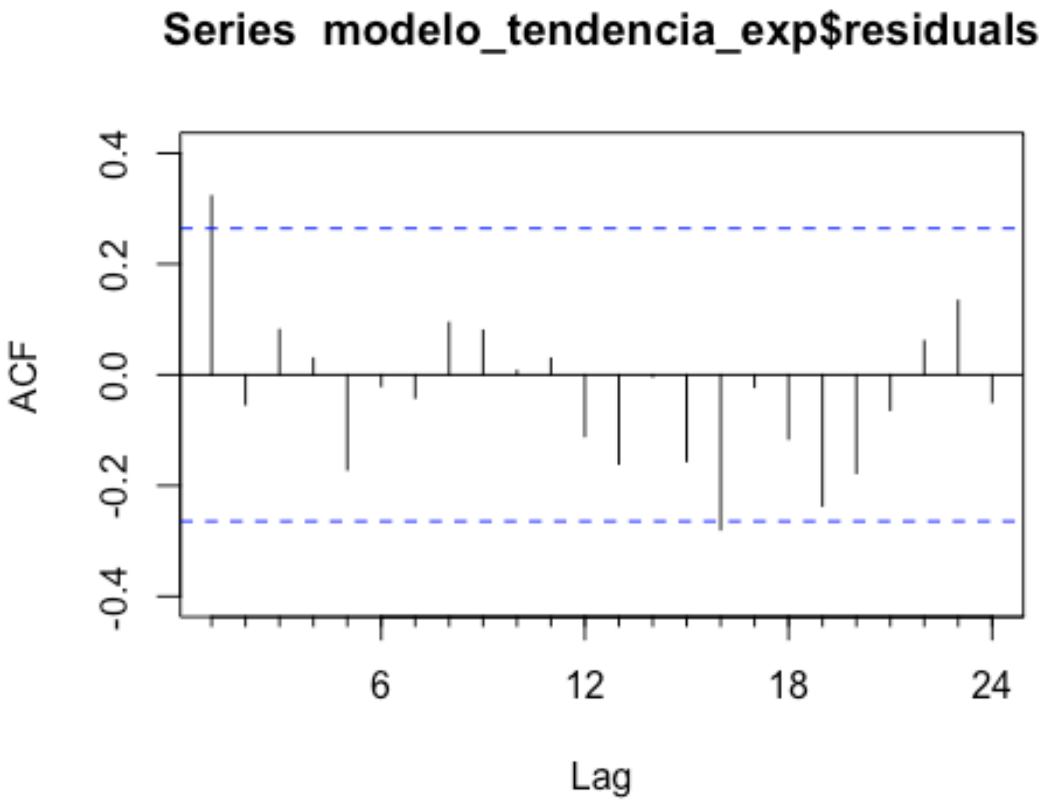
```
#Estima o modelo de tendência exp
modelo_tendencia_exp <- tslm(treinamento_ts ~ trend, lambda=0)

#resumo do modelo
summary(modelo_tendencia_exp)
```

```
##
## Call:
## tslm(formula = treinamento_ts ~ trend, lambda = 0)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1822 -0.2749 -0.0569  0.2597  1.6501
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.895357   0.142909  97.232   <2e-16 ***
## trend        0.001824   0.004440   0.411    0.683
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5227 on 53 degrees of freedom
## Multiple R-squared:  0.003174,   Adjusted R-squared:  -0.01563
## F-statistic: 0.1688 on 1 and 53 DF,  p-value: 0.6829
```

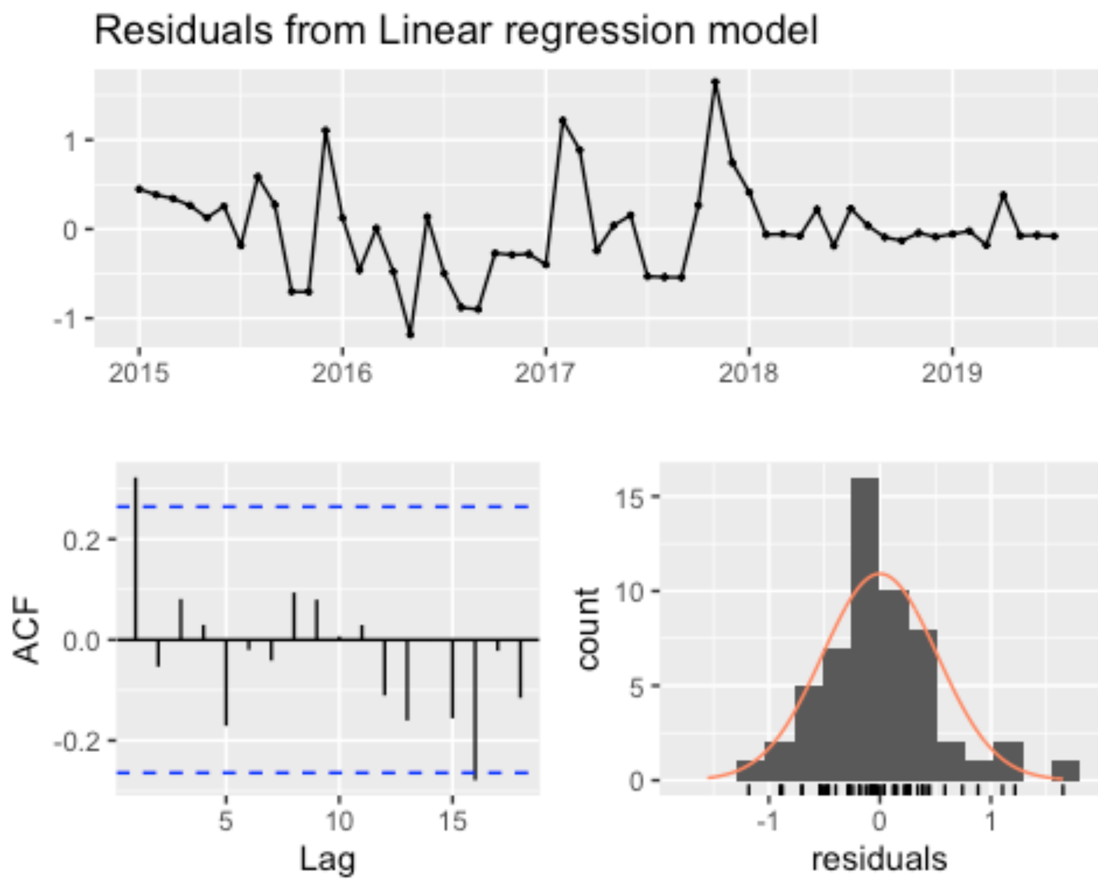
```
#Verificando resíduos

#Plotando os resíduos
plot(modelo_tendencia_exp$residuals, xlab="Tempo", ylab="Resíduos", ylim=
c(-0.5, 0.5), bty="l")
```




```
#calcula a autocorrelação dos resíduos
Acf(modelo_tendencia_exp$residuals)
```




```
#verifica os resíduos com teste de Ljung-Box
checkresiduals(modelo_tendencia_exp, test="LB")
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Linear regression model
## Q* = 9.7166, df = 9, p-value = 0.3739
##
## Model df: 2.   Total lags used: 11
```

```
#CHeca a acuracia
```

Novamente o MAPE foi pior do que o modelo Naïve.

