Um simples modelo ARIMA para o ICMS do RS usando R

Roberto Balau Calazans

A existência de um ambiente de desenvolvimento integrado (IDE, sigla em inglês) para o Software livre R representa um avanço ao trabalho científico, já que permite aumentar a produtividade do pesquisador, reduzindo as tarefas de coleta, tratamento, análise e exposição de dados. O RStudio é a i*nterface* desse *software,* facilitando sua utilização. Ambos são gratuitos e *open source*.

Em diversos sítios eletrônicos (nacionais e internacionais) podem ser encontradas diversas rotinas para a construção de modelos, bem como explicações diversas para os testes de verificação relativos à estacionariedade, à normalidade, ao diagnóstico e à previsão de modelos Arima (p, d, q) para uma dada série temporal. O termo "p" refere-se à ordem autoregressiva do modelo; "d" é relativo à diferenciação; e "q" refere-se às médias móveis.

Neste exercício, procura-se apresentar o script de estimação e previsão de uma série temporal, levando-se em conta as boas práticas estatísticas disponíveis. Obviamente, este *script* não é conclusivo e serve apenas como referência para aqueles se interessam pela aplicação de algoritmos do R em projeções de séries temporais univariadas. Para essa demonstração foi escolhida a série temporal do ICMS nominal do Estado do Rio Grande do Sul, disponível nos seguintes endereços eletrônicos: http://receitadados.fazenda.rs.gov.br/ ou https://financasrs.com.br/estatisticas-fiscais/.