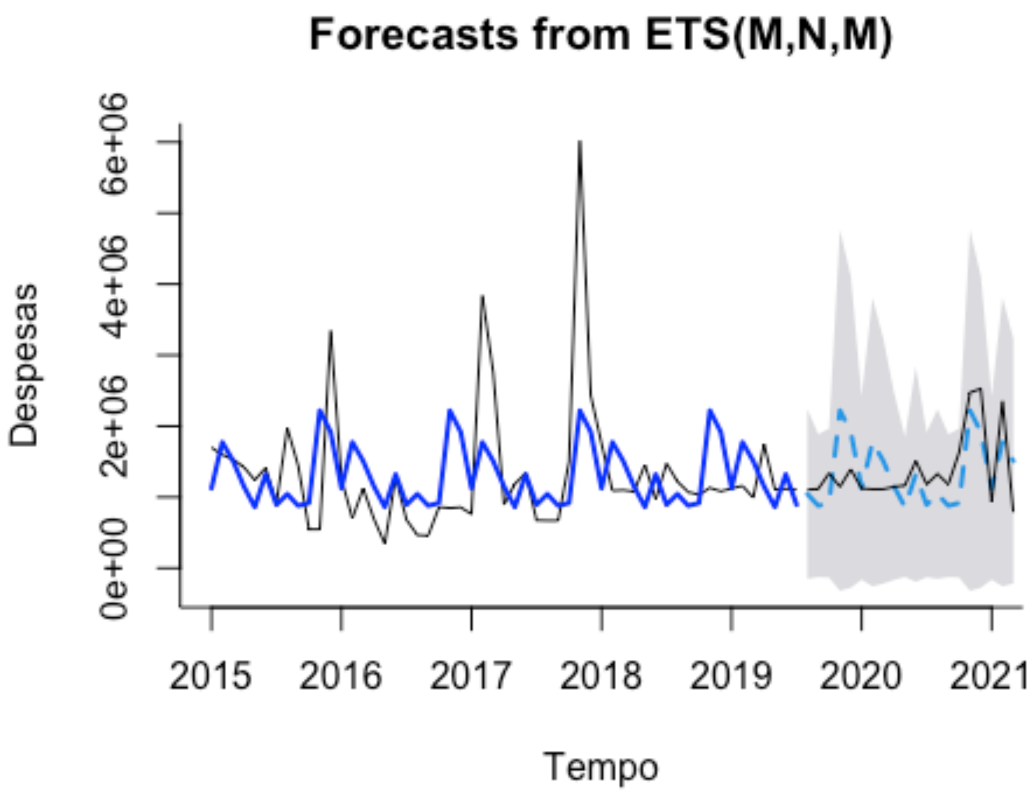
Modelo de suavização exponencial (ZZZ)

```
modelo_ses1 <- ets(treinamento_ts, model = "ZZZ")
summary(modelo_ses1)
```

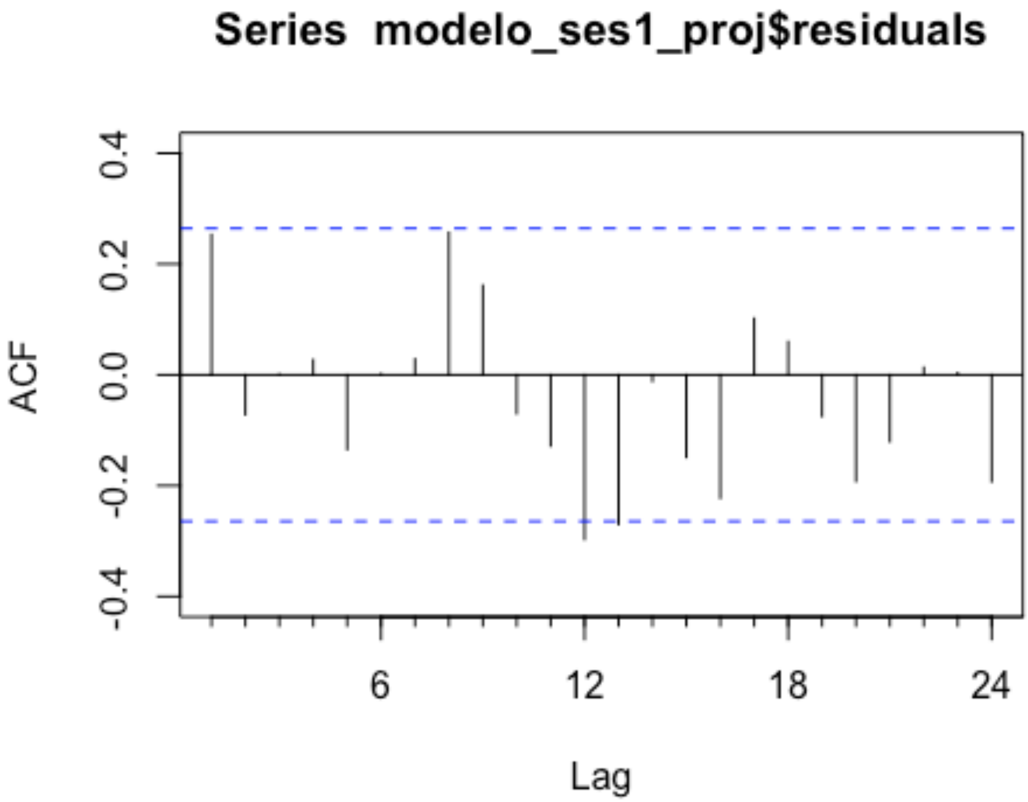
```
## ETS(M,N,M)
##
## Call:
##  ets(y = treinamento_ts, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 1e-04
##     gamma = 1e-04
```

```
##
##   Initial states:
##     l = 1300568.4885
##     s = 1.4752 1.7064 0.7051 0.6755 0.8023 0.686
##            1.0165 0.661 0.8835 1.1596 1.3642 0.8648
##
##   sigma:  0.5835
##
##      AIC     AICc      BIC
## 1717.447 1729.755 1747.557
##
## Training set error measures:
##                    ME    RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 31092.59 835741 561803.4 -22.67877 48.85116 0.5964938 0.2
137468
```

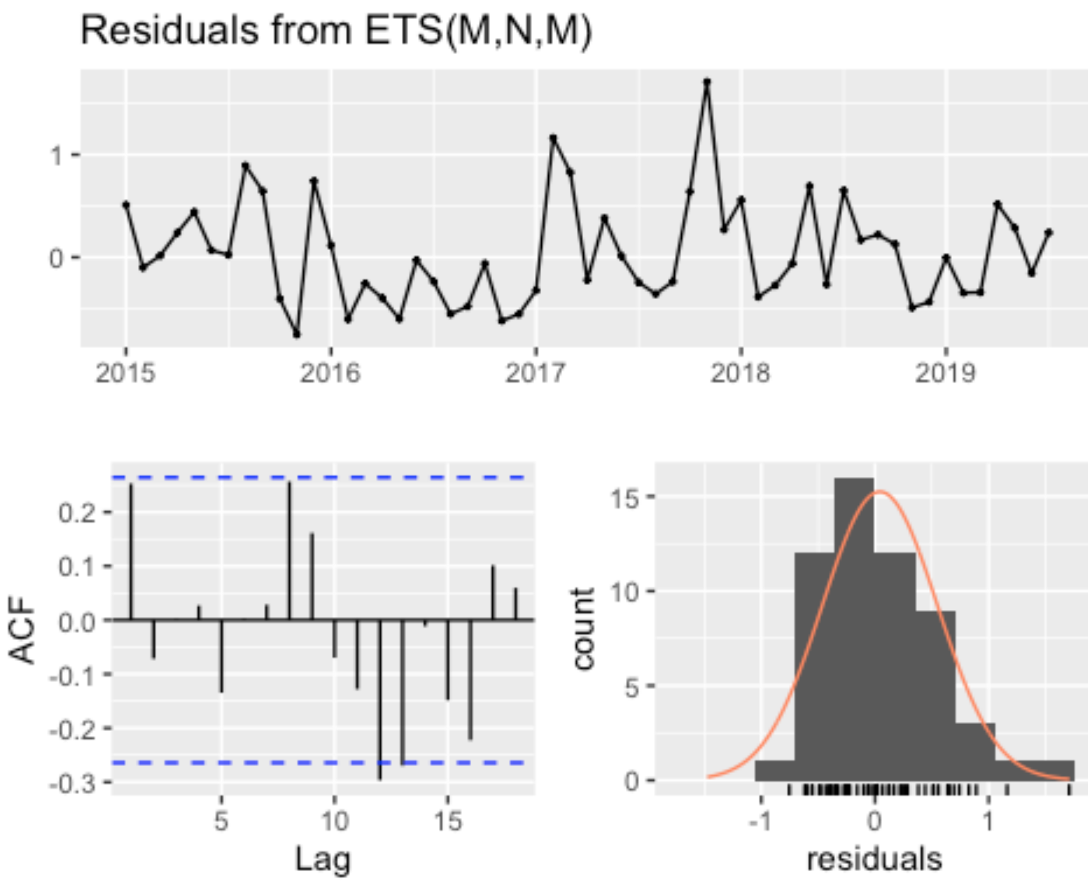
```
modelo_ses1_proj <- forecast(modelo_ses1, h=20, level=0.95)

plot(modelo_ses1_proj, ylim=c(-300000, 6013000), ylab="Despesas", xlab="T
empo", bty="l", xaxt="n", xlim=c(2015,2021), flty=2)
axis(1, at=seq(2015, 2021), labels=format(seq(2015, 2021)))
lines(modelo_ses1$fitted, lwd=2, col="blue")
lines(validacao_ts)
```




```
accuracy(modelo_ses1_proj, validacao_ts)

##                     ME     RMSE      MAE        MPE     MAPE      MASE
ACF1
## Training set 31092.59 835741.0 561803.4 -22.678775 48.85116 0.5964938
0.2137468
## Test set     32893.11 473001.4 390738.7  -3.419636 30.49871 0.4148662
0.1430581
##              Theil's U
## Training set        NA
## Test set     0.8339434

Acf(modelo_ses1_proj$residuals)
```



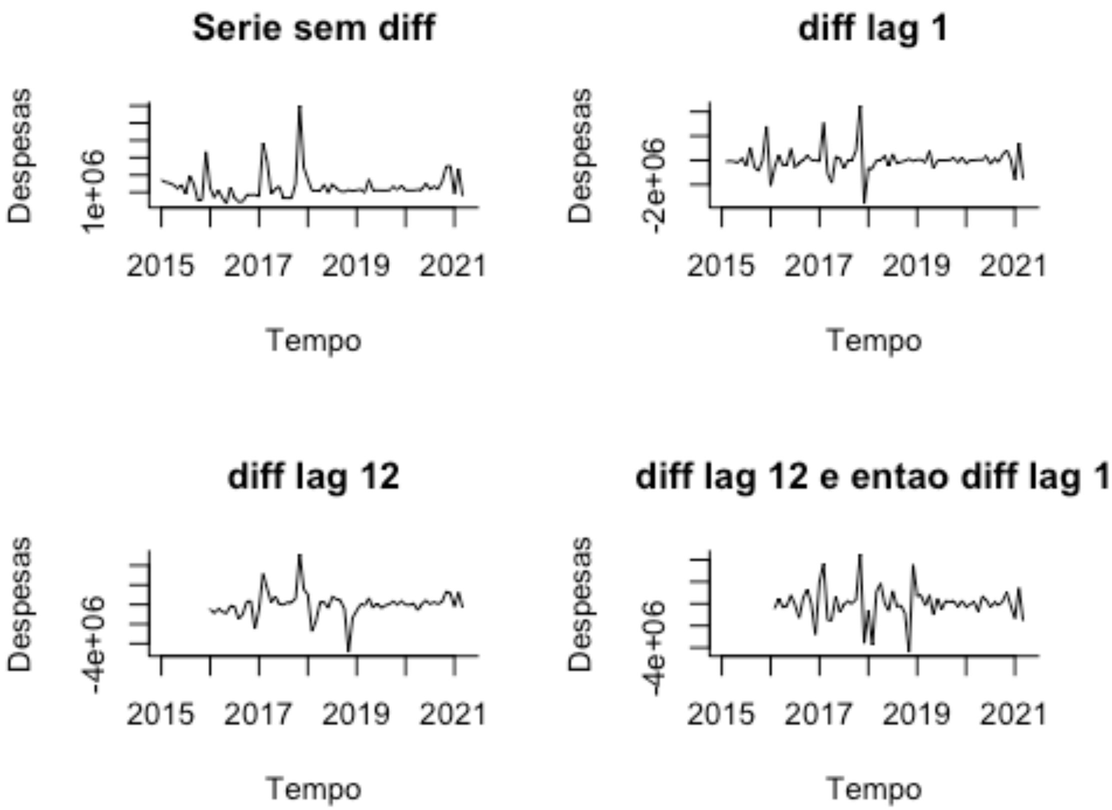

```
checkresiduals(modelo_ses1_proj, test="LB")
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,N,M)
## Q* = 31.379, df = 3, p-value = 7.074e-07
##
## Model df: 14.   Total lags used: 17
```