Segue abaixo a rotina elaborada por este autor. As críticas e os aperfeiçoamentos a sugestão proposta podem ser encaminhados diretamente ao *site*.

```
###################
#Pacotes requeridos
###################
library(tidyverse)
library(forecast)
library(ggplot2)
library(tseries)
library(urca)
library(TSstudio)#ts_plot

######################
#Construindo a série TS
######################

#Buscando dados no Excell
library(readxl)
icmsn <- read_excel("Downloads/icmsn.xlsx")
View(icmsn)
class(icmsn)#ts
```

```
#Definindo um objeto TS
icmsn<-ts(icmsn,start = c(2013,1), freq=12)
summary(icmsn)

##################
#Plotando a série TS
##################
par(mfrow=c(1,1))
#Com TS
plot(icmsn, main = "Icms nominal do RS", ylab="ICMS nominal ", xlab=
"Meses", col=1, lty=1)
grid(col="darkgrey",lwd=1)
ggtsdisplay(icmsn)
#Com Forecast
autoplot(icmsn) + ggtitle("ICMS nominal do RS, 2013-2021") +labs(x="Meses",
y= "R$ milhões")#Forecast
ts_plot(icmsn, line.mode = "lines", title = "ICMS nominal do RS, 2013-2021")

########################################
#Diferenciando, vendo os resíduos e Gráficos
########################################

dicmsn<-diff(icmsn)
plot(dicmsn)
lines(dicmsn-fit.icmsn$residuals, col="red")

ggAcf(dicmsn)+ggtitle("ACF do ICMS nominal do RS")
ggPacf(dicmsn)+ggtitle("PACF do ICMS nominal do RS")

#Gráficos da série em nível e diferenciada
Grafico.icmsn<-cbind(icmsn,dicmsn)
autoplot(Grafico.icmsn, facets=TRUE) + ggtitle("ICMS nominal do RS, nível e
diferenciada") + labs(y="R$ milhões", x="Meses")

#########################
#Dessazonalizando os dados
#########################
library(seasonal)
icmsn.sazonal<-seas(icmsn)
plot(icmsn)
lines(final(icmsn.sazonal), col=2)
legend("topleft", legend = c("com sazonalidade", "sem sazonalidade"), lty =
c(1,1), col= c(1,2))


seasonplot(icmsn, col = rainbow(12), year.labels = TRUE, type = "o", pch=16)
ggseasonplot(icmsn) + ggtitle("ICMS nominal do RS, 2013-2021")
+labs(x="Meses", y= "R$ milhões")
```

```
ggseasonplot(dicmsn) + ggtitle("ICMS nominal do RS, 2013-2021")
+labs(x="Meses", y= "R$ milhões")
ggsubseriesplot(icmsn)
ggsubseriesplot(dicmsn)

############################
#Decompondo a série com TStudio
############################

ts_decompose(icmsn, type = "additive", showline = TRUE)
#Decompondo a série com TS
plot(decompose(icmsn))
par(mfrow=c(1,1))
qqnorm(icmsn)

########################
#Analisando os ACF e PACF
########################

ggAcf(icmsn)+ggtitle("ACF do ICMS nominal do RS")
ggPacf(icmsn)+ggtitle("PACF do ICMS nominal do RS")

#################
#Dickey Fuller Test
#################
library(forecast)
ndiffs(icmsn)
nsdiffs(icmsn)

library(tseries)
adf.test(icmsn)
adf.test(diff(icmsn))
library(urca)
summary(ur.df(icmsn, type="none", lags=1, "AIC"))
summary(ur.df(icmsn, type="drift", lags=1,"AIC"))
summary(ur.df(icmsn, type="trend", lags=1,"AIC"))
summary(ur.df((diff(icmsn)), selectlags = "AIC"))#Rejeita-se Ho:é estacionária

#################
#Phillips Perron Test
#################
library(urca)
pp.test(icmsn)
pp.test(dicmsn)

##########
#KPSS Test
##########
kpps.icmsn<-ur.kpss(icmsn,type="tau",lags = "short")
summary(kpps.icmsn)
```

```
###############
#Outros modelos
###############

fit.icmsn.naive<-snaive(icmsn
print(summary(fit.icmsn.naive))
checkresiduals(fit.icmsn.naive)
plot(fit.icmsn.naive)

fit.icmsn.ets<-ets(icmsn)
print(summary(fit.icmsn.ets))
checkresiduals(fit.icmsn.ets)
plot(fit.icmsn.ets)
forecast(fit.icmsn.ets,h=6, level = c(80,95))
plot(icmsn)
lines(fitted(fit.icmsn.ets) [,1], col="red", lty=2, lwd=3)
legend('topleft', legend=c("ICMS nominal do RS", "com ajuste ETS"), bty=
"n",col=c("black", "red"), lty=c(1,2), lwd=c(1,3))

fit.icmsn.HW<-HoltWinters(icmsn)
fit.icmsn.HW
print(summary(fit.icmsn.HW))
checkresiduals(fit.icmsn.HW)
plot(fit.icmsn.HW)
forecast(fit.icmsn.HW,h=6, level = c(80,95))
plot(icmsn)
lines(fitted(fit.icmsn.HW) [,1], col="red", lty=2, lwd=3)
legend('topleft', legend=c("ICMS nominal do RS", "ajuste com HoltWinter"), bty=
"n",col=c("black", "red"), lty=c(1,2), lwd=c(1,3))


##############
#ARIMA MODEL
##############

#Particionando a série: test data e training data
library(TSstudio)
icmsn.P<-ts_split(icmsn, sample.out = 12)
Treino.icmsn<-icmsn.P$train
Teste.icmsn<-icmsn.P$test
length(Treino.icmsn)
length(Teste.icmsn)

#auto.arima e diagnóstico
arima_diag(Treino.icmsn)
auto.arima(Treino.icmsn, stepwise = FALSE, trace = TRUE)

fit.Treino.icmsn<-arima(Treino.icmsn, order = c(1,1,0), seasonal = c(1,0,1))
```

```
summary(fit.Treino.icmsn)
checkresiduals(fit.Treino.icmsn)
Previsao.icmsn<- forecast(fit.Treino.icmsn,h=12)
test_forecast(actual = icmsn, forecast.obj = Previsao.icmsn, test = Teste.icmsn)
accuracy(Previsao.icmsn, Teste.icmsn)

#Verificando a normalidade da serie particionada
par(mfrow=c(2,2))
plot(resid(fit.Treino.icmsn))
qqnorm(resid(fit.Treino.icmsn))
qqline(resid(ffit.Treino.icmsn))
acf(resid(fit.Treino.icmsn))
pacf(resid(fit.Treino.icmsn))

#Verificando a qualidade do ajustamento
library(lmtest)
coeftest(fit.icmsnA)
par(mfrow=c(1,1))
plot(icmsnA, ylab= "ICMS Regressão-A", main= "ICMS série particionada")
lines(icmsnA-fit.icmsnA$residuals, col= "blue")

######################
#Aplicando Auto.Arima
######################

auto.arima(icmsn, stepwise = FALSE, trace = TRUE, approximation = FALSE)
fit.icmsn<-arima(icmsn,order=c(1,1,0), seasonal = list(order=c(1,0,1)))
summary(fit.icmsn)
print(checkresiduals(fit.icmsn))
forecast(fit.icmsn,h=6, level = c(80,95))
predict(fit.icmsn, n.ahead = 12)

library(stargazer)
stargazer(fit.icmsn, type= "text")

#Verificando os resíduos do modelo
par(mfrow=c(2,2))
plot(resid(fit.icmsn))
qqnorm(resid(fit.icmsn))
qqline(resid(fit.icmsn))
acf(resid(fit.icmsn))
pacf(resid(fit.icmsn))

#Forecast
library(forecast)
par(mfrow=c(1,1))
Prev.icmsn<-forecast(fit.icmsn, h=12)
plot(Prev.icmsn)
autoplot(Prev.icmsn)
```

#Predict

```
par(mfrow=c(1,1))
plot(icmsn, ylab= "R$ milhões", main= "ICMS nominal do RS")
Prev.icmsn1<-predict(fit.icmsn, n.ahead = 12)
Prev.icmsn1
par(mfrow=c(1,1))
plot(icmsn)
lines(Prev.icmsn1$pred, col="red", lty=1)
```

#Ljung-Box Test

```
library(FitAR)
LBQPlot(residuals(fit.icmsn),36)
```

#Raízes Unitárias

```
autoplot(fit.icmsn)
```

#Jarque Bera Test

```
library(tseries)
tsdiag(fit.icmsn)
```