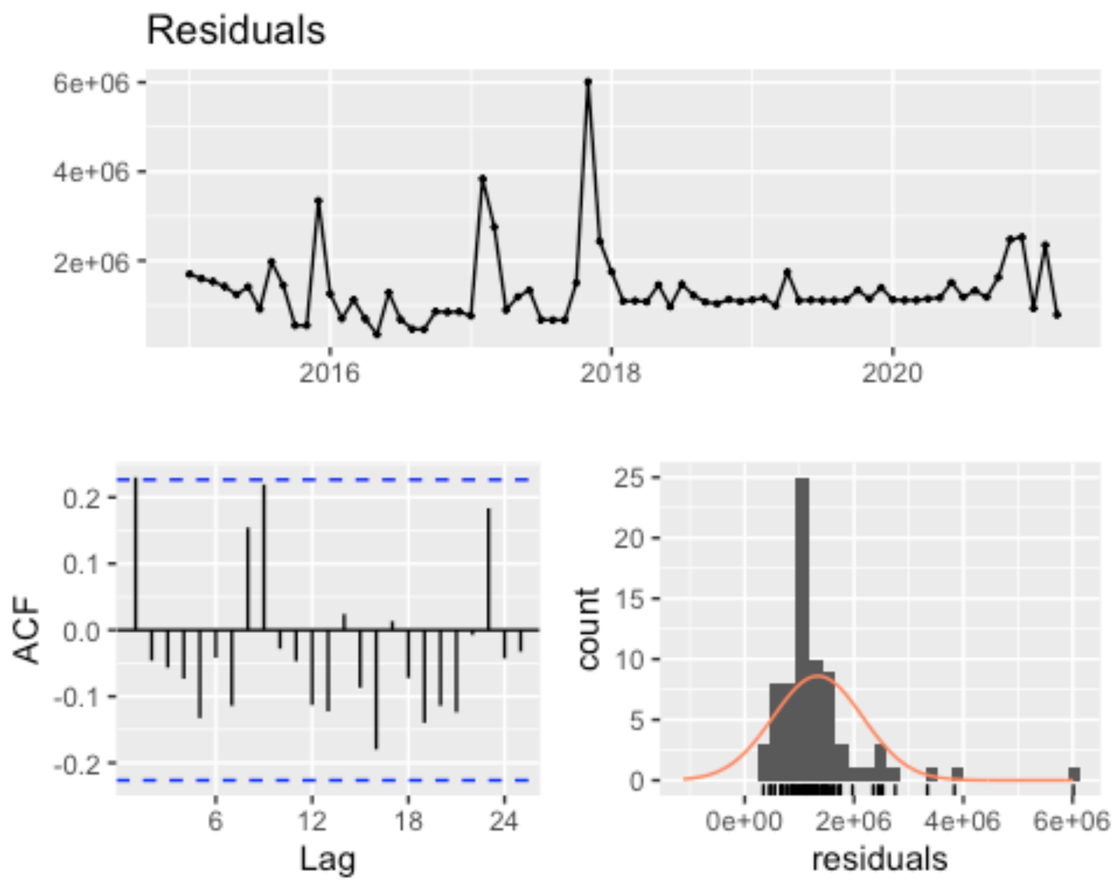
Modelo Arima:

```
#plota o grafica da projecao
par(mfrow=c(2,2))
plot(db_ts, ylab="Despesas", xlab="Tempo", bty="l", xlim=c(2015,2021.25),
main=("Serie sem diff"))
plot(diff(db_ts, lag=1), ylab="Despesas", xlab="Tempo", bty="l", xlim=c(2
015,2021.25), main=("diff lag 1"))
plot(diff(db_ts, lag=12), ylab="Despesas", xlab="Tempo", bty="l", xlim=c(
2015,2021.25), main=("diff lag 12"))
plot(diff(diff(db_ts, lag=12), lag=1), ylab="Despesas", xlab="Tempo", bty
="l", xlim=c(2015,2021.25), main=("diff lag 12 e entao diff lag 1"))
```

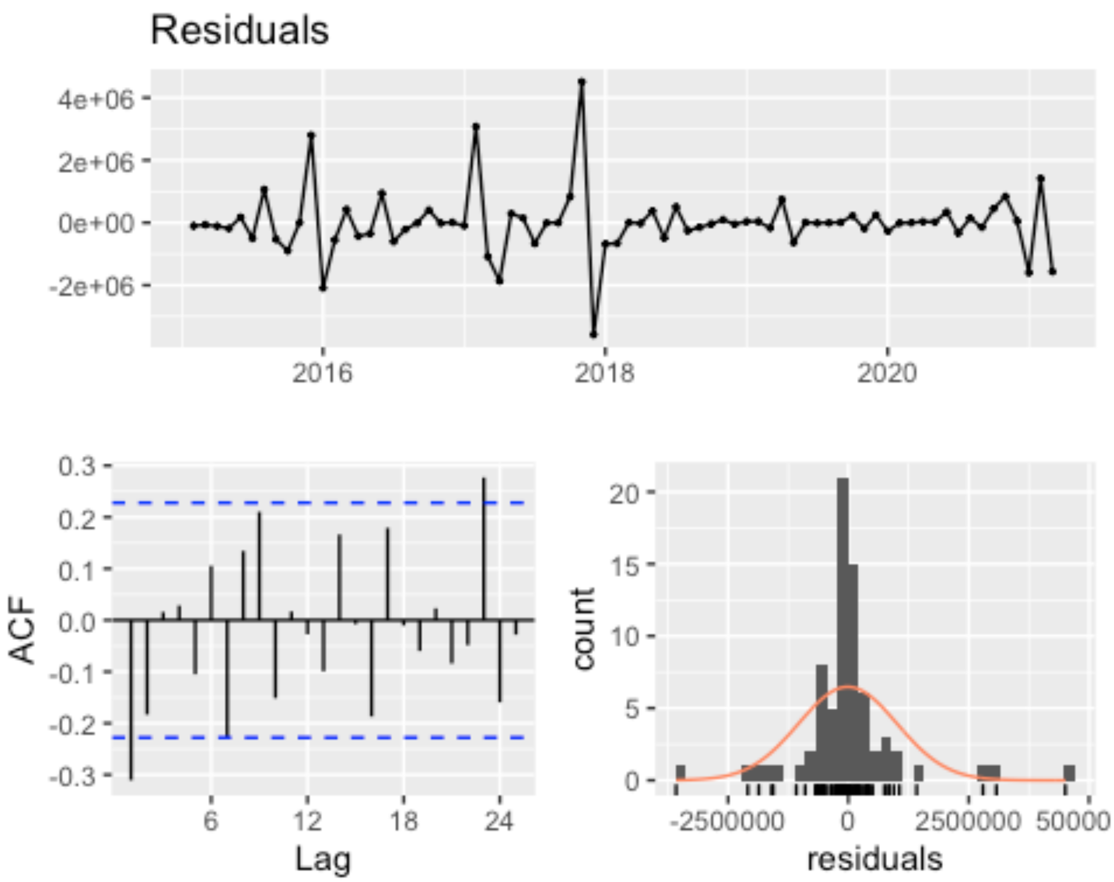



```
#checar estacionariedade
checkresiduals(db_ts)
```

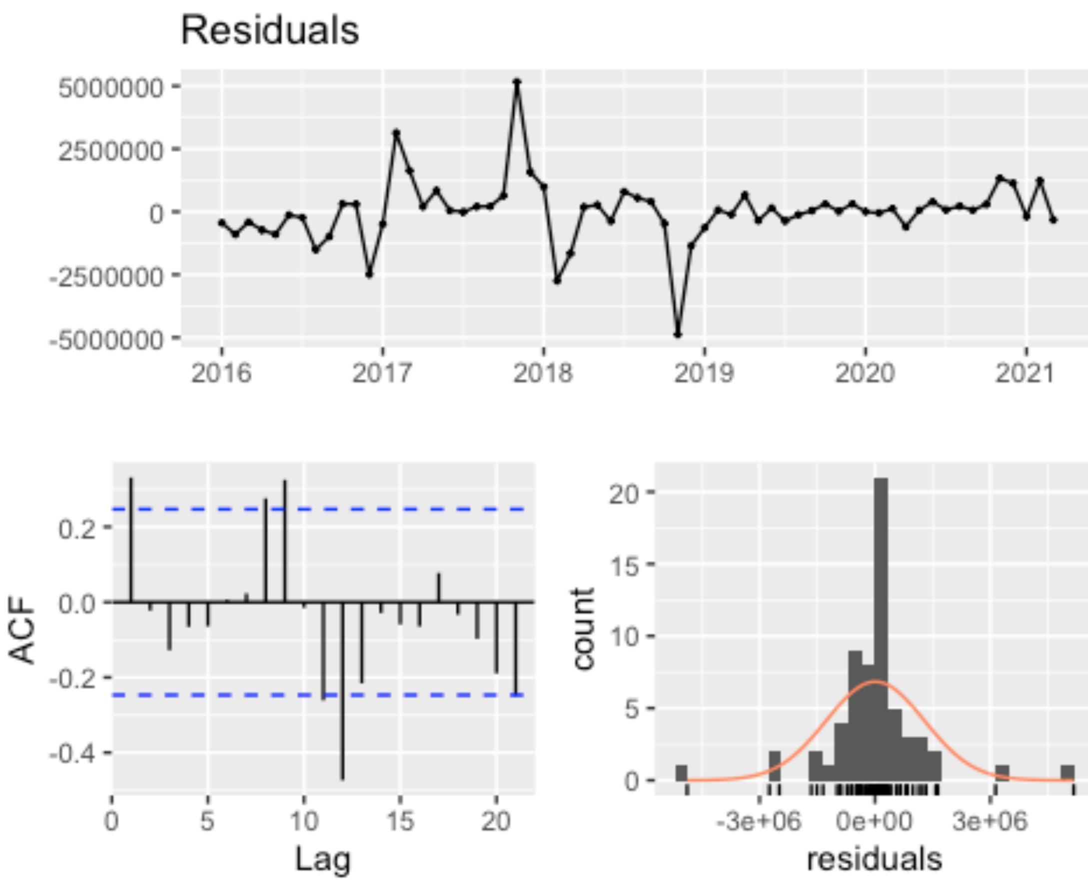
```
## Warning in modeldf.default(object): Could not find appropriate degrees
of
## freedom for this model.
```



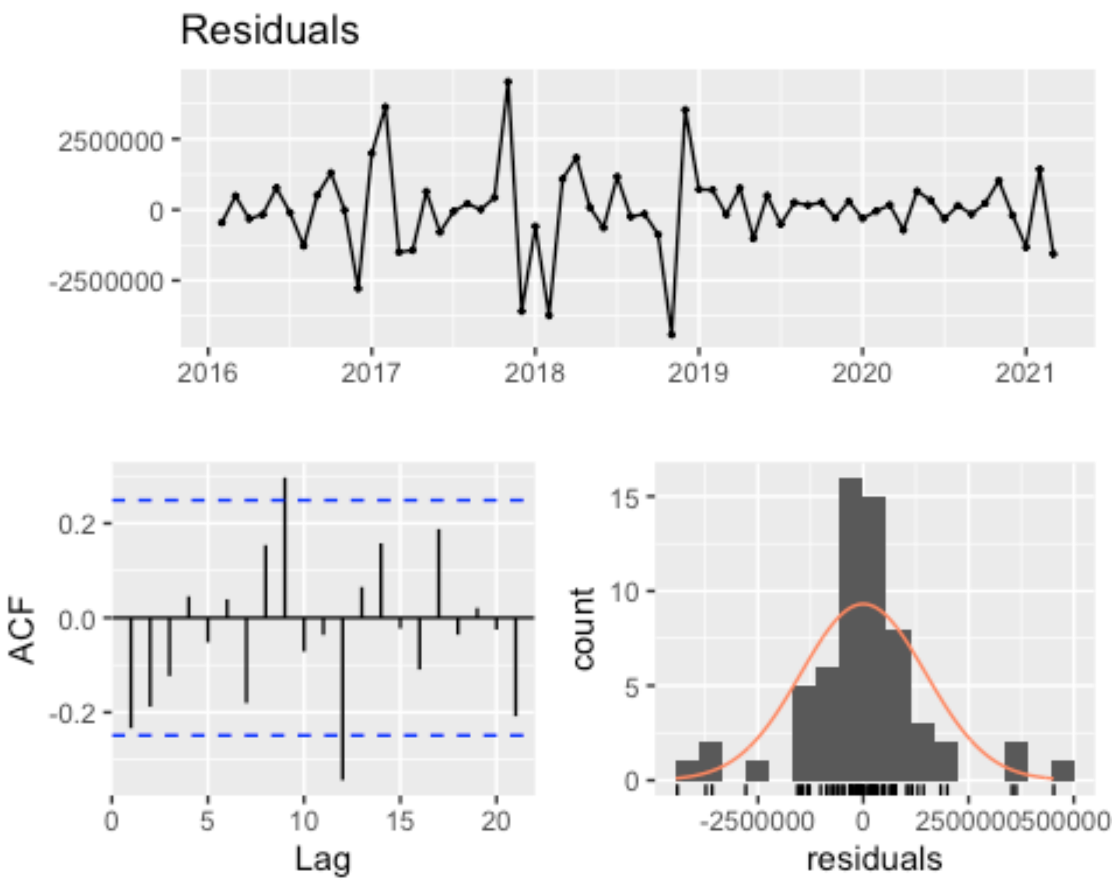

```
checkresiduals(diff(db_ts, lag=1))

## Warning in modeldf.default(object): Could not find appropriate degrees
of
## freedom for this model.
```




```
checkresiduals(diff(db_ts, lag=12))

## Warning in modeldf.default(object): Could not find appropriate degrees
of
## freedom for this model.
```




```
checkresiduals(diff(diff(db_ts, lag=12), lag=1))

## Warning in modeldf.default(object): Could not find appropriate degrees
of
## freedom for this model.
```




```
#diferencia 1 vez
db_ts_diff <- diff(db_ts, lag=1)

#executa o teste de KPSS
summary(ur.kpss(db_ts))

##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 3 lags.
##
## Value of test-statistic is: 0.0621
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739

#executa o teste de KPSS
summary(ur.kpss(db_ts_diff))

##
## #######################
```

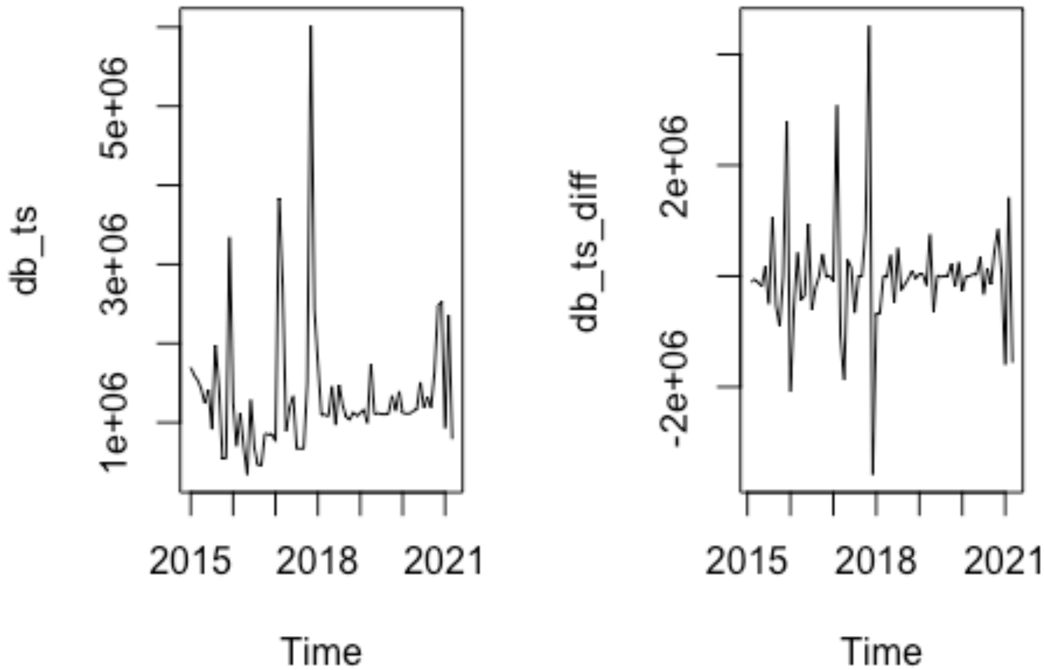
```
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 3 lags.
##
## Value of test-statistic is: 0.0283
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```
#executa o teste de ADF
summary(ur.df(db_ts))
```

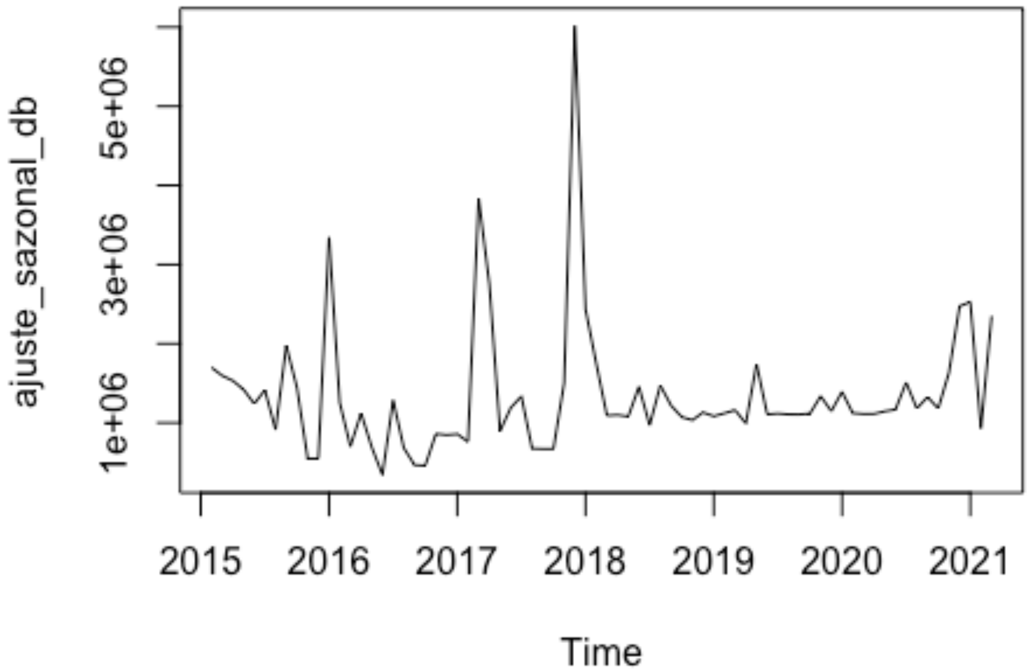
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1654824   -47104   130667   392927  4955334
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1    -0.16690    0.07501  -2.225   0.0293 *
## z.diff.lag -0.23470    0.11726  -2.002   0.0491 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 951900 on 71 degrees of freedom
## Multiple R-squared:  0.1585, Adjusted R-squared:  0.1348
## F-statistic: 6.688 on 2 and 71 DF,  p-value: 0.002182
##
##
## Value of test-statistic is: -2.2251
##
## Critical values for test statistics:
##      1pct  5pct 10pct
## tau1 -2.6 -1.95 -1.61
```

```
#executa o teste de ADF
summary(ur.df(db_ts_diff))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2083031  -346857  -117341   153056  4853907
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1     -1.7326     0.1858  -9.326 6.68e-14 ***
## z.diff.lag   0.3185     0.1162   2.741  0.00777 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 942200 on 70 degrees of freedom
## Multiple R-squared:  0.6862, Adjusted R-squared:  0.6772
## F-statistic: 76.53 on 2 and 70 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -9.3264
##
## Critical values for test statistics:
##      1pct  5pct 10pct
## tau1 -2.6 -1.95 -1.61

#############################################################
# MODELO ARIMA
#############################################################

par(mfrow=c(1,2))
plot(db_ts)
plot(db_ts_diff)
```




```r
par(mfrow=c(1,1))

ajuste_sazonal_db <- db_ts-db_ts_diff

plot(ajuste_sazonal_db)
```



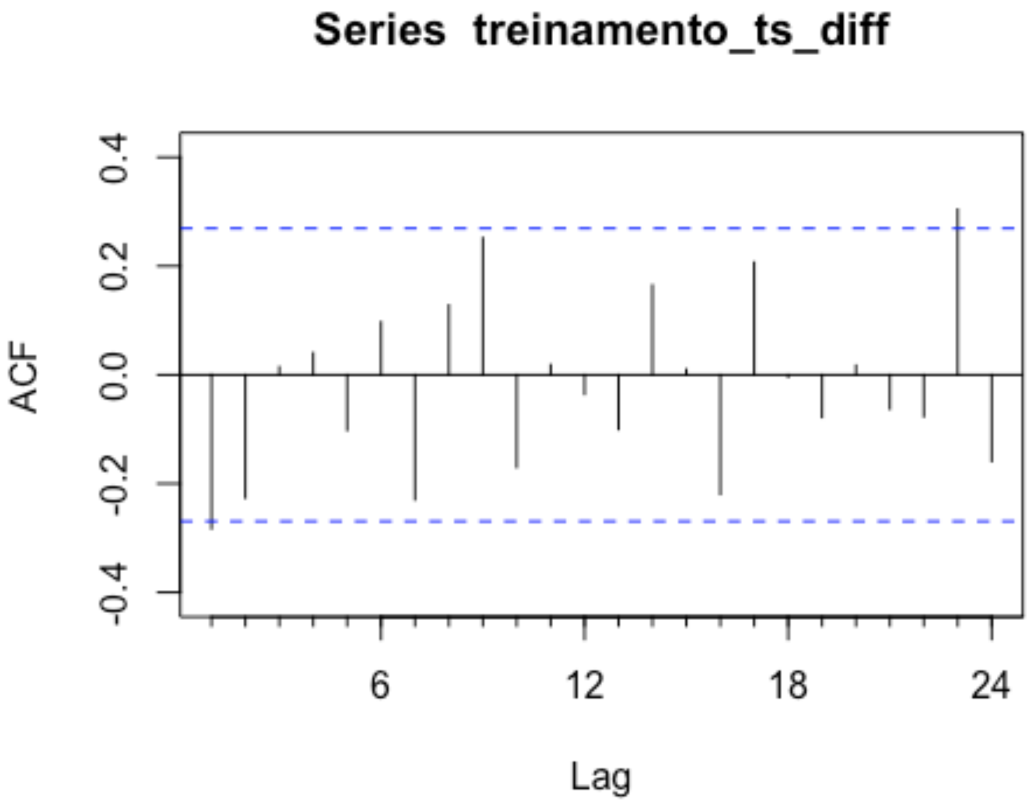

```
#separa as amostras em treinamento e teste

#define o tamanho da amostra de teste

#define o tamanho da amostra de treinamento
amostra_treino <- length(db_ts_diff) - amostra_validacao



#cria a serie temporal de treinamento
treinamento_ts_diff <- window(db_ts_diff, start=c(2015, 1), end=c(2015, a
mostra_treino))

## Warning in window.default(x, ...): 'start' value not changed

#cria a serie temporal de teste
validacao_ts_diff <- window(db_ts_diff, start=c(2015, amostra_treino + 1)
, end=c(2015,amostra_treino + amostra_validacao))


#executa o teste de KPSS
summary(ur.kpss(treinamento_ts_diff))
```

```
## 
## #######################
## # KPSS Unit Root Test #
## #######################
## 
## Test is of type: mu with 3 lags.
## 
## Value of test-statistic is: 0.0368
## 
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

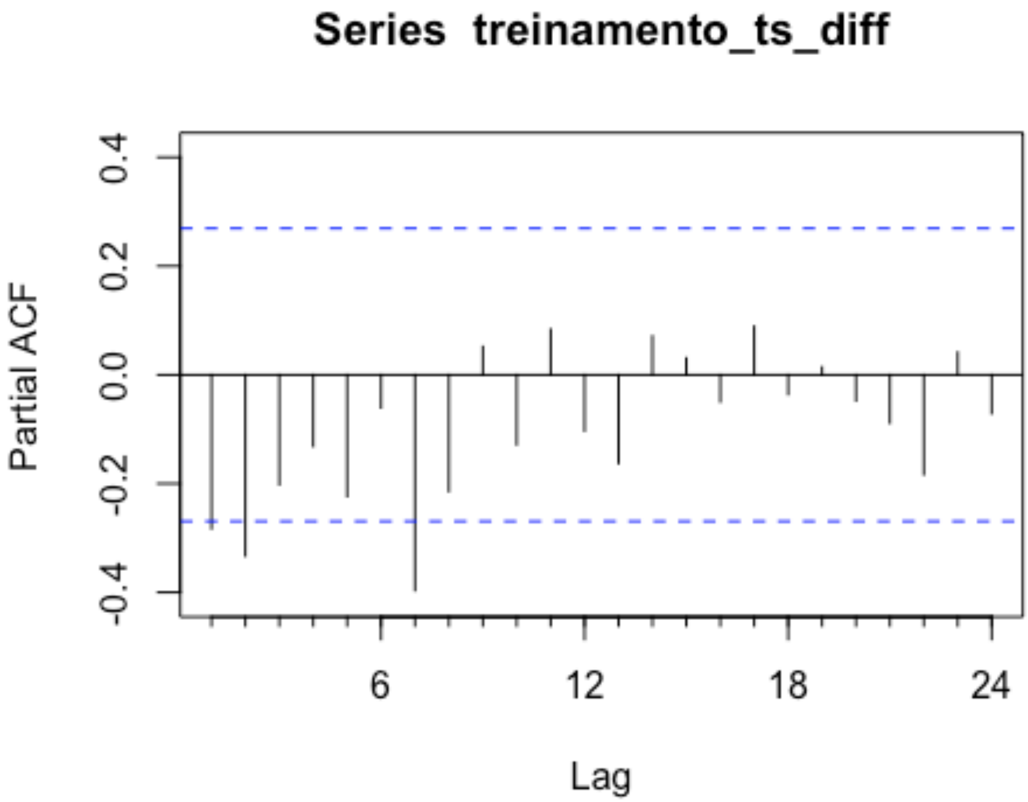
```
#executa o teste de ADF
summary(ur.df(treinamento_ts_diff))
```

```
## 
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
## 
## Test regression none
## 
## 
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -2117586  -433296  -204615    59191  4823541 
## 
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)    
## z.lag.1     -1.7127     0.2161  -7.926 2.48e-10 ***
## z.diff.lag   0.3351     0.1351   2.481   0.0166 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1065000 on 49 degrees of freedom
## Multiple R-squared:  0.6815, Adjusted R-squared:  0.6685 
## F-statistic: 52.42 on 2 and 49 DF,  p-value: 6.713e-13
## 
## 
## Value of test-statistic is: -7.9263 
## 
## Critical values for test statistics: 
##      1pct  5pct 10pct
## tau1 -2.6 -1.95 -1.61
```

```
#calcula a ACF
Acf(treinamento_ts_diff)
```

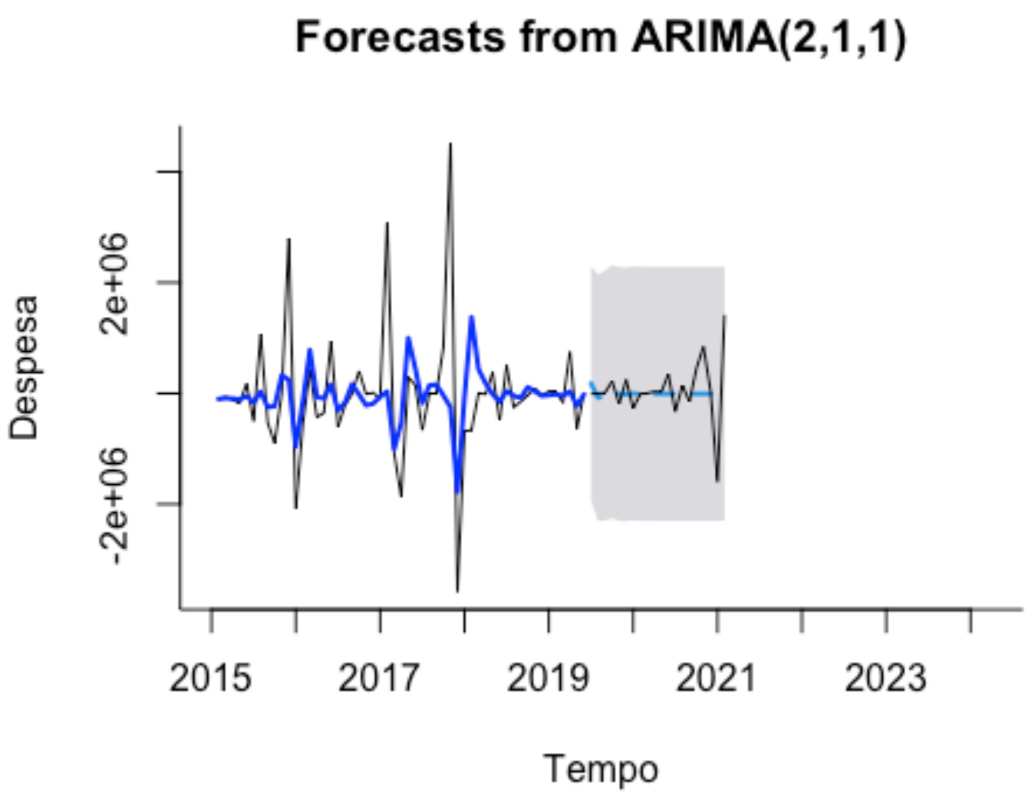



```
#calcula a PCF
Pacf(treinamento_ts_diff)
```




```
#Modelo Arima
Modelo_ARIMA <- Arima(treinamento_ts_diff, order = c(2,1,1))

#resumo modelo
summary(Modelo_ARIMA)

## Series: treinamento_ts_diff
## ARIMA(2,1,1)
##
## Coefficients:
##           ar1      ar2     ma1
##       -0.3578  -0.3104  -1.000
## s.e.   0.1307   0.1290   0.051
##
## sigma^2 estimated as 1.136e+12:  log likelihood=-796.59
## AIC=1601.18   AICc=1602.04   BIC=1608.99
##
## Training set error measures:
##                    ME    RMSE      MAE      MPE     MAPE      MASE
ACF1
## Training set 28465.6 1024914 570511.2 6371.105 6675.536 0.4752428 -0.0
8119995
```

```r
#projeta os proximos 12 meses
modelo_ARIMA_proj <- forecast(Modelo_ARIMA, h=amostra_validacao, level=0.
95)
#plota o grafica da projecao
plot(modelo_ARIMA_proj, ylab="Despesa", xlab="Tempo", bty="l", xaxt="n",
xlim=c(2015,2024.25), flty=2)

axis(1, at=seq(2015, 2024, 1), labels=format(seq(2015, 2024, 1)))

lines(Modelo_ARIMA$fitted, lwd=2, col="blue")

lines(validacao_ts_diff)
```




```r
#verifica precisao
accuracy(modelo_ARIMA_proj, validacao_ts_diff)

##                    ME      RMSE      MAE        MPE      MAPE      MAS
E
## Training set 28465.60 1024914.1 570511.2 6371.10531 6675.5360 0.475242
8
## Test set     64217.44  548342.3 332683.2   17.39917  547.5184 0.277129
1
##                     ACF1 Theil's U
```

```
## Training set -0.08119995        NA
## Test set     -0.38972754  1.276779
```

```
#função auto.arima
auto.arima(treinamento_ts_diff, seasonal = FALSE, stepwise=FALSE, approxi
mation = FALSE)
```

```
## Series: treinamento_ts_diff
## ARIMA(2,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2
##       -0.3706  -0.3230
## s.e.   0.1288   0.1271
##
## sigma^2 estimated as 1.091e+12:  log likelihood=-808.87
## AIC=1623.73   AICc=1624.22   BIC=1629.64
```

O modelo ARIMA teve um desempenho pior entre os modelos apresentados.

Conclusão:

Após todos os modelos testados e analisados conclui-se pela performance do modelo de suavização exponencial. Seus erros RMSE e MAPE são inferiores a todos os outros modelos, e a autocorrelação de seus resíduos não é estatisticamente relevante.